MAK TECHNOLOGIES

VR-Engage User Guide



Version 2.2

ST Engineering North America

Copyright © 2026 MAK Technologies
All rights Reserved. Printed in the United States.

Under copyright laws, no part of this document may be copied or reproduced in any form without prior written consent of MAK Technologies.

VR-Exchange™, VR-TheWorld™, VR-Vantage™, DI-Guy™, and DI-Guy Scenario™ are trademarks of MAK Technologies. MÄK Technologies®, VR-Forces®, RTIspy®, B-HAVE®, and VR-Link® are registered trademarks of MAK Technologies.

GL Studio® is a registered trademark of The DiSTI® Corporation.

Portions of this software utilize SpeedTree® technology (©2008 Interactive Data Visualization, Inc.). SpeedTree is a registered trademark of Interactive Data Visualization, Inc. All rights reserved.

RFView® is a registered trademark of Information Systems Laboratories (ISL), Inc.

SilverLining™ is a trademark of Sundog Software.

Terrain Profiles are based in part on the work of the Qwt project (http://qwt.sourceforge.net).

3ds Max® is a registered trademark of Autodesk, Inc.

All other trademarks are owned by their respective companies.

For third-party license information, visit the MAK ONE Product Support Knowledge Base (at https://www.mak.com/knowledge) and refer to the Third-Party/Open Source List under Frequently Asked Questions (FAQ).

MAK Technologies
10 Fawcett Street, Suite 204
Cambridge, MA 02138 USA

Voice: +1 617.876.8085
Fax: +1 617.876.9208

info@mak.com

www.mak.com

Revision VRN-2.2-02-260107



Contents

PREFACE

How the Manual Is Organized 16
Documentation Set 17
Try the Help 18
Links Between PDFs 18
MAK Products 18
How to Contact Us 21
Telephone 21
E-mail 21
Internet 21
Post 21
Document Conventions 21
Mouse Button Naming Conventions 22
Third-Party Licenses 22

I. INTRODUCTION, INSTALLATION, AND STARTUP

1. Introduction to VR-Engage

1.1. VR-Engage Features 26
1.1.1 VR-Engage Versions 26
1.1.2 Simulation (Role-Playing) Capabilities 27
1.1.3 User Interface 32
1.1.4 Image Generator 33
1.1.5 Terrain 33
1.2. Integration with VR-Forces 33
1.3. Integration with Other MAK ONE Products 34
1.3.1 After-Action Review Using MAK Data Logger 34

1.3.2 Network Interoperability with Other Federates 34
1.4. Extensibility 34
1.5. Licensing 35
1.5.1 Vortex License (for Ground Vehicle Dynamics) 35
1.5.2 RTDynamics License (For Aircraft) 35
1.5.3 License Requirements for Hosting a VR-Forces Session 36

2. Installing VR-Engage

2.1. Installing VR-Engage 38
2.1.1 Installation and Data Packages 38
2.1.2 Installing VR-Engage 38
2.1.3 Installing VR-Engage PDB Files for Debugging Purposes 40
2.1.4 Installing the Toolkit 42
2.1.5 The makData Package 42
2.1.6 Uninstalling from Windows 44
2.2. Upgrading VR-Engage Simulation Model Sets 45
2.3. The VR-Engage Directory Structure 45
2.4. Installing and Setting Up the MAK License Manager 46
2.4.1 Specifying the License Server 47
2.5. Installing an RTI 48
2.5.1 Installing the MAK RTI 49
2.6. Input Devices 50
2.6.1 HMD Support 50
2.7. Factors that Affect Performance 50
2.8. Using MAK ONE Products on Virtual Machines 51
2.9. Localizing the Graphical User Interface 51
2.9.1 Translating Other Interface Files 53
2.9.2 Applying the Language Files 53

3. Running VR-Engage

3.1. Starting VR-Engage and Joining a Session 56
3.1.1 Hosting a VR-Forces Session 57
3.1.2 Starting an Unhosted Session 58
3.1.3 Admin, User, and Lockdown Modes 59
3.1.4 Hosting an Entity in VR-Engage While Participating in a VR-Forces Session 60
3.2. Choosing a Role 63
3.2.1 Choosing an Existing Role 64
3.2.2 Adding an Entity to a Session 65
3.2.3 Using an HMD with an Entity 66
3.2.4 Assigning Roles from VR-Forces 67

3.3. Connecting to VR-Forces 69
3.4. Changing Your Role 70
3.4.1 Changing Your Role in the Choose Role Panel 70
3.4.2 Changing Your Role by Embarking on a Vehicle 70
3.4.3 Switching Roles on a Vehicle 71
3.5. Entity Engagement and Scenario State 71
3.6. The Application Menu 72
3.7. Starting a New Session 73
3.8. Configuring the Simulation Connection 73
3.8.1 HLA Connection Settings 74
3.8.2 DIS Connection Settings 76
3.9. Playing in Windowed Mode 77
3.10. Introduction to Head-Mounted Displays 78
3.10.1 Prerequisites for Using an HMD 78
3.10.2 Resetting the Tracking Sensor 78
3.10.3 Calibrating the Eye Tracking for Varjo 79
3.10.4 VR Anchoring for Mask Tracking 79
3.10.5 Enabling and Disabling Virtual Reality Rendering 80
3.10.6 Configuration Options for Virtual and Mixed Reality 81
3.10.7 Setting Up Mixed Reality 82
3.10.8 Optimizing Performance with HMDs 84
3.11. Creating a Startup Configuration 85
3.11.1 Minion and Boss Configuration 85
3.12. Starting Multiple Simulation Engines 86
3.13. Command-Line Options 87
3.13.1 Command-Line Options for vrEngage 87
3.13.2 Command-Line Options for vrEngageSim 102
3.13.3 Command-Line Options for Player Stations 112
3.14. Managing Plug-ins 112
3.14.1 Loading Plug-ins 113
3.14.2 Adding a Plug-in 114
3.14.3 Specifying the DLLs for a Plug-in 114
3.14.4 Adding a Plug-in Configuration 115
3.14.5 Deleting a Plug-in Configuration 116
3.14.6 Deleting a Plug-in 116

4. Optimizing Performance

4.1. Improving VR-Engage Performance 118
4.1.1 GUI Performance 118
4.1.2 Simulation Engine Performance 119

4.2. Displaying Performance Statistics 120
4.2.1 Displaying VR-Engage Performance Statistics 120
4.2.2 Using the Tracy Client 121
4.3. Configuring Graphics Quality 122
4.3.1 Alpha to Coverage 122
4.3.2 Using the Ambient Occlusion Effect 124
4.3.3 Configuring Ambient Occlusion 125
4.4. DI-Guy Performance Settings 127
4.5. Specifying Terrain Categories 128
4.6. Using the Developer Debug Overlay 128
4.7. Using the VR-Engage Debug Menu 130

II. PLAYER FEATURES

5. Common Player and Entity Features

5.1. The Action Menu 134
5.1.1 Selecting Options on the Action Menu 134
5.2. Using the Radio 135
5.2.1 Using MAK Radio in VR-Forces 136
5.2.2 Using the Standalone Radio Application 136
5.2.3 Using the Intercom or a VOX Radio 138
5.2.4 Configuring the MAK Radio 138
5.2.5 Audio Settings 141
5.2.6 Jamming Radios 142
5.2.7 Enabling the Radio by Default 142
5.3. Text Chat 142
5.3.1 VR-Engage Text Chat Commands 143
5.3.2 Using the Text Chat in VR-Forces 144
5.3.3 Configuring Text Chat Channels for a Scenario 145
5.4. Dynamic Terrain Effects on Movement 147
5.5. Changing the View Mode 147
5.6. Starting VR-Forces Events 147
5.7. Introduction to Sound Effects 148
5.7.1 Enabling Sound Effects 148
5.7.2 Configuring Sound Effects 149
5.7.3 Decibels Configuration 150
5.7.4 Associating Sound Files with Entities and Events 151
5.8. Resetting Entities 151
5.9. HMD Controls 152
5.10. Saved Observer Views 152

5.10.1 Selecting a View 152
5.10.2 Transitioning to a View 153
5.10.3 Resetting the View 153

6. Controlling People

6.1. Introduction 156
6.1.1 Enabling and Disabling Mouse Control 156
6.2. Basic Movements 157
6.2.1 Moving Your Character Around the Terrain 157
6.2.2 Changing Your Character's Posture 157
6.2.3 Looking Around 157
6.2.4 Gesturing and Signaling 157
6.2.5 Climbing Ladders 158
6.3. Navigation 158
6.3.1 Using the Compass 158
6.3.2 Using the Minimap 159
6.4. Health and Stamina 160
6.4.1 Sustaining Damage to Health 161
6.4.2 Experiencing Fatigue and Losing Stamina 161
6.5. Weapons 162
6.5.1 Choosing a Weapon 162
6.5.2 Aiming a Weapon 163
6.5.3 Firing a Weapon 164
6.5.4 Firing Weapons with an HMD Controller 164
6.5.5 Throwing a Hand Grenade 164
6.5.6 Deploying and Stowing Weapons 165
6.5.7 Reloading a Weapon 166
6.5.8 Rocket and Missile Launchers 166
6.6. Using the AN/PED-1 LLDR 171
6.7. Embarking on a Vehicle 172
6.7.1 Embarking on the Stryker’s Interior 173
6.7.2 Disembarking from a Vehicle 174
6.8. Carry or Drag a Person 174
6.9. Parachuting 175
6.10. Tasking Other Entities 176
6.10.1 Disembark Entity 176
6.10.2 Follow Me 176
6.10.3 Move to Location 177
6.10.4 Move to My Location 177
6.11. Turning on Your Flashlight 178

6.12. Using the Binoculars 178
6.13. Changing Visual Sensors 179
6.14. Lasing Targets 179
6.14.1 Defining the Laser Code 181
6.14.2 Displaying a Gimbaled Sensor's View 181
6.14.3 Enabling Laser Designators 181
6.15. Opening and Closing Doors and Windows 181

7. Controlling Ground Vehicles

7.1. Introduction 184
7.1.1 Ground Vehicle Roles 184
7.1.2 Ground Vehicle Display Options 186
7.2. Driving Ground Vehicles 186
7.2.1 Starting the Engine 186
7.2.2 Putting the Vehicle into Gear 186
7.2.3 Controlling the Throttle 187
7.2.4 Steering a Vehicle 187
7.2.5 Looking Around 187
7.2.6 Using the Backup Camera View 187
7.2.7 Releasing and Engaging the Parking Brake 188
7.2.8 Locking and Unlocking the Differential 188
7.2.9 Turning the Headlights On and Off 188
7.3. The Gunner Role 188
7.3.1 Firing a Vehicle's Weapon 189
7.3.2 Choosing a Weapon 189
7.3.3 Moving a Turret or Gun 189
7.3.4 Stabilizing a Vehicle's Gun 189
7.3.5 The LSV MK II Gunner Roles 190
7.4. The Commander Roles 192
7.4.1 Changing Positions 192
7.4.2 Switching Views in the Commander Position 192
7.4.3 Overriding the Gunner 193
7.4.4 Slewing the Gun to Designate a Target 193
7.4.5 Using the Laser Designator 193
7.5. Using the Intercom 194
7.6. Changing the Sensor for a Vehicle Crew Member 194
7.7. Pivoting a Tank 194
7.8. Towing Other Ground Entities 195
7.9. Emergency Lights and Sirens 195

8. Controlling Aircraft

8.1. Introduction 198
8.1.1 Aircraft Roles 198
8.1.2 The Head-Up Display 198
8.1.3 The Instrument Panel 200
8.1.4 Aircraft Display Options 202
8.2. Taking Control of an Aircraft 203
8.3. Using Autopilot 204
8.3.1 Setting the Autopilot Mode 204
8.4. Controlling Landing Gear 205
8.5. Controlling Flaps 205
8.6. Controlling the Engine on Rotary-Wing Aircraft 205
8.7. Selecting a Bullseye 206
8.8. Using the Radar Warning Receiver 207
8.9. Launching Countermeasures (Chaff and Flare) 207
8.10. Choosing a Weapon 208
8.11. Air-to-Ground Combat 209
8.11.1 Targeting an Air-to-Ground Weapon Using MGRS Coordinates 209
8.11.2 Targeting an Air-to-Ground Weapon Using a Laser Code 209
8.12. Air-to-Air Combat 210
8.12.1 Selecting Targets 211
8.12.2 Firing an Air-to-Air Weapon 213
8.12.3 Focusing the Sensor on the Radar Target 214
8.13. Using the Common Operating Picture 214
8.13.1 Displaying COP Information for an Entity 214
8.13.2 Toggling the COP Focus Area 215
8.13.3 Selecting an Entity for the COP 215

9. Controlling Drones

9.1. Overview 218
9.1.1 Drones Included with VR-Engage 218
9.1.2 Drone Flight Controls 219
9.1.3 Drone Camera Displays 219
9.1.4 Video and Images 221
9.1.5 Drone Battery Life 221
9.2. Controlling Deployable Drones 221
9.2.1 Deploying Drones 222
9.2.2 Stopping and Resuming Use of the Drone 222
9.2.3 Landing the Drone 223
9.2.4 Returning the Drone to its Starting Position 223

9.2.5 Recovering the Drone 223
9.2.6 Switching Control Between Character and Spark Quadcopter 223
9.2.7 Setting Control to Manual or Task 224
9.2.8 Changing the Visual Sensor on the LSV Quadcopter 224
9.3. Operating Independent Drones 225
9.3.1 Changing the Visual Sensors 225
9.3.2 Attacking with the Munition Drone 225

10. Sensor Operator Role

10.1. The Sensor Operator Role 228
10.1.1 Focusing the Sensor 229
10.1.2 Changing the Sensor Mode 230
10.1.3 Resetting the Sensor 230
10.1.4 Video and Pictures 230
10.1.5 Defining a Laser Code 230
10.1.6 Lasing Targets 231
10.1.7 Firing Weapons 231
10.2. Creating a Remote Camera Entity 231
10.3. Placing Sensors on Remote Entities 232
10.3.1 Configuring the Location of the Remote Camera 232
10.4. Configuring the Sensor Operator Overlay 234

11. Video Stream and Image Capture

11.1. Introduction 238
11.2. Video Streams 238
11.2.1 Streaming Video as a Sensor Operator 238
11.2.2 Streaming Video from a Drone 238
11.2.3 Specifying the Video Stream File Settings 239
11.2.4 Viewing a Video File 239
11.2.5 Receiving a Video Stream 239
11.3. Capturing Images 240

12. Mouse, Keypad, and Controller Mappings

12.1. Mouse and Keyboard Mappings 242
12.1.1 Application Management 242
12.1.2 Action Menu Controls 242
12.1.3 Human Controls 243
12.1.4 Drone Controls 245
12.1.5 Ground Vehicle Controls 245
12.1.6 Aircraft Controls 247

12.1.7 Sensor Operator Controls 248
12.2. Game Controller Mappings for Using the Action Menu 248
12.3. Game Controller Mappings for Human Control 249
12.4. Game Controller Mappings for Drones 250
12.5. Game Controller Mappings for Vehicles with Weapons 251
12.6. Game Controller Mappings for Cars and Trucks 252
12.7. Racing Wheel Controllers 252
12.8. Flight Controllers 255
12.8.1 HOTAS Warthog Mappings for Fixed-Wing Entities 255
12.8.2 HOTAS Warthog Mappings for Rotary-Wing Entities 258
12.8.3 T.Flight Hotas X Mappings for Fixed-Wing Entities 260
12.8.4 T.Flight Hotas X Mappings for Rotary-Wing Entities 263
12.8.5 Game Controller Flight Control Mappings for Fixed-Wing Entities 266
12.8.6 Game Controller Flight Control Mappings for Rotary-Wing Entities 267
12.8.7 Game Controller Mappings for the Sensor Operator 267
12.9. Head-Mounted Display Controllers 268
12.10. Game Controller Button Numbering 269
12.11. Tips for Logitech Gamepad 272
12.12. Configuring Game Controllers and Keyboard Control 272
12.12.1 Windows Game Controllers 273
12.12.2 The Game Controller Utility 273
12.12.3 File Structure 274
12.12.4 Mappings 275
12.12.5 Input Types 276
12.12.6 Player Controls State 278
12.12.7 Component Mapping Groups 280
12.12.8 List of Actions 281

13. Controllable Entities

13.1. Human Characters 292
13.2. Ground Vehicles 312
13.3. Fixed-Wing Aircraft 324
13.4. Rotary-Wing Aircraft 328
13.5. Other Entities 331

III. CONFIGURATION AND SCENARIO CREATION

14. Adding and Editing VR-Engage Entity Models

14.1. The VR-Engage Simulation Model Set 336
14.2. Extend the VR-Engage SMS 336

14.2.1 Creating a New Simulation Model Set 337
14.2.2 Creating a New VR-Engage-Controllable Entity 337
14.2.3 Editing a VR-Engage-Controllable Entity 338
14.2.4 Deploying the SMS to VR-Engage 339
14.3. VR-Engage Extensions to .entity Files 339
14.3.1 Structure of Role Data in .entity Files 339
14.3.2 The vrEngageRoles and role Entries 339
14.3.3 The parameterOverride Entry 341
14.3.4 Role-Slot Association 342
14.3.5 Roles Included with VR-Engage 342
14.4. Ground Vehicles and Vortex 347
14.4.1 Prerequisites 347
14.4.2 Configuring VR-Engage Ground Vehicles 352

15. Model Customization Tutorial

15.1. Overview of Customizing a Model 362
15.2. Extending VR-Engage.sms 362
15.3. Editing the Entity in the SOE 363
15.3.1 Promote the Entity to your SMS 363
15.3.2 Review the Entity's Systems 364
15.4. Editing the Tank's .entity File 365
15.4.1 Specify a VR-Engage-Capable Platform 365
15.4.2 Add the VR-Engage Tag 366
15.4.3 Add the Driving Role Component 366
15.5. Test Driving Your Tank 366
15.5.1 Copy the SMS Files 367
15.5.2 Launch VR-Engage with Your SMS 367
15.5.3 Drive Your Tank 368
15.6. Creating the Views for Your Tank 368
15.6.1 Create a New Driver Views File 368
15.6.2 Override Tank Driver View with Your View 369

16. Creating VR-Engage Scenarios in VR-Forces

16.1. Installing the VR-Forces/VR-Engage Extensions 372
16.2. Creating a VR-Engage Scenario in VR-Forces 372
16.2.1 Creating a Load-Balanced Scenario 374
16.2.2 Configuring Ground Vehicles on Geocentric Terrains 375
16.2.3 Vortex While Untasked 375
16.3. Running Your Scenario 376
16.3.1 Hosting a Scenario in VR-Forces 376

16.3.2 Hosting a Scenario in VR-Engage 377

17. Customizing Overlays

17.1. Overview of Custom Overlays 380
17.1.1 Types of Overlays 380
17.1.2 Examples of Each Type of Overlay 380
17.2. Adding a QML Overlay 381
17.2.1 Testing Your QML Overlay 382
17.3. Adding a Custom Sensor Overlay 383
17.4. Adding a GL Studio Overlay 383
17.4.1 Creating a GL Studio RSO 384
17.4.2 Creating VR-Vantage GL Studio Updaters 384
17.4.3 Configuring the VR-Vantage GL Studio Model Definition 384
17.4.4 Modifying Role Configuration Scripts 385

18. Activating Scripts in VR-Forces

18.1. Activating Lua Scripts in VR-Forces 388
18.2. Adding a Script to the Action Menu 388
18.3. Activating a Script 389

IV. APPENDICES

A. Troubleshooting

A.1. Performance and Display Issues 394
A.2. Using the Tracy Client 394
A.3. The Fault Tolerant Heap 395
A.4. Graphics Card Binary Compatibility 395
A.5. Issues with VRAM 395
A.6. Remote Camera Placement 396
A.7. Other Limitations 396

B. Embarkation Slot Configuration in VR-Engage

B.1. Overview of Embarkation Slots 398
B.2. Door Point-type Slots for Scripted Doors on Walkable Vehicles 398
B.2.1 Ingress vs. Egress Door Points 398
B.2.2 Example Door Point Slot Configuration 398
B.2.3 Door Point Parameters 399
B.3. Embark Point-type Slots 400
B.3.1 Example Embark Point Slot Configuration 400

B.3.2 Embark Point Parameters 401

C. VR-Engage Support for CIGI

C.1. Introduction to CIGI in VR-Engage 404
C.2. Tutorial for Using VR-Engage as a CIGI Host 404

MAK PRODUCTS GLOSSARY

INDEX



Preface

This manual is for persons who will install or use VR-Engage. The manual assumes that you are familiar with basic administrative tasks and the graphical window environment for your operating system. It also assumes a basic level of familiarity with video game play.

Please see release documentation for information about changes and updates to VR-Engage since this manual was completed.

How the Manual Is Organized

This manual is organized as follows:

- "Section I. Introduction, Installation, and Startup" on page 23
  - "Chapter 1. Introduction to VR-Engage" on page 25 describes the basic features of VR-Engage.
  - "Chapter 2. Installing VR-Engage" on page 37 explains how to install VR-Engage software, the license manager, and an RTI.
  - "Chapter 3. Running VR-Engage" on page 55 explains how to start and stop VR-Engage, load a terrain, and connect to a simulation.
  - "Chapter 4. Optimizing Performance" on page 117 describes issues that may affect performance and how to address them.
- "Section II. Player Features" on page 131
  - "Chapter 5. Common Player and Entity Features" on page 133 describes features that apply to all platform types.
  - "Chapter 6. Controlling People" on page 155 explains how to play a human character, including moving, changing weapons, firing weapons, and embarking and disembarking.
  - "Chapter 7. Controlling Ground Vehicles" on page 183 explains how to drive ground vehicles, including the LAV, M1A2, and civilian vehicles.
  - "Chapter 8. Controlling Aircraft" on page 197 explains how to control aircraft.
  - "Chapter 9. Controlling Drones" on page 217 explains how to deploy and control the quadcopter drone.
  - "Chapter 10. Sensor Operator Role" on page 227 explains the sensor operator role and how to control the sensor interface.
  - "Chapter 11. Video Stream and Image Capture" on page 237 describes how to stream video or save images to a file from a quadcopter drone or sensor operator.
  - "Chapter 12. Mouse, Keypad, and Controller Mappings" on page 241 illustrates the mouse, keyboard, and game controller mappings configured in VR-Engage.
  - "Chapter 13. Controllable Entities" on page 291 lists the entity types that you can control in VR-Forces from VR-Engage if VR-Forces is running a scenario that uses VR-Engage.sms.

- "Section III. Configuration and Scenario Creation" on page 333
  - "Chapter 14. Adding and Editing VR-Engage Entity Models" on page 335 explains how to extend the VR-Engage simulation model set.
  - "Chapter 15. Model Customization Tutorial" on page 361 provides a simple example of how to create customize a model to make it available for player control.
  - "Chapter 16. Creating VR-Engage Scenarios in VR-Forces" on page 371 describes what is involved in creating your own scenarios.
  - "Chapter 17. Customizing Overlays" on page 379 describes how to edit entity configurations to customize what players can see and do.
  - "Chapter 18. Activating Scripts in VR-Forces" on page 387 explains how to add scripts that you can send to VR-Forces.
- "Section IV. Appendices" on page 391
  - "Appendix A. Troubleshooting" on page 393 explains how to resolve some common problems that can arise with VR-Engage.
  - "Appendix B. Embarkation Slot Configuration in VR-Engage" on page 397 provides detailed information about configuring slots on vehicles to allow players to embark on them.
  - "Appendix C. VR-Engage Support for CIGI" on page 403 briefly describes how you can use VR-Engage as a CIGI host and has an example of using VR-Vantage as an IG.

Documentation Set

VR-Engage documentation is provided as manuals in PDF format and online help. Electronic versions of the documentation are in *vrengage/doc*. On Windows, you can access the documentation from the VR-Engage folder on the Start menu. You can also find the most current documentation on the Product Specifications & Dependencies page for VR-Engage: see https://www.mak.com/mak-one-specs. The VR-Engage documentation set is:

- *VR-Engage User Guide* describes how to use the VR-Engage and also provides some basic instructions on customizing how it works.
- *VR-Engage First Experience Guide* provides a brief guided tour of the most important VR-Engage features. It is primarily for new users and persons evaluating VR-Engage.
- Online help. The online help, accessible from the Help menu, presents most of the information in the *VR-Engage User Guide* and Adding Content to MAK ONE Applications Reference Manual.
- *VR-Vantage User Guide*. The VR-Engage Admin mode has many settings that affect the visual quality of your role-playing experience. These and other aspects of the graphical user interface are shared with VR-Vantage and VR-Forces. The *VR-Vantage User Guide* is included to provide documentation for display settings, navigation of the GUI, and other information that may be of use to VR-Engage users, but which is

not central to using VR-Engage.

- *Adding Content to MAK ONE Applications Reference Manual* explains how to add visual models and terrain to VR-Engage.
- *VR-Engage Release Notes* lists system requirements, release-specific requirements, new features and updates, bug fixes, and known problems.

Try the Help

MAK's HTML5 help is the best way to use our documentation. It's easy to navigate, with improved natural language search tools and filters to focus your search. You can click the illustrations to see high-resolution enlargements that show more detail. It's built into the product and installed locally — simply press **Shift+F1** or choose Help from the Help menu to launch the help in your browser. You do not need to be connected to the internet to use it.

> If you attempt to view the help in a browser that does not fully support HTML5, you may experience issues with some features.

Links Between PDFs

Many MAK PDF documents link to each other. For example, the description of a new feature in the release notes might include links to detailed information in a user guide, and there are also many links from user guides to the *Adding Content to MAK ONE Applications Reference Manual* and back. These links between PDFs rely on two things:

- The linked PDF files must be in the same directory. MAK installers put the PDFs for each product in that product's *doc* directory — in Windows, *./vrengage/doc*. If you move or copy a PDF away from that directory or download a new one to another location, any links between that PDF and the others stop working.
- You must use Acrobat Reader to read MAK PDFs. Links between PDFs do not work when you read with a web browser. Acrobat Reader is free; you can get it from Adobe at https://get.adobe.com/reader.

MAK Products

The MAK Technologies product line includes:

- VR-Link® Network Toolkit. VR-Link is an object-oriented library of C++ functions and definitions that implement the High Level Architecture (HLA) and the Distributed Interactive Simulation (DIS) protocol. VR-Link has built-in support for the NETN FOM and RPR FOM and allows you to map to other FOMs. This library minimizes the time and effort required to build and maintain new HLA or DIS-compliant applications, and to integrate such compliance into existing applications.

VR-Link includes a set of sample debugging applications and their source code. The source code serves as an example of how to use the VR-Link Toolkit to write applications. The executables provide valuable debugging services such as generating a predictable stream of HLA or DIS messages, and displaying the contents of messages transmitted on the network.

- MAK RTI. An RTI (Run-Time Infrastructure) is required to run applications using the High Level Architecture (HLA). The MAK RTI is optimized for high performance. It has an API, RTIspy®, that allows you to extend the RTI using plug-in modules. It also has a graphical user interface (the RTI Assistant) that helps users with configuration tasks and managing federates and federations.
- VR-Forces®. VR-Forces is a computer generated forces application and toolkit. It provides an application with a GUI, that gives you 2D and 3D views of a simulated environment.

  You can create and view entities, aggregate them into hierarchical units, assign tasks, set state parameters, and create plans that have tasks, set statements, and conditional statements. You can simulate using entity-level modeling, which focuses on the actions of individual people and vehicles, and aggregate-level modeling, which focuses on the interaction of large hierarchical units.

  VR-Forces also functions as a plan view display for viewing remote simulation objects taking part in an exercise. Using the toolkit, you can extend the VR-Forces application or create your own application for use with another user interface.
- VR-Vantage™. VR-Vantage is a line of products designed to meet your simulation visualization needs. It includes VR-Vantage and the VR-Vantage Toolkit.
  - VR-Vantage displays a realistic, 3D view of your virtual world, a 2D plan view, and an exaggerated reality (XR) view. Together these views provide both situational awareness and the big picture of the simulated world. You can move your viewpoint to any location in the 3D world and can attach it to simulation objects so that it moves as they do.
  - SensorFX. SensorFX is an enhanced version of VR-Vantage that uses physics-based sensors to view terrain and simulation object models that have been materially classified. It is built in partnership with JRM Technologies.
  - The VR-Vantage Toolkit is a 3D visual application development toolkit. Use it to customize or extend MAK's VR-Vantage applications, or to integrate VR-Vantage capabilities into your custom applications. VR-Vantage is built on top of OpenSceneGraph (OSG). The toolkit includes the OSG version used to build VR-Vantage.
- MAK Data Logger. The Data Logger, also called the Logger, can record HLA and DIS exercises and play them back for after-action review. You can play a recorded file at speeds above or below normal and can quickly jump to areas of interest. The Logger has a GUI and a text interface. The Logger API allows you to extend the Logger using plug-in modules or embed the Logger into your own application. The Logger editing features let you merge, trim, and offset Logger recordings.

- VR-Exchange™. VR-Exchange allows simulations that use incompatible communications protocols to interoperate. For example, within the HLA world, using VR-Exchange, federations using the HLA RPR FOM 1.0 can interoperate with simulations using RPR FOM 2.0, or federations using different RTIs can interoperate. VR-Exchange supports HLA, TENA, and DIS translation.
- VR-TheWorld™. VR-TheWorld is a simple, yet powerful, web-based streaming terrain server, developed in conjunction with Pelican Mapping. Delivered with a global base map, you can also easily populate it with your own custom source data through a web-based interface. The server can be deployed on private, classified networks to provide streaming terrain data to a variety of simulation and visualization applications behind your firewall.
- DI-Guy™. DI-Guy is an SDK that allows you to embed the DI-Guy library in your real-time application and populate your world with lifelike human characters. DI-Guy provides high-fidelity human characters that move realistically, respond to simple high-level commands, and travel throughout the environment as directed by the hosting visual application. It comes with thousands of ready-to-use characters, appearances, and motions. DI-Guy character simulation is an integral part of all MAK's visual applications, including VR-Engage, VR-Forces, and VR-Vantage.
- RFView® for MAK ONE. Synthetic Aperture Radar (SAR) imaging is an indispensable tool for today's modern militaries. Developed in partnership with ISL, RFView for MAK ONE acts as a SAR server and delivers simulated SAR images that are both physically accurate and fully correlated with out-the-window views and EO/IR images of MAK ONE synthetic environments. It includes a toolkit that enables you to build a client application to request SAR images from the server.
- VR-Engage. VR-Engage is a multi-role virtual simulator that lets users play the role of a first person human character, a ground vehicle driver, gunner or commander, a sensor operator, or the pilot of a fixed wing aircraft or helicopter. It incorporates the VR-Force simulation engine and the VR-Vantage graphics rendering capabilities.
- WebLVC Server. WebLVC Server implements the server side of the WebLVC protocol so that web-based simulation federates can participate in a distributed simulation. It is part of the WebLVC Suite, which includes the server and several sample applications that work with VR-Forces and VR-Vantage.
- MAK Legion. MAK Legion is a collection of applications and an SDK designed to support simulations with millions of simulation objects.

How to Contact Us

For VR-Engage technical support, information about upgrades, and information about other MAK products, you can contact us in the following ways:

Telephone

Call or fax us at:	Voice:	617-876-8085
	Fax:	617-876-9208

E-mail

Sales and upgrade information:	info@mak.com
Technical support:	support@mak.com

Internet

MAK website home page:	www.mak.com
License key requests:	https://www.mak.com/mak-one/support/portal
Technical support:	https://www.mak.com/mak-one/support/portal
Product version and platform information:	https://www.mak.com/mak-one-specs
For the free, unlicensed MAK RTI:	https://www.mak.com/mak-one/support/help/bonus-material

Post

Send postal correspondence to:	MAK Technologies 10 Fawcett Street, Suite 204 Cambridge, MA 02138 USA

When requesting support, tell us the product you are using, the version, and the platform on which you are running. If you are using VR-Vantage, VR-Forces, VR-Engage, or DI-Guy, please also include your data compatibility version and date.

Document Conventions

This manual uses these typographic conventions:

Sans Serif Bold	Indicates a key on the keyboard or menu options. Mentions of letter keys do not require you to press the **Shift** key. If Shift is necessary, that will be included, for example, **Shift**+**Q**. Also indicates parameters.
`Monospaced`	Indicates values you enter such as commands or arguments.
`Monospaced Italic`	Indicates command variables that you replace with appropriate values.
{ }	Indicates required arguments.
[ ]	Indicates optional arguments.

|	Separates options in a command where only one option may be chosen at a time.
(|)	In command syntax, indicates equivalent alternatives for a command-line option, for example, `(--help | -h)`.
/	Indicates a directory. Since MAK products run on both Linux and Windows PC platforms, we use the / (slash) for generic discussions of pathnames.
Italic	Indicates a file name, pathname, or a class name.
➤	Indicates a procedure.
Menu > Option	Indicates a menu choice. For example, an instruction to select the Save option from the File menu appears as: Choose **File > Save**.
	Indicates supplemental or clarifying information.
	Indicates additional information that you must observe to ensure the success of a procedure or other task.
	Indicates helpful information.

Directory names are preceded with dot and slash characters that show their position with respect to the VR-Engage home directory. For example, the directory *./vrengage2.2/doc* appears in the text as *./doc*.

For anything in the data directory, the path is shown starting with *./SharedData/19/latest* to be clear that it is not in the VR-Engage home directory. That is the default installation path. If you installed the data in a different directory, substitute your path.

Mouse Button Naming Conventions

An instruction to click the mouse button refers to clicking the primary mouse button, usually the left button for right-handed mice and the right button for left-handed mice. The context-sensitive menu, also called a popup menu or right-click menu, refers to the menu displayed when you click the secondary mouse button, usually the right button on right-handed mice and the left button on left-handed mice.

Third-Party Licenses

MAK software products may use code from third parties. For an up-to-date list of supported third-party and open source libraries and the associated license information, please visit the MAK ONE Product Support Knowledge Base (at https://www.mak.com/knowledge) and refer to the Third-Party/Open Source List under Frequently Asked Questions (FAQ).



I. Introduction, Installation, and Startup

VR-Engage is designed for simulation training and is a part of the MAK ONE suite.



1. Introduction to VR-Engage

This chapter is a brief introduction to VR-Engage.

1.1. VR-Engage Features 26
1.1.1 VR-Engage Versions 26
1.1.2 Simulation (Role-Playing) Capabilities 27
1.1.3 User Interface 32
1.1.4 Image Generator 33
1.1.5 Terrain 33
1.2. Integration with VR-Forces 33
1.3. Integration with Other MAK ONE Products 34
1.3.1 After-Action Review Using MAK Data Logger 34
1.3.2 Network Interoperability with Other Federates 34
1.4. Extensibility 34
1.5. Licensing 35
1.5.1 Vortex License (for Ground Vehicle Dynamics) 35
1.5.2 RTDynamics License (For Aircraft) 35
1.5.3 License Requirements for Hosting a VR-Forces Session 36

1.1. VR-Engage Features

VR-Engage was developed for use in training simulations or laboratory experimentation. Users can play the role of a human character; a vehicle driver, gunner or commander; a sensor operator; a drone operator; or the pilot of an airplane or helicopter.

VR-Engage is built on MAK's mature, proven technologies. It utilizes the simulation engine from VR-Forces, the game-quality 3D graphics from VR-Vantage, and the network interoperability from VR-Link.

VR-Engage includes:

- A high-fidelity vehicle physics engine needed for accurate vehicle motion.
- Ground vehicles, rotary- and fixed-wing aircraft (both manned and unmanned), and a full library of friendly, hostile, and neutral DI-Guy characters.
- Radio and voice communications over DIS and HLA.
- Sensors, weapons, countermeasures, and behavior models for air-to-air, air-to-ground, on-the-ground, and person-to-person engagements.
- Vehicle- and person-specific interactions with the environment (open and close doors, move, destroy, and so on.)
- Terrain agility, supported by VR-Vantage and VR-Forces, you can use the terrain you have or take advantage of innovative streaming and procedural terrain techniques.
- Support for virtual reality and mixed reality using head-mounted displays, such as Oculus Rift, Varjo, HTC Vive, Valve Index, and XTAL.

1.1.1 VR-Engage Versions

VR-Engage has these run-time versions:

- VR-Engage. Allows you to control air, ground, and human entities.
- VR-Engage with SensorFX. Includes the SensorFX plug-in, which gives you access to more sophisticated infrared and night vision settings.

Both configurations are installed as part of the standard VR-Engage installation process. Your access to them depends on your VR-Engage license. For simplicity, the procedures in this manual assume you are using VR-Engage.



- VR-Engage also includes a Remote Display Engine (RDE). Refer to the Section II of the *VR-Vantage User Guide* to learn how to use the RDE.
- Terrains and models require special configuration to display in SensorFX. The terrains and models included with VR-Engage are configured to display correctly. If you need to create your own, please refer to the chapters on material classification in the *Adding Content to MAK ONE Applications Reference Manual*.

1.1.2 Simulation (Role-Playing) Capabilities

VR-Engage lets you choose which entity you want to simulate. Said another way, you can choose the role you want to play. You can play:

- Dismounted soldiers and other human characters.
- A member of the crew of a ground vehicle.
- A helicopter or fixed-wing aircraft pilot.
- The operator of a gimbaled sensor on a UAV or a mounted camera.
- The pilot of a small drone.

- If appropriate for your training simulation, you can prevent players from selecting a role. In those cases, an instructor using VR-Forces can assign players to roles. See "3.2.4 Assigning Roles from VR-Forces" on page 67.
- The instructor cannot assign tasks to entities controlled by VR-Engage players.

Dismounted Soldiers and Other Human Characters

If you take the role of a dismounted soldier or other human character, you have access to the full DI-Guy library of friendly, hostile, and neutral characters. Your character can:

- Move (look, run, walk, crawl, crouch, lie prone).
- Interact with the environment (open and close doors or windows, climb ladders, experience fatigue, and lose stamina).
- Interact with other characters (occupy different seats in a vehicle, shoot at, be shot at and sustain damage to health, drag or carry wounded people, task CGF entities, interact with other players).
- Use sensors (binoculars and night vision goggles (NVG)).
- Use weapons (rifles, handguns, grenades, rocket and missile launchers).
- Use carried gear, such as a flashlight, range finder, or compass.
- View a 2D minimap.
- Lase targets for other players or simulated entities.
- CGF Assist (passenger, fast-roping). When you are running a VR-Forces scenario, you can give up control of your character and let the simulation engine manage complex behaviors.
- Embark and disembark. Enter and leave vehicles and take on the roles of vehicle crew members.
- Parachute from an aircraft.

Figure 1. Dismounted infantry

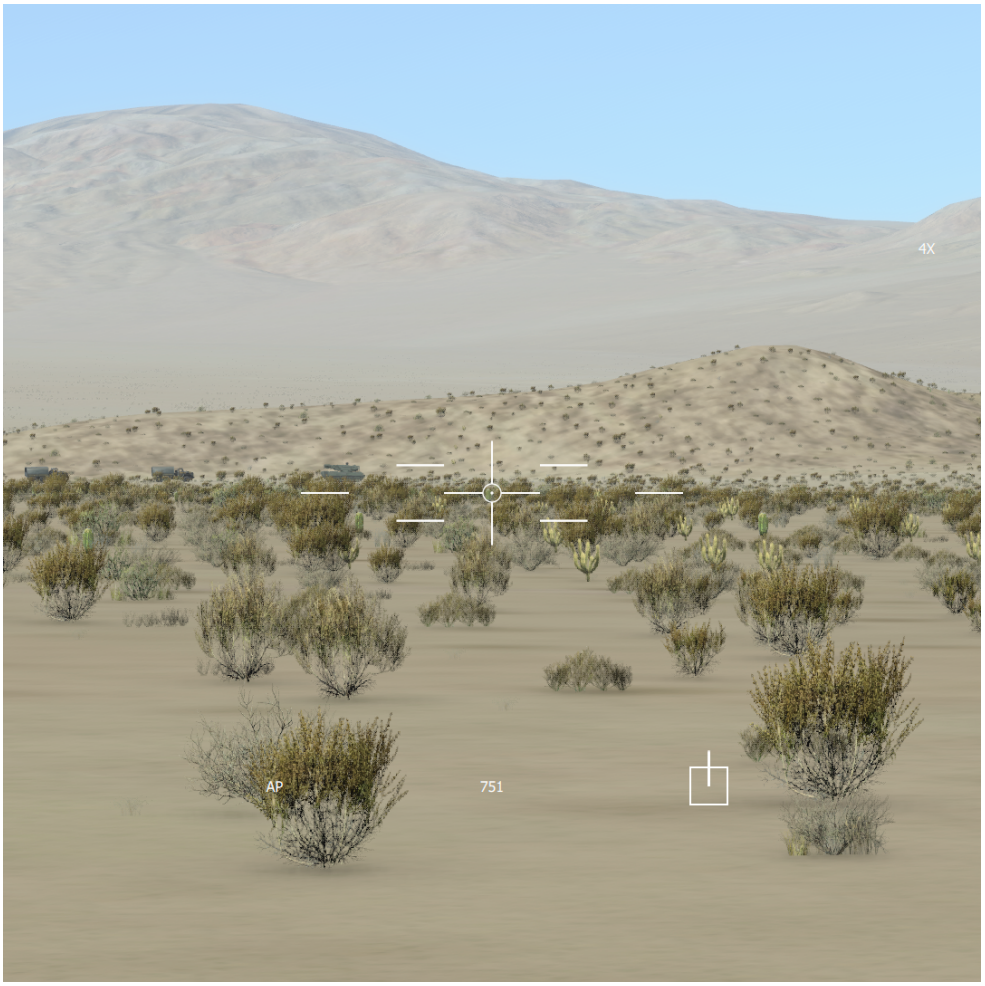


Ground Vehicle Crew

VR-Engage allows you to take on roles such as driver, gunner, or commander of a ground vehicle that:

- Has physics-based ground vehicle dynamics model.
- Can interact with environment (collide, drive over, push, destroy).
- Perform ground-to-ground engagements:
  - Sensors (EO, IR, NVG).
  - Weapons (main gun, machine gun).
  - Signals/Countermeasures (smoke grenades).
- Operate vehicle controls (headlights, tail lights, emergency lights and sirens).
- CGF Assist (drive, gun). In a VR-Forces scenario, as you switch roles, the simulation engine can take over the tasks and plans that manage the other roles for the vehicle. For example, if you are the gunner on an M1126 Stryker ICV with M2, VR-Forces can drive the vehicle.

Figure 2. Bradley gunner



You can easily switch roles and also have the ability to disembark from the vehicle to assume the role of a dismounted soldier.

Pilot

As a pilot, you can take off, fly, and land an aircraft with:

- Physics-based fixed- and rotary-wing dynamics models.
- Head-up display (HUD).
- Multi-function cockpit display. Flight instruments, map/chart, imagery.
- Air-to-Air Engagements:
  - Sensors (radar scope, radar warning receiver).
  - Weapons (missiles).
  - Countermeasures (chaff, flares).

- Air-to-Ground Engagements:
  - Sensors (IR targeting).
  - Weapons (missiles, guns, bombs).
  - Countermeasures (chaff, flares).

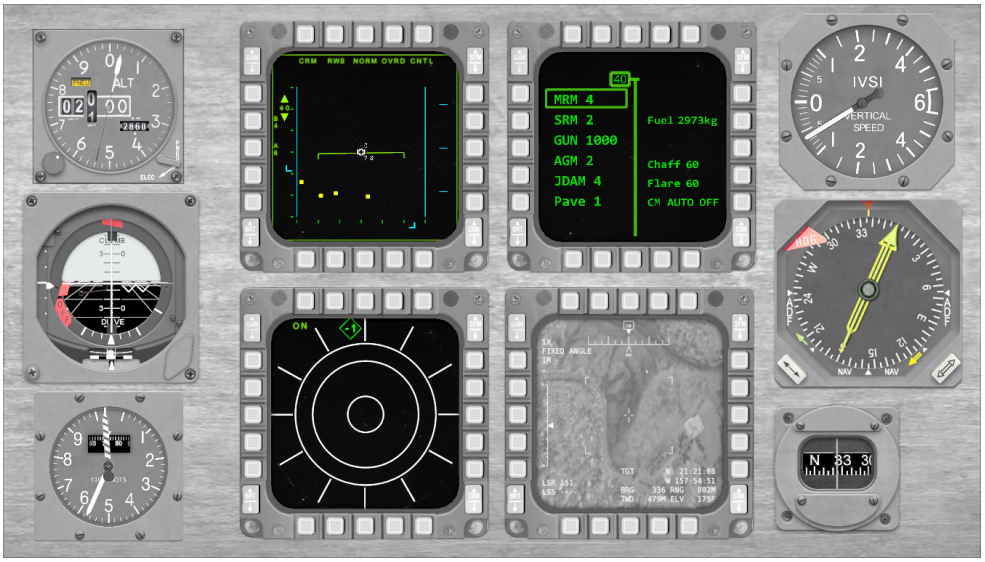
Pilots use their out-the-window views and head-up display (HUD) (Figure 3) to understand their environment and control their aircraft.

- Single screen. Out-the-window view, on-screen controls, and HUD.
- Dual screen. Out-the-window view, on-screen controls, and HUD.
- On-screen multi-function display (IR) (Figure 4).
- Slaved IG (requires VR-Vantage Remote Display Engine).
- Head-mounted display (HMD). Oculus Rift, Varjo, HTC Vive, Valve Index, and XTAL.

Figure 3. Head-up display with instruments



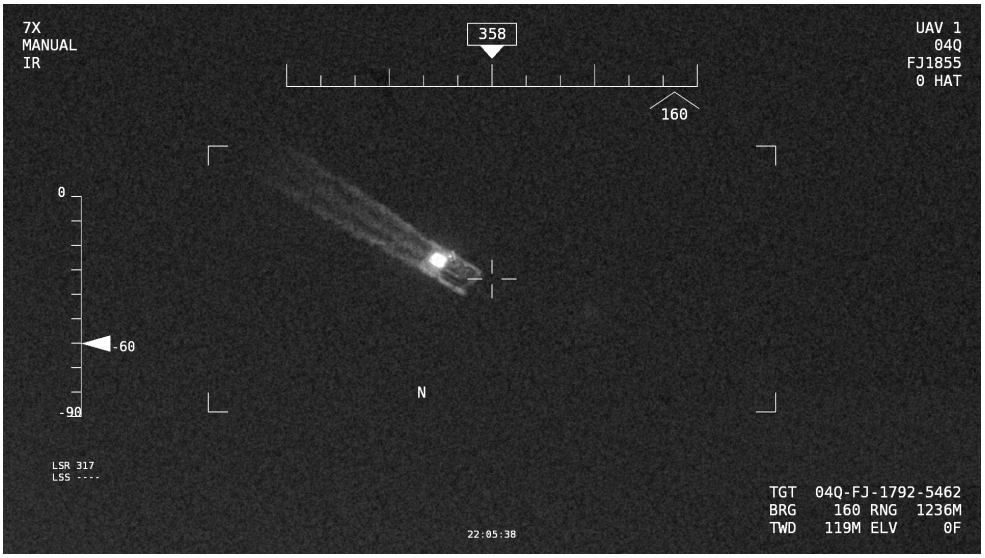
Figure 4. Instrument panel



Sensor Operator

In the sensor operator role, you can perform common surveillance and reconnaissance tasks, such as tracking fixed and moving targets, using a simulated E/O camera or IR sensor. You can control gimbaled sensors using joysticks or game controllers or configure VR-Engage to work with sensor-specific hand controller devices. VR-Engage lets you attach cameras to entities and structures and set their initial heading and pitch. VR-Engage can also automatically attach cameras to remotely simulated (that is, non-VR-Forces) entities.

Figure 5. Sensor view



Drone Operator

As a civilian drone operator, you can deploy a small drone equipped with a camera and then fly it around. This role is designed to train people how to safely fly a drone without risking an actual device. You can use these in many different types of scenarios, such as simple infrastructure inspection or surveillance and reconnaissance.

Figure 6. Drone operator



VR-Engage includes the Light Strike Vehicle MK II (LSV), which has a drone operator role. The drone is equipped with a camera that has different sensor modes: visual, NVG, and IR white hot or black hot. You can fly the drone even when the vehicle is moving.

There are also independent drone entities, such as the Parrot Anafi Quadcopter and DJI Mavic 3 Small Munition, that you can place in scenarios and operate.

1.1.3 User Interface

You can control your character using the keyboard, mouse, and game controllers. The graphical user interface provides command menus and cockpit and crew cabin displays. Other interfaces include:

- Radio (push-to-talk, radio channels, MAK Data Logger support) and text chat with other players and instructors.
- Control with role-specific devices (driver's steering wheel and pedals, pilot's HOTAS).
- Display system support (monitors, projectors, mixed and virtual reality headsets).
- Audio (engine sounds, environmental noise, event effects, warning tones).
- Sensor and drone information (compass, ownship, drone position, speed, and battery charge).

1.1.4 Image Generator

VR-Engage includes a built-in image generator, based on VR-Vantage, that can support multiple display channels, to the limits of your installed graphics card. It can also be extended to fill larger multi-channel display systems by adding VR-Vantage Remote Display Engines. This is an example of the flexibility of MAK's product architecture that allows you to configure a system of any size.

Regardless of which role your VR-Engage simulator is playing, VR-Engage delivers realistic scene rendering (terrain, entities, interaction effects, weather, clouds, time-of-day, lighting, shadows, high dynamic range (HDR) lighting.)

The image generator supports:

- Single screen.
- Dual screen.
- Multi-channel distributed rendering (requires VR-Vantage Remote Display Engine). Spread the view across multiple monitors run by multiple computers.
- IR and NVG Sensors using the included CameraFX or optional SensorFX.
- On screen Multi-function Display (IR).
- Head-mounted displays.

> ⚠ When you use remote display engines with VR-Engage, you must connect to any remote display engine before hosting or joining a session. If you attempt to connect to a remote display engine after a session has started, VR-Engage leaves the session and can end up in a state where it constantly makes unsuccessful attempts to rejoin the session.

1.1.5 Terrain

VR-Engage, like VR-Vantage and VR-Forces, allows you to use the terrain you have or take advantage of innovative streaming and procedural terrain techniques to simulate on large, complex terrains.

- Terrain agile (all types supported by VR-Vantage and VR-Forces).
- Dynamic terrain. Destroyable buildings, bridges, and props.
- Dynamic ocean. Waves, spray, wakes, wind.

VR-Engage includes a set of useful geo-specific terrains and options for building more. See the *Adding Content to MAK ONE Applications Reference Manual* for information about terrain support. (The manual is included with VR-Engage.)

1.2. Integration with VR-Forces

When VR-Engage is used with a compatible version of VR-Forces and other MAK ONE products, you can reap the additional benefits of a common system architecture:

- Common representation of the environment across player and CGF stations, including synchronized weather, time-of-day, and dynamic terrain.
- Build terrains, models, and configurations once, and deploy them across VR-Engage player stations, VR-Forces simulation engines and GUIs, and any other applications that use VR-Vantage.
- Role-play multiple entities at a time by switching between manual and CGF control on-the-fly. Take control of VR-Forces-driven entities.
- Manage entity assignments and roles for multiple VR-Engage players from VR-Forces.
- VR-Forces users can talk with VR-Engage players using MAK Radio or text chat.
- Configure entities with the VR-Forces Simulation Object Editor.
- Create scenarios in VR-Forces and load them in VR-Engage for standalone use.
- Automatic WebLVC integration (tablet-based IOS, and so on).
- Publish entity state properties and appearance bits.

VR-Engage 2.2 is compatible with VR-Forces 5.2d.

1.3. Integration with Other MAK ONE Products

Previous sections have discussed VR-Engage's integration with VR-Forces and VR-Vantage. Additionally, like all MAK ONE products, VR-Engage uses VR-Link for network interoperability and supports DIS and HLA. It integrates seamlessly with our other Link products, such as MAK Data Logger for recording and after-action review (AAR), the MAK RTI for HLA support, and VR-Exchange for data interchange in heterogeneous simulation environments.

1.3.1 After-Action Review Using MAK Data Logger

Whether in standalone mode or when playing with other simulation applications, VR-Engage publishes to the DIS or HLA network. Therefore, you can use the MAK Data Logger to record your simulation and play the recording back for AAR.

1.3.2 Network Interoperability with Other Federates

VR-Engage is natively compliant with DIS and HLA, allowing it to participate in any topology of networked simulators and simulation support systems, which includes multi-player classroom environments. It can interoperate with existing simulation applications and third-party simulators, semi-automated forces (SAFs), and computer-generated forces (CGFs).

1.4. Extensibility

VR-Engage is ready to use out-of-the-box. It can be deployed as a trainee simulator, as a role player station, an instructor aid, a desktop simulation game, or even as a VR headset experience.

Because MAK focuses on the needs of system integrators, we designed flexibility into VR-Engage. It can be customized and extended to meet program-specific requirements, and can also be integrated into a diverse range of system configurations. MAK has a pricing and support model that has proven to be effective with system integrators both before and after they choose MAK ONE products.

1.5. Licensing

The VR-Engage application requires a VR-Engage run-time license in order to run. Internally, VR-Engage relies on VR-Vantage capabilities for its 3D visualization, VR-Forces capabilities for its ownship simulation, VR-Link for its networking, and so on. These capabilities are part of VR-Engage and do not require separate run-time licenses of these MAK ONE products.

To learn about VR-Forces, see "1.5.3 License Requirements for Hosting a VR-Forces Session" on the next page.

> You can run a limited version of VR-Engage—air, ground, or sensor—using the `--appMode` command-line option as described in "3.13.1 Command-Line Options for vrEngage" on page 87. If you only have a license for one of these limited versions and you run the full version of vrEngage.exe without one of the command-line options, you can see other features but not use them.

1.5.1 Vortex License (for Ground Vehicle Dynamics)

VR-Engage uses Vortex from CM Labs for its ground vehicle dynamics. This built-in runtime capability is included with VR-Engage. You do not need a separate Vortex license to run VR-Engage.

If you want to use the Vortex Editor to edit VR-Engage's built-in vehicle dynamics models or author new ones, you must purchase a copy of Vortex Studio directly from CM Labs. The right to execute the resulting custom content in VR-Engage is covered by the VR-Engage license and does not require a separate Vortex runtime license.

> For more information, see "14.4. Ground Vehicles and Vortex" on page 347.

1.5.2 RTDynamics License (For Aircraft)

VR-Engage uses the RotorLib and FixedWingLib products from RTDynamics for its aircraft vehicle dynamics. This built-in runtime capability is included with the VR-Engage license. You do not need a separate RotorLib or FixedWingLib license key to run VR-Engage.

If you want to use RTDynamics's RIME editor to edit VR-Engage's built-in flight dynamics models or author new ones, you must purchase a copy of RIME or the RTDynamics SDKs. You can purchase these through MAK; contact your account representative or sales@mak.com. The right to execute the resulting custom content in VR-Engage is covered by the VR-Engage license, and does not require a separate RTDynamics runtime license.

1.5.3 License Requirements for Hosting a VR-Forces Session

With only a VR-Engage license, you can run VR-Engage, choose a terrain database to simulate on, create and place one entity, and control that entity. VR-Engage can only simulate and control a single entity at a time. When you change roles and create a different entity to simulate, the entity you were previously simulating is deleted and replaced by the new one.

If you are running a copy of VR-Forces on your network (which requires a separate VR-Forces license), then VR-Engage can interact with VR-Forces:

- VR-Engage allows you to join a VR-Forces session and take manual control of one of the VR-Engage controllable entities in the VR-Forces scenario. If you are controlling an entity, and then change roles and ask to control a different entity, the original entity continues to exist, as it is owned by VR-Forces.
- VR-Engage can initiate and receive changes to the terrain, for example, opening doors or destroying buildings. (VR-Forces acts as the dynamic terrain server.)
- VR-Engage can reflect changes to time-of-day and weather that are published by VR-Forces.
- VR-Engage allows you to assign certain tasks to entities that are simulated by VR-Forces (for example, tell a subordinate to follow you, or move to a location you designate).

To simplify the startup workflow for certain use cases, VR-Engage has the ability to launch a VR-Forces simulation engine directly from the VR-Engage user interface, and to instruct the sim engine to load and host a scenario of your choice. This allows you to avoid separately launching VR-Forces from the Start menu or command line, and automatically ensures that VR-Forces uses the same interoperability parameters as VR-Engage.

A VR-Forces simulation engine license is always required to run VR-Forces, whether you launch it from the VR-Engage user interface or launch it separately from the Start menu or command line. If you do not have a VR-Forces simulation engine license available, VR-Engage does not allow you to launch VR-Forces and host a VR-Forces scenario.

When you start VR-Engage, it looks for a running instance of VR-Forces and, if it finds one, it connects to it. If it does not find one, it offers you the option to host and load a VR-Forces scenario or start an unhosted session. If you host a VR-Forces scenario, you consume a VR-Forces license, which prevents you from starting VR-Forces independently if you have only one license. If you start an unhosted session, you can start or join a session only as an individual player—you cannot load and run a scenario.

To learn about how VR-Engage connects to a VR-Forces session or allows you to create a hosted or unhosted session, see "3.1. Starting VR-Engage and Joining a Session" on page 56.



2. Installing VR-Engage

This chapter explains how to install the VR-Engage applications and data. You must also install the MAK license manager. To use VR-Engage with HLA, you must install an RTI.

2.1. Installing VR-Engage 38
2.1.1 Installation and Data Packages 38
2.1.2 Installing VR-Engage 38
2.1.3 Installing VR-Engage PDB Files for Debugging Purposes 40
2.1.4 Installing the Toolkit 42
2.1.5 The makData Package 42
2.1.6 Uninstalling from Windows 44
2.2. Upgrading VR-Engage Simulation Model Sets 45
2.3. The VR-Engage Directory Structure 45
2.4. Installing and Setting Up the MAK License Manager 46
2.4.1 Specifying the License Server 47
2.5. Installing an RTI 48
2.5.1 Installing the MAK RTI 49
2.6. Input Devices 50
2.6.1 HMD Support 50
2.7. Factors that Affect Performance 50
2.8. Using MAK ONE Products on Virtual Machines 51
2.9. Localizing the Graphical User Interface 51
2.9.1 Translating Other Interface Files 53
2.9.2 Applying the Language Files 53

2.1. Installing VR-Engage

2.1.1 Installation and Data Packages

MAK ONE installations are provided in one or more files (packages) that you can download from the MAK website.

There are four types of installation products for MAK ONE:

- Product installers – Each product has a single main installer for each operating system.
- Optional installers – Some products have installers for developer options.
- Required data packages – MAK ONE applications (VR-Vantage, VR-Forces, VR-Engage, SensorFX, and RFView for MAK ONE) and the DI-Guy SDK have a collection of data packages that are required for those products' basic functions.
- Recommended data packages – Additional data packages (if available) are recommended to access the full capabilities of MAK ONE applications.

MAK ONE applications share the same data installation, therefore the same set of data packages is all you need for any or all applications (VR-Vantage, VR-Forces, VR-Engage, SensorFX, RFView for MAK ONE, and the DI-Guy SDK).

Dependencies

Licensing tools and dependent products have their own installers. For details, see "2.4. Installing and Setting Up the MAK License Manager" on page 46.

For interoperability using HLA, see "2.5. Installing an RTI" on page 48.

Before you install VR-Engage, please read the release notes to see if there are any special instructions for installation.

Build Information

If you need to know the details about the MAK ONE application you are using, you have two options:

- While running the application, choose **Help > About** and then read the information in the About dialog.
- In the installation directory, look at a build information text file.

2.1.2 Installing VR-Engage

Windows versions of VR-Engage are provided as downloaded files.

➤ To install VR-Engage, run the installer. Follow the instructions in the installation wizard. (You install the data used by VR-Engage separately, as described in "2.1.5 The makData Package" on page 42.)

Note the following:

- The installers are named to indicate the compiler used to build that version of VR-Engage.
- When you install large applications on Windows, there may be a delay of up to several minutes from the time you try to run the setup program to the time that an installation dialog box displays. This is due to how Windows scans setup programs before it executes them. If you experience this problem, turning off User Access Control can reduce or eliminate this delay.

Installing the VR-Forces Extensions

When the VR-Engage installer completes the data installation, it informs you of the extensions installer, which installs VR-Engage plug-ins for VR-Forces. Among other things, the extensions allow VR-Engage players to control VR-Forces entities that are configured to be controlled by VR-Engage.

If you have VR-Forces installed and want to install the extensions, run the *vrEngage2.2-win64-vc##-YYYYMMDD.VRF5.2d-extensions.exe* installer. This is packaged separately from the VR-Engage installer.

The extensions make the following changes to VR-Forces:

- Installs the *./data/simulationModelSets/VR-Engage.sms* and custom models. It also adds the scenarios included with VR-Engage to *./userData/scenarios/Sample*.
- Replaces some VR-Forces level data or configuration files. In those cases, it saves the original version is preserved by the installer by copying it to <*original file name*>*.vrengage-addons-backup*. If you uninstall the VR-Engage extensions, any VR-Forces files that the installer modified are restored by copying those **.vrengage-addons-backup* files to their original names.
- Adds the VR-Engage Manager Panel.
- Adds the VR-Engage revision to the About dialog box.

You must install the extensions on top of a compatible version of VR-Forces. Check the version number in the executable name. The installer attempts to check that the version of VR-Forces is compatible with your version of the VR-Engage extensions. You can also refer to the VR-Engage release notes to learn which version of VR-Forces you can use with your version of VR-Engage. If you install the extensions on an incompatible version of VR-Forces, it is not guaranteed to run reliably or correctly (if at all).

Default Windows Installation Directory

The default root directory for installing MAK ONE products is *C:/MAK*.

You can use an environment variable, MAK_INSTALL_DIR, to specify a different location where all MAK ONE products and data should be installed by default. For example, if you define it as MAK_INSTALL_DIR = D:\MyProject, then all MAK ONE products automatically install to that location. The data packages and products would then install in that directory by default: *D:/MyProject/SharedData*, *D:/MyProject/vrlink5.10*, and so on.

Installing the Product Documentation

Due to the limitations of Windows installers, the user and developer documentation for VR-Engage are in a separate installer. To access the documentation locally, you must run the product's **.docu.exe* installer and install the files in the product install directory.

➤ **To install the documentation:**

1. Run the product's documentation installer, *vrengage#.#.docu.exe*.
2. The wizard finds the location where you installed the product. Accept that location to ensure that links to the documentation work correctly.
3. Optionally, change the Start menu location for the shortcuts to the documentation. If you want, you can choose a location separate from product installation directory. For example, you might want to put MAK ONE documentation shortcuts in a central location. This change does not affect the documentation shortcuts folder that is already located in the product installation directory.
4. Complete the installation wizard. The product documentation will be available on your machine.

2.1.3 Installing VR-Engage PDB Files for Debugging Purposes

In cases where a MAK ONE application crashes, it can be difficult to determine the cause. The application generates a *.dmp* file that contains valuable information about the application's state at the time of the crash. MAK technical support can analyze this file to help diagnose the issue.

In addition, with the application's PDB (program database) files, the application generates a call stack log file (*.callstack.log*) of the application at the time of the crash. The call stack log provides information in plain text about the sequence of functions that were called leading up to the crash, which is also helpful in identifying the specific cause.

By providing the *.dmp* or *.callstack.log* files to technical support, you help support personnel to resolve the issues. They use specialized debugging tools to analyze the files and determine the cause of the crash, and that information aids in the development of a solution to prevent future crashes.

Due to security reasons or organization policies, some customers cannot send the *.dmp* file to technical support. Because the call stack log file is a plain text file, it is easier to inspect and analyze than the *.dmp* file. Users can easily attach it to an email and pass it out of secure areas with visual inspection, making it a convenient option for customers with restrictions on sharing binary files.

It is important to note that the call stack log file alone might not be sufficient for support personnel to fully diagnose the issue. They may require more information to gain a complete understanding of the issue. A detailed procedure for reproducing the crash often helps them solve the problem you encountered.

- Support for the generation of the call stack log file was introduced in MAK ONE 2024. It is supported only in MAK ONE products part of that release and later versions.
- The PDB files are part of the Linux installation. You do not need to install them separately.

Installing PDB Files

Before installing the PDB files on Windows, you must install VR-Engage. Next, make sure that the PDB files you are installing are from the same build as the application. When you download the installer from the MAK ONE Product Downloads page, the appropriate PDB installer is also available for download.

➤ **To install the PDB files:**

1. After installing VR-Engage, run the PDB installer for that product.
2. The installer looks for the matching product directory by searching the registry and then it automatically sets the install directory to the appropriate path. If that is not correct for your product, see the note below.

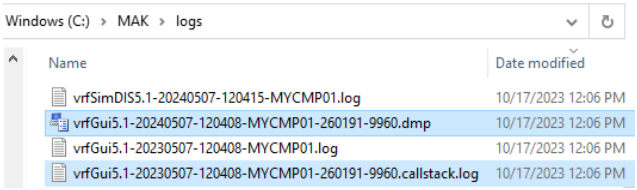
   For an installation of VR-Forces that will be used with VR-Engage, you should install the VR-Engage PDBs. This requires you to manually change the installation directory to the VR-Forces directory with no subdirectory, for example, `C:\MAK\vrforces5.2d`. (You do not need to install the VR-Forces PDBs because they are included in the VR-Engage PDB installer.)
3. Complete the installation wizard. The necessary PDB files are installed in the product's *./bin64* directory.

- The default installation directory is *C:\MAK*. See "Default Windows Installation Directory" on page 39 to learn about setting the default to other locations.
- Be sure that the folder in the Select Destination Location page of the installation wizard is correct. It should be the top-level folder for the product installation.
- If the PDB installer displays a warning that the SVN revisions do not match, make sure that you have the correct PDB installer for the version of VR-Engage that you installed.

Reporting an Issue Using the *.callstack.log* and *.dmp* Files

Send the *.callstack.log* and *.dmp* files (or only the *.callstack.log* if you are unable to provide the *.dmp* file) to MAK support through the Support Portal (https://www.mak.com/mak-one/support/portal) or support@mak.com. The files will have similar names and are located in the central directory for MAK ONE log files (Figure 7). Please also provide information on anything you experienced when the issue occurred.

Figure 7. Example of *.callstack.log* and *.dmp* files for VR-Forces



Specifying the Location for *.log* and *.dmp* Files

The default directory for MAK ONE log files is *C:/MAK/logs*. You can customize the location by specifying a different path in the MAK_LOG_DIR environment variable.

2.1.4 Installing the Toolkit

The SDK toolkit is an API that allows developers to create and add functionality to VR-Engage, and it also includes examples about how to use the SDK.

The VR-Engage installer includes an option to install the toolkit. This requires that you purchase a separate license, which comes with its own installation code.

If you skipped the toolkit installation when you installed VR-Engage but you now have a toolkit installation code, you must rerun the application installer and enter the code when prompted. Be sure to install it in the correct directory. In the Select Components page of the installation wizard, simply clear the runtime application check box and select the toolkit check box.

Required Software for Toolkit Users

If you plan to build VR-Engage applications using the VR-Engage Toolkit, you must download and install VR-Link 5.10.

To build VR-Engage applications, you must link with VR-Link 5.10. If you plan to build the VR-Engage GUI, you must also download and install Qt 5.15.2 (Windows VC++ 15 and Linux Red Hat 8). You can download it from the Bonus Downloads page on the website (https://www.mak.com/mak-one/support/help/bonus-material).

Qt is also available as a free download under the LGPL version 2.1 license at https://www.qt.io. This version should be satisfactory for most VR-Engage customers. If you need a Qt commercial license, you must purchase the license from Qt.

2.1.5 The makData Package

The MAK Data Packages are used by multiple MAK ONE applications. The applications need to be able to find the data, wherever you install it. VR-Engage looks for data in a default location or uses a collection of environment variables to find the data in the location where you installed it. You have several options for installing the data and telling VR-Engage where to look for it.

MAK Data Installation Directory

The shared data files are stored in a directory file hierarchy. This structure allows for multiple versions of data to be installed for use with different versions of VR-Engage.

The directory looks like *shared-data-root/data-compatibility-version/stem/data-folders*, where:

- *shared-data-root* is the location of the shared data directory. The default is *C:/MAK/SharedData* on Windows and *~/MAK/SharedData* on Linux.
- *data-compatibility-version* is a number, such as 18 or 19.
- *stem* is the directory holding the data files, for example, *latest* for the most recent installation of that compatibility version or the date of a previous installation, such as *2024-03-14*.
- *data-folders* hold the various files that make up the MAK data files (models, audio files, fonts, and so on).

Advanced Windows users may want to install MAK Data Packages in a very specific way, for example, installing the data on a shared drive and configuring any machines with MAK ONE applications to look there for the data. Refer to the *MAK ONE Installation Guide* for instructions.

Installing the makData Package

➤ **To install the data:**

1. Install the MAK ONE Software Manager for your operating system.
2. Run the MakOneSoftwareManager executable in the .*bin64* folder.

   On Windows, you can also open the Start menu and then choose **MAK Technologies > MAK ONE Software Manager**.
3. Click Select and Install Data Packages in the welcome page.
4. Specify the directory where you saved the data files. The default is your Downloads folder, but you can click the folder icon to navigate to another folder.
5. Select the data files that you want to install.

   > MAK Technologies recommends that you install all of the data files if you have the space and are not absolutely sure which files you require.
   >
   > You must install the Core data for VR-Engage to function. To have access to other terrains and to see anything other than notional models, install the other data packages provided by MAK. If you do not install any other data, it is assumed that you are providing your own models and terrains.

6. Optionally, choose the location where you want to install the data.

   MAK recommends that you install the files in the default location. While MAK ONE products allow for diverse installations, the typical use is the standard installation. If you require a more complex installation, refer to the *MAK ONE Installation Guide*.
7. If you have a previous installation of data for the same compatibility version, you are prompted to choose whether to back up the previously installed data, delete it, or overwrite it. See "Options When Installing a Data Package" on the next page.

8. Click Install. The MAK ONE Software Manager shows the progress of the data installation.
9. When the installation is complete, click Exit.

Options When Installing a Data Package

When installing a data package using the MAK ONE Software Manager, if you have no preexisting installations of that compatibility version, it simply installs the data files in the *./SharedData/##/latest* folder (where *##* is the compatibility version).

If there is already a previous release of the same compatibility version installed, however, you are prompted to choose whether to back up, delete, or overwrite the data from the previous release.

- Backup. If you install a newer release and choose to create a backup of the previous release, the files in the *latest* directory are moved to a new directory named for the data version's date in the same compatibility folder and the new files are then installed in the latest folder. For example, the old files move to *./SharedData/18/2024-03-14* and the new files are installed in *./SharedData/18/latest*.

  Creating a backup is the recommended option in most circumstances. It allows all MAK ONE applications to use the newest data, while also giving you the option to revert to a previous release using command-line options or setting a Windows environment variable. See the *MAK ONE Installation Guide* for details.
- Delete. The installer deletes the old data from the *latest* directory and then installs the new data.
- Overwrite. The installer overwrites the previous data files in the *latest* directory with the newer ones you chose to install. It also adds files that are new and ignores files that were previously installed if the package you are installing does not include that file.

2.1.6 Uninstalling from Windows

MAK ONE product installations include an uninstaller application (*unins000.exe*), which you can find in the installation directory. Run that program to uninstall the product.

When you run other installers for the product, such as the PDBs and documentation, there are additional uninstallers. If you run the uninstallers from the highest number down (for example, *unins002.exe* and then *unins001.exe*), running *unins000.exe* uninstalls everything else and removes the install folder.

When you uninstall VR-Engage, files such as scenarios or SMSs that reside in the product directory are also deleted. Before uninstalling, copy any of your custom files from the product directory to another folder.

The uninstaller does not remove the data package. That is in a separate directory (for example, *./SharedData/19/latest*).

If you run only the product uninstaller, you can then simply delete the rest of the product's install folder. Note that if you do not run all of the uninstallers, running only the product uninstaller may not remove all of the shortcuts from the Windows Start menu.

2.2. Upgrading VR-Engage Simulation Model Sets

When you upgrade to new, compatible versions of VR-Engage and VR-Forces, you must upgrade your simulation model set, as well. Refer to the documentation for the compatible version of VR-Forces, specifically the *VR-Forces Migration Guide* and "Upgrading Old Simulation Model Sets" in the *VR-Forces Users Guide*.

2.3. The VR-Engage Directory Structure

The installation process creates the VR-Engage directory structure shown in Figure 8. On PCs, the default installation directory is *C:\MAK\vrengage#.#.*)

Figure 8. VR-Engage directory structure

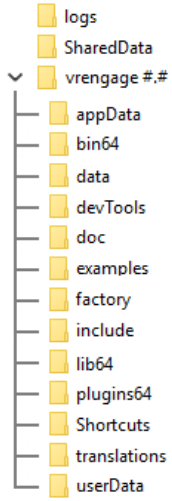


Table 1 describes the contents of the directories.

Table 1. Contents of VR-Engage directories

Directory	Directory contents
./appData	Subdirectories contain data required by the VR-Engage applications, such as schema, model definitions, settings, configuration files, and other data. Also contains data required by the SensorFX plug-in for VR-Engage.
./bin64	Product and utility executables, dynamic link libraries (DLLs). Also where PDBs are installed.

Table 1. Contents of VR-Engage directories (continued)

Directory	Directory contents
./data	The *./data* directory has several subdirectories that contain: • Bitmaps and icons • Simulation model sets • Terrain database source files • Visual model data • Other data required by the application
./devTools	Files to support toolkit users.
./doc	VR-Engage documentation. This is the user documentation, both PDF and HTML5.
./examples	Sample code for toolkit users.
./factory	Backups of the application data and settings provided by MAK. Also includes the default settings for SensorFX (for reverting to "factory" settings).
./lib64	Library files.
./plugins64	Plug-in shared libraries.
./Shortcuts	Contains the shortcuts to the documentation and tools, which are used for the Windows Start menu.
./translations	Translation files for localizing VR-Engage menus.
./userData	Terrain and scenario files.

2.4. Installing and Setting Up the MAK License Manager

Before you can use a MAK ONE product, you must obtain a valid license file, install the MAK License Manager, and configure the license server and client machines. The License Manager uses a client-server architecture, so you do not need to install the License Manager on every computer on which you install MAK ONE products. You only need to install it on the computer that you will use as the license server.



If you have already installed the License Manager for another MAK ONE product, you do not have to install it again. You only need licenses for your newly installed products.

The License Manager installer is included with MAK installation media. It is separate from the product installers. You can download the installers from the MAK ONE Product Downloads page at https://www.mak.com/mak-one-downloads.

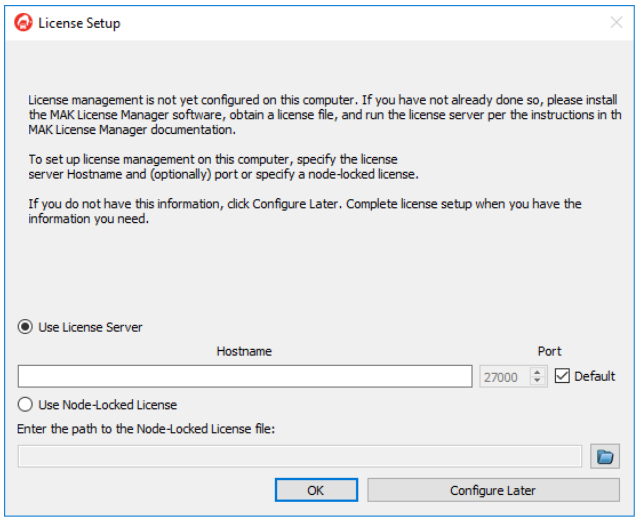
Complete installation and configuration instructions are included with the License Manager installer. The *MAK ONE Installation Guide* is available at https://mak.com/mak-one-installation-guide.

Some customers use dongle licenses instead of running a license server. Instructions for using dongles are also in the *MAK ONE Installation Guide*.

2.4.1 Specifying the License Server

The first time you run a MAK ONE product on a particular computer, the License Setup dialog box opens (Figure 9). It prompts you to enter the hostname of the license server and, optionally, a port number.

Figure 9. License Setup dialog box



If you do not know the hostname of the license server, click Configure Later. When you have the hostname, you can start the application again and complete the dialog box. You will not be able to run any MAK ONE products until you set up license management.

If you know the hostname, enter it in the Hostname box and then click OK. The application will start.

Under limited circumstances, MAK issues node-locked licenses. If you have a node-locked license, select the Use Node Locked License option and enter the path to the license file.

- If you are running MAK ONE products on the license server machine, it is also a client, so you must specify the license server on that machine, too.
- If you change the license server, the saved configuration will no longer be valid and the License Setup dialog box will open the next time you start a MAK ONE product.
- If you have changed the license server, you may want to clear the license cache. You can clear the saved license configuration by deleting the cache file. On Windows, it is *C:\Users\user_name\AppData\Roaming\MAK\licenses<n>.txt*. (The *AppData* directory is hidden by default.) On Linux, it is *.mak/license<n>.txt*. (There may be more than one cache file, for example, *licenses1.txt* and *licenses2.txt*.) You can also clear the cache in the RTI Assistant.

Setting the MAKLMGRD_LICENSE_FILE Environment Variable

As an alternative to using the license setup procedure described in the previous section, you can configure the license server or node-locked license in an environment variable. The MAKLMGRD_LICENSE_FILE environment variable identifies the server machine. If you set this environment variable, it overrides the settings stored by the license setup procedure.

For license servers, the syntax for the environment variable is: `@server_name`. For example, if the server machine is oak, set the environment variable to `@oak`.

The following sections explain how to set environment variables on the different platforms that MAK ONE products run on.

Windows

➤ **To set MAKLMGRD_LICENSE_FILE in Windows:**

1. On the Start menu, click the Settings button (⚙). The Settings window opens.
2. In the Search box, enter `environment`. A list of choices displays.
3. Select Edit the System Environment Variables. The System Properties dialog box opens to the Advanced tab.
4. Click the Environment Variables button. The Environment Variables dialog box opens.
5. Below the System Variables list, click New. The New System Variable dialog box opens.
6. In the Variable Name field, enter `MAKLMGRD_LICENSE_FILE`.
7. In the Variable Value field, enter `@server_name`, where *`server_name`* is the name of the license server.
8. Click OK to save out of each dialog box and set the variable.

Linux

On Linux, you set environment variables in your *.cshrc* (or equivalent startup file).

➤ To specify a license server, set the variable similarly to:

```
setenv MAKLMGRD_LICENSE_FILE @oak
```

If you are using the sh or bash shells, you set environment variables in your *.profile* file (or *.bashrc*). Set the variable similarly to:

```
MAKLMGRD_LICENSE_FILE=@oak
export MAKLMGRD_LICENSE_FILE
```

Do not put spaces around the equal (=) sign.

You are ready to run the license server and use your new licenses or MAK ONE products.

2.5. Installing an RTI

An RTI is a software library (and perhaps supporting executables) that implements an HLA Interface Specification. In HLA, applications exchange FOM data through RTI calls, which means that all HLA applications need to use an RTI. MAK Technologies has the MAK

RTI, but MAK ONE products do not require that you use the MAK RTI.

> Because of differences in the low-level network mechanisms used by different RTI implementations (which include, but are not limited to, packet layout), applications that want to interoperate in the same federation execution must use the same RTI implementation.

Because RTIs are usually provided as dynamic libraries that implement a fixed API, a federation can often switch from one RTI implementation to another between runs (without even recompiling the applications), but during each run, all participants must agree on which RTI to use, much as they must also agree on which FOM to use.

For the most recent information about the RTI versions supported by MAK ONE products, please visit the Product Specifications & Dependencies page (https://www.mak.com/mak-one-specs).

The following instructions explain how to install the MAK RTI. If you are using an RTI provided by another vendor, please refer to that RTI's documentation for installation and setup instructions.

2.5.1 Installing the MAK RTI

To install the MAK RTI, follow the instructions in Chapter 2 of *MAK RTI Users Guide*. To install another RTI, see the vendor's documentation.

Configuring Your System to Use the MAK RTI

The RTI dynamic libraries must be located somewhere on your dynamic library search path. The path is specified in the PATH environment variable. The path is indicated by an environment variable:

- Linux — LD_LIBRARY_PATH.
- Windows — PATH.
- The MAK RTI needs to know where to find these configuration files:
  - The federation configuration file (*.fed*, *.fdd*, or *.xml*) for your federation execution (required).
  - The RID file (*rid.mtl*) (optional).

Put the configuration files in the directory from which you are running, or set the environment variable RTI_CONFIG to the directory that contains them.

Running Applications with the MAK RTI

➤ **To run a MAK ONE product with the MAK RTI:**

1. Ensure that the license server is running.
2. Start the application. The RTI Assistant prompts you to choose an RTI configuration.
3. Choose a configuration. If necessary start the rtiexec.
4. Click Connect. The application should run.

In many cases, you do not need to run the rtiexec to use the MAK RTI, however you can run the rtiexec if you want to. (It is required to use certain features of the MAK RTI.) For more information, please see your RTI documentation.

2.6. Input Devices

VR-Engage can support any device that Windows recognizes as a game controller. VR-Engage is tested with these game controllers and has configurations for them:

- Xbox controller
- Logitech F310 Gamepad
- Logitech F710 wireless Gamepad
- Logitech G27 Steering Wheel
- Logitech G29 Driving Force game steering wheel
- Thrustmaster Hotas Warthog
- Thrustmaster T-Flight Hotas X

The mappings for these devices and the keyboard are described in "Chapter 12. Mouse, Keypad, and Controller Mappings" on page 241. Please contact support (https://www.mak.com/mak-one/support/portal) for assistance configuring other devices.

2.6.1 HMD Support

VR-Engage 2.2 supports a number of virtual and mixed reality devices. See "3.10. Introduction to Head-Mounted Displays" on page 78 and also the chapter *Using Head Mounted Displays for Virtual and Mixed Reality* in the *VR-Vantage User Guide* to learn more.

2.7. Factors that Affect Performance

The performance of a visualization product such as VR-Engage depends on:

- Processor speed.
- Video card. (Always use the latest drivers for your video card.)
- Available memory.
- Size of the terrain database.
- The number and complexity of models loaded and how well they have been optimized for performance. For information about optimizing models, see "Best Practices for Creating Models" in the *Adding Content to MAK ONE Applications Reference Manual*.
- Number of objects in the simulation and how active they are.
- Use of model instancing.



To learn how to improve performance, see "Chapter 4. Optimizing Performance" on page 117.

2.8. Using MAK ONE Products on Virtual Machines

You can run MAK ONE products on virtual machines. For more information, see the *Running MAK ONE Products on Virtual Machines or Docker Containers* guide on our website at https://www.mak.com/docs/products/MAK_ONE_Running_On_Virtual_Machines.pdf.

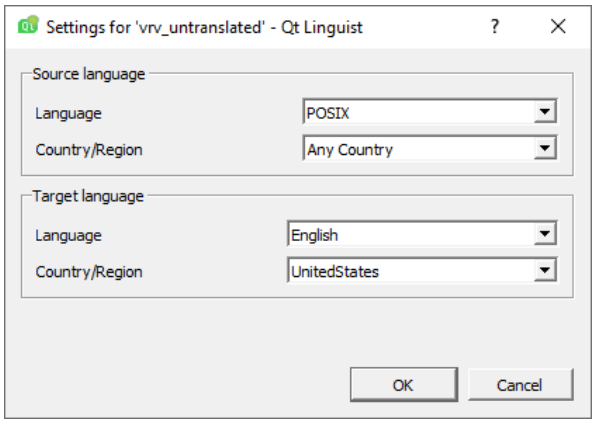
2.9. Localizing the Graphical User Interface

You can localize your MAK ONE product's graphical user interface so that it uses languages other than English. Your product includes a file called *./translations/product_untranslated.ts*, for example, *vre_untranslated.ts* or *vrv_untranslated.ts*. The file contains entries for all of the application-specific menu and dialog box text. The Qt Linguist tool, which is included with VR-Engage, lets you map the English text provided by MAK to your chosen language.

➤ **To edit a resource file:**

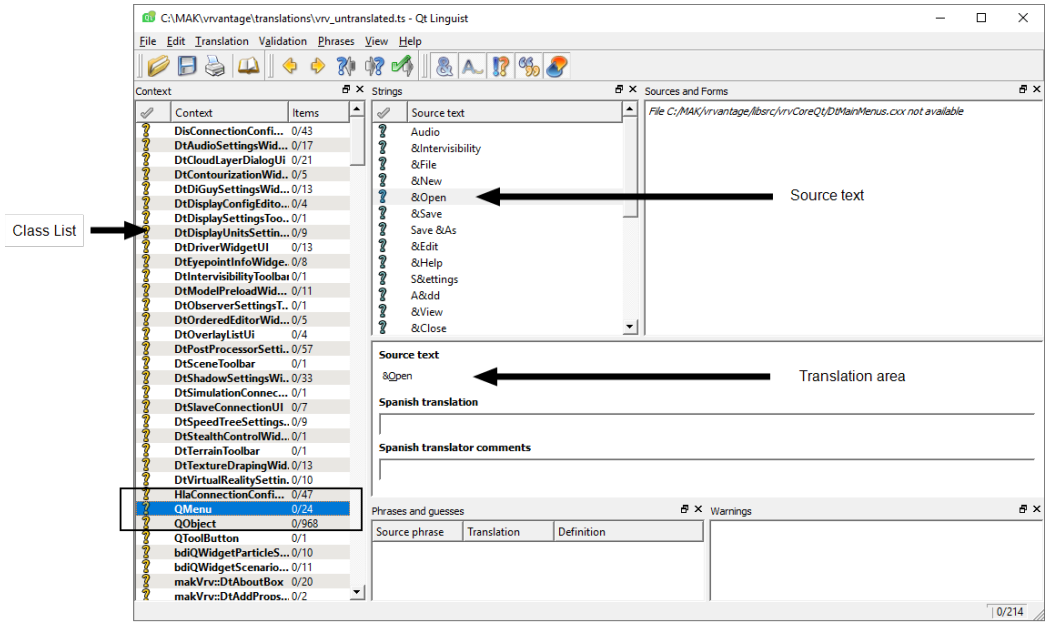
1. Start the Qt Linguist utility. It is *./translations/linguist.exe*
2. In the Qt Linguist window, choose **File > Open**. Select the file *./translations/product_untranslated.ts*. The Settings for *filename* dialog box opens (Figure 10).

Figure 10. Linguist settings dialog box



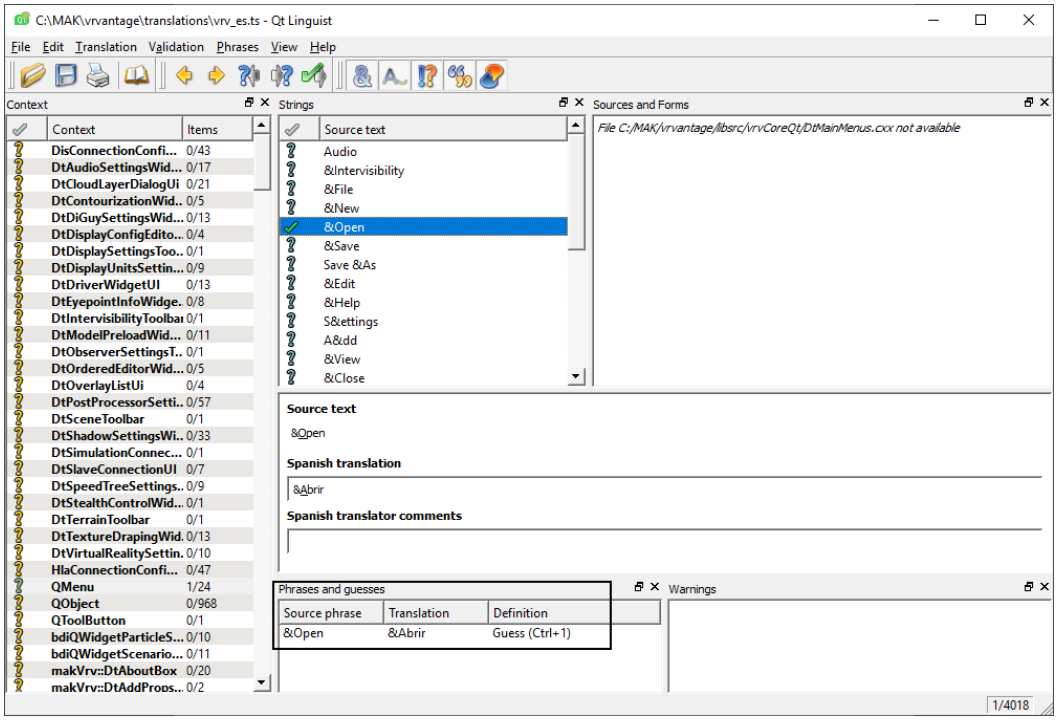
3. Select values for the source language, target language, and target region or country.
4. Click OK. The translation file opens and a list of classes displays on the left side of the window (callout 1 in Figure 11).

Figure 11. Linguist window



5. To translate text, select each class in turn and translate the text associated with that class. You do not have to understand anything about what the classes mean; you only need to translate the text. In Figure 11, **QMenu** is selected. The Strings window lists text that can be translated (callout 2). Text that has not been translated has a blue question mark in the checkmark column.
6. In the Source Text list, select the word you want to translate. The text displays under Source Text in the translation area (callout 3 in Figure 11).
7. Place the cursor in the text box below *language* Translation, where *language* is the target language that you selected when you opened the translation file.
8. Type the translation text.
9. When you are done and you are certain of the translation, select the Done and Next button () on the toolbar. A green check mark replaces the question mark (Figure 12). If you move to a different text string without clicking Done and Next, a yellow question mark displays for the string you translated. When you translate all elements for a class, a green check mark replaces the question mark next to the class name.

Figure 12. Completed translation for Open



10. Repeat this process for each element of source text for each class.
11. Choose **File > Release**. The file is saved in the format *filename.qm*.
12. Rename the file to a name that indicates the language to which you have translated the file, using the two-character abbreviation typically used with your language. For example, a Spanish translation would be renamed *product_es.qm*.
13. Save your translated *.ts* file.

2.9.1 Translating Other Interface Files

The *vre_untranslated.ts* or *vrv_untranslated.ts* file contains all the text used in the VR-Engage interface. However, there may be other text and messages that are not specific to VR-Engage that are generated by the software that implements the VR-Engage GUI. To translate this text, you need to translate the file *qt_untranslated.ts*. Follow the same procedure as you did for *vre_untranslated.ts*.

2.9.2 Applying the Language Files

➤ **To put the language file into effect, use one of the following methods:**

- Set your computer's environment to the proper language (LANG environment variable).
- Set the locale to the proper language.
- Start VR-Engage with the `(--locale | -G)` *`language`* command-line option, where *`language`* is the two-letter code for the local language.



3. Running VR-Engage

This chapter describes how to start up VR-Engage, connect to a session, and choose a role.

3.1. Starting VR-Engage and Joining a Session 56
- 3.1.1 Hosting a VR-Forces Session 57
- 3.1.2 Starting an Unhosted Session 58
- 3.1.3 Admin, User, and Lockdown Modes 59
- 3.1.4 Hosting an Entity in VR-Engage While Participating in a VR-Forces Session 60

3.2. Choosing a Role 63
- 3.2.1 Choosing an Existing Role 64
- 3.2.2 Adding an Entity to a Session 65
- 3.2.3 Using an HMD with an Entity 66
- 3.2.4 Assigning Roles from VR-Forces 67

3.3. Connecting to VR-Forces 69

3.4. Changing Your Role 70
- 3.4.1 Changing Your Role in the Choose Role Panel 70
- 3.4.2 Changing Your Role by Embarking on a Vehicle 70
- 3.4.3 Switching Roles on a Vehicle 71

3.5. Entity Engagement and Scenario State 71

3.6. The Application Menu 72

3.7. Starting a New Session 73

3.8. Configuring the Simulation Connection 73
- 3.8.1 HLA Connection Settings 74
- 3.8.2 DIS Connection Settings 76

3.9. Playing in Windowed Mode 77

3.10. Introduction to Head-Mounted Displays 78
- 3.10.1 Prerequisites for Using an HMD 78
- 3.10.2 Resetting the Tracking Sensor 78
- 3.10.3 Calibrating the Eye Tracking for Varjo 79

3.10.4 VR Anchoring for Mask Tracking 79
3.10.5 Enabling and Disabling Virtual Reality Rendering 80
3.10.6 Configuration Options for Virtual and Mixed Reality 81
3.10.7 Setting Up Mixed Reality 82
3.10.8 Optimizing Performance with HMDs 84
3.11. Creating a Startup Configuration 85
3.11.1 Minion and Boss Configuration 85
3.12. Starting Multiple Simulation Engines 86
3.13. Command-Line Options 87
3.13.1 Command-Line Options for vrEngage 87
3.13.2 Command-Line Options for vrEngageSim 102
3.13.3 Command-Line Options for Player Stations 112
3.14. Managing Plug-ins 112
3.14.1 Loading Plug-ins 113
3.14.2 Adding a Plug-in 114
3.14.3 Specifying the DLLs for a Plug-in 114
3.14.4 Adding a Plug-in Configuration 115
3.14.5 Deleting a Plug-in Configuration 116
3.14.6 Deleting a Plug-in 116

3.1. Starting VR-Engage and Joining a Session

When you start VR-Engage, it looks for an instance of VR-Engage or VR-Forces that is running, to check for a session to join. A session manages the relationship between the simulation engine, or back-end, and the GUI, or front-end. For more information, see the chapter *VR-Forces Application Concepts* in the *VR-Forces User Guide*.

If VR-Engage finds a session, it immediately connects to it, loads the terrain it is using, and then shows you a list of entities that you can control. If it does not find a session, it stops and prompts you to choose whether to host a VR-Forces session or start an unhosted session.

You can set some of the startup behaviors with a startup configuration script. For details, see "3.11. Creating a Startup Configuration" on page 85.

If you join a session hosted by either VR-Forces or another instance of VR-Engage, your instance of VR-Engage does not create a sim engine by default.

To run a simulation engine for an instance of VR-Engage when joining another session, make the configuration changes described in "Configuring VR-Engage to Start a Sim Engine" on page 61.

3.1.1 Hosting a VR-Forces Session

VR-Engage includes scenarios that were created using VR-Forces. When you host a VR-Forces session, you can load one of these scenarios or start an empty scenario on any installed terrain. If you run an existing scenario, you can control any of the entities in them. You can also add and control one other entity. If you start an unhosted (empty) scenario, you can only add and control one entity.

If other instances of VR-Engage are running, they can connect to the session. Each instance can only control one entity.

- You cannot save scenarios in VR-Engage. The principal benefit of hosting a VR-Forces session is that a VR-Forces session supports global weather and dynamic terrain. If you want to create and save scenarios for use with VR-Engage, you must run VR-Forces with the VR-Engage extensions.
- When you host a VR-Forces session, you use a VR-Forces license. If you only have one VR-Forces license and you host a VR-Forces session from VR-Engage, you will not be able to start the VR-Forces application.
- By default, the list of scenarios comes from those saved in the *./userData/scenarios* directory, including any subfolders. You can change this by editing the USER_DIR environment variable or the `defaultScenariosDirectory` definition in the *./appData/scripts/playerStation.lua* file.

➤ **To start VR-Engage and host a VR-Forces session:**

1. On the Start menu, choose **MAK Technologies > VR-Engage 2.2**. The VR-Engage window opens.
2. In the Choose Connection screen, select the connection that you want to use and then click Connect. The Searching for a Session screen displays.

   Optionally, you can change the connection settings. For details, see "3.8. Configuring the Simulation Connection" on page 73.

3. If VR-Engage finds a VR-Forces session, it connects to it.

   If it does not find a session, the Searching for Session screen provides you with the options to host a VR-Forces session or start an unhosted session (Figure 13).

Figure 13. Searching for Session



> By default, the VR-Engage window fills the screen, which we call full-screen mode. Admin mode shows title bar and menu bar typical of most windowed applications. We call this windowed mode. Full-screen mode does not have menus or other window controls. Although full-screen mode is the default for controlling characters, you can also control them in windowed mode. For details, see "3.9. Playing in Windowed Mode" on page 77.

4. Click Host VR-Forces Session. The Choose a Scenario for this Session dialog box opens, where you can load a scenario.
5. Select the scenario you want to load. Any information about the scenario displays at the right. If you are in Admin mode, the simulation model set also displays below the scenario information.
6. Click Load Scenario. A loading screen displays while opening the scenario and starting a sim engine. Once the terrain and scenario load, VR-Engage displays the scenario in 3D View mode and opens the Choose Role Panel (Figure 16 on page 63).
7. Choose a role, as described in "3.2. Choosing a Role" on page 63.

3.1.2 Starting an Unhosted Session

In an unhosted session, you cannot load scenarios. You can load an installed terrain and create a single entity on it to control. If there are other VR-Engage unhosted sessions running on the network, you can interact with them.

- Unhosted sessions do not support dynamic terrain. Destructible buildings and buildings with switchable state windows or doors do not function in an unhosted session. As a result, players cannot damage buildings and cannot open and close doors or windows on buildings.
- By default, the list of terrains comes from those saved in the *./SharedData/19/latest/TerrainData/TerrainConfiguration* directory. You can change this by editing the SHARED_DATA_DIR environment variable or the `defaultTerrainsDirectory` definition in the *./appData/scripts/playerStation.lua* file.

➤ **To start VR-Engage and host a VR-Forces session:**

1. On the Start menu, choose **MAK Technologies > VR-Engage 2.2**. The VR-Engage window opens.
2. In the Choose Connection screen, select the connection that you want to use and then click Connect. The Searching for Session screen displays (Figure 13 on the previous page).
3. Click Start Unhosted Session. The Choose Terrain list displays.
4. Select a terrain. Any information about the terrain displays at the right. If you are in Admin mode, the simulation model set also displays below the terrain information.
5. Click OK. VR-Engage displays the terrain in 3D View mode and the Choose Role Panel opens (Figure 16 on page 63).
6. Create an entity to control, as described in "3.2.2 Adding an Entity to a Session" on page 65.

3.1.3 Admin, User, and Lockdown Modes

There are three modes for VR-Engage: Admin (administrator), User, and Lockdown:

- User mode, which is the default, is meant for role players that are allowed to set the DIS/HLA networking parameters, choose the sessions and entities that they engage with, and select their roles. This mode allows users to host a VR-Forces session (assuming you have a VR-Forces license), join an existing VR-Forces session, or add a new VR-Engage-owned entity to an exercise hosted by VR-Engage. Users can play and pause the scenario that they are hosting, or are participating in an unhosted scenario.

  User mode does not allow the player to access the Qt-based menus and toolbars that VR-Engage inherits from VR-Vantage to set advanced visualization and application settings. It also omits administrative functionality, such as Reset and Reset in Place. You can switch to Admin mode to access those functions.
- Admin mode is for developers and administrative staff, allowing them to configure, debug, customize, and tune the performance of VR-Engage. It includes all of the capabilities of User mode, and adds the ability to edit Display Engine configurations as well as access many of the VR-Vantage rendering engine's menus and toolbars. From those UI elements, you can control and monitor visuals such as shadows,

high-dynamic range rendering (HDR), lighting, ocean settings, time and weather, and much more. Admin mode also allows you to reset and restore an entity you are controlling if it is destroyed or otherwise inoperable.

- Lockdown mode is intended for deployed training systems where the scenarios will be loaded and roles assigned by an instructor/operator using VR-Forces. It is more restrictive than User mode because it disables the Choose Session, Choose Role, and Create Entity functions. The player can only play the roles assigned to them through the instructor/operator station, scripts, or command-line options.

There are three methods for specifying the mode:

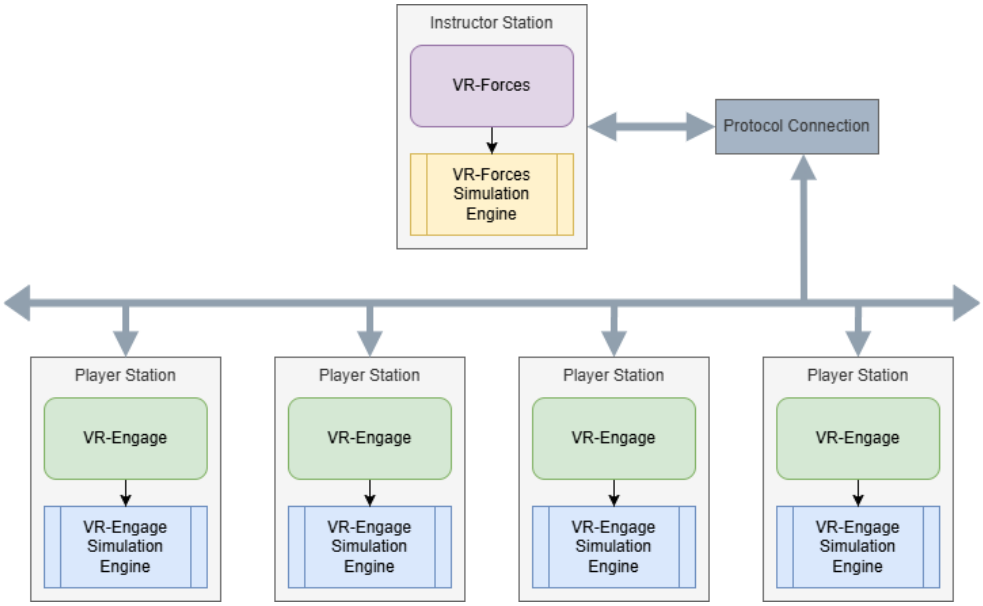
- Use the `--appMode` command-line option to specify which mode you want to use: `user`, `admin`, or `lockdown`.
- At runtime, toggle between the Admin and User modes by pressing **Shift+F12**. This does not work in Lockdown mode by default. You can choose to allow for Admin mode in Lockdown by using the `--enableAdminMode` command-line option.
- Specify the mode to use in the *./appData/scripts/playerStation.lua* file's `applicationMode` parameter.

3.1.4 Hosting an Entity in VR-Engage While Participating in a VR-Forces Session

If your setup requires multiple players to connect to a scenario hosted in VR-Forces, you can choose to configure each player's instance of VR-Engage to run its own simulation engine where it will host that player's entity. This improves performance for the overall simulation, partly because the VR-Forces sim engine does not manage the VR-Engage entity's inputs and statuses while it simulates CGF entities in the scenario. In more advanced use cases, this also allows each VR-Engage sim engine to use its own plug-ins that are not required for the VR-Forces scenario.

Figure 14 shows a setup where the VR-Forces sim engine does not simulate any VR-Engage entities. Each VR-Engage player station hosts one entity that the player controls. (You do not need additional VR-Forces licenses to host a single entity in an instance of VR-Engage.)

Figure 14. Architecture for multiple sim engines



Configuring VR-Engage to Start a Sim Engine

➤ **To configure VR-Engage to start a sim engine when joining a session hosted in VR-Forces:**

1. Edit the *./appData/scripts/playerStation.lua* file.
2. Uncomment the `runOwnshipOnJoinedSession` parameter and set it to `true`.
3. Uncomment the vreSimAppId parameter and give it a unique number for that player station. You must use a different number for each station.

   The updated code looks similar to:

```
-- Control when to run local simulation engine instances.
-- Default: only run local sim when joining unhosted sessions.
-- Set to true to also run local sim engine when joining
-- hosted sessions.
runOwnshipOnJoinedSession = true;

-- VR-Engage simulation site identifier for DIS networking.
-- Used to distinguish between different simulation sites
-- in multi-site exercises.
-- Valid range: 0-65535. Default: 1 if not specified.
-- vreSimSiteId = 1;

-- VR-Engage simulation application identifier for DIS networking.
-- Used to distinguish between different applications within
-- a site.
-- Valid range: 0-65535. Default: random number between 3200-4000
-- if not specified.
vreSimAppId = 3800;
```

```
-- Session ID for matching with specific VR-Forces instances.
-- When set, VR-Engage will only connect to VR-Forces with
-- the same session ID.
-- Useful for coordinating multiple simulation instances.
-- vrfSessionId = 1;
```

Manually Load Balancing the Scenario in VR-Forces

When you load the scenario in VR-Forces, specify that you want to distribute the entity load. You then assign entities to VR-Engage sim engines. Note that:

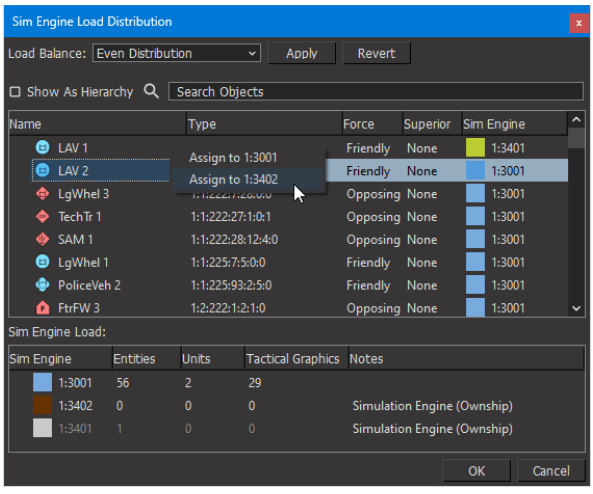
- The dialog box lists all entities in the scenario. It is not limited to those that can be controlled by VR-Engage players.
- Assigning a VR-Engage entity to a sim engine is not the same as assigning the player to a role or even that entity. See "3.2.4 Assigning Roles from VR-Forces" on page 67.

> You can also create and save a scenario with the desired load distribution. See "16.2.1 Creating a Load-Balanced Scenario" on page 374.

➤ **To load-balance a existing scenario when you open it:**

1. Start VR-Forces and the instances of VR-Engage on their respective stations. Ensure that they are connected.
2. In VR-Forces, choose **File > Load Scenario**. The Load Scenario dialog box opens.
3. Select the Show Load Balancing Dialog Before Scenario Load check box.
4. Navigate to and select your scenario.
5. Click Open. The Sim Engine Load Distribution dialog box opens, listing the entities in the scenario as well as the VR-Forces and VR-Engage simulation engines.
6. For each entity that should be controlled by a player, right-click on the entity and assign it to the correct sim engine (Figure 15).

Figure 15. Sim Engine Load Distribution



> Each VR-Engage sim engine is in Ownship mode, which means that it is limited to hosting one entity. Once you assign an entity to a sim engine, the line grays out.

7. After assigning all VR-Engage entities as desired, click OK. The scenario loads and the VR-Engage entities are simulated by the specified VR-Engage sim engines.

> You can also transfer ownship of entities after loading the scenario. In VR-Forces, choose **Simulation > Load Distribution** and then change the load balance as described above. At this time, this is supported only in HLA Evolved and HLA 4.

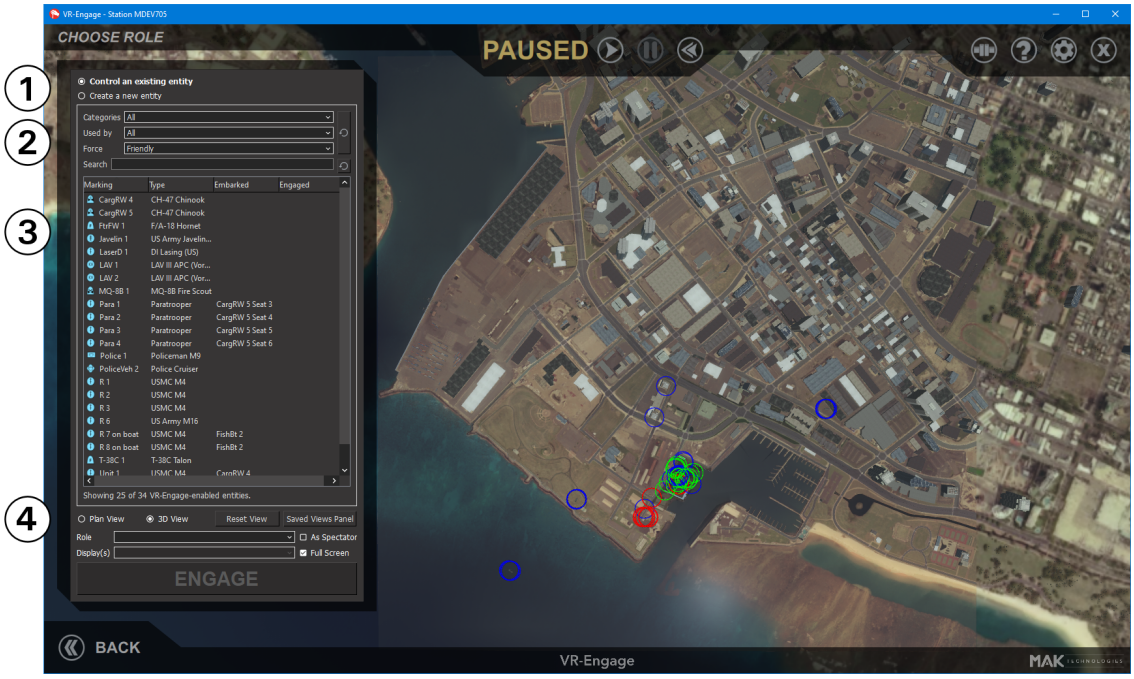
For more information on loading and running scenarios in VR-Forces, refer to the VR-Forces help or *VR-Forces User Guide*.

3.2. Choosing a Role

When you connect to or start a session, you can then choose a role to play. The VR-Engage window lists the available roles in the Choose Role Panel (Figure 16). If another player station is controlling a role, that is indicated in the Engaged column and you cannot take control of that role.

If you load a scenario or connect to a simulation, you can choose one of the entities in the simulation or create a new entity (except in lockdown mode). If you start an unhosted session, you must create a new entity and choose that entity to control.

Figure 16. Choose Role Panel



- 1 – Action type
- 2 – Filters
- 3 – Entity list
- 4 – View mode

VR-Engage users typically control a character in the simulation. However, you can also participate as a spectator. A spectator attaches to an entity in the same way as a player, but cannot control the entity. The entity is controlled by another player or by another application, such as VR-Forces. While the scenario is running, the spectator's display indicates that they are not actively playing the role.

When you host a session and engage in a role, the scenario begins to run. If you join a session hosted by VR-Forces or another instance of VR-Engage, the user of that application controls when the scenario is running or paused. If you create an unhosted session, the scenario does not begin to run until you click Play.

> Multiple players should not try to control the same vehicle or person in the same role. If multiple players take control of an entity in the same role, and then one of the players relinquishes control by returning to the Choose Role Panel, the remaining player also loses control of that entity. If this happens, the remaining player can regain control of the entity by going through the Choose Role process.
>
> Note that two players can concurrently play different roles in the same vehicle, for example, one player is Driver and the other is Gunner or Commander.

3.2.1 Choosing an Existing Role

If you load a scenario or connect to a simulation that is publishing entities, you can select one of them as your role.

➤ **To choose a role:**

1. In the Choose Role Panel, select Control an Existing Entity. The panel lists the entities in the loaded scenario.
2. Optionally, filter the list by category, used by (country), force, or search string.
3. Optionally, switch from 3D view to the Plan View display mode (2D) by selecting an option at the bottom of the panel.
4. Select an entity by choosing it from the list or clicking it on the terrain.
5. If the selected entity has multiple roles, such as those on a vehicle, select an option from the Role list.
6. If the selected entity is an aircraft or a vehicle with different display options, select a display option from the Display(s) list. (For information about aircraft display options, see "8.1.4 Aircraft Display Options" on page 202.)
7. If you just want to observe the simulation from the perspective of the selected character but not actually control it, select the As Spectator check box.
8. If you want to use VR-Engage in full-screen mode, select the Full Screen check box.

If you want to use VR-Engage in windowed mode, clear the Full Screen check box.

9. Click Engage. You can now control your character and participate in the scenario (or watch, if you chose to be a spectator).

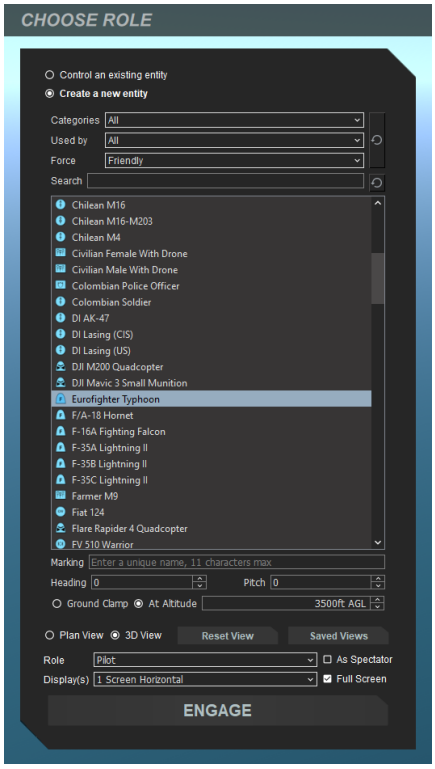
3.2.2 Adding an Entity to a Session

When you are hosting a session or running an unhosted session, you can create a single entity and control it. That entity can participate in the simulation but, if you change roles, it is removed from the simulation.

➤ **To create an entity:**

1. In the Choose Role Panel, select the Create a New Entity option. The window lists the entity types that you can create.
2. Optionally, filter the list by category, country (used by), force, or search string.
3. Optionally, switch from the 3D view to the Plan View display mode (2D) by selecting an option at the bottom of the panel.
4. Select an entity in the list (Figure 17).

Figure 17. List of entities to create



5. Optionally, in the Marking box, you can type a name for the character. The maximum length is 11 characters.
6. Optionally, specify a heading.

7. Optionally, specify an altitude. By default, humans and ground vehicles are placed on the ground. Aircraft are placed at the altitude specified in the Altitude box. If you want an aircraft to be placed on the ground, you can change the specified altitude or select the Ground Clamp option. Similarly, if you want to create a paratrooper in the air, select the At Altitude option and specify an altitude.
8. Click on the terrain where you want to place the entity. (You can navigate the Plan View or 3D view using your mouse and keyboard. For details about navigating the terrain, see the chapter *Moving the Observer* in the *VR-Vantage User Guide*.)
9. If the selected entity has multiple roles, such as those on a vehicle, select an option from the Role list.
10. If the selected entity is an aircraft or a vehicle with multiple views, select a display option from the Display(s) list. (For information about aircraft display options, see "8.1.4 Aircraft Display Options" on page 202.)
11. If you simply want to observe the simulation from the perspective of the selected character but not actually control it, select the As Spectator check box.
12. If you want to use VR-Engage in full-screen mode, select the Full Screen check box.

    If you want to use VR-Engage in windowed mode, clear the Full Screen check box.
13. Click Engage. You can now control your character and participate in the scenario (assuming you did not choose to be a spectator).

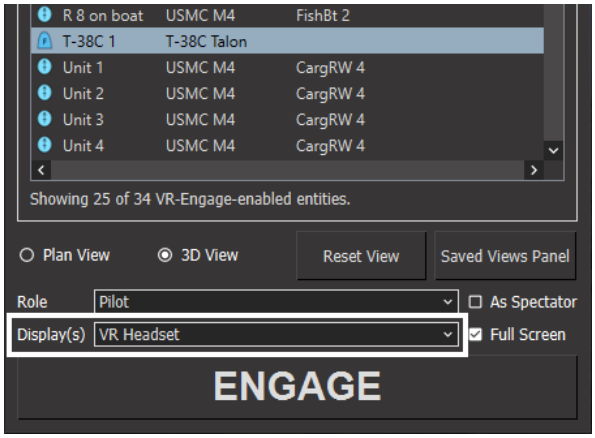
3.2.3 Using an HMD with an Entity

VR-Engage supports use of head-mounted displays to provide a virtual reality or mixed reality experience when you are controlling an entity. VR-Engage includes some entities that are configured for virtual or mixed reality. For entities that are configured for it, you can choose to use a VR or MR headset when one is connected. For details about configuring HMDs, see "3.10. Introduction to Head-Mounted Displays" on page 78.

➤ **To use a head-mounted display to control an entity:**

1. Start VR-Engage.
2. Join a session or load a scenario.
3. Create a new entity or select an existing entity.
4. On the Displays list, select VR Headset (virtual reality) or MR Headset (mixed reality), as appropriate.

Figure 18. Displays list



5. Engage the entity.

3.2.4 Assigning Roles from VR-Forces

If you install the VR-Engage extensions for VR-Forces in a compatible version of VR-Forces, you can assign roles from VR-Forces. This includes:

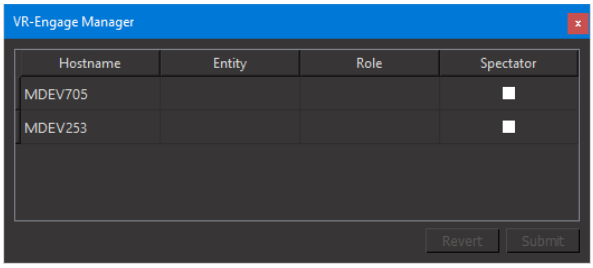
- Assigning an initial role for the player and starting the scenario.
- Specifying that the player be a spectator.
- Changing the role to None, which causes the VR-Engage session to revert to the Plan View role selection GUI.
- Changing the role to a different entity as the scenario progresses.

Managing entity assignments from VR-Forces does not prevent the VR-Engage player from using the VR-Engage GUI to select roles unless their instance of VR-Engage is in Lockdown mode. If the VR-Engage player selects a role, that decision is reflected in the VR-Engage Manager Panel in VR-Forces. To learn about the different modes, see "3.1.3 Admin, User, and Lockdown Modes" on page 59.

➤ **To manage role assignments from VR-Forces:**

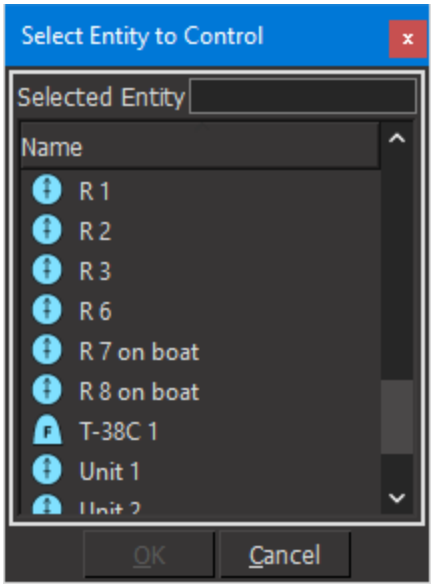
1. Start a version of VR-Forces that is compatible with your version of VR-Engage and has the VR-Engage extensions for VR-Forces installed.
2. Load the scenario you want to run.
3. Choose **View > VR-Engage Manager Panel**. The VR-Engage Manager Panel opens (Figure 19).

Figure 19. VR-Engage Manager Panel



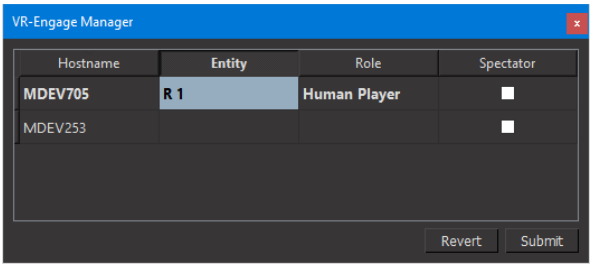
4. Start VR-Engage. It automatically connects to the VR-Forces session. The VR-Engage Manager Panel displays the hostname of the machine running VR-Engage. (If you have multiple instances of VR-Engage running, they are listed on additional rows in the panel.)
5. Click the Entity cell for the VR-Engage host to which you want to assign a role. The Select Entity to Control dialog box opens (Figure 20).

Figure 20. Select Entity to Control dialog box



6. Select the entity that you want the VR-Engage player to control. If the player has a role and you want to set it to no role, select <NONE>.
7. Click OK. The entity is listed in the table cell and the role is listed in the Role cell.

Figure 21. VR-Engage Manager Panel with entity selected



8. If the entity supports multiple roles, such as the LAV, you can select which role you want it to use. To select a role, click the Role cell. It becomes a list.
9. Select the role you want in the list. The panel automatically fills in the Station Name column.
10. If you want the player to participate as a spectator, select the Spectator check box.
11. Click Submit. The assignment is sent to the host. The player is automatically engaged in the assigned role on the host machine.

3.3. Connecting to VR-Forces

VR-Engage can interact with any DIS or HLA simulation. However, it has a particular ability to participate in simulations that use VR-Forces, MAK's computer generated forces application. This allows you to switch between player and CGF control while the scenario is running. You can also use VR-Forces to create scenarios that include VR-Engage controllable entities in addition to CGF simulation objects.

By default, when you start VR-Engage, it looks for a VR-Forces session to join. If it connects to a complementary VR-Forces build that has the appropriate VR-Engage plug-ins installed, you can take control of entities being simulated by VR-Forces. You can assume control of VR-Forces entities and give up control as desired.

When you take control of a VR-Forces entity, the entity pauses the plan or task it is executing. When you give up control of the entity, control returns to VR-Forces and the entity resumes whatever task or plan it was executing before you took control of it.

For a list of entities that can be controlled by VR-Engage players out of the box, see "Chapter 13. Controllable Entities" on page 291.

To learn how to create your own VR-Forces scenarios, see "Chapter 16. Creating VR-Engage Scenarios in VR-Forces" on page 371.

- You can only control entities that are in the VR-Engage SMS or your own SMS that extends that SMS. The VR-Engage SMS includes most human characters, ground vehicles configured to use the Vortex libraries, and aircraft configured to use RotorLib or FixedWingLib.
- For more information on connecting and troubleshooting separate instances of MAK ONE products, please refer to the *MAK ONE Interoperability Guide*.

3.4. Changing Your Role

If a scenario has multiple entities, you can change the role you are playing to be any entity in a scenario. You can change your role by exiting the simulation and choosing a new role. You can also change a role by embarking on a vehicle and choosing an available role.

3.4.1 Changing Your Role in the Choose Role Panel

You can change your role by returning to the Choose Role Panel. When you engage characters or exit a character to the Choose Role Panel, the state of the scenario may change. For details, see "3.5. Entity Engagement and Scenario State" on the facing page.

➤ **To change your role:**

1. Press **Esc**. The Application Menu displays.
2. Click Choose Role. You return to the Choose Role Panel (Figure 16 on page 63).
3. Select the role you want, as described in "3.2. Choosing a Role" on page 63.
4. Click Engage.

3.4.2 Changing Your Role by Embarking on a Vehicle

You can change your role dynamically while you are participating in a simulation. For example, the LAV has several roles associated with it: Driver, Gunner, Commander, Passenger, and Driver/Gunner. Other vehicles can have different sets of roles available. If you embark on a LAV while playing a human character, you can assume any of these roles except for Driver/Gunner. (If you embark on a LAV from the Choose Role panel, you cannot select Passenger.)

➤ **To change your role by embarking on a vehicle:**

1. Walk towards a vehicle that supports embarkation. As you near the vehicle, the Embark menu opens.

Figure 22. Embark LAV menu



> If the Embark menu closes before you make a selection, back up your character a few steps and then approach the vehicle again.

2. Select a role using one of the following methods:
   - Use the **Up** and **Down arrow** keys or the up and down buttons on your D-pad to select a role. Then press the **Right arrow** key or right button on the D-pad to embark using the selected role.
   - Press the number key associated with the embarkation option you want.
   - Click the embarkation option you want. You can only use the mouse when mouse control is disabled.

3.4.3 Switching Roles on a Vehicle

Ground vehicles configured with more than one role allow the player to change roles when they are embarked. Using the Action menu or the square bracket keys (**[** or **]**), they can change to any of the roles (other than Passenger, even when that is one of the available roles).

➤ **To change roles while embarked using the Action menu:**

1. Open the Action menu and then select Switch Role.
2. In the Open Roles Positions submenu, use an Action menu command to select the desired role.

➤ **To change roles while embarked using the keyboard, you can:**

- Press **[** to change to the previous role.
- Press **]** to change to the next role.

3.5. Entity Engagement and Scenario State

The state of a scenario—running or paused—varies depending on whether you are actively engaged in controlling a character and which application started the scenario. The possible cases are:

- If VR-Engage is hosting a session:
  - When a player clicks Engage to control an entity, it puts the scenario in running state. The word "Running" displays briefly at the top of the screen.
  - When a player presses **Esc**, the Application Menu displays. The player can click the pause button () to pause the scenario. The word "Paused" displays at the top of the screen.

- If VR-Engage joined a session hosted by VR-Forces or another instance of VR-Engage:
  - When a player clicks Engage to control an entity, it does not change the scenario's state.
  - When a player presses **Esc** to bring up the Application Menu, it does not change the scenario's state.
- If VR-Engage started an unhosted session:
  - When a player clicks Engage to control an entity, it does not change the scenario's state. The player must start the scenario.
  - When a player presses **Esc** to bring up the Application Menu, it does not change the scenario's state.

When the scenario is not running, the word "Paused" displays at the top of the screen. Players can only act while the scenario is running, not while it is paused.

If a player uses VR-Engage to join a session hosted by VR-Forces or another instance of VR-Engage and the user on the hosting application closes the scenario, a message informs the player that the session has ended and returns the player to the Searching for Session screen.

3.6. The Application Menu

When you are engaged in a role and press **Esc**, you see the Application Menu. This screen contains a number of options.

Figure 23. Application Menu



- 1 – Click Return to Role to exit the Application Menu and return to the role you were playing.
- 2 – Click Choose Role to return to the Choose Role panel.

- 3 – Click Choose Session to leave the session and return to the Choose Session screen.
- 4 – Click Exit VR-Engage to close the application.
- 5 – The current status of the scenario, either Running or Paused.
- 6 – Play the scenario.
- 7 – Pause the scenario.
- 8 – Rewind the scenario.

> The Play, Pause, and Rewind buttons are available only when you are hosting the session.

- 9 – Click Back to exit a submenu, such as the Connection Menu. If you are not in a submenu, clicking Back exits the Application Menu and returns you to the role you were playing.
- 10 – Display your connection information.
- 11 – View the control mappings.
- 12 – Access the About dialog box. If you are in Admin mode, you can also access some advanced settings.
- 13 – Close VR-Engage.
- 14 – The Status is information about your character and the scenario.

3.7. Starting a New Session

If the instance of VR-Engage that you are using is hosting the session, you can exit the session you are participating in and join or start a new session.

➤ **To change the scenario:**

1. Press **Esc**. The Application Menu displays.
2. Click Choose Session.
3. Click Yes in the message to confirm that you want to leave the session. The Searching for Session screen displays.
4. Follow the procedure for starting a session, as described in "3.1. Starting VR-Engage and Joining a Session" on page 56.

3.8. Configuring the Simulation Connection

To participate in a simulation, you must connect VR-Engage to a simulation network. VR-Engage includes connectors for DIS, HLA Evolved (IEEE 1516-2010, also referred to as HLA 1516e), and HLA 4 (IEEE 1516-2025).

In User and Admin modes, you can change the configuration values of the default connectors and you can add connectors. You connect VR-Engage to a network at startup. To change the connection, click the Connection button at the top right.

For complete details about connection settings and parameters, see the *VR-Vantage Users Guide*.

➤ **To set up the simulation connection at startup:**

1. Start VR-Engage.
2. In the top left of the Connection Menu, select a connection.
3. At the bottom right of the Connection Menu, review the connection settings.

Figure 24. Connection Menu



4. Change the connection parameters as needed. If necessary, click the down arrows to access the advanced settings. Those connection settings are described in Table 2 below and Table 3 on page 76.
5. If you want to use those connection parameters in future sessions, click Save.
6. Click Connect to connect to the network.

3.8.1 HLA Connection Settings

Table 2 describes the connection settings for HLA 1516e and HLA 4.

Table 2. HLA connection settings

Parameter	Description
Federation Name	Specifies the federation execution name.
FED File Name	Specifies the FED file. Default: the federation name with *.xml* appended.
Federate Type	Federate Type can be used to distinguish different categories of federates, and its only practical use is in save-and-restore.

Table 2. HLA connection settings (continued)

Parameter	Description
Advanced	
Federate Name	Specifies a unique name that can be assigned to the federate. No two federates within the same federation can have the same name. If no name is specified, the RTI will automatically assign a unique name for the federate.
Local Settings Designator	Specifies an optional RTI-specific string used for configuration.
Site ID	Specifies the site number for the observer. The Site ID and Application Number uniquely identify the observer.
Application Number	Specifies the application number for the observer.
Use RPR FOM	Specifies use of a version of the RPR FOM.
RPR FOM Version	Specifies the RPR FOM version. VR-Engage has built-in support for RPR FOM version2.0. You must specify the correct FED file for the version you use. If you use a different FOM, you must specify the correct FED file and FOM Mapper.
Use Custom FOM	Specifies use of a FOM other than the RPR FOM.
FOM Mapper Library	Specifies the name of a FOM Mapper library.
Initialization String	Specifies initialization data for the specified FOM Mapper, if required. It is your responsibility to know if your FOM Mapper needs initialization data.
FOM Modules	Lets you specify FOM Modules. Consult your RTI documentation for information about FOM Modules.
Publish Observers	Specifies whether to publish an entity for the observer that is using this connection.
Override Global View Control Setting	Apply the setting in this connection instead of the setting in the GUI. If this is used, the GUI setting is ignored.
Listen for View Controls	Specifies whether to respond to view control messages sent by other applications.
Ignore Advisories	Enables or disables use of HLA advisory messages.
Implicit Geographic DDM	Enables or disables use of implicit DDM. Valid only with the MAK RTI.

Table 2. HLA connection settings (continued)

Parameter	Description
Use Absolute Time Stamps	Enables or disables use of absolute timestamps.
Use Topo for Flat Earth Vector Conversion	Enables or disables the use of the pre-2.0 conversion method for Flat Earth Vectors.

3.8.2 DIS Connection Settings

Table 3 describes the connections settings for DIS connections.

Table 3. DIS connection settings

Parameter	Description
Port	The DIS port to connect to.
Exercise ID	Specifies the exercise ID for VR-Engage.
Site ID	Specifies the site number for VR-Engage.
Application Number	VR-Engage can send PDUs to the network to communicate the observer's position and orientation. By default, these PDUs are sent to the host machine's broadcast address. You may want to direct VR-Engage data to a unicast address so it is received by only one application.
DIS Version	Specifies the DIS version: 7.
Device Address	On computers with multiple network cards, the address of the network card to use.
Advanced	
Destination Address	Specifies the destination address for outgoing PDUs. Default: the computer's broadcast address.
Subnet Mask	Specifies a subnet mask. VR-Engage can usually figure out the subnet mask from the IP address. However, if it does not do this correctly, you can explicitly specify the subnet mask using this command-line option.
Send Buffer Size	Increasing the send and receive buffer size can reduce dropped packets. If the parameters are set to 0, the system default setting is used.
Receive Buffer Size	Increasing the send and receive buffer size can reduce dropped packets. If the parameters are set to 0, the system default setting is used.
Multicast TTL	Specifies the number of routers that a message can pass through when PDUs are sent to multicast addresses.

Table 3. DIS connection settings (continued)

Parameter	Description
Enable Timeout Processing	Enables or disables applying a timeout value to discovered objects.
Timeout Interval	The number of seconds to wait without receiving an updated before timing out an object.
Use Asynchronous IO	Enables or disables asynchronous IO. Use of asynchronous IO can reduce dropped packets.
Use Absolute Time Stamps	Enables or disables use of absolute timestamps.
Show Self Published Objects	Broadcast the location of objects that VR-Engage creates - primarily the observer.
Publish Observers	Specifies whether to publish an entity for the observer that is using this connection.
Override Global View Control Setting	Apply the setting in this connection to instead of the setting in the GUI. If this is used, the GUI setting is ignored.
Listen for View Controls	Specifies whether to respond to view control messages sent by other applications.
Use Topo for Flat Earth Vector Conversion	Enables or disables the use of the pre-2.0 conversion method for Flat Earth Vectors.

3.9. Playing in Windowed Mode

By default, when you click Engage to control a character, you go into full-screen mode. There are no menus or other window controls available. If you want access to menus (Admin mode only) or the title bar without escaping to the Choose Role Panel, you can exit full-screen mode and still control your character. This is called windowed mode.

Whenever you go from full-screen to windowed mode, VR-Engage releases mouse control so that you can use the mouse to access menus or other applications. You can also toggle mouse control on and off. (For more information about mouse control, see "6.1.1 Enabling and Disabling Mouse Control" on page 156.)



To access menus, you must be in Admin mode. See "3.1.3 Admin, User, and Lockdown Modes" on page 59.

➤ To change from full-screen mode to windowed mode, press **Ctrl**+**Enter**. To return to full-screen mode, press **Ctrl**+**Enter** again.

3.10. Introduction to Head-Mounted Displays

VR-Engage is integrated with several virtual reality head-mounted displays (HMDs). VR-Engage can render stereo views, which are presented inside the HMD. The head position and orientation tracking of the HMD are reflected in the observer as the wearer of the HMD moves their head.

VR-Engage uses the OpenVR, Oculus SDK, Varjo SDK, and VRgineers libraries to integrate with virtual reality systems. Any system that is capable of using OpenVR should work with VR-Engage.

VR-Engage has been tested with:

- Oculus Rift and Rift S
- HTC Vive and HTC Vive Pro
- Valve Index
- Varjo XR-1, XR-3, XR Elite, VR-2, and VR-3
- VRgineers XTAL 2 and XTAL 3
- Meta Quest 3

3.10.1 Prerequisites for Using an HMD

To use an HMD, you need specific software and VR-Engage setup:

- SteamVR. Required for the Vive and other small-type OpenVR headsets. It is installed through the Steam digital distribution platform (https://store.steampowered.com).
- The Oculus Rift requires that you install the Oculus Runtime.
- For a Varjo headset, you must install Varjo Base. The Varjo XR-range devices support mixed reality. The XR-1 requires that you also install Thunderbolt 3. XR-3 and later require only USB 3.

  For information on Varjo system requirements, see https://support.varjo.com.
- XTAL headsets require the VRgineers software.
- If you are using Microsoft Mixed Reality devices, the "Windows Mixed Reality for SteamVR" add-on must be installed through Steam.

3.10.2 Resetting the Tracking Sensor

For user comfort, you can reset the origin of the positional tracking performed by the HMD. This allows you to restart the head tracking once you have put on the HMD and assumed a comfortable position. You can reset the sensor at any time by pressing the **F8** key.

You can also reset the tracking position:

- Choose **Reset VR Head Tracking** on the Action menu.
- Press the right thumbstick on a game controller.
- Press button 5 on the HOTAS Warthog joystick.

You can change the keyboard assignment by editing the *./data/simulationModelSets/myVR-EngageSMS/input/common/keyboard.lua* file:

```
DeviceType="keyboard";
MappingGroups = {
  {
    Group="entity-control";
    Mappings={
      ...
      {type="key"; key="<F8>"; action="reset-vr-head-position"};
```

3.10.3 Calibrating the Eye Tracking for Varjo

Varjo devices allow applications that use eye tracking to start multi-dot calibration for more accurate eye tracking data. By default, it calibrates each time a user dons the headset. You can access the Varjo's eye tracking calibration by pressing the Menu button on the headset or using Varjo Base.

3.10.4 VR Anchoring for Mask Tracking

You can use an anchor when configuring your VR display. This allows you to position a mesh cutout, such as for a cockpit display. You use the tracker to reposition the entire view of the virtual world to match it to the real world. This was designed to work with the Vive Tracker and a Varjo headset.

The calibration is calculated relative to the Vive Tracker. As long as the tracker remains in the same location relative to your masked area, the mask remains in the correct position.

➤ **To enable and calibrate VR anchoring:**

1. Open SteamVR and pair a Vive Tracker.
2. Launch VR-Vantage with Virtual Reality.
3. Press the **F4** key to open the Developer Debug Overlay and then expand VR Anchoring.
4. Select the Enable check box. Other options become available.
5. Select the Calibrate check box. The virtual world no longer moves when as you position the headset, allowing you to align the mask with the real world.
6. When you successfully align the mask, clear the Calibrate check box.
7. Click Save. The calibration is saved to *./data/VR/anchorConfig.json*. The configuration loads the next time you start VR-Engage.

3.10.5 Enabling and Disabling Virtual Reality Rendering

By default, when you start VR-Engage with an HMD connected, rendering to the HMD is enabled at startup. If needed, you can enable or disable rendering to the HMD at runtime. You must be in Admin mode to manage the virtual reality settings. If you cannot see the menu bar, press **Shift+F12** to enter Admin mode.

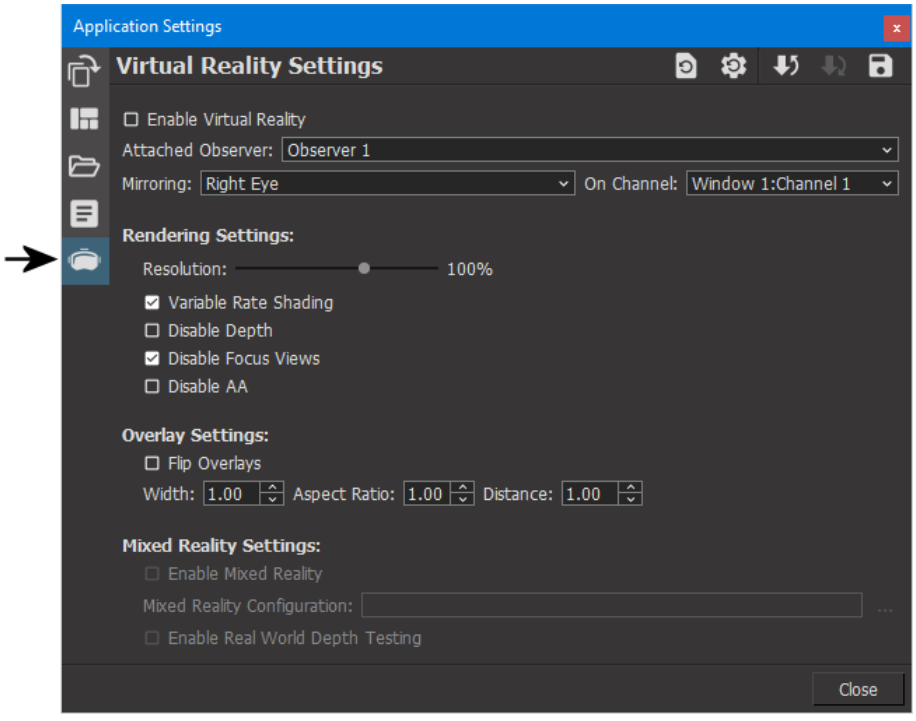
> ⚠ To use an HMD, you must connect and power on the device before starting VR-Engage. Failure to do so can prevent the HMD from working properly with a VR-Engage player role.

➤ **To enable or disable virtual reality rendering:**

1. Choose **Settings > Application**. The Application Settings dialog box opens.
2. Select the Virtual Reality Settings page (Figure 25).

> ⓘ The Virtual Reality Settings page is available only when the virtual reality plug-in is loaded. The default settings for VR-Engage load this plug-in.

Figure 25. Virtual Reality Settings page



3. Select or clear the Enable Virtual Reality check box.

> You can also enable or disable virtual reality by choosing **Settings > Enable Virtual Reality**.

3.10.6 Configuration Options for Virtual and Mixed Reality

You can configure these options for virtual and mixed reality:

- Attached Observer. This is the observer that the HMD will attach to and render the perspective from. The default is to attach to Observer 1.
- Mirroring. This copies the image rendered for a given eye to the selected channel in the main display. The default is to render the right eye to the main channel. Mirroring disables the normal render pass of the attached channel to help with performance.

  > ⓘ If mirroring is enabled and the HMD is attached to the observer that is driving a displayed channel, then the main view will be rendered from the HMD perspective. This affects performance, because the scene will be rendered a third time.

- Rendering Settings.
  - Rendering Resolution. The slider bar allows you to dynamically adjust the resolution. Overlays remain at their native resolution.
  - Variable Rate Shading. Improves performance in VR by lowering the precision and accuracy of GPU fragment operations in certain areas of the frame.
  - Disable Depth. Some headsets support depth-based warping, which is a way to make the image appear smooth with lower frame rates. Currently, this applies to Varjo headsets and the Oculus Rift. When you select Disable Depth, you disable the depth-based warp (VR-Engage does not pass the depth buffer to the headset SDKs). If the depth warp is causing cockpit windows to display artifacts and otherwise look incorrect, selecting Disable Depth can help.
  - Disable Focus Views. This applies only to Varjo devices and is selected by default. Select it to force the Varjo provider to render only the context views. The result is a single multi-view channel, which improves performance, instead of the default two channels.

    If you use a Varjo headset older than the XR-4, you should also enable Simple Rendering in Varjo Base. It is in the Headset tab of Varjo Base. XR-4 and later use only one channel per eye.
  - Disable AA. This disables multisampling, which improves performance by saving on rendering time. Note that it does so by disabling alpha to coverage and therefore can cause transparency issues and visual artifacts.
- Overlay Settings.
  - Flip Overlays. Currently this applies only to the HTC Vive. Enabling this option flips the overlays vertically. Select this check box when overlays display upside down.

- Mixed Reality Settings.
  - Enable Mixed Reality, MR Configuration, and Enable Real World Depth Testing. These options do not apply to VR-Engage.

➤ **To set virtual reality configuration settings:**

1. Choose **Settings > Application**. The Application Settings dialog box opens. (If the menu bar is not visible, press **Shift+F12** to enter Admin mode.)
2. Select the Virtual Reality Settings page (Figure 25 on page 80).
3. Change the settings as desired.

3.10.7 Setting Up Mixed Reality

VR-Engage supports the use of the forward-facing cameras on the Vive, Vive Pro, Varjo XR range, and XTAL headsets to add real elements such as a joystick, throttle, or other physical controls to the virtual world. Mixed reality modes are available for human roles and aircraft pilot roles for aircraft configured for VR. When you select a compatible entity, MR Headset is an option in the Display(s) list. Select this option to use mixed reality mode.

To use Mixed Reality mode in VR-Engage, you need a Vive, Vive Pro, Varjo XR, or XTAL, and the appropriate software (described in "3.10.1 Prerequisites for Using an HMD" on page 78). If you are using a Vive HMD, you also need a green screen and you must start SteamVR and turn on the HMD before you start VR-Engage. We strongly recommend that you test the Vive HMD camera before each use. For details, see "Testing the Vive Camera" on the facing page.

For the Vive and Vive Pro, green screen masks out real-world elements that should not appear in the simulation. For the VR-Engage pilot role with mixed reality, the green screen is expected to cover an area approximately 120 degrees wide at -90 degrees elevation extending up to 60 degrees wide at 0 elevation, from the player's point of view. If seated at a table or desk, the desk should be covered with a green screen approximately five feet wide and extending up to at least seated eye level. The only controls that should be visible on top of the green screen are those that you want to see in the mixed reality view. The green screen should be well-illuminated with soft, diffused light so that the green color is clear and uniform.

Mixed Reality Masks

The MR_VR_Flight_Demo scenario contains examples of both VR and MR configurations in a T-38 fixed wing aircraft. VR-Engage allows you to specify the mask file used for a particular display layout. In this example, it's a mask of the lower part of the T-38's cockpit.

In the *./data/simulationModelSets/VR-Engage/roles/t38Pilot.lua* file, there is an MR Headset displayLayout. This demonstrates how to make a Mixed Reality displayLayout. To use a different mask for your own layout, refer to the "cameraProjectionGeometry" field. This is where the mixed reality mask definition file for the displayLayout is specified. For more information, see "Configuration for Mixed Reality" in the *VR-Vantage User Guide*.

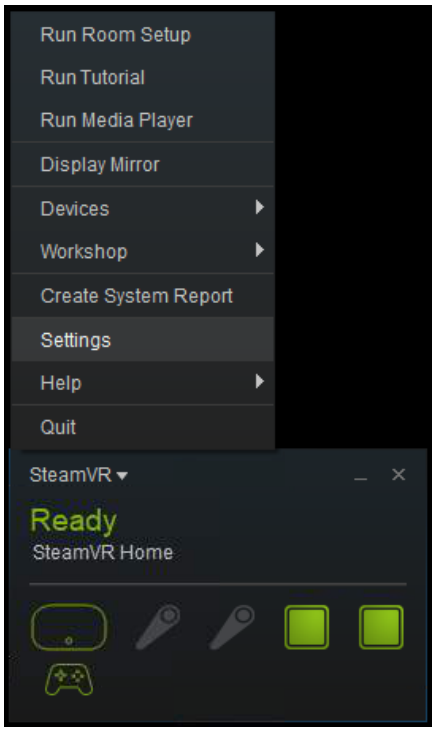
Testing the Vive Camera

If you are using the Vive HMD, you must turn it on before you start VR-Engage. We also recommend that you test the camera every time SteamVR or the computer is restarted, and prior to running VR-Engage. In some cases, the camera does not work unless you initialize it before starting VR-Engage.

➤ **To test the Vive camera:**

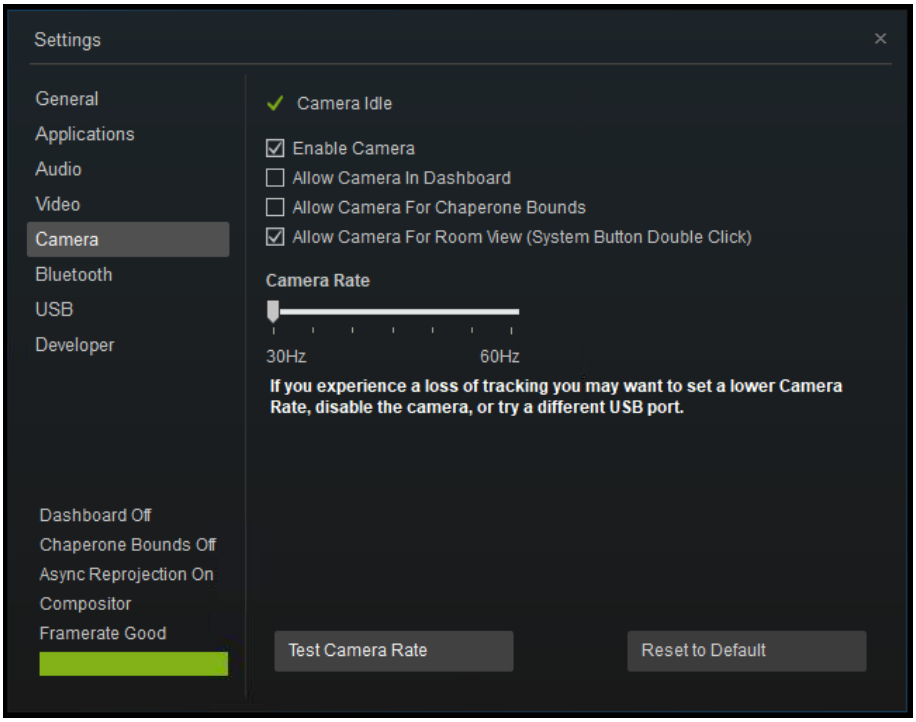
1. Turn on the Vive HMD.
2. Start SteamVR.
3. On the SteamVR menu, choose **Settings** (Figure 26). The Settings dialog box opens.

Figure 26. SteamVR menu



4. Select the Camera page (Figure 27).

Figure 27. Camera page



5. Click Test Camera Rate.
6. Confirm that the camera is working. (Wait until three green check marks appear.) If a red x appears, or if the Test Camera Rate button is not available, verify that the Vive is connected to a USB-3-capable port, and reboot the Vive headset and/or the computer.
7. If the test fails, restart the Vive HMD and SteamVR.

3.10.8 Optimizing Performance with HMDs

Virtual reality headsets require a high frame rate for a comfortable viewing experience. There are many factors that determine frame rate, such as complexity of the scene being rendered, terrain, special effects such as HDR and shadows, and environmental effects such as clouds. You may need to make trade-offs when selecting a terrain database, entity models, and applying certain visual effects at runtime.

For convenience, we have included an overlay installer that makes a number of these trade-offs to optimize performance in VR in the context of air entities. The VR-Engage installation includes an installer named *MR_VR_FlightSettingsOverlay.exe*, located at the top level folder. Please see the associated *README_MR_VR_FlightSettingsOverlay.txt* for details on the settings it modifies.

> To improve performance, VR-Engage disables ambient occlusion (HBAO+) by default when you use a virtual reality device (HMD).

3.11. Creating a Startup Configuration

You can create a startup configuration to automate some aspects of VR-Engage startup. VR-Engage includes a fully commented example startup configuration script called *exampleStartupConfig.lua* in *./appData/scripts*.

You can specify the following startup parameters in a script:

- **mode**. Tells VR-Engage to only `join` a running session, immediately `host` a session, or immediately `start` an unhosted session.
- **scenario**. Optional. If the **mode** is `host`, specifies the scenario to host. If you do not specify a scenario, VR-Engage uses the value for **terrain** or prompts you to specify a terrain.
- **terrain**. Specifies the terrain to load in `start` mode, or in `host` mode if you do not specify a scenario.
- **guiTerrain**. Optional. Specifies the terrain to use for visualization if you want to visualize on a terrain that is different from the simulation terrain.
- **attach**. Specifies an entity to control or attach to as a spectator.
- **entity**. Optional. If **attach** is not specified, this parameter specifies an entity to create.
- **minion**. Causes this instance of VR-Engage to connect to the entity being controlled by another instance of VR-Engage.

To use a startup configuration script, create a shortcut or batch file with the command:

```
vrEngage.exe --initSettingsFile startupConfigFile.lua
```

where:

- vrEngage.exe is the VR-Engage executable.
- *`startupConfigFile`* is the name of your startup configuration file, including the *.lua* extension. If you use a simple file name, VR-Engage looks for the file in *./appData/scripts*. You can also specify a relative path from the *./bin64* directory or a fully qualified path.

3.11.1 Minion and Boss Configuration

The **minion** startup configuration configures one station to watch the entity engagement status of another station, and automatically engage with that same entity. You would use it in conjunction with other startup options to create a multi-head station, where you have more than one instance of VR-Engage, on one or more machines, to create a full player interface.

The most common use case is for creating aircraft stations, although it applies to other entity types. You would have one or more VR-Engage stations running an out-the-window aircraft HUD display, and another one running the instrument panels.

On a single machine, a single instance of VR-Engage is able to create a separate instrument panel window, so this can be done with only one VR-Engage instance running. For performance reasons, however, you might want to use multiple machines. In that case, you

need one instance of VR-Engage on one machine to run the out-of-window view, and a second instance of VR-Engage on another machine to run the instrument panel window. One of these is the Boss, which takes player input and controls the entity, and the other is the Minion running as Spectator. The function of Minion mode then is to make these two stations behave as one when switching which entity is engaged.

> The terms Master and Slave are technically a different context, which is why we use the terms Boss and Minion to differentiate.

3.12. Starting Multiple Simulation Engines

VR-Forces scenarios can be designed to use multiple simulation engines to improve performance, although this is no longer recommended. The preferred method for optimizing the performance of scenarios is to manage multiprocessors usage as described in "Optimizing Multiprocessor Usage for Simulation Engine" in the *VR-Forces User Guide*.

When using multiple simulation engines, each sim engine simulates a subset of the simulation objects in the scenario. This is called load balancing. If a scenario is designed to use multiple simulation engines, VR-Engage can automatically start up the required number of simulation engines.

By default, use of multiple simulation engines is disabled and the maximum number of simulation engines to use is set to 1. To enable use of multiple simulation engines, you can either:

- Start VR-Engage with the `--enableMultipleBackends` command-line argument.
- In *./appData/scripts/playerStation.lua*, uncomment the `enableMultipleBackends` parameter and set it to true.

By default, the maximum number of simulation engines that VR-Engage can start is `number of logical processors/2`. The minimum is always 1. You can change the maximum number of simulation engines to start by uncommenting the `MaxBackends` parameter in *./appData/scripts/playerStation.lua* and setting it to your desired value.

If you enable multiple simulation engines but do not have enough licenses for the number of simulation engines that a scenario is configured to use, VR-Engage simulates all entities on one simulation engine.

> If you start a scenario in VR-Forces and do not have the configured number of simulation engines running, VR-Forces prompts you to remap the entities to the existing simulation engines. For information about load balancing and remapping scenarios, see the *VR-Forces User Guide*.
>
> You must have a VR-Forces license for each simulation engine you start.

3.13. Command-Line Options

3.13.1 Command-Line Options for vrEngage

Table 4 lists VR-Engage's command-line options.

VR-Engage also allows you to use startup and player station configuration files to determine some options when the program launches. See "3.11. Creating a Startup Configuration" on page 85 and "3.13.3 Command-Line Options for Player Stations" on page 112.

Table 4. VR-EngagevrEngage GUI command-line options

Option	Description
`--airMode`	Launches VR-Engage in Air mode. You can create and control only aircraft and humans.
`--antiAliasing` *`level`*	Set the anti-aliasing level (0, 2, 4, 8, or 16). Default: 4. Depending on your graphics card and driver, some combinations of anti-aliasing and bit settings (`--depthBits`, `--stencilBits`, and `--alphaBits`) may be incompatible. For example, if you set stencil bits to 8, the depth must be set to 24 and anti-aliasing may need to be set to 0 or 2.
`--appCacheDir` *`directory`*	Use `--appCacheDir` to specify a directory for the cache. This allows you to run multiple remote display engines (RDEs) or other instances of VR-Engage from the same installation. Running two RDEs on the same machine helps compensate for the limited threading in VR-Engage.
`--appDataDir` *`directory`*	Specifies the location of application data. If not specified, VR-Engage uses the default: *./appData*.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
`--appMode [user|admin|lockdown]`	Controls the features that are available to the user. • User mode does not display the menus when you start VR-Engage. • Admin mode shows the menus when you start VR-Engage. • Lockdown mode disables the ability to toggle to Admin mode. User mode is the default behavior. In VR-Engage, you can toggle between the User and Admin modes by pressing Shift+F12.
`(--autoConnect | -C) {`*`filepath | connector_name`*`}`	Automatically connect to the network using the specified connector. Can be used multiple times.
`--autoReconnectPeriod` *`period`*	Specifies the amount of time, in seconds, to wait before trying to attach to disconnected remote displays.
`(--bindObservers | -B)` *`"observer1,...,observern,../saved_views_file" "../terrain_file"`*	If you load a scene that has multiple observers, this command assigns the views in the specified saved views file to the specified observers. The first saved view is assigned to the first observer, the second to the second observer, and so on. If the observers are not specified as arguments, only the first observer in the display engine configuration is assigned a view. The entire string must be quoted. There are no spaces after commas. You must specify a terrain database for this command to work. You would ordinarily also specify a display configuration that contains the observers with `--dispSetting`. For more information about using this command-line option, see "Attaching the Observer at Startup" in the VR-Vantage User Guide, the attaching the observer to an entity at startup tutorial.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
`--customSettingsFile` *filename*	Name of the your settings file that you want to load to change startup configuration settings defined in the original *playerStation.lua* file. For more information, refer to that file. You can also override individual settings in that file using the `--setting` command-line option, described below (page 98).
`--dataDir` *directory*	Specifies the location of the application's data directory. If not specified, VR-Engage uses the default, *./data*.
`--depthBits` *bits*	Sets the depth of graphics to 24 bits or 32 bits. The default is 24, This has no effect when reverse Z is enabled, which is the default for VR-Engage. The number of depth bits in reverse Z is always 32 bits float.
`--dis`	Start in DIS mode, using DIS command-line arguments to specify connection parameters. DIS-specific arguments do not have to follow `--dis`; they can be in any order.
`--disableBufferClearUsesCompute`	Do not use this option unless directed to do so by MAK technical support.
`--disableDynamicTerrain`	Disables the dynamic terrain engine. May slightly speed up terrain loading and paging.
`--disableErrorDialog`	Prevents error messages from displaying. Errors print to the console instead.
`--disableEscapeKey`	Disable use of the **Esc** key while the player is engaged in a role.
`--disableGlBufferTracking`	Disable GL buffer tracking, which can improve performance in cases such as CDB terrains.
`--disableIndirectEntities`	Disables indirect rendering of entities. This is primarily for debugging and fixing any regression with the old path. It may or may not match the behavior with the default path (which is indirect).

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
`--disableIndirectExternalRenderer`	Disables the indirect external renderer, which minimizes state changes by default. When this is used, state changes are not minimized.
`(--disableKDTrees | -K)`	Disable creation of KD trees for intersections.
`--disableModelSets "`*`model-set-number`*`"`	Disables one or more model sets. Model set numbers are: • `0` - 3D Model Set • `1` - Colorized Models Model Set • `4` - Minimap • `5` - 2D Icons Model Set To disable multiple model sets, separate them with commas. For example: `--disableModelSets "0, 1"` disables the 3D and XR model sets `--disableModelSets "0, 1, 5"` disables all model sets
`(--disableReverseZBuffer | -z)`	Disables use of reverse Z buffering, which is enabled by default. Leaving reverse Z buffering enabled can reduce Z-fighting when viewing the terrain from a distance. (3D observer modes only.)
`--disableTerrainObserverViews`	Disables the loading of terrain observer view files. Loading a terrain does not clear the observer views and the terrain's observer views do not load.
`--discoveredModelPreloadFileName` *`filename`*	Specifies a file to which the list of models discovered during a session that were not preloaded is written.
`--displayEnginePort` *`port`*	Specifies the display engine port used by a remote display engine. Used only if `--slave` is specified.
`--enableAdminMode`	When VR-Engage is in Lockdown mode, Admin mode is disabled by default. This option enables the Admin mode hot keys, allowing you to access that mode.
`--enableDofConstraints`	Enables the use of degree(s) of freedom constraints on simulation object models.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
`--enablePersistentShadowTechnique`	Use this command-line option to enable persistent shadows. By default, shadows are disabled: the shadow camera traversal is skipped but the shaders remain the same, preventing additional shader sets in SpeedTree and instant switching. If the VP for the shadow pass is always installed, SpeedTree (and other drawables) do not regenerate VP shaders/user shaders.
`--enableSensorFX`	Load the SensorFX plug-in. SensorFX uses more advanced rendering to display infrared and night vision.
`--enableTracyProfiling`	Enables the Tracy Client when you launch the application.
`--enableVSync`	Turns on VSync.
`--entDispSetting` *filename*	Specifies the entity display setting to load.
`--entityDefsDir` *directory*	Loads the *.hier* and *.leaf* files in the specified directory to the entity element definition manager.

`( --entityGridCellSize` *x y z* `|`
`--entityGridCellSize` *x,y,z* `|`
`--entityGridCellSize` *integer*`)`
	Use this option for debugging, particularly looking for bottlenecks in the code. Entities (in indirect) are divided into a grid of cells. The cells have the same size. This is a regular (sparse) grid that allows for efficient, higher-level culling. The default cell size is 5000 5000 5000. You can specify three values or one, for example:
	• 2000 2000 2000
	• 2000
	• 2000 200 10000
	• -1 -1 -1 (one cell, infinite cell size)
	• -1 (one cell, infinite cell size)

`(--entTypeMap | -E)` *filename*	Load the specified object type mapping file. The filename extension determines the type of mapping.
`--envSetting` *filename*	Specifies the environment configuration file to load.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
`--exConnConfigFile` *filename*	Specifies the exercise connection configuration file to use. Use this to override settings in the VR-Link exercise connection configuration file, *./appData/settings/connections/MAK-ONE-YYYY-Config.xml*, which defines the default connections settings. For more information, refer to the *MAK ONE Interoperability Guide*.
`( --extraArgs | -e)` *string* | "*string1 ... stringn*"	Allows you to send arbitrary strings to the initializer, which can be queried for by plug-ins. This option works around the restriction on adding new command-line arguments. It can take one argument or multiple arguments. Multiple arguments must be wrapped in double quotes.
`--factoryRootDir` *directory*	Set the root directory used to restore settings to factory defaults.
`--forceTextShaping`	Force all text to use HarfBuzz shaping. Use this option if you need support for right-to-left languages. It may affect performance.
`(--frameRate | -F)` *rate*	If rate is 0, the frame rate is variable. Otherwise, specifies a fixed frame rate for rendering.
`--fullScreen`	Start with the window in full screen mode.
`--groundMode`	Run VR-Engage in Ground mode. You can create and control only ground vehicles and humans.
`--guiWindowRectangle "`*width height x y*`"` `--guiWindowRectangle` *width*`,`*height*`,`*x*`,`*y*	Specifies the window size and location at startup.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
`(--hasNTPSync | -O)`	Enables the synchronization of all time-based features between different VR-Vantage applications, such as instances of VR-Vantage or the VR-Forces GUI. Note that system clocks must be synchronized. Time-based features include the rendering of light point animations, continuous flipbook animations, ocean waves, clouds, and SpeedTrees.
`--highestPriority`	Changes the priority of the application and the main render thread to the highest available priority. This command may starve other processes.
`--hla1516e`	Start in HLA Evolved mode, using HLA command-line arguments to specify connection parameters. HLA-specific arguments do not have to follow `--hla1516e`; they can be in any order.
`--hla4`	Start in HLA 4 mode, using HLA command-line arguments to specify connection parameters. HLA-specific arguments do not have to follow `--hla4`; they can be in any order.
`(--ignoreZoomForSpeedtreeLod | -Z)`	Disable zoom for SpeedTree LOD.
Use the following texture limiting command-line options to limit and debug the memory usage and then observe the total GPU memory used.	
`--limitEarthSplattingTextures "`*`width height`*`"`	Restricts the size of an earth file's splatting textures to the specified resolution. The width and height must be a power of 2, for example, `"16"`. If you do not enter a power of 2, VR-Engage converts it to the next highest power of 2.
`--limitEarthTerrainTextures "`*`width height`*`"`	Restricts the size of an earth file's terrain textures to the specified resolution. The width and height must be a power of 2, for example, `"16"`. If you do not enter a power of 2, VR-Engage converts it to the next highest power of 2.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
`--limitEarthVegetationTextures "width height"`	Restricts the size of an earth file's vegetation textures to the specified resolution. The width and height must be a power of 2, for example, "16". If you do not enter a power of 2, VR-Engage converts it to the next highest power of 2.
`--limitEntityTextures "width height"`	Uses only mipmap levels at or below the resolution for entities. Use two integers for the maximum width and height of the texture mipmap to use, for example, "1024 1024" or "2k 2k".To turn off the texture limit function, use `--limitEntityTextures "-1 -1"`.
`--limitEntityTexturesExclusions "texture"`	Excludes textures from texture reduction limits for indirect. Specify the texture(s) that you want to exclude, for example, `--limitEntityTexturesExclusions "SurfaceCVN-75HarrySTruman, SurfaceCruiseShip, FixedWingF-16, FixedWingRQ-1Predator"`.
`--limitTerrainTextures "width height"`	Uses only mipmap levels at or below the resolution for terrain. Use two integers for the width and height, for example, "1024 1024" or "2k 2k".
ⓘ In most situations, using `--limitEntityTextures` and `--limitTerrainTextures` prevents crashes with high VRAM. If you continue to experience crashes, use `--limitGpuMemoryPercentage` in addition to these command-line options.	
`--limitGpuMemoryPercentage integer`	Set the cutoff percentage as a value between 0 and 100. (Typical values are between 85 and 100.) Beyond the specified percentage, VR-Engage does not make new textures or buffers as a percentage of the available video RAM. This applies only to VR-Engage-side textures: terrain insets, entities, indirected osgEarth layers (feature_ geom and buildings). It does not apply to SilverLining, Triton, or osgEarth tiles. The latter are controlled by the *.earth* file options.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
Typically, you would use `--limitGpuMemoryPercentage` in conjunction with `--limitEntityTextures` and `--limitTerrainTextures` when you see crashes with high VRAM.	
(`--loadObserverViews` | `-L`) *filename*	Loads the saved observer view states in the specified file. You can use this command multiple times, for example: `--loadObserverViews file1 --loadObserverViews file2`
(`--locale` | `-G`) *language*	Specifies the locale (for localization).
(`--logFileName` | `-l`) *filename*	Specifies the log file name.
`--mainScreenNum` *num*	Specifies the display that the main window should use. The screen number can be a number from 0 to the number of screens -1. This number matches the screen number used in the Display Engine Configuration Editor for a window.
`--mergeEntityTypeMap` {*directory* | *filename*}	Merges all entity type mapping files (**.metx*) in a directory or individual entity type mapping files. Can be used multiple times. See note below.
`--mergeHierarchyEntityDef` {*directory* | *filename*}	Merges the *.hier* files in the specified directory or the individual *.hier* file into the default hierarchy (usually loaded from *./SharedData/19/latest/ModelData/Configuration/Original/Definitions*). Merging is additive only. No nodes in a previously loaded hierarchy are replaced. Can be used multiple times.
`--mergeHierarchyTacticalGraphicsDef` {*directory* | *filename*}	Merges the specified *.hier* file(s) into the default hierarchy (usually loaded from *./SharedData/19/latest/ModelData/Configuration/Original/Definitions*). Merging is additive only. No nodes in a previously loaded hierarchy are replaced. Can be used multiple times.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
`--mergeHierarchyUnitDef {`*`directory`* `|` *`filename`*`}`	Merges the specified *.hier* file(s) into the default hierarchy (usually loaded from *./SharedData/19/latest/ModelData/Configuration/Original/Definitions*). Merging is additive only. No nodes in a previously loaded hierarchy are replaced. Can be used multiple times.
`--mergeLeafEntityDef {`*`directory`* `|` *`filename`*`}`	Merges the leaf files in the specified directory or the individual leaf file into the default entity element definitions. Can be used multiple times.
`--mergeLeafTacticalGraphicsDef {`*`directory`* `|` *`filename`*`}`	Merges the leaf files in the specified directory or the individual leaf file into the default tactical graphics element definitions. Can be used multiple times.
`--mergeLeafUnitDef {`*`directory`* `|` *`filename`*`}`	Merges the leaf files in the directory or the specified individual leaf file into the default unit element definitions. Can be used multiple times.
`--mergeModelDefs {`*`directory`* `|` *`filename`*`}`	Merges all model definition files (**.ommx*) in the specified directory or individual model definition files. Can be used multiple times.
`--mergeTacticalGraphicsTypeMap {`*`directory`* `|` *`filename`*`}`	Merges all tactical graphics type mapping files (**.mtgx*) in a directory or individual tactical graphics type mapping files. Can be used multiple times. See note below.
`--mergeUnitTypeMap {`*`directory`* `|` *`filename`*`}`	Merges all unit type mapping files (**.magx*) in a directory or individual unit type mapping files. Can be used multiple times. See note below.
`--modelPreloadFileName` *`filename`*	Specifies a file that has a list of models to preload.

The options for merging type maps (--mergeEntityTypeMap, --mergeTacticalGraphicsTypeMap, and --mergeUnitTypeMap) allow you to merge your type mapping with those that already exist. By default, that does not overwrite the existing mappings. Your merged mappings will exist alongside those that were already loaded.

If you want to overwrite the existing ones, use the merge option alongside the --overwriteMappingsOnMerge command-line option.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
`--noAppDataEntityDefs`	Do not load the entity element definitions in *./SharedData/19/latest/ModelData/Configuration/Definitions*.
`--noAppDataEntityTypeMap`	Do not load entity type mappings in *./SharedData/19/latest/ModelData/Configuration/Definitions*.
`--noAppDataModelDefs`	Do not load model definitions in *./SharedData/19/latest/ModelData/Configuration/Definitions*.
`--noAppDataTacticalGraphicsDefs`	Do not load the tactical graphics element definitions in *./SharedData/19/latest/ModelData/Configuration/Definitions*.
`--noAppDataTacticalGraphicsTypeMap`	Do not load tactical graphics type mappings in *./SharedData/19/latest/ModelData/Configuration/Definitions*.
`--noAppDataUnitDefs`	Do not load the unit element definitions in *./SharedData/19/latest/ModelData/Configuration/Definitions*.
`--noAppDataUnitTypeMap`	Do not load unit type mappings in *./SharedData/19/latest/ModelData/Configuration/Definitions*.
`--noGui`	Start without the graphical user interface.
(`--notifyLevel` | `-n`) *notification_level*	Specifies the notification level for application messages, where *notification_level* is a number from 0 through 4. The least verbose message level is 0, the highest is 4. The default is 2.
`--overwriteMappingsOnMerge`	Any mapping merge specified on the command line overwrites mappings in the default files. Use in conjunction with the command-line options for type maps (above).
`--plugin` *filename*	Load the specified GUI plug-in. Can be used multiple times.
`--pseudoFullScreen`	Use pseudo full screen for full screen mode. This may improve performance when using dynamic ocean.
(`--realizeMinimap` | `-m`)	Realize the minimap model set.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
`--sensorMode`	Run VR-Engage Sensor Operator. You can create and control only sensors and humans.
`--setting`	You can use the new `--setting` command-line option to override a specific parameter in the *playerStation.lua* file. For example, you can specify how VR-Engage connects to an existing simulation session. (You can also use the `--customSettingsFile` option to override the entire file.) For details, refer to *playerStation.lua*.
`--sharedDataDir` *directory_path*	Use this to specify the location of the shared data directory if you installed it somewhere other than the default, which is *./SharedData*.
`--sharedDataRoot` *directory_root*	Specifies the location of the shared data root. This combined with the stem and data version creates the shared data path.
`--sharedDataStem` *directory_stem*	Specifies the location of the shared data stem. This combined with the data root and data version creates the shared data path.
`(--showConsole | -c)`	Display the console window.
`(--slave | -s)`	Run as a remote display engine.
`--stationName`	Station name of application as seen by the VR-Engage manager.
`--stencilBits` *bits*	Sets the number of stencil bits (0 or 8). Default: 0.
`(--suppressHatIntersectOnAttach | -S)`	Suppress the periodic HAT intersection when attached. This setting can improve performance if you need to attach the observer to ground vehicles a lot, but do not need to attach to air vehicles.
`--tacticalGraphicsDefsDir` *filename*	Loads the *.hier* and *.leaf* files in the specified directory to the tactical graphics element definition manager.
`--tcpSendTimeout`	Sets the timeout (in seconds) for the master display engine to wait for a remote display engine before giving up on a send. Default: 30 seconds.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
(*terrain_MTF_file* | *scene_MSF_file*)	Specifies a terrain (MTF) or scene (MSF) file to load. By default, VR-Engage looks in the expected data location for terrain or scene files in either the default location, *C:\MAK\SharedData*, or the one specified by the environment variables MAK_SHARED_DATA_ROOT and MAK_SHARED_DATA_STEM. (If the file is in another location, the path must be a relative path from *./bin64* to the file, for example, *../../myTerrains/MTF_file*.)
--unitDefsDir *filename*	Loads the *.hier* and *.leaf* files in the specified directory to the unit element definition manager.
--unplug *filename*	Do not load the specified plug-in. Can be used multiple times.
--userDataDir *directory*	Specifies the location of user data directory. If not specified, VR-Engage uses the default: *./userData*.
--windowedMode	Starts VR-Engage in windowed mode instead of full screen. Also clears the Full Screen option in the Choose Role Panel.
(--withDisplayEngine | -W) *port@netAddress*, *displayConfigFilenameN* Legacy Windows syntax: (--withDisplayEngine | -W)*port@netAddress*; *displayConfigFilenameN* (semicolon separator) Legacy Linux syntax: (--withDisplayEngine | -W)*port@netAddress*: *displayConfigFilenameN* (colon separator)	Connect to a display engine running on another computer. For more information with an example, see "Loading Multiple Display Engine Configurations from the Command Line" under "Loading a Display Engine Configuration" in *VR-Vantage User Guide*. Can be used multiple times.
(--ignore_rest | --)	Ignores all command-line options that are listed after this argument. This option is useful for disabling part of a command line in a script for test purposes so that you do not have to retype all of the command options at a later point.
(--version | -v)	Displays version information.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
(--help | -h)	(Displays command-line usage information in a console window.)
HLA Only	**These options are available when you specify --hla1516e or --hla4.**
(--appNumber | -a) *number*	Specifies the application number.
(--execName | -x) *exec_name*	Specifies the name of the federation execution. Default: MAK-RPR-2.0.
(--federateName | -N) *federate_name*	Specifies a unique name that can be assigned to the federate. No two federates within the same federation can have the same name. If no name is specified, the RTI automatically assigns a unique name for the federate. Default: VR-Forces. (HLA Evolved and HLA 4 only)
(--federateType | -p) *type*	Federate Type can be used to distinguish different categories of federates, and its only practical use is in save-and-restore.
(--fedFileName | -F) *fedfile_name*	FED file name.
--fileTransporterReceivePort *port*	Port for the file transporter. This is the port used to receive order of battle and plan files.
--fomMapperInitData *data*	FOM Mapper initialization data.
(--fomMapperLib | -f) *libname*	FOM Mapper library name.
--fomModules *module*	Specifies an HLA Evolved or HLA 4 FOM module. Can be used multiple times.
(--hostAddressString | -H) *address*	Specifies the host address. This is usually the same as the device address.
--ignoreAdvisories	Enables or disables HLA advisory messages. Default: true.
(--localSettingsDesignator | -S) *designator*	Specifies a string that is passed to the RTI during federate connect when using HLA Evolved. It is usually used to provide additional RTI configuration, but every RTI uses it differently. Please consult your RTI documentation for proper usage.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
`--reportedHostAddressString` *address*	Specifies how VR-Forces reports its IP address to another VR-Forces application for the purpose of file transfers. It is only important when you have a GUI or remote controller on one side of a router and a simulation engine on the other, and there is some kind of network address translation occurring whereby they have to connect to one another using an IP address other than the one they recognize themselves at locally.
`--rprFomRevision` *revision_number*	RPR FOM revision for RPR FOM 2.0, draft 17.
`--rprFomVersion` *version_number*	RPR FOM version. (0.5, 0.7, 0.8, 1.0, 2.0006, 2.0014, 2.0017, or 2.0)
(`--sessionId` | `-i`) *ID*	Specifies the session ID for this instance of the GUI.
(`--siteId` | `-s`) *ID*	Specifies the site ID.
`--useAbsoluteTimeStamps`	Enables use of absolute timestamps. Otherwise uses relative timestamps.
`--useGeographicDdm`	Specifies use of geographic DDM if you are using the MAK RTI. Default: false.
DIS Only	**These options are available when you specify `--dis`.**
(`--appNumber` | `-a`) *number*	Specifies the application number.
`--appIdRange` *range*	The application ID range for VR-Forces sim engines.
`--defaultObjectTimeout` *timeout*	Specifies the timeout interval for objects.
(`--destAddrString` | `-A`) *address*	Specifies the destination address.
`--deviceAddress` *address*	Specifies the address of the network card to use for UDP traffic.
(`--disPort` | `-P`) *port*	DIS port.
`--disVersion` *version*	The DIS version. Default: 7.
(`--exerciseId` | `-x`) *ID*	Exercise ID.
`--fileTransporterReceivePort` *port*	Port for the file transporter. This is the port used to receive order of battle and plan files.
(`--hostAddressString` | `-H`) *address*	Specifies the host address. This is usually the same as the device address.

Table 4. VR-EngagevrEngage GUI command-line options (continued)

Option	Description
`--mcastTtl` *time*	Specifies the multicast time-to-live. Default: -1.
(`--multicastAddresses` | `-S`) *address1 ... addressn*	Specifies one or more multicast addresses.
`--recvBufferSize` *size*	Receive buffer size.
`--reportedHostAddressString` *address*	Specifies how VR-Forces reports its IP address to another VR-Forces application for the purpose of file transfers. It is only important when you have a GUI or remote controller on one side of a router and a simulation engine on the other, and there is some kind of network address translation occurring whereby they have to connect to one another using an IP address other than the one they recognize themselves at locally.
`--sendBufferSize` *size*	Send buffer size. Default: -1.
(`--sessionId` | `-i`) *ID*	Specifies the session ID for this instance of the GUI.
(`--siteId` | `-s`) *ID*	Specifies the site ID.
`--useAbsoluteTimeStamps`	Enables use of absolute timestamps. Otherwise uses relative timestamps.
`--subnetMask` *mask*	Specifies a subnet mask.
`--useAsyncIO`	Use asynchronous I/O. Default: False.
`--useIpv6`	Specifies that the DIS exercise connection use the IPV6 protocol, if available.

3.13.2 Command-Line Options for vrEngageSim

Table 5 describes the command-line options for vrEngageSim. You can use these when you start a sim engine for VR-Engage.

Table 5. vrEngageSim command-line options

Option	Description
`--additionalSystemScriptsDirectory` *directory*	Specifies a directory where VR-Engage will look for system scripts when it populates the Scripted Task dialog box script list.
`--appDataDir` *directory*	Specifies the location of application data. If not specified, VR-Engage uses the default: *./appData*.
`--appIdRangeMax` *max*	Specifies a range maximum to roll the application number to the next ID in the range. When the upper limit of the entity number is reached, the application number for the next object created will be increased and the entity number restarted at 0. If the application number range max is reached no more objects will be created on the sim engine until some are deleted.
(`--appNumber` | `-a`) *number*	Specifies the application number.

Table 5. vrEngageSim command-line options (continued)

Option	Description
`--attributePduStrategy`	This flag controls how VR-Engage sends DIS attribute PDUs. For the GUI to control VR-Forces Entities, the Simulation Engine must supply additional attributes through DIS. Some applications cannot correctly interpret coupled attribute PDUs in some DIS versions. In those cases, changing the default behavior may be required.The possible values are: • 0: DtAttributePduStrategy_ Unspecified • 1: DtAttributePduStrategy_ NoAttributePdu • 2: DtAttributePduStrategy_Couple • 3: DtAttributePduStrategy_ DoNotCouple • 4: DtAttributePduStrategy_ DoNotCoupleAndSendAttributeFirst The default is 0, which means that VR-Engage chooses based on the selected version of DIS. For DIS 7, it uses 2. Mode 1 will not send any additional attributes, so VRF GUI will not recognize the published entities as VRF entities and will not be able to control them.
`(--batchScenarioFileName | -B)` *batch_file*	Runs the specified batch file (for use with one sim engine only).
`--clockMode` *mode*	For Windows only, specifies whether to use the performance counter or the system clock for absolute time stamping. This only applies when Use Absolute Time Stamps is enabled. **0** - Performance counter (default), highly accurate internal timer **1** - Continual synchronization with system clock
`--connectionless`	Run VR-Forces without a connection.
`--countryCodesMappingFile` *file_name*	Specifies an alternative file that maps country codes to country names and DIS enumeration values.

Table 5. vrEngageSim command-line options (continued)

Option	Description
`--dataDir` *directory*	Specifies the location of the application's data directory. If not specified, VR-Engage uses the default, *./data*.
(`--destAddrString` | `-A`) *address*	Specifies the destination address.
`--deviceAddress` *address*	Specifies the address of the network card to use for UDP traffic.
`--dis5Timestamps`	In DIS 7, there is a discrepancy in timestamps that can lead to some of them appearing to take place in the future. Use this option to enable the time management method from DIS 5.
`--diGuyAnimationsFile` *file_name*	Specifies an additional configuration file for DI-Guy animations.
`--diGuyCharacterDataFile` *file_name*	Specifies an additional configuration file for DI-Guy character data.
`--disableParallelPublisherTick`	Turn off ticking of publishers in parallel.
`--disableParallelTick`	Execute all tick code serially.
`--disableRemoteDiscovery`	Disable discovery of remote objects.
`--doNotLoadVrfPlugins`	Prevents loading of plug-ins in the *./plugins* directory.
(`--doNotUseConsole` | `-q`)	Specifies that all *vrfSim* output go to the log file rather than the console (quiet mode). If you are using a high level of notification, sending output to the console can degrade performance. Running in quiet mode prevents this degradation of performance. This command only applies to sim engines running on Windows. To create a log file on Linux, redirect the output of vrfSim, for example: `vrfSim > mylog.txt`
`--enableChannel` *channel*	Enables a specific channel for the channel-specific output, even if it is disabled in *channelSettings.mtl*.
`--enablePduProfiling` *string*	Enables the entity PDU flood check of a given time or a number of packets over that time.
`--entityVisEnable`	Enable or disable the Entity Visualizer tool for graphviz.

Table 5. vrEngageSim command-line options (continued)

Option	Description
`--entityVisFileOut` *`filename`*	Name of the output file for the Entity Visualizer tool.
`--entityVisSMS` *`sms_name`*	Name of the simulation model set (SMS) used with Entity Visualizer tool.
`--entityVisType` *`entity_name`*	Default entity to use for the Entity Visualizer tool.
`--exConnConfigFile` *`filename`*	Specifies the exercise connection configuration file to use. Use this to override settings in the VR-Link exercise connection configuration file, *./appData/settings/connections/MAK-ONE-YYYY-Config.xml*, which defines the default connections settings. For more information, refer to the *MAK ONE Interoperability Guide*.
`--exitOnTestResultCommand`	If a plan in the simulation includes the Test Result statement, and the simulation engine was started with this command, then the simulation sends the message to all sim engines to close after the Test Result statement.
`--fileNotifyLevel` *`level`*	Notify level in to use in the log file.
`--frontEndPID` *`PID`*	The process ID for the GUI when running in combined mode.
`--fullForcesMode`	Start the VR-Engage sim engine as a full VR-Forces sim engine. It requires a VR-Forces sim engine license.
`--fullyLoadTerrain`	For paged terrains, load all pages.
`--hostAddressString` *`address`*	Specifies the host address. This is usually the same as the device address. Called Device Address in the Launcher.
`--loadPlugin` *`plugin_filename`*	Specifies a plug-in to load. Accepted multiple times. The filename can be a DLL or SO or an XML file that specifies the plug-ins to load. The order in which you list multiple plug-ins on the command line does not determine the order in which they are loaded. For details, see "3.14. Managing Plug-ins" on page 112.
`--logFileName` *`filename`*	Specifies the log file name.

Table 5. vrEngageSim command-line options (continued)

Option	Description
`--logFrameRateStatistics`	Starts the Tracy Client connection to capture performance statistics for the sim engine. See "A.2. Using the Tracy Client" on page 394 to learn more.
`--mcastTtl` *time*	Specifies the multicast time-to-live. Default: -1.
`--msdlSIDCMappingFile` *file_name*	Specifies an alternative file for mapping SIDC codes to DIS enumerations.
(`--notifyLevel` | `-n`) *notification_level*	Specifies the notification level for messages written to the console or the VR-Engage log file (*vrf.log*), where *notification_level* is a number from 0 through 4. The least verbose message level is 0, the highest is 4. The default is 2.
`--nonVrfObjectsUUIDScheme` *scheme*	Sets the UUID scheme for non VR-Forces objects. Values are `entity-identifier`, `global-identifier`, `marking-text`.
`--numCallbackThreads` *threads*	Number of threads in the callback queue.
`--numNetworkCallbackThreads` *threads*	Number of network threads in the callback queue. Zero means run network in a thread serially (unless it is turned off). If publishing is in the tick thread, this will be shared with publishers.
`--outputInterval` *interval*	Interval for output.
`--outputLogFile` *file_name*	The file to use to write out application output.
`--recvBufferSize` *size*	Receive buffer size.
(`--scenarioFileName` | `-L`) "*scenario_pathname*"	Loads the specified scenario. The pathname can be absolute or relative from the *./bin* directory. Mutually exclusive with the `-T` option.
`--sendBufferSize` *size*	Send buffer size. Default: -1.
(`--sessionId` | `-i`) *ID*	Specifies the session ID for this instance of the GUI.
(`--settingsFile` | `-d`) *file_name*	Specifies a configuration file to load instead of *vrfSimSettings.xml*.

Table 5. vrEngageSim command-line options (continued)

Option	Description
`--setMainThreadToHighPriority`	Sets the VR-Engage main thread to high priority. When enabled, other threads may be starved if there are insufficient processors available. If running on hardware with limited processor time, this may decrease overall system performance.
`--sharedDataDir` *directory_path*	Use this to specify the location of the shared data directory if you installed it somewhere other than the default, which is *./SharedData*.
`--sharedDataRoot` *directory_root*	Specifies the location of the shared data root. This combined with the stem and data version creates the shared data path.
`--sharedDataStem` *directory_stem*	Specifies the location of the shared data stem. This combined with the data root and data version creates the shared data path.
`--sharedSettingsDirectory` *directory*	Specifies the directory where shared settings are stored. The default directory is *$(APP_DIR)/settings*. If the `--useUserSettingsDirectory` flag is set, then the user settings directory is used to store the settings instead. If not, the application's configuration location (system-dependent) is used.
`--simulationModelSet` *sms*	Specifies the simulation model set. It must be declared separately for GUI and sim engine, even in combined mode.
(`--siteId` | `-s`) *ID*	Specifies the site ID.
(`--startInRunMode` | `-r`)	Starts the sim engine in run mode. If you start the sim engine without specifying a scenario (with `-L`) or a terrain database (with `-T`), the sim engine starts in pause mode.
`--startMinimized`	Minimizes the vrfSim console at startup.

Table 5. vrEngageSim command-line options (continued)

Option	Description
`--subnetMask` *mask*	Specifies a subnet mask. VR-Engage can usually figure out the subnet mask from the IP address. However, if it does not do this correctly, you can explicitly specify the subnet mask using this command-line option.
`--suppressDataAvailableException`	Do not throw an exception when dataAvailable is not checked in non-blocking terrain calls.
`--targetFrameRate` *rate*	Specifies the frame rate above which vrfSim will yield CPU time to other processes.
(`--terrainDatabase` | `-T`) *terrain_database*	Specifies a terrain database to load in the sim engine. Mutually exclusive with the `-L` option.
`--testOutputFilename` *filename*	The name of the file to which you want to log scenario autotest outputs.
`--tickPublishersDuringEntityTick`	Tick publishers during the entities' ticks.
`--tracyPort` *integer*	Allows you to specify a sim engine port to profile with the Tracy Client. You can then profile a different sim engine port than the default, and also profile sim engines separately from GUIs. Be sure to include the colon before the port number. The default port is **:10101**.
`--useAbsoluteTimeStamps`	Enables use of absolute timestamps. Otherwise uses relative timestamps.
⚠ If you select `--useAbsoluteTimeStamps`, it is your responsibility to ensure that the time clocks of exercise participants are synchronized. If they are not synchronized, VR-Engage may not work properly.	
`--useAsyncIO`	Use asynchronous I/O. Default: False.
`--useIpv6`	Specifies that the DIS exercise connection use the IPV6 protocol, if available.
`--userDataDir` *directory*	Specifies the location of the user data directory for the sim engine.

Table 5. vrEngageSim command-line options (continued)

Option	Description
`--useUserSettingsDirectory`	Indicates that the system should store settings in the user login settings directory, rather than the application settings directory. Use in conjunction with `--sharedSettingsDirectory`.
`--vrengageOwnshipSim`	This causes the VR-Engage sim engine to start as an ownship sim engine. It does not require a VR-Forces sim engine license and is limited to only one entity.
`--waitQueue`	Turn off main thread participation in parallel algorithms.
`(--ignore_rest | --)`	Ignores all command-line options that are listed after this argument. This option is useful for disabling part of a command line in a script for test purposes so that you do not have to retype all of the command options at a later point.
`(--version | -v)`	Displays version information.
`(--help | -h)`	Displays command-line usage information in a console window.
HLA Only	**These are the options specific to vrEngageSimHLA1516e and vrEngageSimHLA4.**
`--disableRequestUpdates`	Disables the VR-Link Request for updates upon object discovery
`(--execName | -x)` *`exec_name`*	the name of the federation execution Default: VR-Link.
`(--federateName | -N)` *`federate_name`*	Specifies a unique name that can be assigned to the federate. No two federates within the same federation can have the same name. If no name is specified, the RTI automatically assigns a unique name for the federate. Default: VR-Forces. (HLA Evolved and HLA 4 only)
`(--federateType | -p)` *`type`*	Federate Type can be used to distinguish different categories of federates, and its only practical use is in save-and-restore.
`(--fedFileName | -F)` *`fedfile_name`*	FED file name.
`--fomMapperInitData` *`data`*	FOM Mapper initialization data.
`(--fomMapperLib | -f)` *`libname`*	FOM Mapper library name.

Table 5. vrEngageSim command-line options (continued)

Option	Description
`--fomModules` *`module`*	Specifies an HLA Evolved or HLA 4 FOM module. Can be used multiple times.
`--ignoreAdvisories`	Enables or disables HLA advisory messages. Default: true.
(`--localSettingsDesignator` | `-S`) *`designator`*	Specifies a string that is passed to the RTI during federate connect when using HLA Evolved. It is usually used to provide additional RTI configuration, but every RTI uses it differently. Please consult your RTI documentation for proper usage.
`--mimModules` *`HLAE_MIM`*	Specifies an HLA Evolved MIM module.
`--netnFomRevision` *`revision_number`*	Specifies the internal revision of the implementation of the NETN FOM version as an integer. The default (and currently the only supported revision) is 1.
`--netnFomVersion` *`version_number`*	Specifies the NETN FOM version as a double. The default (and currently the only supported version) is 3.0.
`--noRtiCompilerCheck`	Disable RTI Compiler version check.
`--rprFomRevision` *`revision_number`*	RPR FOM revision for RPR FOM 2.0, draft 17.
`--rprFomVersion` *`version_number`*	RPR FOM version. (0.5, 0.7, 0.8, 1.0, 2.0006, 2.0014, 2.0017, or 2.0)
`--setFomModuleList`	Overwrite FOM Modules with the listed modules. Accepts a comma-separated list of module names; the best practice is to list them in hierarchical order to prevent any issues related to dependencies.
`--timeManagement`	Enables time management for the sim engine.
DIS Only	**These are the options specific to vrEngageSimDIS.**
`--appIdRange` *`lower-upper`*	Specifies a range of application IDs that represent objects generated by VR-Forces.
`--defaultObjectTimeout` *`timeout`*	Specifies the timeout interval for objects.

Table 5. vrEngageSim command-line options (continued)

Option	Description
(--disPort | -P) *port*	DIS port. If you specify a DIS port that is less than 3000 or greater than 4000, discovered objects might not be seen as VR-Forces objects. If you want to use a DIS port outside this range, specify it for the GUI and sim engine with the `--appIdRange` command-line argument.
--disVersion *version*	The DIS version. Default: 7.
(--exerciseId | -x) *ID*	Exercise ID.
(--multicastAddresses | -S) *address1 ... addressn*	Specifies one or more multicast addresses.
--suppressSelfReflect	Represses self reflection of the DIS exercise connection.

3.13.3 Command-Line Options for Player Stations

You can specify additional command-line options by editing *./appData/scripts/playerStation.lua* and adding command-line options to `additionalBackendCommandLineParameters`. For example:

```
additionalBackendCommandLineParameters = --outputLogFile my-log-file
    --startInRunMode --userDataDir my-data-directory
```

Please note:

- The connection settings in the VR-Engage GUI's current connection options. A warning displays in the GUI console and in *vrEngage.log* for each ignored parameter, and what the replacement is.
- If you supply a DIS-specific command to HLA, or vice versa, the application will not start correctly.

3.14. Managing Plug-ins

One way to extend the functionality of VR-Engage is by loading plug-ins that have been written for it by MAK or other software vendors. You can manage which plug-ins are loaded when VR-Engage starts. You can also view information about the plug-ins that have been loaded. This is primarily used for sim engine plug-ins.

Plug-ins are implemented as one or more dynamic linked libraries (DLL or SO files). The DLLs can be release or debug and for a specific protocol or protocol-independent. Each association of an application, platform, build type, and transport type is called a configuration. Use the Plug-ins Editor to specify the DLLs for use with each of the desired configurations.

- Changes to plug-in configurations do not take effect until the next time you start the affected VR-Engage application.
- You can also specify plug-ins on the command line using the `--loadPlugin` *`plugin_filename`* option. Be sure that you do not load the same plug-in multiple times.
- To access the Settings menu, you must be in Admin mode. See "3.1.3 Admin, User, and Lockdown Modes" on page 59.

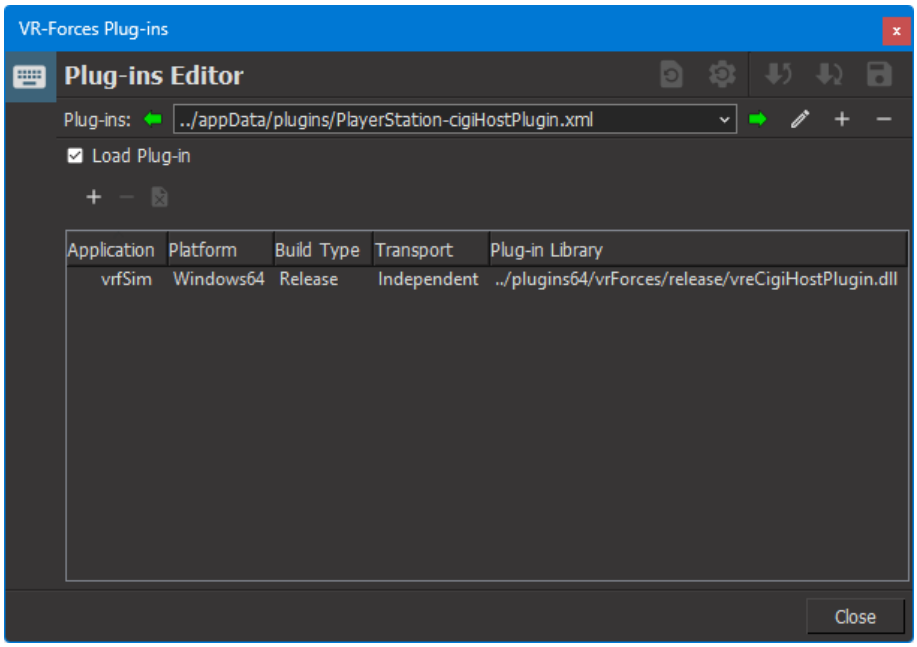
3.14.1 Loading Plug-ins

You can specify which plug-ins to load using the Plug-ins dialog box. Those plug-ins load the next time VR-Engage starts.

➤ **To specify that a plug-in be loaded or not loaded in the Plug-ins dialog box:**

1. Choose **Settings > Plug-ins**. The Plug-ins dialog box opens (Figure 28). (To access the Settings menu, you must be in Admin mode.)

Figure 28. Plug-ins Editor



2. In the list of plug-ins at the top of the page, select the one that you want to edit. The configurations for that plug-in are listed in the window.
3. Select or clear the Load Plug-in check box.

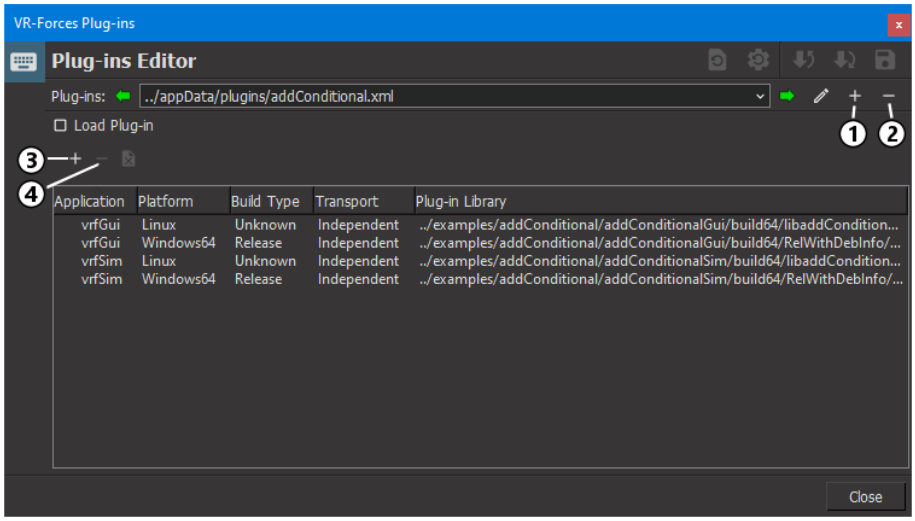
3.14.2 Adding a Plug-in

When you add a plug-in, VR-Engage creates an XML file (in *./appData/plugins*) that contains all of the information VR-Engage needs to load the plug-in. New plug-ins are added with entries for each of the configurations (application, platform, build type, transport). After you add a plug-in, you must specify the DLLs for each configuration that you want to support.

➤ **To add a plug-in:**

1. Choose **Settings > Plug-ins**. The Plug-ins dialog box opens (Figure 28 on the previous page).
2. Click the Add button (+) at the upper right corner of the page (#1 in Figure 29). The New Plug-in dialog box opens.
3. Type a name for the plug-in.
4. Click OK. The plug-in is added to the Plug-ins list. It has configurations for each of the combinations of application, platform, build type, and transport (Figure 29) as shown in the figure.

Figure 29. New plug-in



5. Specify the DLL for each configuration that you want to support, as described in "3.14.3 Specifying the DLLs for a Plug-in" below.

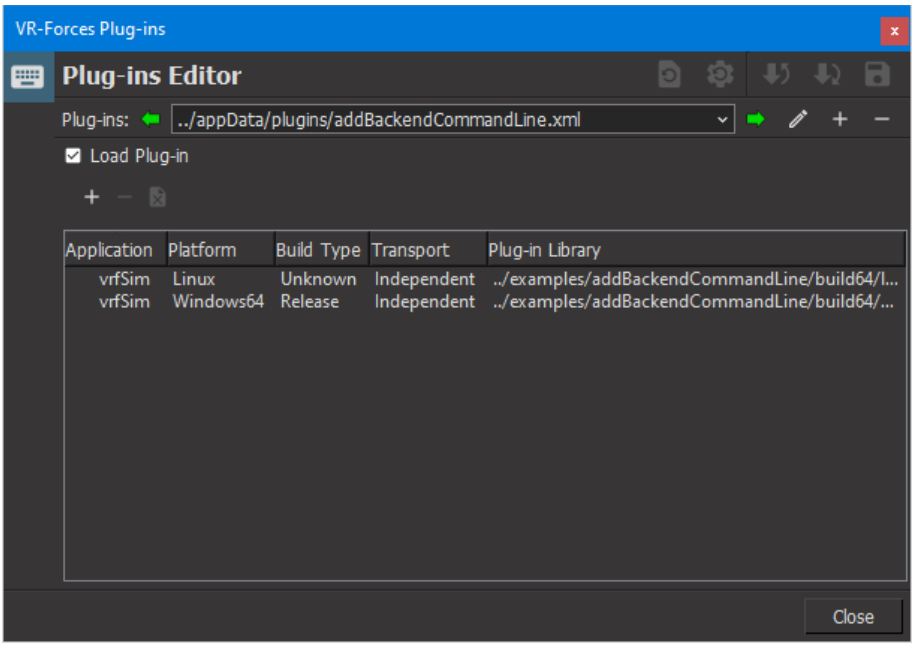
3.14.3 Specifying the DLLs for a Plug-in

Each plug-in configuration that you want to be able to load must specify the DLL to load for that configuration.

➤ **To specify the DLL for a plug-in configuration:**

1. Choose **Settings > Plug-ins**. The Plug-ins dialog box opens (Figure 30).

Figure 30. Plug-ins Editor



2. In the list of plug-ins at the top of the page, select the one that you want to edit.
3. Click the Plug-in Library column for the configuration that you want to edit. The Select Plug-in dialog box opens.
4. Select the DLL that you want to load for this configuration.
5. Click Open. The value changes.

Removing a Plug-in's DLL

You can remove the DLL assignment for a plug-in configuration without removing the configuration.

➤ **To remove the DLL assignment for a plug-in:**

1. Choose **Settings > Plug-ins**. The Plug-ins dialog box opens.
2. Select the Plug-ins Editor page (Figure 30).
3. Select the configuration whose DLL assignment you want to remove.
4. Click the Erase button ( ).

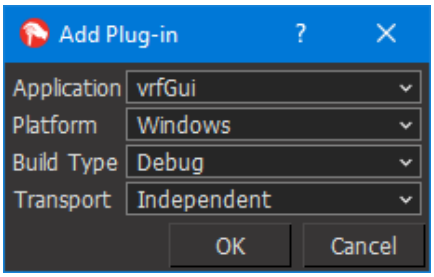
3.14.4 Adding a Plug-in Configuration

A plug-in configuration is a row in the Plug-ins Editor window. It specifies an application, platform, build type, transport type, and plug-in library.

➤ **To add a plug-in configuration:**

1. Choose **Settings > Plug-ins**. The Plug-ins dialog box opens (Figure 28 on page 113).
2. In the list of plug-ins at the top of the page, select the one that you want to edit.
3. Click the Add button (+) above the list of configurations (#3 in Figure 29 on page 114). The Add Plug-in dialog box opens (Figure 31).

Figure 31. Add Plug-in dialog box



4. For each parameter (Application, Platform, Build Type, and Transport), select a value from the list.
5. Click OK. The configuration is added to the list.
6. Specify a DLL for the configuration, as described in "3.14.3 Specifying the DLLs for a Plug-in" on page 114.

3.14.5 Deleting a Plug-in Configuration

➤ **To delete a plug-in configuration:**

1. Choose **Settings > Plug-ins**. The Plug-ins dialog box opens (Figure 28 on page 113).
2. In the list of plug-ins at the top of the page, select the one that you want to edit.
3. Select the configuration that you want to delete.
4. Click the Delete button (−) that is above the list of configurations (#4 in Figure 29 on page 114).

3.14.6 Deleting a Plug-in

When you delete a plug-in, you delete the XML configuration file in *./appData/plugins* that specifies the plug-in configurations and whether to load it.

➤ **To delete a plug-in:**

1. Choose **Settings > Plug-ins**. The Plug-ins dialog box opens (Figure 28 on page 113).
2. In the list of plug-ins at the top of the page, select the one that you want to delete.
3. Click the Delete button (−) to the right of the plug-ins list (#2 in Figure 29 on page 114).



4. Optimizing Performance

This chapter briefly explains how to monitor and improve performance. For more information, refer to the *VR-Vantage User Guide*.

Note that debug menus display over the interface, and clicks in the debug menu also register on the interface. Be sure to position the debug menu so that it is not over any buttons.

4.1. Improving VR-Engage Performance 118
4.1.1 GUI Performance 118
4.1.2 Simulation Engine Performance 119
4.2. Displaying Performance Statistics 120
4.2.1 Displaying VR-Engage Performance Statistics 120
4.2.2 Using the Tracy Client 121
4.3. Configuring Graphics Quality 122
4.3.1 Alpha to Coverage 122
4.3.2 Using the Ambient Occlusion Effect 124
4.3.3 Configuring Ambient Occlusion 125
4.4. DI-Guy Performance Settings 127
4.5. Specifying Terrain Categories 128
4.6. Using the Developer Debug Overlay 128
4.7. Using the VR-Engage Debug Menu 130

4.1. Improving VR-Engage Performance

4.1.1 GUI Performance

To improve performance in the VR-Engage GUI (the player station, or front end), review the suggestions in this chapter and in "A.1. Performance and Display Issues" on page 394. Improving performance can decrease the quality of visuals. You must determine which compromises to make in order to achieve the performance needed for your use case and hardware.

In addition to the configuration options that are common for VR-Vantage applications, there are additional ways you can improve the performance of the VR-Engage GUI.

- Use a dedicated machine for player stations. Run the VR-Engage GUI on one machine and the sim engine on a separate machine. While VR-Engage can host its own session, this is typically used only for small scenarios or testing. To scale up to larger scenarios and maintain good performance in the GUI, provide dedicated hardware.

  The basics of connecting player stations to simulations is covered in "3.1. Starting VR-Engage and Joining a Session" on page 56.

- Limit texture resolutions. Use the `--limitEntityTextures "width height"` and `--limitTerrainTextures "width height"` command-line arguments to reduce the risk of running out of GPU memory in cases where GPU usage is too high. For example:

  ```
  --limitTerrainTextures "256 256" --limitEntityTextures "256 256"
  ```

  This is useful for simulations focused on ground-based player entities where players frequently observe a lot of details at close range. Depending on your 3D content, you might be able to reduce texture memory resolution without much of a discernible difference.

  To learn about these and other command-line options, see "3.13. Command-Line Options" on page 87.

- Use remote display engines. This distributes the work of rendering across multiple machines. For some roles, such as the pilot role, there is the challenge of a fast-moving entity continuously paging in large amounts of terrain data, and often the need for a wide field of view for the role player. To maintain a good frame rate for the player station, you can use one machine to host the VR-Engage role player and supplement it with one or more remote display engines to provide additional channels.

  To learn about remote display engines, see the chapter on *Windows, Channels, and Display Engines* in the *VR-Vantage User Guide*.

4.1.2 Simulation Engine Performance

To improve sim engine performance, the guidance for VR-Forces performance applies to VR-Engage, as well. This is because the VR-Engage sim engine is a VR-Forces sim engine. For details, refer to the *Optimizing Performance* chapter of the *VR-Forces User Guide* or help.

Below are some additional suggestions specific to VR-Engage that can help you address performance issues related to the sim engine. Many of these involve ways of hosting entities on their own sim engines. To learn more, see "3.1.4 Hosting an Entity in VR-Engage While Participating in a VR-Forces Session" on page 60.

- Host the simulation with VR-Forces. For larger, complex scenarios with potentially multiple role players, have VR-Forces host the simulation on its own software. For details, refer to the *Running VR-Forces* chapter of the *VR-Forces User Guide* or help.
- Dedicated sim engine for each role player. For a better role-playing experience, simulate the ownship on its own sim engine. Maintaining a high frame rate for ownship entities provides a more responsive experience for the player.
- Host Vortex entities on a dedicated sim engine. To ensure a consistent frame rate of at least 20 FPS, it is often best to simulate Vortex entities on their own sim engine.
- Minimize the number of CGF entities with Vortex. Your use case may require that a CGF vehicle be promoted as a full-fledged Vortex entity while not actively tasked. While this enables high-fidelity collision interactions with other Vortex vehicles, it is also more "expensive" than simulating it as a simple CGF entity. If you do not need these interactions, consider keeping this option disabled. (It is disabled by default.) For details, see "16.2.3 Vortex While Untasked" on page 375.
- Simplify terrain collision geometry in the sim engine. In general, this is a good practice, but it is particularly important in the case of Vortex vehicles. As Vortex entities move around the terrain, they are continuously paging in nearby terrain, transforming it, and using it to inform the vehicle's interaction with the terrain. If the entity encounters a terrain tile with detailed geometry, this can result in long frames and sim engine stalls. In those situations, the sim engine can stall and make the vehicle unstable. (The Vortex engine is sensitive to long frames.) Use the techniques described in the *Terrain Performance and Configuration* chapter of the *Adding Content to MAK ONE Applications Reference Manual* to create simplified versions of terrain geometry objects.

4.2. Displaying Performance Statistics

VR-Engage can display performance statistics and a function profiler (Figure 32).

Figure 32. VR-Engage performance statistics and function profiler



Displaying performance statistics can affect performance.

4.2.1 Displaying VR-Engage Performance Statistics

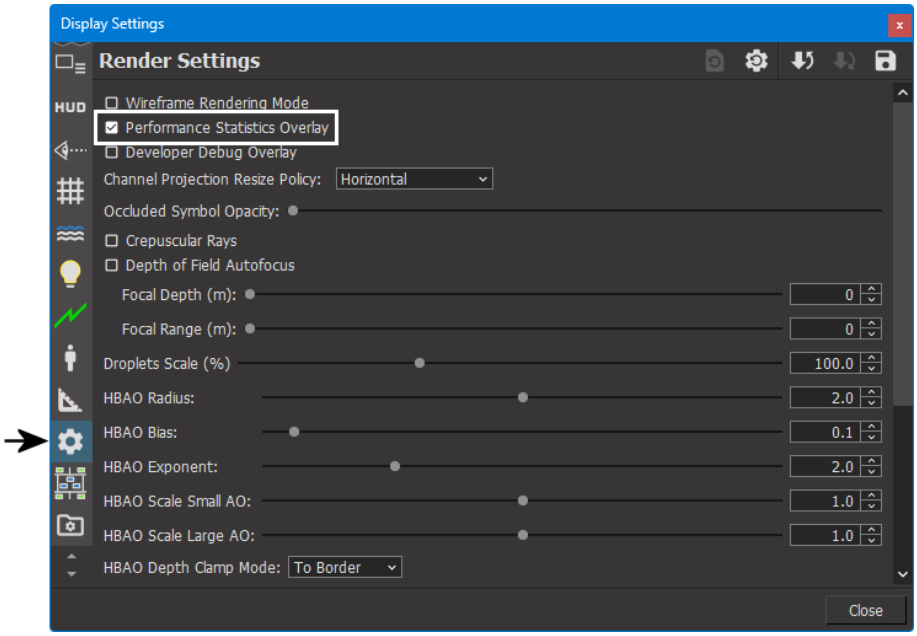
The VR-Engage Performance Statistics shows information about the scene, while the Function Profiler is primarily for the use of developers who are debugging VR-Engage applications. For information about how to integrate the profiler into your code, please see Chapter 11, *Using the Function Profiler*, in the *VR-Vantage Developers Guide*.

For information about another debugging utility, see "4.6. Using the Developer Debug Overlay" on page 128. You can also use the VR-Engage Debug Menu.

➤ **To display VR-Engage performance statistics:**

1. Ensure that you are in Admin mode (**Shift+F12**). Note that, if you are currently engaged in a role, you might need to use the **Ctrl** key to use the cursor. For more information, "3.1.3 Admin, User, and Lockdown Modes" on page 59.
2. Choose **Settings > Display**. The Display Settings dialog box opens.
3. Select the Render Settings page (Figure 33).

Figure 33. Render Settings page



4. Select the Performance Statistics Overlay check box. A statistics display is added to the window (Figure 32 on the previous page).

4.2.2 Using the Tracy Client

VR-Engage includes the Tracy Client, which you can use to evaluate performance or save a file to send to MAK Support to help debug performance issues. To learn more about using the Tracy Client, read the documentation at https://bitbucket.org/wolfpld/tracy/downloads.

> ⚠ VR-Engage is compatible with the Tracy Client version that was included in the installation package. If you update Tracy Client, it might no longer be compatible.

➤ **To run the Tracy Client and save a file:**

1. Ensure that you are in Admin mode (**Shift+F12**).
2. Choose **Settings > Display**. The Display Settings dialog box opens. Select the Render Settings page.
3. Select the Performance Statistics Overlay check box. A statistics display is added to the window.
4. Next to Collect Stats, select the Tracy check box.
5. To run Tracy Client, from the Start menu, choose **MAK Technologies > VR-Engage 2.2 Tools**. In the Tools folder, double-click the Tracy shortcut.
6. Verify the client address and then click Connect. The Tracy Client displays the information from VR-Engage.
7. To save the file, click Connection ( ). The Connection dialog opens.

8. Click Save Trace.
9. Save the file.



You can also open the Performance Statistics Overlay by pressing **F2** twice and then selecting the Tracy check box next to Collect Stats. (Note that, if you do this while the Choose Role Panel is visible, it might be hidden behind that panel. Simply move the overlay.)

4.3. Configuring Graphics Quality

VR-Engage has command-line and display settings options that allow you to configure graphics quality. In particular, depending on the capabilities of your graphics card, you may need to adjust the default settings for anti-aliasing using `--antiAliasing` *`level`*. Anti-aliasing is a technique used to reduce the jagged edges of digital graphics. You can set the level to 0, 2, 4, 8, or 16. Default: 4. On older graphics cards, you may need to set anti-aliasing to 0 to improve performance.

You can also make adjustments to the horizon-based ambient occlusion. See "4.3.3 Configuring Ambient Occlusion" on page 125.

4.3.1 Alpha to Coverage

Alpha to coverage is a way of handling objects with transparency that improves rendering of objects that have a sharp transition between opaque and transparent. Grass, trees, and fences are a few examples of when alpha to coverage is a good option, as shown in Figure 34. In addition to improving visualization of some objects, there is a slight performance improvement when alpha to coverage is enabled.

Figure 34. Alpha to coverage - on and off



You can set alpha to coverage for models and terrains. For models, you can enable it globally or in specific model definitions. For terrains, you can enable it globally or for specific terrain patches.

By default, alpha to coverage is enabled globally for models and terrains.

If alpha to coverage is enabled, you can enable a dither shader that modifies the alpha with a noise pattern so that there are no hard edges between different alpha levels. The tradeoff for this setting is that it adds a small amount of noise as the object or camera moves.

VR-Engage uses render bin 1 for the alpha to coverage render bin. You can change the render bin with the VRV_ALPHA_TO_COVERAGE_RENDERBIN environment variable. (Render bins are an OpenSceneGraph concept that affects the order in which scene objects are rendered. Do not set the render bin environment variable unless instructed to do so by a plug-in developer.)

Enabling Alpha to Coverage for Models

You can enable alpha to coverage globally for models and you can specify it for specific models. When you enable alpha to coverage globally, it affects all models that do not have it specified specifically in their model definition.

To enable or disable alpha to coverage for a specific model, add the alphaToCoverageEnabled parameter to the model definition and set it to true or false. For information about editing model definitions, see "Editing a Model Definition" in the *Adding Content to MAK ONE Applications Reference Manual*.

➤ **To enable or disable alpha to coverage globally for models:**

1. Ensure that you are in Admin mode (**Shift+F12**).
2. Choose **Settings > Display**. The Display Settings dialog box opens.
3. Select the Render Settings page (Figure 33 on page 121).
4. Select or clear the Models Alpha to Coverage check box.

Enabling Alpha to Coverage for Terrains

You can enable alpha to coverage globally for terrains or for specific terrain patches. The global setting affects all terrain patches that do not have an alpha to coverage option specified in their MTF file.

You enable alpha to coverage for a terrain patch when you add it. For details, see "Adding Elevation Data to a Terrain" in the *Adding Content to MAK ONE Applications Reference Manual*.

➤ **To enable or disable alpha to coverage globally for terrains:**

1. Ensure that you are in Admin mode.
2. Choose **Settings > Display**. The Display Settings dialog box opens.
3. Select the Render Settings page (Figure 33 on page 121).
4. Select or clear the Terrain Alpha to Coverage check box.

Enabling the Dither Shader for Alpha to Coverage

If alpha to coverage is enabled, you can enable a dither shader that modifies the alpha with a noise pattern so that there are no hard edges between different alpha levels. The trade-off for this setting is that it adds a small amount of noise as the object or camera moves.

➤ **To enable or disable the dither shader for alpha to coverage:**

1. Ensure that you are in Admin mode.
2. Choose **Settings > Display**. The Display Settings dialog box opens.
3. Select the Render Settings page (Figure 33 on page 121).
4. Select or clear the Alpha to Coverage Dither check box.

4.3.2 Using the Ambient Occlusion Effect

Horizon-based ambient occlusion (HBAO) adds screen space ambient occlusion lighting effects to the scene. This improves the realism by approximating the amount of ambient light that impacts a scene, which provides more realistic lighting of three-dimensional objects.

HBAO is disabled in Figure 35 and enabled in Figure 36. The differences are subtle. Some differences you might notice are the light inside the building as seen through the window on the right side of that building, the lighting on the food in the stands, and the shadows on the wall of the building behind the stands and grass.

Figure 35. HBAO disabled



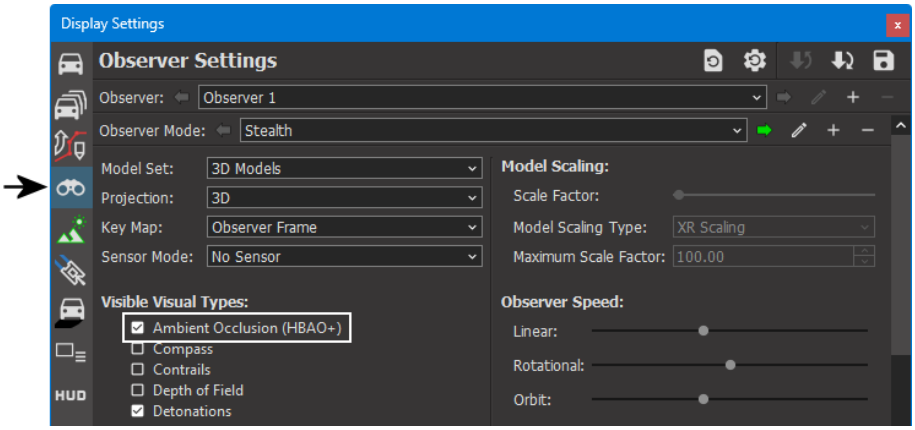
Figure 36. HBAO enabled



➤ **To enable or disable HBAO:**

1. Ensure that you are in Admin mode (**Shift+F12**).
2. Choose **Settings > Display**. The Display Settings dialog box opens.
3. Select the Observer Settings page (Figure 37).

Figure 37. Observer Settings page



4. Select the 3D observer mode for which you want to enable HBAO.
5. Select or clear the Ambient Occlusion (HBAO+) check box.

4.3.3 Configuring Ambient Occlusion

There are parameters that adjust how ambient occlusion appears.

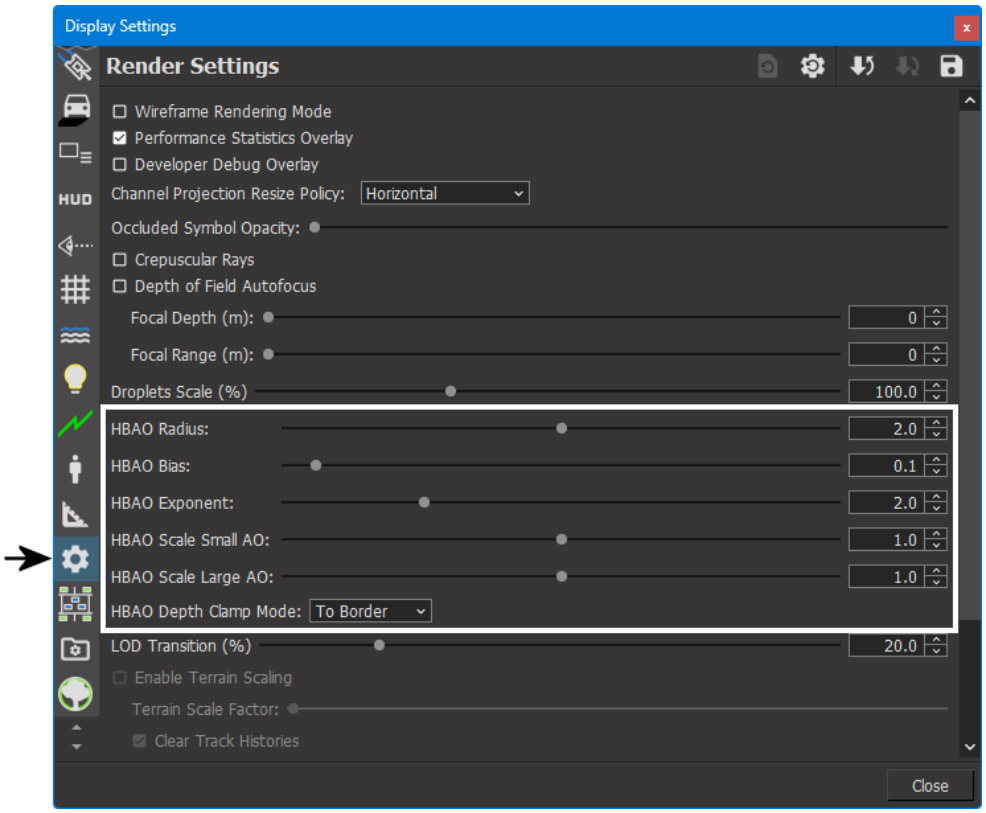
- HBAO Radius. The distance from occluding objects where the HBAO effect is visible. Higher values cause the effect to extend further from occluding objects.

- HBAO Bias. The bias for thin items. At higher values, larger objects cause the most noticeable effect on surrounding objects. This can reduce polygon edge artifacts and artifacts caused by thin geometry.
- HBAO Exponent. The intensity of the HBAO effect. Higher values create a more intense effect.
- HBAO Scale Small AO. The intensity of the HBAO effect on objects that are close to each other. Higher values have a more intense effect on the small corner areas of objects and also objects that are close together.
- HBAO Scale Large AO. The intensity of the HBAO effect on objects that are at the edges of the specified HBAO Radius. Higher values make the area for objects that are around the HBAO radius distance from each other show a more intense effect.
- HBAO Depth Clamp Mode. The depth clamp mode to use. To Border uses a method that reduces the false occlusion of objects along the edge of the screen. To Edge uses a method that correctly applies the occlusion effect to objects along the edge of the screen.

➤ **To configure HBAO:**

1. Ensure that you are in Admin mode (**Shift+F12**).
2. Choose **Settings > Display**. The Display Settings dialog box opens.
3. Select the Render Settings page (Figure 38).
4. Adjust the HBAO settings as needed.

Figure 38. Render Settings page



To improve performance, VR-Engage disables ambient occlusion (HBAO+) by default when you use a virtual reality device (HMD).

4.4. DI-Guy Performance Settings

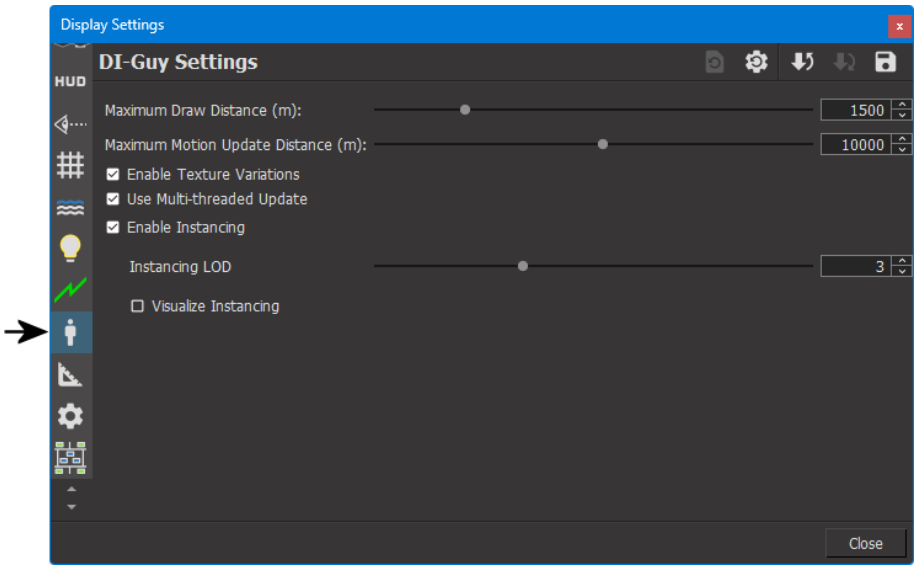
These settings affect performance of DI-Guy human characters:

- Maximum draw distance. VR-Engage will not render characters that are farther away from the observer than this distance.
- Maximum motion update distance. The distance at which VR-Engage stops animating DI-Guys unless they are changing posture.
- Use multi-threaded update. Developers who are debugging DI-Guy may want to disable multi-threaded update to make it easier to debug.
- Enable instancing. Enabling instancing improves performance by reusing model data for similar characters. You can set the LOD beyond which instancing is used. You can visualize instanced characters.

➤ **To specify DI-Guy performance settings:**

1. Ensure that you are in Admin mode (**Shift+F12**).
2. Choose **Settings > Display**. The Display Settings dialog box opens.
3. Select the DI-Guy Settings page (Figure 39).

Figure 39. DI-Guy Settings page



4. To set the maximum draw distance, adjust the Max Draw Distance slider or change the value in the box.
5. To change the distance at which DI-Guys stop animating, adjust the Maximum Motion Update Distance slider or change the value.

6. Optionally, select or clear the Use Multi-threaded Update check box.
7. Optionally, select or clear the Enable Instancing check box.
8. If instancing is enabled, adjust the Instancing LOD slider or change the value in the box.
9. Optionally, select or clear the Visualize Instancing check box.

4.5. Specifying Terrain Categories

To optimize terrain intersections that VR-Engage performs for different purposes and for different entity types, you can adjust the terrain categories on a per-platform (OPE file) basis. Note that this applies to only CGF-controlled entities.

Each platform file specifies **los-terrain-category-mask** and **movement-terrain-category-mask**. The value for these parameters can be specified to be any combination of:

- TerrainSkin
- Structure
- Vegetation

Any category of terrain not included in the mask will potentially be ignored for that type of intersection by entities using the platform file in question. Note that some terrain data sources do not specify a category, which means that they are always included because they have no category that VR-Engage can recognize.

You should specify the minimum categories for each platform type that are needed for your scenario and expected behavior. For example, the default VR-Forces entity-level SMS does not include vegetation for the **movement-terrain-category-mask** for humans because it is not expected that humans will walk on trees. This allows VR-Engage to avoid paging in vegetation data near humans as they move.

Similarly, the default configuration for fixed-wing aircraft platforms includes only TerrainSkin for both line-of-sight (LOS) and movement masks. Because these are typically fast-moving entities with long range sensors, terrain intersection checks and terrain paging can be quite costly for them. The entity-level SMS that ships with VR-Engage assumes that these entities do not need to land or taxi on buildings or trees, therefore these are not in the movement mask. VR-Engage also assumes that buildings and trees will not be primary blocks for LOS tests performed by fixed-wing aircraft entities. While this choice makes the terrain paging much faster, there may be scenarios where buildings do need to block the LOS for their sensors and therefore it should be added.

For more information, please refer to the *VR-Forces Developers Guide*.

4.6. Using the Developer Debug Overlay

VR-Engage uses the ImGui API to create an overlay that developers can use to test changes to some of its visual features. VR-Engage includes an overlay for testing ocean settings, shadow settings, and other visual features. It also includes an ImGui test user interface that demonstrates some of its features.

Figure 40. Developer debug overlay



This feature is provided "as is" for the use of interested developers. For complete information about how to use ImGui, please see documentation at https://github.com/ocornut/imgui and other internet resources.

➤ **To enable the developer debug overlay:**

1. Ensure that you are in Admin mode (**Shift+F12**).
2. Choose **Settings > Display**. The Display Settings dialog box opens.
3. Select the Render Settings page (Figure 33 on page 121).
4. Select the Developer Debug Overlay check box. The Developer Debug Overlay opens. It has menus and options for Ocean Settings, SpeedTree Settings, osgEarth Network Monitor, and so on.

   You can also open the developer debug overlay by pressing the **F4** key.

5. To view shadow settings:
   a. In the Display Settings dialog box, select the Shadow Settings page.
   b. Select the Enable Shadow check box.

4.7. Using the VR-Engage Debug Menu

The VR-Engage Debug Menu is a general purpose diagnostic window for VR-Engage entities, providing insight into the current player state at runtime.

➤ To open the VR-Engage Debug Menu, press **Shift+F4**.

Figure 41. VR-Engage Debug Menu





II. Player Features

The chapters in this section explain how you can engage and participate in a scenario in the role of a human character; a vehicle driver, gunner or commander; a sensor operator; a drone operator; or the pilot of an airplane or helicopter.



5. Common Player and Entity Features

This chapter describes actions that are common to all types of entities in a scenario.

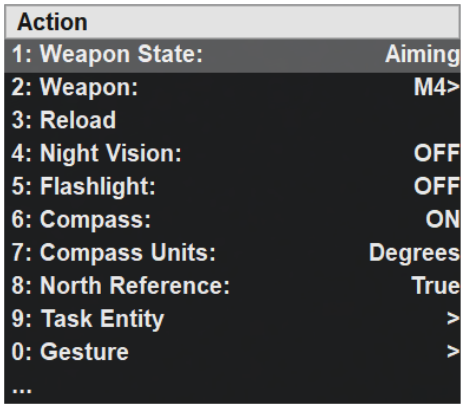
5.1. The Action Menu 134
5.1.1 Selecting Options on the Action Menu 134
5.2. Using the Radio 135
5.2.1 Using MAK Radio in VR-Forces 136
5.2.2 Using the Standalone Radio Application 136
5.2.3 Using the Intercom or a VOX Radio 138
5.2.4 Configuring the MAK Radio 138
5.2.5 Audio Settings 141
5.2.6 Jamming Radios 142
5.2.7 Enabling the Radio by Default 142
5.3. Text Chat 142
5.3.1 VR-Engage Text Chat Commands 143
5.3.2 Using the Text Chat in VR-Forces 144
5.3.3 Configuring Text Chat Channels for a Scenario 145
5.4. Dynamic Terrain Effects on Movement 147
5.5. Changing the View Mode 147
5.6. Starting VR-Forces Events 147
5.7. Introduction to Sound Effects 148
5.7.1 Enabling Sound Effects 148
5.7.2 Configuring Sound Effects 149
5.7.3 Decibels Configuration 150
5.7.4 Associating Sound Files with Entities and Events 151
5.8. Resetting Entities 151
5.9. HMD Controls 152
5.10. Saved Observer Views 152
5.10.1 Selecting a View 152
5.10.2 Transitioning to a View 153

5.10.3 Resetting the View 153

5.1. The Action Menu

The Action menu lets you turn several features on and off (toggles) and lets you assign tasks to other entities if you are hosting a VR-Forces scenario or connected to a VR-Forces session. The Action menu is context-sensitive.

Figure 42. Action menu



➤ **To open the Action menu, you can either:**

- Press **~**.
- Press the menu button on your game controller or head-mounted display (HMD) controller (the Back button on game controllers).

The Oculus Rift, HTC Vive, and Varjo HMDs configure the Action menu differently. If the Action menu is displayed upside down, you must change the configuration. For details, see "3.10.6 Configuration Options for Virtual and Mixed Reality" on page 81.

5.1.1 Selecting Options on the Action Menu

You can choose options on the Action menu using the keyboard (arrow keys or number keys), the D-pad on a game controller or HMD controller, or your mouse.

➤ **To choose an option on the Action menu using the arrow keys or D-pad:**

1. Press the **Up arrow** and **Down arrow** keys on your keyboard or the up and down buttons on a game controller's D-pad to highlight an option.
2. Press the **Right arrow** key or the right button on the D-pad to toggle the option on or off or to display a submenu.

When you choose an option with the mouse, if it is a toggle, the state changes immediately. If it displays a submenu, the menu changes immediately. If it is a task, the entity executes that task immediately. Therefore, if a task requires input, such as a location, do not use the mouse to choose it. Use the arrow keys.

➤ To choose an option on the Action menu using a mouse, click the menu option.

Each option on the Action menu has a number. When you choose an option with a number key, if it is a toggle, the state changes immediately. If it displays a submenu, the menu changes immediately. If it is a task, the entity executes that task immediately. Therefore, if a task requires input, such as a location, do not use a number key to choose it. Use the arrow keys.

> Numbers are reassigned dynamically if you scroll the Action menu or menu commands are removed due to some change in the state of the character. Therefore, you cannot assume that a particular number always controls the same command.

➤ To choose an option on the Action menu using a number key, press the number associated with the menu option.

5.2. Using the Radio

VR-Engage supports voice communications over radios and intercoms. You can communicate with anyone using the radio application that is in the same exercise. There are 2 radios, each of which has 2 channels. (The radios are configurable. For details, see "5.2.4 Configuring the MAK Radio" on page 138.) Two applications can communicate only when the radios are tuned to the same frequency and that frequency is not being jammed.

Ground vehicles have an intercom, which has one channel. The intercom allows the crew to communicate among themselves. For information about the intercom, see "5.2.3 Using the Intercom or a VOX Radio" on page 138.

Audio data from the radio system is sent as signal interactions/PDUs on the simulation network. It can be received in other applications that can receive standard signal interactions/PDUs.

Radio communications are perfect. Radio signals are not affected by terrain, weather, distance, or other factors. However, you can use the VR-Forces radio panel or the standalone radio application included with VR-Engage to jam radios. For more information about jamming, see "5.2.6 Jamming Radios" on page 142.

The radio uses the audio devices configured in the Audio Settings dialog box for capture and playback. See "5.7.2 Configuring Sound Effects" on page 149.

➤ **To use the radio:**

1. Open the Action menu.
2. Choose **Radio** to toggle it to one of the available radios, for example, Radio 1, Radio 2, or Intercom (if one is configured).
3. When you select a radio, the channel, frequency, and other settings display. You can toggle those settings using Action menu commands.
4. To talk, hold down the **T** key or, if you are controlling a human character and using a game controller, press the A button. To hear others, release the key or button.

- You cannot hear other people talking while you are pressing the talk key or button.
- By default, radios are disabled and must be turned on manually. You can configure the roles to default to enabling the radios when a player engages in that role. See "5.2.7 Enabling the Radio by Default" on page 142.

5.2.1 Using MAK Radio in VR-Forces

The MAK Radio feature in VR-Forces allows VR-Forces users to use the radio to talk to VR-Engage players or anyone running the radio application.

To learn more about using the MAK Radio in VR-Forces, please refer to the *VR-Forces User Guide* or help.

5.2.2 Using the Standalone Radio Application

VR-Engage includes a standalone radio application. It lets you communicate with VR-Engage players without running VR-Engage. The application has different versions for the protocols supported by VR-Engage.

The default key for Push to Talk is **T**, but you can specify other keys in the Audio Settings. See "5.2.5 Audio Settings" on page 141.

➤ **To use the VR-Engage standalone radio:**

1. In a console window or File Explorer, change to *./bin64*.
2. Run the executable for the protocol you are using:

   DIS:

   `makRadioAppDIS.exe`

   HLA Evolved:

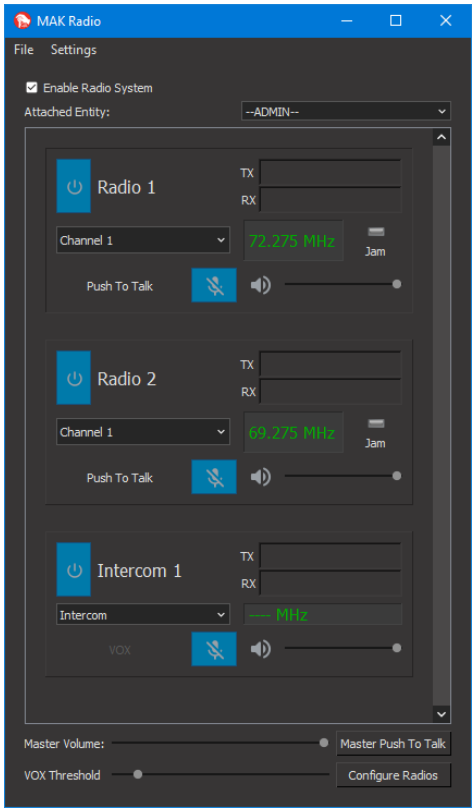
   `makRadioAppHLA1516e.exe`

   HLA 4:

   `makRadioAppHLA4.exe`

   MAK Radio opens (Figure 43).

Figure 43. MAK Radio



3. Optionally, adjust the Master Volume slider to adjust the volume of all radios.
4. Optionally, adjust the VOX Threshold slider to adjust the level of sound needed to enable voice activated transmission. (This is usually used for intercoms, not radios.)
5. Optionally, click Master Push to Talk to transmit on all radios that are not muted.
6. To operate a radio head:
    a. Make sure the radio head power button is enabled ( ).
    b. Optionally, select a channel from the list.
    c. Click Push to Talk or press **T** (or other key) to talk (if this is not a Vox radio).
    d. Click the microphone button ( ) to mute the microphone or enable it.
    e. Click the speaker button ( ) to mute the speaker or enable it.
    f. Adjust the slider to change the radio's volume.
    g. Click the Jam button ( ) to jam the radio frequency.
7. Click Configure Radios to edit the radio configuration. See "5.2.4 Configuring the MAK Radio" on the next page for details.
8. Optionally, choose **Settings > Audio Settings** to select the Playback Device (speaker) and Recording Device (microphone).

5.2.3 Using the Intercom or a VOX Radio

The intercom and VOX radios are voice activated and transmit any sound greater than the VOX threshold.

➤ **To use the intercom or a Vox radio:**

1. In the MAK Radio Application, select the Attached Entity.
2. In the application, make sure the power button is enabled ( ).
3. Make sure that the microphone is not muted.
4. If you are using a Vox radio, you can select a channel.
5. Start talking.

5.2.4 Configuring the MAK Radio

You can configure the number of radios and channels, the channel frequencies, the sample rate, and the format. The Modulation and Crypto settings for radios are as defined for the Transmitter PDU in the IEEE DIS standard.

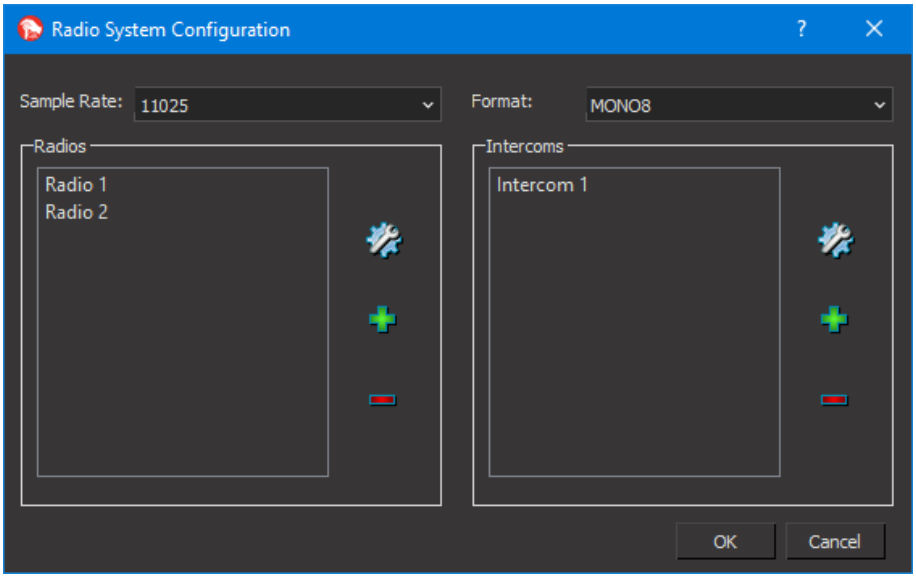
If there is a mismatch between two or more radios in the Transmission, Modulation, and Crypto settings, they cannot talk to each other. The settings must match between communicating radios for the signals to be processed and the transmissions to be heard.

Changes that you make persist from one session of VR-Engage and MAK Radio to another, and the settings are the same for both.

➤ **To configure the MAK Radios:**

1. In the MAK Radio application, click Configure Radios. The Radio System Configuration dialog box opens (Figure 44).

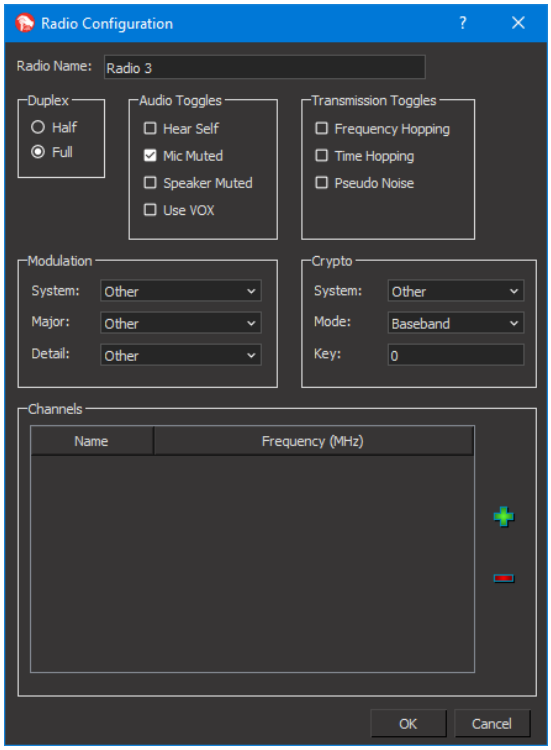
Figure 44. Radio System Configuration



2. To change the sample rate for all radios, select the rate from the list.

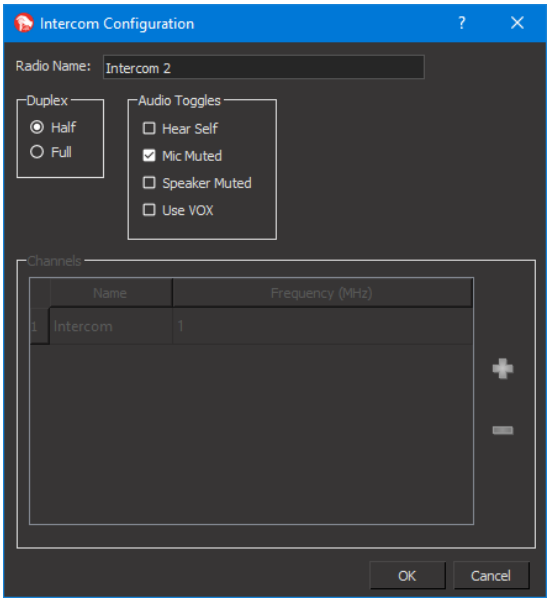
3. To change the format for all radios, select the format from the list.
4. To add or edit a radio:
    a. Click the Add Radio ( ) or Edit button ( ) next to the list of radios. The Radio Configuration dialog box opens.
    b. Configure the radio as needed. Table 6 on the next page describes the settings.

Figure 45. Radio Configuration



    c. Click OK.
5. To add or edit an intercom:
    a. To add an intercom, click the Add Intercom ( ). A row is added.
    b. To edit a new or existing intercom, select the row and then click the Edit button ( ) next to the intercom list. The Intercom Configuration dialog box opens (Figure 46).
    c. If desired, edit the name of the intercom.
    d. Configure the intercom as needed. Table 6 provides information about the settings. Note that only Duplex and Audio Toggles settings are available for intercoms.

Figure 46. Intercom Configuration



e. Click OK.

6. When you are finished, click OK in the Radio System Configuration dialog box. You return to the Radio Communications dialog box. If you added or removed a radio or intercom, the list updates.

Table 6. Radio and Intercom settings

Setting	Options
Radios and Intercoms	
Duplex	— Half. Only one user can talk at a time. — Full. All users can speak simultaneously.
Audio Toggles	— Hear Self. Users can hear themselves over the speaker. — Mic Muted. Microphone defaults to muted. — Speaker Muted. Speaker defaults to muted. — Use Vox. Radio is voice-activated.
Radios only	
Transmission Toggles	— Frequency Hopping. Switches frequencies during transmission to reduce interference and avoid interception. — Time Hopping. Varies the period and duty cycle to achieve anti-jamming or low probability of intercept. — Pseudo Noise. Introduces a deterministically generated random sequence of bits.

Table 6. Radio and Intercom settings (continued)

Setting	Options
Modulation	— System. Radio system type: Generic, SINGCGARS, and so on. — Major. Modulation type: Amplitude, Pulse, and so on. — Detail. Additional information about the modulation type (options based on the Major setting).
Crypto	— System. Cryptographic system for the radio: VINSON, KY-100, and so on. — Mode. Crypto mode: Baseband, Diphase, and so on. — Key. A simulated cryptographic key. This key is a non-zero integer value for simulating secured communication. Set to 0 when simulating plain communications.
Channels	— Use the Add or Delete buttons to manage the radio's channels. — In the rows, double-click to edit each channel's name and frequency as needed.

5.2.5 Audio Settings

In the MAK Radio standalone application, you can access the Audio Settings. These settings allow you to specify the Playback Device (speaker), Recording Device (microphone), and one or more push-to-talk buttons.

The options are saved in *./appData/settings/makRadio/radioAudioSettings.csv*. You can copy the CSV file to other workstations that should have the same settings.

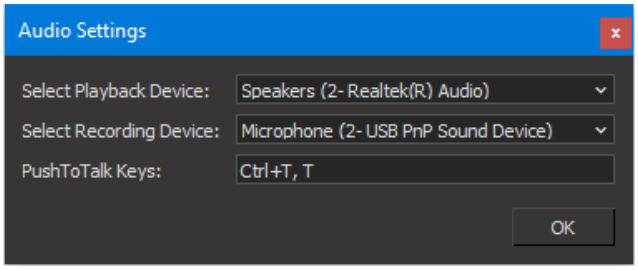
➤ **To choose a Playback Device or Recording Device:**

1. In the MAK Radio standalone application, choose **Settings > Audio Settings**. The Audio Settings dialog box opens.
2. Choose the speaker or microphone from the appropriate list.
3. Click OK.

➤ **To set the default push-to-talk key:**

1. If the Audio Settings dialog box is not open, choose **Settings > Audio Settings**.
2. Click on the Push To Talk Keys box to give it focus.
3. Press the desired key or key combination. If you want to use more than one key, continue pressing keys. Each key or key combination is separated by a comma (Figure 47).
4. Click OK.

Figure 47. Audio Settings for MAK Radio



5.2.6 Jamming Radios

The MAK Radio Application allows you to jam radio frequencies. When you click the Jam button on a radio head, it creates another radio that jams the frequency of the currently selected channel. The users will not be able to hear the radio communications, and will not see an indication that their radio is jammed.

➤ To jam a radio frequency, click the Jam button (Jam).

5.2.7 Enabling the Radio by Default

You can add the `defaultRadio` parameter in the *./data/simulationModelSets/VR-Engage/roles/devices/radio.*[*role*]*.lua* files, where [*role*] is the name of the role. You can enter the name of a configured radio, such as Radio 1, to automatically enable the radio for entities who engage in that role.

The information goes inside the `components` block. This is an example of the *radio.human.lua* file set to enable Radio 1:

```
components={
  ["radioLogic"] = {
    componentType = "DtRadioLogic";
    ...
  };

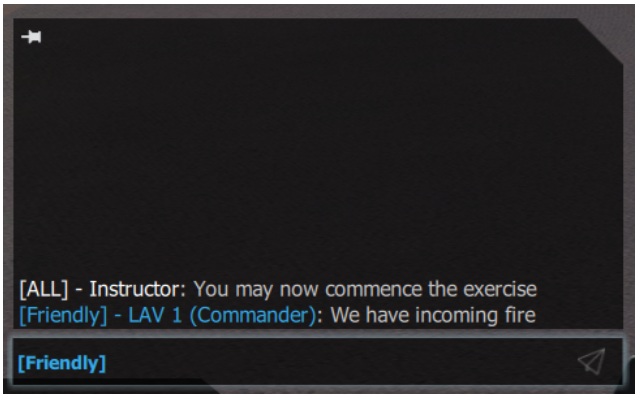
    defaultRadio="Radio 1";

    radios={
      ...
    }
```

5.3. Text Chat

The text chat feature enables players to communicate with each other, and also allows an instructor to communicate with the players. The chat messages can represent communication that takes place in the simulated world, and they can also be administrative.

Figure 48. Chat window in VR-Engage



5.3.1 VR-Engage Text Chat Commands

The chat window displays messages sent by other users in the scenario that the player can access, such as those sent to channels configured for the same force or for all entities. The player can also send messages to one of the channels available to them.

If someone sends a message when the chat window is not open, the message displays briefly and then fades. You can choose to pin the chat window open using the pin icon. This causes the window to persist when you remove focus from the chat window.

In Table 7, commands in bold sans-serif font are key presses and commands in monospace font are those that you enter in the message field and then send to get the results.

Table 7. Text chat commands

Command	Description
/	Opens the chat window and gives it focus, allowing the player to send messages and see available messages.
Alt+**/**	Toggles the pin for the chat window.
Enter	After typing your message, you can press **Enter** or click the send icon ().
Esc	Removes focus from the chat window, allowing you to use the keyboard to interact with the application. (If it is pinned, it remains open instead of fading away until the next message.)
`/username` *name*	Changes the name that displays in the chat window. All users can see this name in their chat windows. The default name when the player is not engaged in a role is the computer name. When the player is in a role, the default is the entity name and, if it has a role such as driver or pilot, then that is included in parentheses. If you enter a custom name, that displays.
`/username clear`	Resets the username. It uses the defaults described above.
`/list`	Lists all available channels.

Table 7. Text chat commands (continued)

Command	Description
`#ChannelName`	Switches the chat window to the specified channel.
`#ChannelName` *`message`*	Sends a message to the specified channel, but does not change the current channel.
Tab	Cycles through the channels available to the user. While in a role, this is limited to the universal All channel and any channels available to the force of the entity they are playing. In the Choose Role screen, the player can access other channels. Force-specific channels are colored for that force.
`/clear`	Deletes all of the messages from your chat history.
`/help`	Displays a list of the available chat commands, along with a brief description of what each command does.

If there are no custom channels configured for the scenario, the only channels that are available are the All channels for the forces in the scenario. The `/list` command shows only "All" for each force available to the player. Use the **Tab** key to cycle through them to find the force color to which you want send your message.

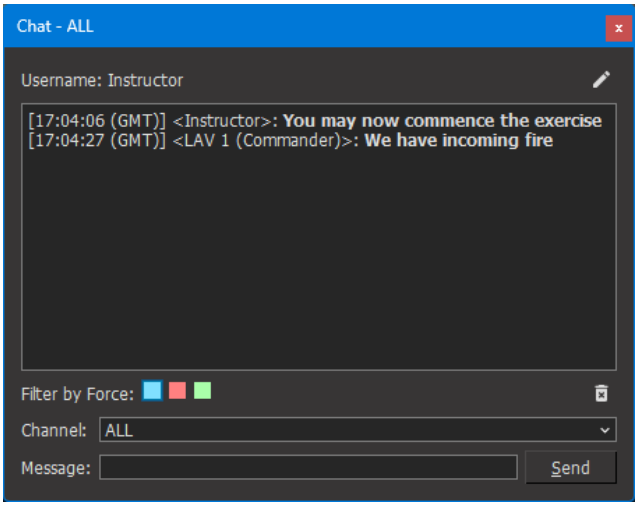
5.3.2 Using the Text Chat in VR-Forces

The text chat feature allows instructors and players to communicate with each other. The timestamp shows the current time on the computer; it is unrelated to the simulation time.

- To change the name that displays in the chat window, use the Change Username button (). In Figure 49, the user changed their name to "Instructor". All users can see this name in their chat windows.
- To see the channels specific to one or more forces, select the force color next to Filter by Force. You can select multiple forces to access those channels.
- To see messages sent to a particular channel, choose a channel from the list. The default channels are All, which displays messages sent to any channel and is visible to all users, and channels for any selected forces, which include all users who are participating in that force. There might be channels defined for a scenario, which override the default channels as described in "5.3.3 Configuring Text Chat Channels for a Scenario" on the facing page.
- To remove all messages from the currently selected channel, click the Clear Chat button ().

  Note that All messages are cleared from the window when you rewind a scenario.

Figure 49. Chat window in VR-Forces



➤ To open a chat window, choose **View > New Chat Window**. You can open multiple windows if you want to have separate ones for different channels.

➤ To send a message, select a channel, type in the Message field, and then click Send.

5.3.3 Configuring Text Chat Channels for a Scenario

The default for channels in the chat window is to have one for each force and one for all participants. You can create a file for the scenario with channels specific to the scenario's purpose, such as channels for specific entities. For example, you might want channels for the members of each unit in the scenario, or for a particular task, such as a pilot and a JTAC.

The VR-Engage installer and the VR-Engage extensions installer for VR-Forces include an example configuration file for the Test Drive scenario, *./userData/chatConfigs/VR-Engage_Test_Drive_ChatConfig.mtl*. You can use it as a guideline for creating a configuration file for your scenario. There is also an example below for a different scenario.

- You must save a copy of the file in the *./userData/chatConfigs* folder for each installation that will use the channels. This is where VR-Forces and VR-Engage look when loading the scenario file. Additionally, the file name must be *scenario-file-name_ChatConfig.mtl*, where *scenario-file-name* is the exact name of the scenario file. The channels defined in those files are available to VR-Forces and VR-Engage users.
- You can configure channels that are specific to a force using its ID. The force IDs are defined in the Forces Editor, which is in the VR-Forces Simulation Object Editor. (Refer to the VR-Forces user documentation to learn about that tool.) The default values are 0 for all forces, 1 for friendly, 2 for opposing, and 3 for neutral. Only users in a force can see messages sent to a channel configured for that force. If VR-Forces is configured to hide ground truth for a force (in Spot Report Settings), the VR-Forces user cannot access chats specified for that force. Setting the force for a channel also determines the color of the channel name text displayed in VR-Engage.

A scenario-specific configuration file overrides the default channels. If you want force-specific channels, or one for all participants, you must define them in your configuration file. We did so in the example below.

- Channels must include one or more entities using entity names in the scenario for VR-Engage entities to be able to use the channel. (You cannot use the unit name to include all entities aggregated into that unit; you must use the individual entity names.) To include all entities, use the * wildcard. In the example below, the ALL channel includes all entities in the scenario, and the Blue Force and Red Force channels include all entities in the friendly and opposing forces, respectively.

This example is a configuration file for the VR-Engage_First_Experience scenario. It has a channel for only the paratroopers who start on the CH-47 Chinook CargRW 5, as well all the roles on that aircraft. (We also specified a force for that channel because, while only the specified entities can access that channel, we wanted VR-Forces users to be able to filter it by force and for the channel name to display in blue font in VR-Engage.) It is a partial example; a real configuration file would have channels for the other groups in the scenario we planned for users to play in the simulation.

```
;; The filename is VR-Engage_First_Experience_ChatConfig.mtl
(chat-config
  (channels
    (channel
      (channel-name "CargoRW 5 Crew")
      (force 1)
      (entities
        (entity "Para 1")
        (entity "Para 2")
        (entity "Para 3")
        (entity "Para 4")
        (entity "CargRW 5")
      )
    )
    (channel
      (channel-name "ALL")
      (force 0)
      (entities
        (entity "*")
       )
    )
    (channel
      (channel-name "Blue Force")
      (force 1)
      (entities
        (entity "*")
       )
    )
    (channel
      (channel-name "Red Force")
```

```
        (force 2)
        (entities
          (entity "*")
         )
      )
    )
  )
```

5.4. Dynamic Terrain Effects on Movement

Dynamic terrain events cause changes to parts of the terrain, such as explosions that create craters in the ground. Those changes affect entities moving across that area of the terrain, impeding and possibly preventing their movement. For example, a human character or a ground vehicle can hide in a ditch, go over a berm, or become trapped in a crater.

5.5. Changing the View Mode

By default, when you assume a role, you are in first-person view mode. That is, the eyepoint is placed as if you were looking out of the eyes of a human character or from the appropriate position of a character in a vehicle, such as the driver or pilot. Using the keyboard, you can switch to third-person view so that you can see the character you are playing in its context in the terrain. While driving a vehicle, you can have up to six views of the vehicle from different angles available if they are configured.

➤ **To change the view mode, do one of the following:**

- For human characters and aircraft, press the **V** key.
- For ground vehicles (typically only the Driver role), on the keyboard, press **V** or a number **1** through **6**. If you press a number for which no view is defined, then the view does not change.

  You can also use the Y button on a game controller.

ⓘ Ground vehicle views are defined for the vehicle and role. For information, see "Chapter 15. Model Customization Tutorial" on page 361.

5.6. Starting VR-Forces Events

VR-Forces scenarios can include scenario events, which can consist of text, audio, video, graphics, or some combination of these. A scenario event represents an incident or condition that exercise participants can respond to, for example a political event, a natural disaster, or some other novel event that might require a participant to respond. Events can also be used to satisfy the conditions in a trigger.

If you are in Admin mode, are controlling a character in a VR-Forces scenario, and the scenario has scenario events, you can trigger them. The scenario events are displayed in

the VR-Forces GUI. They are not displayed in the VR-Engage window. If VR-Engage is hosting a scenario (and there are no VR-Forces GUIs in the exercise), the events are listed on the Action menu, but triggering them may have no visible effect.

The Action menu lists all of the events that have been created for the scenario, but does not indicate their status. A VR-Engage character can only start an event. If an event is active, you cannot stop it from VR-Engage.

➤ **To start a scenario event:**

1. Be sure that you are in Admin mode (**Shift+F12**).
2. Open the Action menu.
3. Choose **Trigger Event**.
4. Choose the event you want to trigger.

5.7. Introduction to Sound Effects

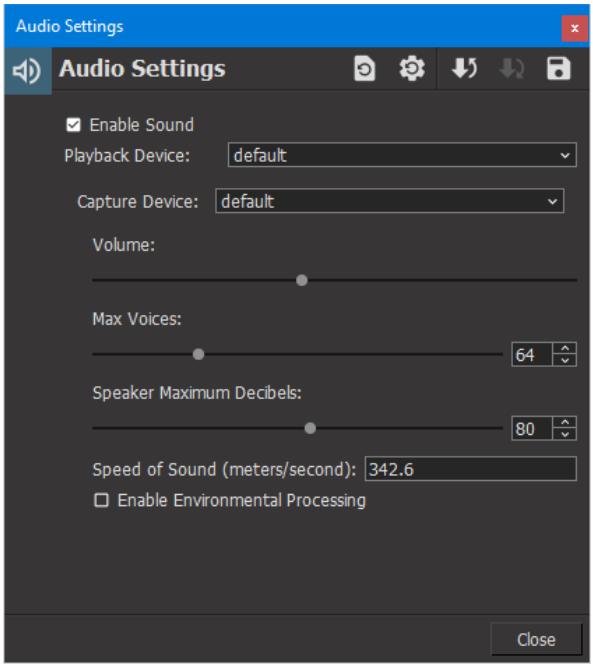
VR-Engage can associate sound effects with entities, fire events, and detonation events. You can enable and disable use of sound effects and you can specify that the player will hear sounds only when their character is within a certain distance of the entity or event that generates them.

5.7.1 Enabling Sound Effects

➤ **To enable sound effects in Admin mode:**

1. Choose **Settings > Audio Settings**. The Audio Settings dialog box opens (Figure 50).

Figure 50. Audio Settings page



2. Select the Enable Sound check box.
3. Optionally, specify the playback device (speaker) and capture device (microphone).

5.7.2 Configuring Sound Effects

You can configure the sound settings in Admin mode.

- Playback Device. Select the speaker for the audio output.
- Capture Device. Select the microphone to use. This is typically for use with MAK Radio.
- Volume. This is the volume of the system.
- Max Voices. You can have from 1 to 256 voices, or maximum simultaneous playing sounds. Sounds that play at the same time are mixed into a single sound wave prior to being sent to the speaker. This mixing has a CPU cost. A high number of allowable voices can affect performance when there are many audio sounds trying to play.
- Speaker Maximum Decibels. Calibrates the audio system for the speakers that the system is played on. This affects how sounds at a distance fall off so that the sound loudness is realistically replicated. The default setting of 80 db represents headphones or a desktop PC's speakers. See "5.7.3 Decibels Configuration" on the next page for more information.
- Speed of Sound. This is the speed of sound in the atmosphere in meters/second. The default value is derived by using a standard day as parameters (20° C, 50% relative humidity, 1013.25 mbar/29.92 inHg). This affects the delay in playing sounds at a distance. For example, an explosion 2 km away has a several second delay from the time you can see the visual effect to when you can hear the audio effect.
- Enable Environmental Processing. Enables low pass filtering. As sound propagates through air, higher frequency sounds are more absorbed the further you move from the source of the sound. This setting replicates air absorption of a sound based on the distance the sound is from the listener.

➤ **To configure sound settings:**

1. Choose **Settings > Audio Settings**. The Audio Settings dialog box opens (Figure 50 on the previous page).
2. To select the speaker, choose a device from the Playback Device list.
3. To select the microphone, choose a device from the Capture Device list.
4. To set the volume, adjust the Volume slider.
5. To specify the maximum number of voices, adjust the Max Voices slider or change the value in the text box.
6. To set the speaker maximum decibels, adjust the Speaker Maximum Decibels slider or change the value in the text box.

7. To set the speed of sound, type a value in the Speed of Sound box.
8. To enable lowpass filtering, select the Enable Environmental Processing check box.

5.7.3 Decibels Configuration

As a feature, the VR-Engage audio system uses a sound decibel scaling system to compensate for the deployed sound systems such as headphones, computer speakers, and amplifiers. This helps provide a similar audio experience regardless of the system that you use.

You can configure each individual entity sound to match real world characteristics. For example:

- A whisper or rustling of leaves registers at 20 dB.
- A conversation at home is approximately 50 dB.
- A passenger car traveling on the freeway at 50 feet away is 70 dB.
- Inside a helicopter cabin 100 dB.

When configuring the decibels (dB) for a sound, you should select a decibel level that represents the level near the sound source. The dB setting should be the loudest level that you would ever want to hear the sound, which might differ from the real-life level. This allows the audio system to calibrate the sound levels automatically.

For each sound, based on the Speaker Maximum Decibels setting, the audio system computes a sound "reference" distance and "falloff" distance:

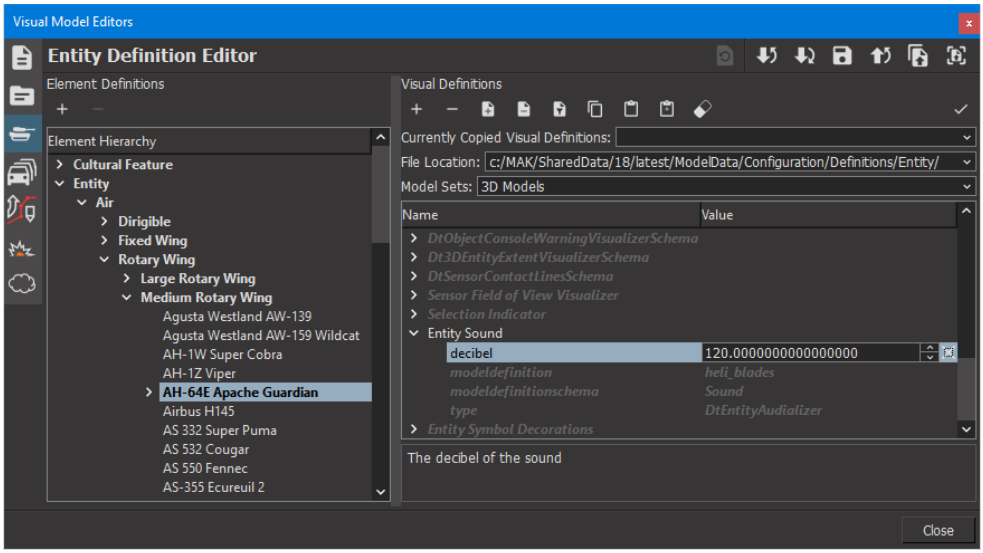
- The reference distance is the distance at which the sound is the loudest it can be heard.
- The falloff distance is the distance at which the sound can no longer be heard.

These examples show when a VR-Engage entity's sound is configured to be 80 dB and the observer is close to the entity:

- If the Speaker Maximum Decibels is set to 80 dB or lower, then the sound will be played and heard at its maximum volume.
- If the Speaker Maximum Decibels is set to 120 dB, then the sound will be played at a lower volume because the speaker system is expected to increase the heard volume to 80 dB.

Other sound effects settings are covered in "5.7.2 Configuring Sound Effects" on the previous page.

Figure 51. Example of entity decibel attribute in Entity Definition Editor



5.7.4 Associating Sound Files with Entities and Events

VR-Engage includes a set of sound files (WAV format) for generic entity platforms and for fire and detonation events. The files are in *./SharedData/19/latest/Audio/Sounds*. You add and edit mappings between entity types, fire events, or detonation events and sound files in the Visual Models Editor. Sound files are associated with model definitions. For example, the **filename** parameter for the heli_blades model definition specifies *$(SHARED_DATA_DIR)/Audio/Sounds/heli_blades.wav*. The Rotary Wing element definition has a parameter called **Entity Sound**. The value for the **model definition** attribute is **heli_blades**.

You can quickly see which model definitions are available for sounds by filtering the list of the Visual Model Editors, Model Definitions Editor page for the Sound schema.

5.8. Resetting Entities

In Admin mode, you can reset entities, which can be useful during tests or other situations. When you reset an entity, their health is restored, their resources are replenished, such as fuel and munitions, and, in the case of vehicles, they are set upright. You might want to reset the entity if you die, roll a vehicle, crash a plane, expend all of your ammunition, or are otherwise rendered unable to continue performing the desired task. To learn about Admin mode, see "3.1.3 Admin, User, and Lockdown Modes" on page 59.

There are two types of reset:

- Reset. Restores the entity and returns it to the location it had when you took control. If you are controlling a human entity that disembarked from a vehicle role, it resets to the location at which it disembarked.
- Reset in Place. Restore the entity at its current location.

> When you reset an entity to the location it had when you took control, none of the other entities in the scenario will reset. As a result, the location might no longer make sense in the context of the overall scenario.

➤ To reset an entity, press **F5** (Reset) or **Ctrl+F5** (Reset in Place).

You can also reset an entity using the Action menu.

➤ To reset an entity, open the Action menu and then choose **Reset** or **Reset in Place**.

5.9. HMD Controls

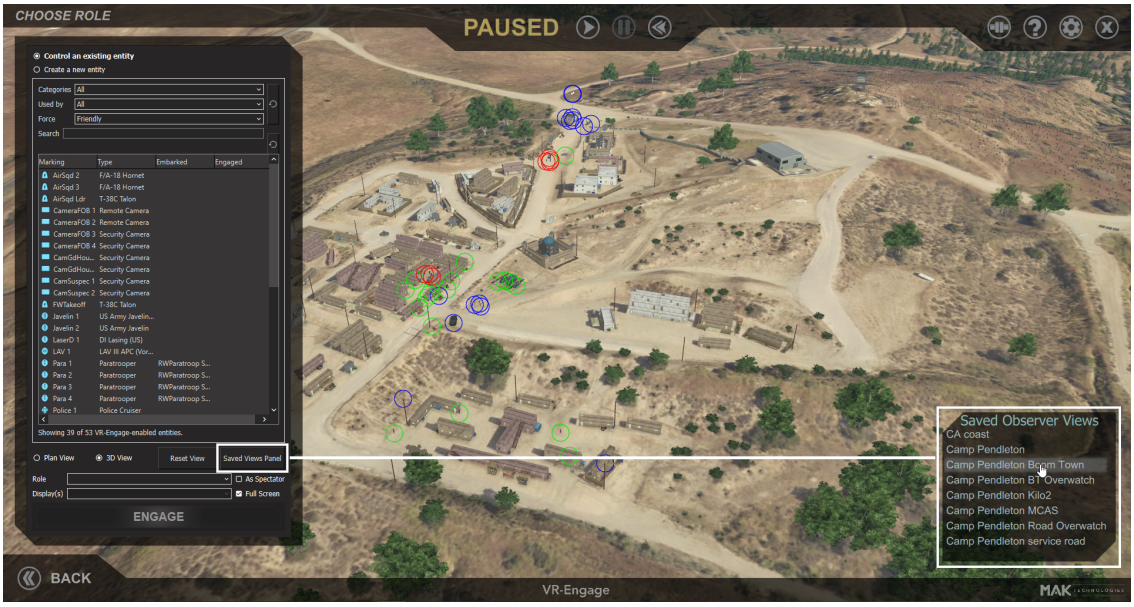
There are some minor differences for head-mounted displays.

- The Play and Pause controls are located in the Action menu.
- Pressing **Esc** returns you to the main menu and exits VR mode. You return to VR when you select Resume from the menu.

5.10. Saved Observer Views

At any given moment, the state of the observer is called its view. Most terrains included with VR-Engage include saved observer views. Saved views are listed in the Observer Views Panel, which you open by clicking Saved Views Panel in the Choose Role panel.

Figure 52. Saved Observer Views



5.10.1 Selecting a View

➤ To recall a view, select it in the Saved Observer Views panel. The view changes immediately.

5.10.2 Transitioning to a View

Normally, when you select an observer view, the observer jumps immediately to the new view. However, VR-Engage can smoothly transition from the current observer view to a saved view.

➤ To transition between observer views, hold down the **Ctrl** key and click the view in the Saved Observer Views panel that you want the observer to move to.

5.10.3 Resetting the View

You can change the observer position from the default using the standard observer keys and the mouse. You can reset to the default view for the selected observer view.

➤ To reset the view to the saved observer view, click Reset View.



6. Controlling People

This chapter describes how to control human characters and interact with the environment and other characters.

6.1. Introduction 156
6.1.1 Enabling and Disabling Mouse Control 156
6.2. Basic Movements 157
6.2.1 Moving Your Character Around the Terrain 157
6.2.2 Changing Your Character's Posture 157
6.2.3 Looking Around 157
6.2.4 Gesturing and Signaling 157
6.2.5 Climbing Ladders 158
6.3. Navigation 158
6.3.1 Using the Compass 158
6.3.2 Using the Minimap 159
6.4. Health and Stamina 160
6.4.1 Sustaining Damage to Health 161
6.4.2 Experiencing Fatigue and Losing Stamina 161
6.5. Weapons 162
6.5.1 Choosing a Weapon 162
6.5.2 Aiming a Weapon 163
6.5.3 Firing a Weapon 164
6.5.4 Firing Weapons with an HMD Controller 164
6.5.5 Throwing a Hand Grenade 164
6.5.6 Deploying and Stowing Weapons 165
6.5.7 Reloading a Weapon 166
6.5.8 Rocket and Missile Launchers 166
6.6. Using the AN/PED-1 LLDR 171
6.7. Embarking on a Vehicle 172
6.7.1 Embarking on the Stryker's Interior 173

6.7.2 Disembarking from a Vehicle 174
6.8. Carry or Drag a Person 174
6.9. Parachuting 175
6.10. Tasking Other Entities 176
6.10.1 Disembark Entity 176
6.10.2 Follow Me 176
6.10.3 Move to Location 177
6.10.4 Move to My Location 177
6.11. Turning on Your Flashlight 178
6.12. Using the Binoculars 178
6.13. Changing Visual Sensors 179
6.14. Lasing Targets 179
6.14.1 Defining the Laser Code 181
6.14.2 Displaying a Gimbaled Sensor's View 181
6.14.3 Enabling Laser Designators 181
6.15. Opening and Closing Doors and Windows 181

6.1. Introduction

VR-Engage allows you to control individual human characters. VR-Engage includes models for several types of dismounted infantry and civilian characters, including:

- Dismounted infantry
- Dismounted infantry with lasing capability
- Dismounted infantry with missile launchers
- Paratroopers
- Police officers

6.1.1 Enabling and Disabling Mouse Control

When mouse control is enabled, you can use the mouse to turn a character, fire, and other actions. When it is disabled, you can access menus (in windowed mode) and do other typical mouse actions.

When you take control of a character, mouse control is automatically enabled. When you switch to windowed mode, it is automatically disabled. You can toggle mouse control on and off at any time in both full-screen mode and windowed mode.

➤ To enable or disable mouse control, press **Ctrl**. You can also double-click in the application to re-enable mouse control.

6.2. Basic Movements

As a player, you can move your character around and perform gestures.

6.2.1 Moving Your Character Around the Terrain

You can move your character around the terrain using the keyboard, game controller, or head-mounted display (HMD) controller. You can look around using the keyboard, game controller, HMD controller, or mouse. You can increase and decrease the speed at which you move.

6.2.2 Changing Your Character's Posture

You can change your character's posture. The default is to be standing upright, but you can also crouch or lie prone. Crouching helps you move into spaces with a low clearance or hide behind obstacles. Lying prone makes you less of a target for enemies and improves your accuracy when aiming a weapon.

An icon indicates your character's posture.

➤ **To change your character's posture:**

- On the keyboard:
  - Press **E** to move down to a crouching or prone position.
  - Press **Q** to move up to a crouching or standing position.
  - Press **Z** to lay prone from any position.
  - Press **C** to stand from any position.
- On the game controller, press the B button to cycle between standing, crouching, and lying prone.

6.2.3 Looking Around

You can look around by turning only your character's head, instead of turning the entire body. While you are looking, you cannot fire a weapon.

➤ **To look without moving your body:**

1. Press and hold **.** (period) or the right thumbstick.
2. Use the mouse, the right thumbstick, or the **H**, **K**, **U**, or **J** keys to look around you.

6.2.4 Gesturing and Signaling

Your character can perform a gesture or signal, such as pointing or surrendering. Some signals have military meanings, which are only available to military characters.

➤ **To make a gesture:**

1. Open the Action menu.
2. On the Action menu, choose **Gesture**.

3. Use an Action menu control to show the submenu. The Gesture menu opens.
4. Choose a gesture.

Your character performs the gesture. People looking at your character will see the gesture. (If you are in the first-person view, you will not see yourself performing the gesture. You can see it in third-person view.)

6.2.5 Climbing Ladders

If there are ladders in the terrain, you can climb them.

➤ **To climb a ladder:**

1. Move your character to the base or top of the ladder. When your character mounts the ladder, the ladder fills the view.
2. Climb up or down the ladder:
   - To climb up the ladder, move forward.
   - To climb down the ladder, move backward.

6.3. Navigation

While playing a human character, you can use the compass and the minimap to help you navigate the terrain.

6.3.1 Using the Compass

The compass at the top allows you to see which direction your character is facing. The compass is horizontal and shows the direction where your character is looking.

While in the human role, you can choose whether the compass should show degrees or NATO-MIL. You can also choose whether you want the compass to reference True, Magnetic, or Grid north. True north, also known as geodetic north, is the direction pointing to the geographic location of the North Pole. Magnetic north, on the other hand, is the point where the magnetic field points down. Lastly, grid north is the direction indicated by the vertical grid lines on ordnance survey maps.[1]

[1] In VR-Engage, magnetic north is determined by the World Magnetic Model (WMM). The current parameters are valid for 2020–2025. (The date used in a simulation is the date used for the magnetic model information.) To learn more about the WMM, visit https://www.ngdc.noaa.gov/geomag/WMM.

Figure 53. Compass



➤ **To enable or disable the compass:**

1. Open the Action menu (Figure 42 on page 134).
2. On the Action menu, choose **Compass**.
3. Use an Action menu control method to toggle the compass on or off.

To toggle the compass on or off, you can also press **G**.

➤ **To change the units displayed in the compass:**

1. Open the Action menu.
2. Use an Action menu control method to toggle the Compass Units to either NATO-MIL or Degrees.

➤ **To specify which north the compass should use:**

1. Open the Action menu.
2. Use an Action menu control method to toggle the North Reference to True, Magnetic, or Grid.

The default settings for the compass are based on the 3D Compass and Heading specified in the Display Units Settings; see "Specifying Display Units" in the *VR-Vantage User Guide* to learn how to adjust these settings while in Admin mode.

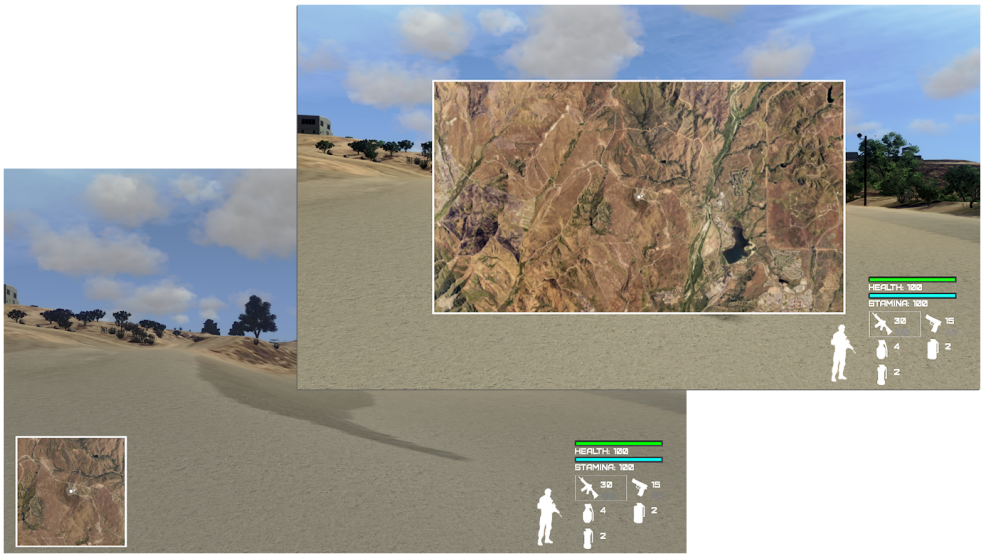
6.3.2 Using the Minimap

You can open a minimap that shows where your character is in relation to the terrain.

The minimap is hidden by default. Use the Action menu or the keyboard to select the display format. Using the keyboard, you can also zoom in or out of the minimap.

You can configure the available map sizes and the minimap's border in *./data/simulationModelSets/VR-Engage/role/devices/minimap.lua*.

Figure 54. Minimap - corner and center



➤ **To show or hide the minimap display using the Action menu:**

1. Open the Action menu (Figure 42 on page 134).
2. On the Action menu, choose **Minimap**.
3. Use an Action menu control method to toggle the setting: hidden, corner, or center.

➤ **To control the minimap display using the keyboard:**

- Press **M** to cycle through the corner, center, and hidden views.
- Press **+** on the number pad to zoom in on the minimap.
- Press **-** on the number pad to zoom out of the minimap.
- Press **0** on the number pad to reset the minimap to the default zoom.

6.4. Health and Stamina

The human entity overlay includes bars that indicate the current health and stamina for your character (Figure 55). Health and stamina are depleted as your character sustains damage and experiences fatigue. Tracking human entities' health and stamina adds realism to your simulation.

Figure 55. Health and stamina bars



6.4.1 Sustaining Damage to Health

Unless your character is set to be invulnerable, you will sustain damage when other characters attack you. The health bar indicates the state of your health (Figure 55 above). As you take damage and your health is depleted, you will have trouble moving around and aiming weapons. After a certain amount of damage, you fall to the ground in pain. At that point, your view lowers to ground level, and you cannot move or aim a weapon. Other characters in the scenario also sustain damage and experience the same effects (if they are not invulnerable).

The location on the body where a character is injured determines how much damage a character sustains. For example, a shot in the leg or arm is less lethal than a shot to the torso or head.

As you take damage, the screen shows some red. When you are hit from the side, the corners of that side of the screen are highlighted in red. If you are shot from the front or the back, all corners of the screen are highlighted. In addition, as your health is depleted, the screen takes on a red tint and your aiming will be less accurate.

Human characters that are only slightly injured and sustain no further damage will recover to full health after 30 seconds.

6.4.2 Experiencing Fatigue and Losing Stamina

Sprinting, crawling, walking in a crouched position, and carrying or dragging a wounded human are more challenging and therefore they cause you to use your stamina and experience fatigue. The stamina bar displays your current level of stamina (Figure 55 above). As you grow fatigued and your stamina reserves are reduced, you cannot move quickly and you will also see a greater margin of error when you fire a weapon.

When you walk or stop moving, you regain your stamina. You recover faster if you stop moving than if you walk.

6.5. Weapons

Infantry characters have a rifle, a handgun, and several types of grenades. The name indicates a character's primary weapon; for example, characters with an M4 in their name have an M4 rifle. A graphic in the lower right corner of the screen shows the current weapon (Figure 56).

Figure 56. Weapon overlay for humans



6.5.1 Choosing a Weapon

➤ To cycle through the weapon list, use the Select Weapon button on your game controller or press the **Spacebar**.

➤ To choose a specific weapon using the keyboard, press the number key that correlates to the weapon.

➤ **To choose a weapon using the Action menu:**

1. Open the Action menu (Figure 42 on page 134).
2. On the Action menu, choose **Weapon**.
3. Use an Action menu control method to show the submenu. The Weapons menu opens.
4. Choose a weapon.

6.5.2 Aiming a Weapon

You can aim a weapon using the reticle or the sight (Figure 57 and Figure 58).

The reticle indicates the accuracy of your shots. If you are moving, fatigued, or wounded, there is a greater margin of error in your accuracy and the reticle increases in size. To improve your accuracy, you can hold still or change to a prone position. The reticle shrinks to indicate a lesser margin of error.

Figure 57. Aiming with the reticle



Figure 58. Aiming with sight



➤ To aim a weapon using the reticle, turn and raise or lower the weapon using the mouse (in mouse-look mode) or the right thumbstick on the game controller, or by pressing the **H**, **K**, **U**, or **J** keys.

➤ **To aim a weapon using the sight:**

1. Hold down the right mouse button (in mouse-look mode), the left trigger on your game controller, or the **,** (comma) key.
2. Aim the sight using the mouse or the right thumbstick, or by pressing **H**, **K**, **U**, or **J**.

6.5.3 Firing a Weapon

Most guns shoot once each time you fire the weapon. The AK-47 fires in three-shot bursts.

➤ **To fire a weapon, you can either:**

- Click the left mouse button.
- Press the right trigger on your game controller.

6.5.4 Firing Weapons with an HMD Controller

If you are using a head-mounted display (HMD) with hand controllers, you can fire weapons using your controller. By default, the right hand controller is the dominant controller. You can change this by changing the **rightHanded** parameter in *./data/simulationModelSets/VR-Engage/roles/displayLayouts/humanVrLayouts.lua* from true to false.

- If you have a pistol, hold the controller in front of you to aim. Press the trigger to fire. The pistol only tracks the position of the dominant hand.
- If you have a rifle, align the two controllers as if you were holding a rifle, with the dominant hand closer to your body. Aim the rifle and fire.

Figure 59. Firing a rifle using HMD hand controllers



6.5.5 Throwing a Hand Grenade

Soldiers can throw fragmentation grenades, smoke grenades, and flash bang grenades. Grenades are thrown at a target point. (If the soldier is aiming towards an area with a light background, the target point may be hard to see.)

Figure 60. Grenade target point



➤ **To throw a grenade:**

1. Use the mouse or the right thumbstick to aim the grenade.
2. Click the left mouse button (in mouse-look mode) or press the right trigger on your game controller.

To use the keyboard to aim the grenade, press the **H**, **K**, **U**, or **J** keys.

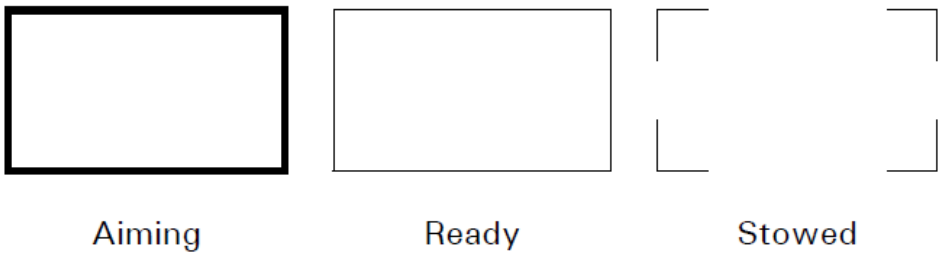
6.5.6 Deploying and Stowing Weapons

When a weapon is deployed (Aiming state), you see it in first-person view and you can fire it. When it is in the Stowed or Ready state, you cannot see it in first-person view and you cannot fire it.

The weapon overlay indicates the status of the weapon, as follows and as illustrated in Figure 61:

- Aiming. Weapon icon has a thick rectangle around it.
- Ready. Weapon has a thin rectangle.
- Stowed. Weapon has a bracketed rectangle.

Figure 61. Weapon status



➤ **To deploy or stow a weapon:**

1. Open the Action menu.
2. On the Action menu, choose **Weapon State**.
3. Use an Action menu control method to toggle between states.

➤ To cycle the weapon state, press ' (apostrophe).

➤ To move a weapon from the Ready state to the Aiming state, press the left mouse button or right trigger on the controller.

6.5.7 Reloading a Weapon

As you fire a weapon, the system tracks the amount of ammunition left in the magazine. When the magazine is depleted, your character automatically pauses to reload. At any point, you can also manually reload the weapon. The character reloads a magazine's worth of ammunition.

Figure 62. Magazine and ammunition counts



- 1 - Remaining in magazine
- 2 - Available to load

➤ **To reload your weapon with a fresh magazine, you can either:**

- Press **R**.
- Open the Action menu and then choose **Reload**.

6.5.8 Rocket and Missile Launchers

Entities configured for shoulder-mounted weapons can be equipped with different launchers in either VR-Forces or VR-Engage's Admin mode at runtime. You can equip them with any of the launchers available in VR-Forces by setting the launcher equipment resources on the entities.

Out of the box, VR-Engage has two characters that are already equipped with Javelin missile launchers: one with a shoulder-mounted Javelin missile launcher (US Army Javelin) and one that can deploy a tripod-mounted Javelin (US Army Javelin (Tripod)).

Equipping Soldiers with Launchers

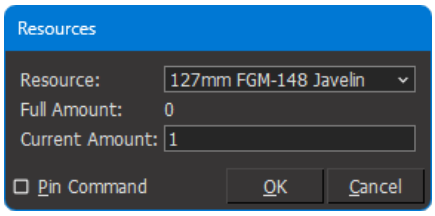
You can equip soldiers and police officers with shoulder-mounted weapons and rockets or missiles at runtime, either in VR-Forces or as a VR-Engage player in Admin mode.

Equipping an Entity in VR-Forces

➤ **To equip a human character with a weapon in VR-Forces:**

1. In VR-Forces, select the entity and then choose **Set > Disposition > Resources**. The Resources parameters open.

Figure 63. VR-Forces Resources dialog box with weapon selected



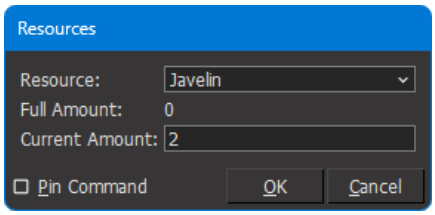
2. Select the weapon from the list.
3. In the Current Amount box, enter 1 to give the entity the selected weapon.
4. Click OK.

You can supply the entity with munitions using the same set data request. By default, entities that are configured for shoulder-mounted weapons already have resources for those weapons.

➤ **To supply a human character with munitions for the weapon in VR-Forces:**

1. In VR-Forces, select the entity and then choose **Set > Disposition > Resources**. The Resources parameters open.

Figure 64. VR-Forces Resources dialog box with munition selected



2. Select the appropriate munition from the list.
3. In the Current Amount box, enter the number of munitions that you want the entity to have.
4. Click OK.

Equipping an Entity in VR-Engage

In Admin mode, you can simply equip your character with the desired weapon.

➤ **To equip a human character in VR-Engage as a player:**

1. Make sure you are in Admin mode (**Shift+F12**).
2. Engage as a soldier or police character.
3. Open the Action menu and then choose **Load Resource**. The Load Resource submenu opens.
4. Use an Action menu command to choose one of the available weapons.

Using the Shoulder-Mounted Missile Launcher

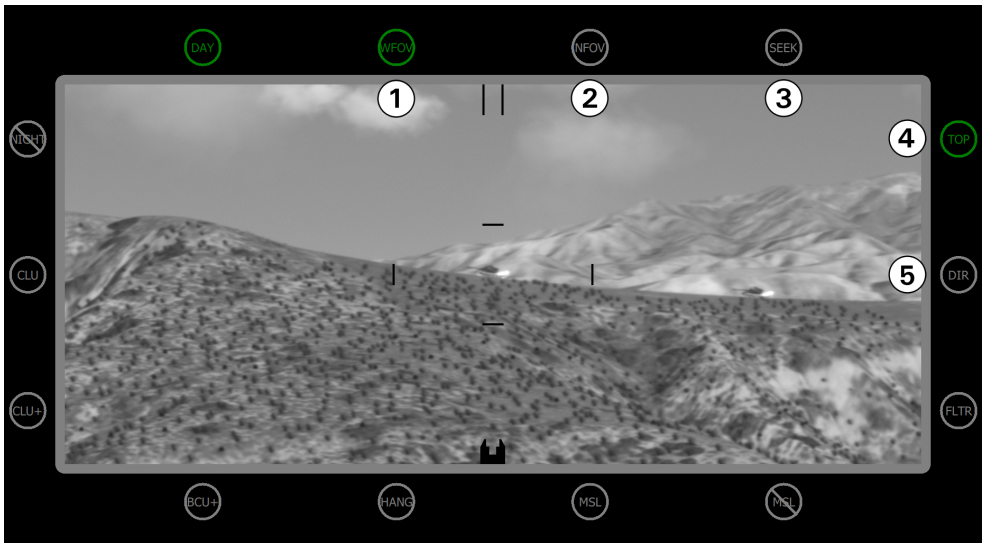
The shoulder-mounted missile launcher simulates an FGM-148 Javelin. It has two methods of firing: direct or top attack. The minimum range is 65 m for direct attacks and 150 m for top attacks. It is configured with a maximum range of 4 km. You must seek and lock onto a legitimate target before firing the weapon.

By default, the US Army Javelin entity is equipped with the Javelin as one of its weapons. If you want another entity to use a Javelin, see "Equipping Soldiers with Launchers" on the previous page.

The Javelin Command Launch Unit Display

The Javelin's command launch unit (CLU) display shows the status indicators.

Figure 65. Javelin CLU display



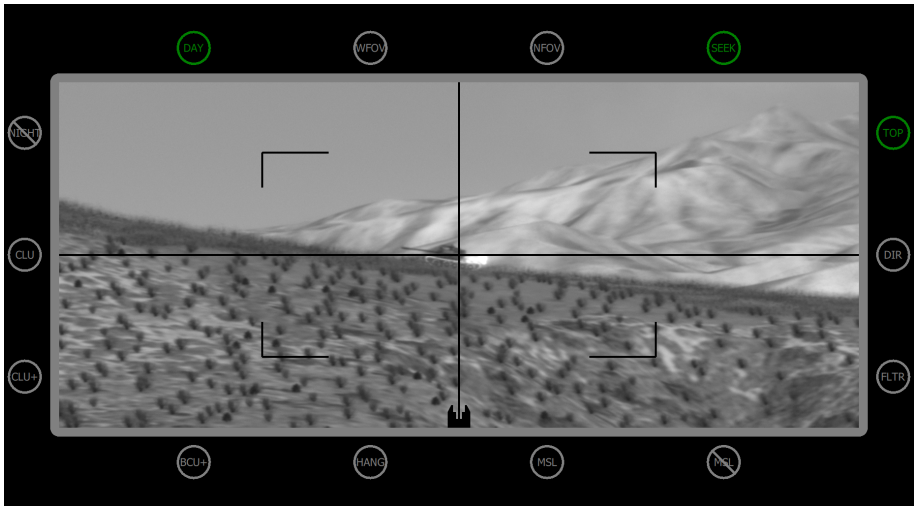
- 1 – WFOV. Indicates when you are using the wide field-of-view.
- 2 – NFOV. Indicates when you are using the narrow field-of-view.
- 3 – Seek. Indicates whether you are in Seek mode.
- 4 – Top. Indicates that the Javelin is set to top attack.
- 5 – Dir. Indicates that the Javelin is set to direct attack.

Aiming and Firing the Javelin

➤ **To use the shoulder-mounted missile launcher:**

1. While engaged as a character equipped with a Javelin, select that weapon using one of the standard weapon selection methods.
2. Aim the Javelin using one of the weapon aiming methods. The view changes to show the Javelin CLU, zoomed in where you are looking.
3. Optionally, use the mouse wheel to change between NFOV (scroll forward) and WFOV (scroll backward).
4. Optionally, you can toggle between top attack and direct attack by pressing the **M** key.
5. Press the **T** key or the left shoulder button to enter Seek mode, where you can detect legitimate targets. The cross-hairs display and the Seek icon is green (Figure 66).
   - If there is no target, the cross-hairs and reticle blink.
   - When you have a target, nothing blinks.

Figure 66. CLU display, target found



6. Fire the weapon using a standard weapon firing method.

Using the Tripod-Mounted Missile Launcher

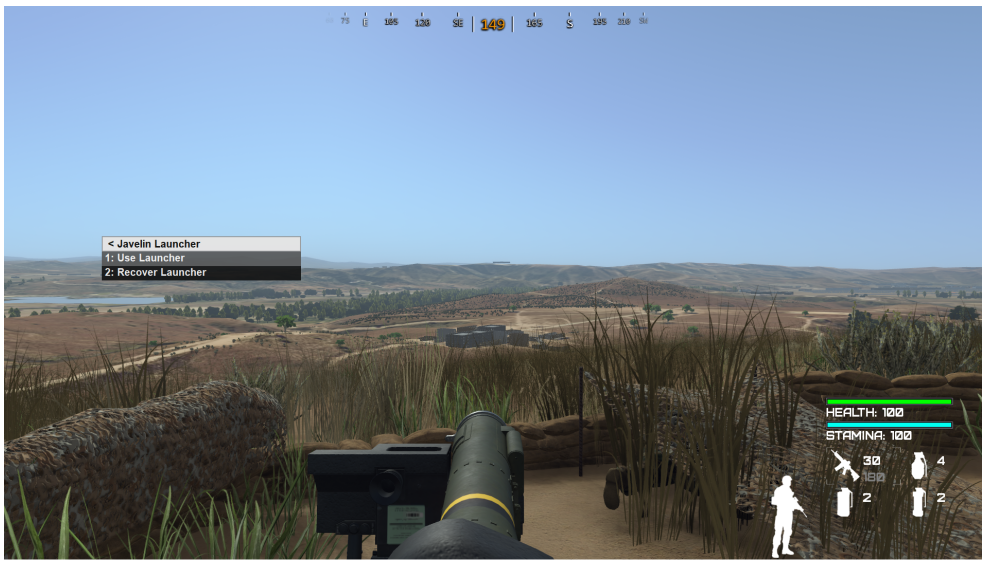
The tripod-mounted missile launcher is an object that the US Army Javelin (Tripod) character can deploy. Until it is deployed, it is not part of the scenario. Once it is deployed, it is an independent entity in the scenario. Other characters can fire it. Only the character that originally deployed it can recover it; that is, remove it from the scenario.

A tripod-mounted missile launcher has an azimuth (horizontal range) of 45 degrees to the left and right. It has an elevation (vertical range) of 30 degrees. When a tripod-mounted missile launcher is deployed, it faces in the direction of the entity that deployed it. Once it is deployed, you cannot change the tripod's heading.

➤ **To deploy the tripod-mounted Javelin:**

1. Add or select the US Army Javelin (Tripod) character.
2. Engage the character. The default presentation is that of a US Army M4 character.
3. Walk to the location where you want to deploy the missile launcher.
4. Face the heading that you want the missile launcher to be deployed to.
5. Optionally, stow your rifle.
6. On the Action menu, choose **Deployable Objects > Deploy Javelin Launcher**. The Javelin Missile Launcher is added to the scenario. The Action menu changes to show Javelin Launcher menu options.

Figure 67. Tripod-mounted Javelin



Using a Tripod-Mounted Missile Launcher

Once a missile launcher is deployed, any character can use it.

➤ **To use the missile launcher:**

- If your character is close to the missile launcher, on the Action menu, choose **Javelin Launcher > Use Launcher**.
- If your character is not close to the missile launcher, walk towards it until the Javelin Launcher menu displays and then choose **Use Launcher**. The display changes to show the CLU (Figure 65 on page 168).

The tripod-mounted missile launcher aims and fires using the same methods as the shoulder-mounted one. See "The Javelin Command Launch Unit Display" on page 168 and "Using the Shoulder-Mounted Missile Launcher" on page 168 for instructions.

Recovering a Tripod-Mounted Missile Launcher

When you recover a tripod-mounted Javelin, you remove it from the scenario. It is not available for use until you deploy it again. Only the character that deployed the Javelin can recover it.

➤ To recover the missile launcher, choose **Javelin Launcher > Recover Launcher** on the Action menu. If you are not within a few feet of the missile launcher, the Javelin Launcher menu option is not available. If you walk towards the missile launcher, it opens automatically.

Using a Rocket Launcher

Soldiers and police officers can be equipped with rocket launchers. These are simpler weapons than the Javelin, in that they do not detect legitimate targets. Players simply point and fire the weapon.

➤ **To use a rocket launcher:**

1. While engaged as a character equipped with a rocket launcher, select that weapon using one of the standard weapon selection methods.
2. Aim the launcher using one of the weapon aiming methods. The HUD displays the simple iron sight. (Aiming is optional.)
3. Fire the weapon using a standard weapon firing method.

6.6. Using the AN/PED-1 LLDR

You can use the AN/PED-1 LLDR (Lightweight Laser Designator Rangefinder AN/PED-1) in VR-Engage for forward observer training. A player operating the LLDR can look around, lase targets that are within line of sight, use the radio, and change the camera on the LLDR.

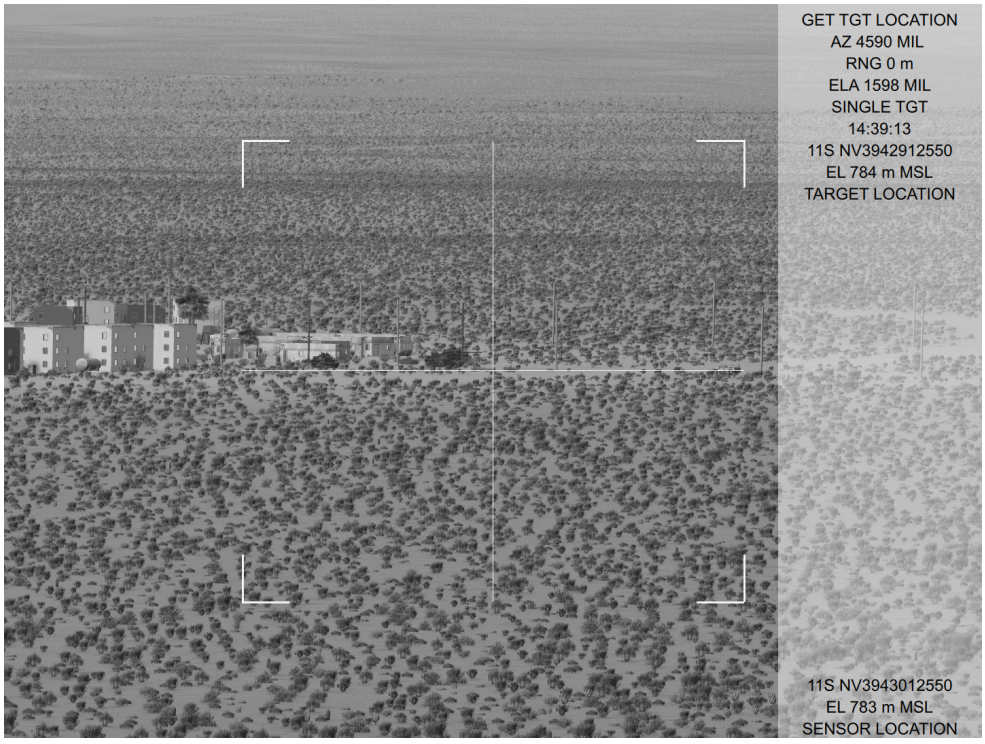
The device is a standalone entity, not attached to any other character. You can include it in a scenario that includes human characters for them to use, or create it as an entity and operate it directly from the Choose Role panel.

ⓘ You can use the LLDR with VR devices.

➤ **To operate the LLDR as a human character:**

1. Approach the AN/PED-1 LLDR. When the LLDR menu opens, use an Action menu command to choose **Use LLDR**. Your view changes to the LLDR HUD (Figure 68).
2. Move the view to a point of interest.

Figure 68. LLDR HUD



➤ To lase a target, open the Action menu and toggle Lasing to On. The HUD indicates that you are lasing a target: the top line changes from ELRF_IDLE to GET TGT LOCATION.

➤ **To define a laser code:**

1. In the Action menu, choose **Laser Code**. The Set Laser Code dialog box opens.
2. Enter a laser code and then press **Enter**.

6.7. Embarking on a Vehicle

If a vehicle supports embarkation, characters can embark on it and move with it as it moves. If the vehicle has interior geometry, your character can look around and walk around on the floor of the vehicle. Among the vehicles provided with VR-Engage, the 5-ton truck, the Stryker, the police car, and the CH-47 Chinook and CH-149 Cormorant helicopters have interior geometry. Other vehicles, such as the LAV and M1A2, support embarkation but do not have interior geometry.

➤ **To embark on a vehicle:**

1. Have your character walk up to the vehicle. When the character is within a meter or so of the vehicle, the embarkation menu for the vehicle displays.

Figure 69. Embark menu



2. Select a role or seat.
3. Press the **Right arrow** key or right D-pad button to initiate the action. The character is placed on the vehicle.

6.7.1 Embarking on the Stryker's Interior

The M1126 Stryker ICV with M2 has an interior that allows you to board and choose an available seat.

➤ **To embark on the Stryker from the interior:**

1. Have your character walk up to the back of the vehicle. When the character is within a meter or so of the vehicle, the Stryker menu displays.
2. If the door is not open, select **Door**. The door opens, providing a ramp into the Stryker.
3. Walk your character up the ramp and into the interior. Your character automatically shifts to a crouching posture.

   As you navigate the interior, the seat closest to your character displays in the Stryker menu. Different seats display in the menu based on your position.
4. Choose the seat from the menu. This can be a seat in the back, the Driver Seat, or the Commander seat. (You can also embark as the driver or commander directly from the Stryker menu as described in "6.7. Embarking on a Vehicle" on the previous page.)

6.7.2 Disembarking from a Vehicle

There are a couple of ways that you can disembark from a vehicle, depending on how the vehicle is configured.

➤ **To disembark from a vehicle, you can either:**

- On the Action menu, choose Disembark. The character is placed outside of the vehicle.
- Have the character walk off the back of the vehicle.

6.8. Carry or Drag a Person

In some situations, it is necessary to carry or drag a person, such as someone who is injured. VR-Engage allows you to replicate that in a scenario. If there is a vehicle in a scenario, you can embark the person you are carrying on that vehicle.

Figure 70. Drag and carry



- If a human character is dead, you cannot carry or drag them.
- The ability to drag or carry an entity is currently limited to some of the US soldier entities.

➤ **To drag or carry a character:**

1. Approach the injured person.
2. Open the Action menu.
3. Select the character and then press the **Right arrow** key or right D-pad button.
4. Choose the method you want to use:
   - Armpit Drag. Put your hands under the injured person's armpit to move them.
   - Strap Drag. Take the injured person by the strap on the back of their vest to move them.
   - Fireman Carry. Put the injured person over your shoulder to carry them.

The camera switches to third-person view.

5. Move in the direction where you want to take the injured person.
6. At any point, you can open the Action menu and choose **Drop Injured** to place the person on the ground.

➤ To place the person in a vehicle, choose **Embark Injured** from the Action menu.

6.9. Parachuting

The paratrooper character can parachute out of aircraft. The descent starts in free fall. The paratrooper's weapon is hidden. After the parachute is deployed, the drift of the paratrooper is affected by wind speed and direction. You can turn the parachute to the left or right and you can slow the rate of descent. When the paratrooper lands, the parachute is hidden and the paratrooper's weapon is again available. It returns to whichever view mode it was in before it jumped.

Figure 71. Parachute deployed



➤ To jump from an aircraft, walk out of an open cargo door. If the door is not open, you can open it.

➤ To deploy a parachute, press **Spacebar** or the A button on your controller.

➤ To turn left or right while the parachute is deployed, press the **A** or **D** key, or the shoulder buttons on the game controller.

➤ To slow the rate of descent while the parachute is deployed, press **A+D** or both shoulder buttons simultaneously.

6.10. Tasking Other Entities

While you are playing a character, you can assign tasks to other entities in the scenario.

> When you assign a task, you are assigning what VR-Forces calls an independent task. If the entity to which you are assigning the task is currently executing a task either as an independent task or as part of a plan, when you assign the new independent task, the entity abandons the old task or plan.

6.10.1 Disembark Entity

The Disembark Entity task causes the selected entity to disembark from the vehicle that it is on. You can only task an individual entity, not a unit.

➤ **To issue the Disembark Entity task:**

1. Open the Action menu.
2. Choose **Task Entity**.
3. Use an Action menu control method to show the submenu. The Select Entity menu opens.
4. Select the entity that you want to give a task to.
5. Use an Action menu control method to show the submenu. The Task Entity menu opens.
6. Choose **Disembark Entity**.
7. Use an Action menu control method to initiate the action.

6.10.2 Follow Me

The Follow Me task causes the selected entity or unit to follow your character.

> If you task an entity or unit to follow you, it will do so indefinitely. To make it to stop following, task the entity or individual entities to move to a location.

➤ **To issue the Follow Me task:**

1. Open the Action menu.
2. Choose **Task Entity**.
3. Use an Action menu control method to show the submenu. The Select Entity menu opens.
4. Select the entity or unit that you want to give a task to.
5. Use an Action menu control method to show the submenu. The Task Entity menu opens.
6. Choose **Follow Me**.

7. Use an Action menu control method to initiate the action. The entity or unit moves to a location behind your character.

6.10.3 Move to Location

The Move to Location task causes the selected entity to move to the specified location. You can only task an individual entity, not a unit.

➤ **To issue the Move to Location task:**

1. Open the Action menu.
2. Choose **Task Entity**.
3. Use an Action menu control method to show the submenu. The Select Entity menu opens.
4. Select the entity that you want to give a task to.
5. Use an Action menu control method to show the submenu. The Task Entity menu opens.
6. Select **Move to Location**. VR-Engage places a waypoint symbol at the point your character is looking at.
7. Use the mouse or right thumbstick on your game controller to move the waypoint to the location you want the other entity to move to.
8. Use an Action menu control method to initiate the task. The tasked entity moves to the location.

6.10.4 Move to My Location

The Move to My Location task causes the selected entity to move so that it is standing next to your character. You can only task an individual entity, not a unit.

➤ **To issue the Move to My Location task:**

1. Open the Action menu.
2. Choose **Task Entity**.
3. Use an Action menu control method to show the submenu. The Select Entity menu opens.
4. Select the entity that you want to give a task to.
5. Use an Action menu control method to show the submenu. The Task Entity menu opens.
6. Choose **Move to My Location**.
7. Use an Action menu control method to initiate the action. The selected entity moves to a position next to your character.

6.11. Turning on Your Flashlight

The flashlight casts a circle of light on the terrain.

➤ **To turn on the flashlight:**

1. Open the Action menu.
2. Choose **Flashlight**.
3. Use an Action menu control method to toggle the flashlight on or off.



You can also toggle the flashlight on and off by pressing the **F** key.

6.12. Using the Binoculars

You can zoom in and out in five fixed steps. At the highest zoom level (binoculars), VR-Engage displays a range finder graphic as an M22 binoculars reticle with 10 mil markers.

You cannot fire a weapon when the binoculars are in use, nor can you use binoculars while your weapon is in the Aiming position. If you are controlling a character that has lasing capability, pressing the Fire button lases the target.

➤ **To zoom the view in and out, do any of the following:**

- Scroll the mouse wheel forward and back.
- On the Action menu, choose **Binoculars** and change the status. Binoculars go immediately to the highest zoom level.
- Press the **B** key (for binoculars view). Binoculars go immediately to the highest zoom level. Press **-** to zoom out or **=** to zoom in. The view zooms one step for each key press
- Press the X button. Binoculars go immediately to the highest zoom level. Use the left shoulder button to zoom out or the right shoulder button to zoom in.
- Press **=**, right shoulder button, or scroll the mouse wheel forward. Binoculars go immediately to the highest zoom level.



If you return to the Choose Role panel while using binoculars, the entity stops using the binoculars and you must use one of the methods to start using them again if you reengage the same entity.

6.13. Changing Visual Sensors

Human characters can simulate the effect of wearing night vision goggles (NVG).

> The default visual sensors use CameraFX, which just applies a filter to the scene. It does not take into account the materials of the viewed objects. For details, see the *VR-Vantage User Guide*.
>
> If you want a visual sensor capability that takes into account the materials of the viewed objects, you can install SensorFX. SensorFX is an extra cost option. Please contact info@mak.com for details.

➤ **To turn on the night vision goggles for human characters:**

1. Open the Action menu.
2. Choose **Night Vision**.
3. Use an Action menu control method to toggle night vision on or off.

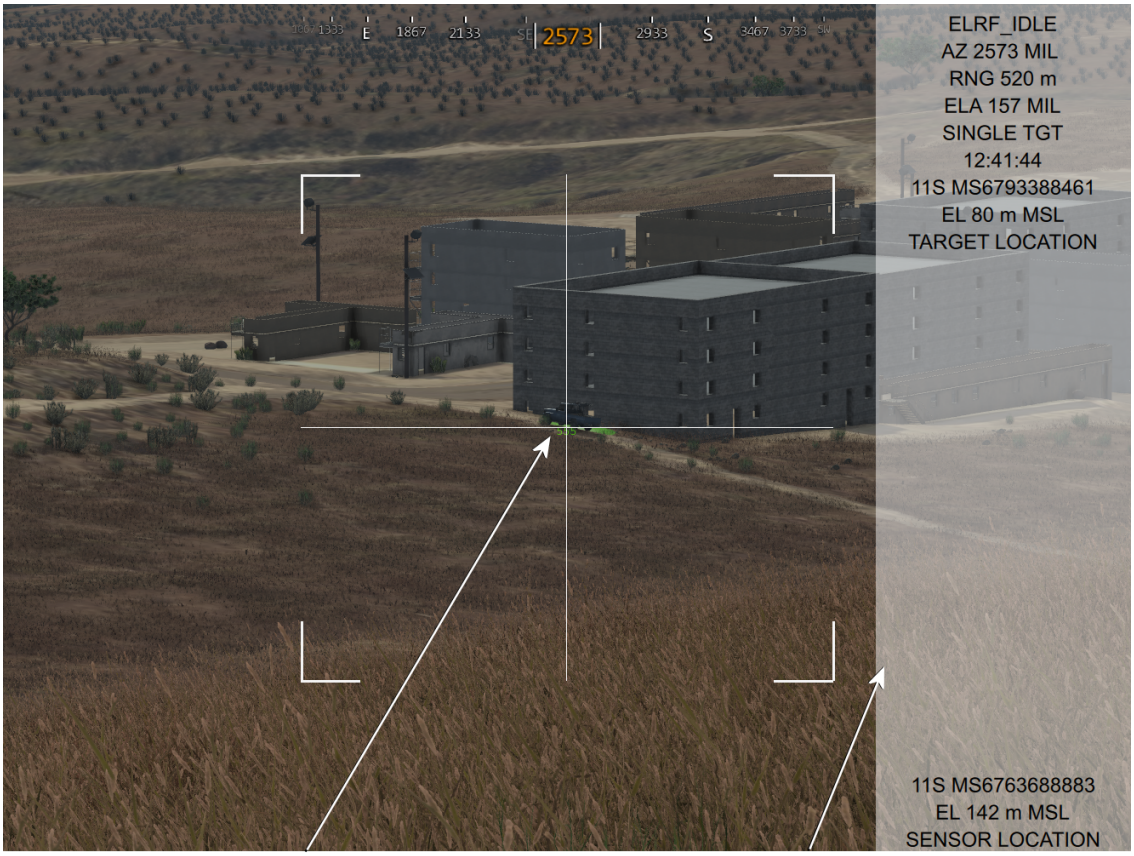
> You can also toggle night vision by pressing the **N** key or D-pad down on a game controller.

6.14. Lasing Targets

Soldiers that have a lasing capability can lase targets if they have line of sight to the target and it's within range of their laser designator. VR-Engage includes two characters with lasing capability: DI Lasing (US) and DI Lasing (CIS).

When you lase a target, the display shows that the laser is on. If you enable laser designators, a green circle displays on the lased location.

Figure 72. Lasing a target



You cannot select an entity to lase. You can only lase locations on the terrain.

➤ **To lase a target:**

1. Do one of the following to enable binoculars:
   - On the Action menu, toggle **Binoculars** to On.
   - Press the X button on a game controller.
   - Press the **B** key.
2. Point the cross-hairs of the binoculars at the point you want to lase.
3. Press and hold the left mouse button or the right trigger on a game controller.

6.14.1 Defining the Laser Code

You can define the laser code used when you lase a target. Laser codes can be up to four digits long. When you lase a target, that code is used.

➤ **To define the laser code:**

1. On the Action menu, choose **Forward Air Control**.
2. In the Forward Air Control menu, choose **Laser Code**. The Set Laser Code dialog box opens.
3. Enter a laser code and then press **Enter**.

6.14.2 Displaying a Gimbaled Sensor's View

You can display the sensor view for gimbaled sensors in the scenario. You cannot control the view, only see it.

➤ **To view the sensor on another entity:**

1. On the Action menu, choose **Forward Air Control**.
2. On the Forward Air Control menu, choose the entity whose sensor view you want to display.

6.14.3 Enabling Laser Designators

A laser designator is a green circle that shows where a laser is being pointed. It includes the laser code. Laser designators are disabled by default. You must be in Admin mode to enable laser designators.

➤ **To enable or disable laser designators:**

1. If you are not in Admin mode, press **Shift+F12**.
2. Choose **Settings > Laser Designators**.

6.15. Opening and Closing Doors and Windows

If a terrain has doors and windows that can open and close, when a human character approaches within about a meter of an opening, a menu is displayed that lets you choose to open or close the opening.

Figure 73. Door menu



The ability to open and close doors requires a dynamic terrain server. Therefore, it only works when you are connected to a VR-Forces session.

➤ To open or close a door or window, when the appropriate menu displays, make a selection from the menu.



7. Controlling Ground Vehicles

This chapter explains how to control and crew ground vehicles.

7.1. Introduction 184
7.1.1 Ground Vehicle Roles 184
7.1.2 Ground Vehicle Display Options 186
7.2. Driving Ground Vehicles 186
7.2.1 Starting the Engine 186
7.2.2 Putting the Vehicle into Gear 186
7.2.3 Controlling the Throttle 187
7.2.4 Steering a Vehicle 187
7.2.5 Looking Around 187
7.2.6 Using the Backup Camera View 187
7.2.7 Releasing and Engaging the Parking Brake 188
7.2.8 Locking and Unlocking the Differential 188
7.2.9 Turning the Headlights On and Off 188
7.3. The Gunner Role 188
7.3.1 Firing a Vehicle's Weapon 189
7.3.2 Choosing a Weapon 189
7.3.3 Moving a Turret or Gun 189
7.3.4 Stabilizing a Vehicle's Gun 189
7.3.5 The LSV MK II Gunner Roles 190
7.4. The Commander Roles 192
7.4.1 Changing Positions 192
7.4.2 Switching Views in the Commander Position 192
7.4.3 Overriding the Gunner 193
7.4.4 Slewing the Gun to Designate a Target 193
7.4.5 Using the Laser Designator 193
7.5. Using the Intercom 194
7.6. Changing the Sensor for a Vehicle Crew Member 194

7.7. Pivoting a Tank 194
7.8. Towing Other Ground Entities 195
7.9. Emergency Lights and Sirens 195

7.1. Introduction

VR-Engage allows you to control ground vehicles that use the Vortex physics libraries, which provide high-fidelity movement behaviors.

For a list of vehicles included with VR-Engage, see "Chapter 13. Controllable Entities" on page 291. You can add more ground vehicles. For details, see "Chapter 14. Adding and Editing VR-Engage Entity Models" on page 335.

By default, vehicles do not use Vortex physics unless they are driven by a VR-Engage player. This improves performance. If your scenario needs to allow for full-physics collisions between player-driven and CGF ground vehicles, you can configure them with that capability. See "16.2.3 Vortex While Untasked" on page 375 for details.

Some vehicles equipped with weapons allow a player to use the main gun while VR-Forces drives the vehicle.

If the vehicle's primary gun is modeled in Vortex, however, Vortex takes active control of the vehicle and all of its dynamics when a player engages in the gunner role. In those cases—such as tanks with 120 mm guns (M1A1, M1A2, and Leopard 2A4) and APCs with 22 mm guns (ASLAV-25, LAV 6 ISC, and LAV III APC)—CGF cannot drive the vehicle.

7.1.1 Ground Vehicle Roles

All ground vehicles support the Driver role, and some vehicles support other roles that you can select from the Choose Role Panel. You can change roles once embarked, as described in "3.4.3 Switching Roles on a Vehicle" on page 71.

Some vehicles also allow you to embark as a passenger in a particular seat when you walk up to them as a human character, but not from the Choose Role Panel. See "6.7. Embarking on a Vehicle" on page 172 for details.

To learn what roles are available for each vehicle, see Chapter 13 on page 291.

Driver

The Driver role allows you to:

- Drive the vehicle, which involves:
  - Turning the engine on and off.
  - Engaging gears.
  - Increasing and decreasing throttle.
  - Steering.
- Manage headlights and any emergency lights or sires configured for the vehicle.

Driver/Gunner

Some vehicles have a Driver/Gunner role. In addition to the driver functions, the Driver/Gunner role allows you to:

- Rotate the turret. (You cannot see this happen in the first person view.)
- Fire weapons that are configured on the vehicle. (For details, see "7.3.1 Firing a Vehicle's Weapon" on page 189.)

Gunner

The Gunner role allows you to:

- Fire weapons that are configured on the vehicle.
- Rotate a turret and raise and lower the gun. (Some vehicles allow you to perform these actions while the CGF drives the vehicle.)
- Stabilize the gun.

Commander

The Commander role allows you to:

- Look around.
- Change the sensor view.
- Zoom in and out.
- Slew or take full control of the gun (only specific vehicles).
- Designate a target with a laser (only specific vehicles).

Drone Operator

The Drone Operator role on the Light Strike Vehicle MK II (LSV) allows a player to deploy and control a quadcopter, even when the entity is in motion. For information, see "Chapter 9. Controlling Drones" on page 217.

7.1.2 Ground Vehicle Display Options

Most ground vehicles are designed for a single screen display. Some vehicles have support for a 3D interior or VR headsets.

- 1 Screen Horizontal. This is the standard interface for most ground vehicles. For ground vehicles, the driver HUD includes a fuel gauge, gear indicator, speedometer, and tachometer. The Light Strike Vehicle MK II (LSV) also has an inclinometer that indicates the pitch (Θ (theta)) and roll (Ψ (psi)).
- 3D Cockpit. This display shows a more detailed view of the interior for some vehicles.
- VR Headset. Select this option if you are using a head-mounted display and have chosen to engage one of the vehicles that supports VR. (This is relevant only if you have an HMD connected.)

➤ **To choose a display option for vehicles with multiple displays:**

1. In the Choose Role Panel, select the vehicle you want to control.
2. Choose the Role.
3. In the Display(s) list, select the display option. Different displays are available only on some vehicles and only for particular roles.

7.2. Driving Ground Vehicles

You can operate vehicles using a game controller, a racing wheel controller, or the keyboard. You have more nuanced control over your speed and steering when using a controller rather than the keyboard.

7.2.1 Starting the Engine

➤ **To start (or stop) the engine, do one of the following:**

- On the Action menu, toggle Engine: On/Off.
- On a game controller, press the Start button.
- On a Racing Wheel Controller, press the Engine On button.
- Press the **X** key on the keyboard.

7.2.2 Putting the Vehicle into Gear

Vehicles in VR-Engage have automatic transmissions. Vehicles have different sets of gears. For example, the LAV has eight forward gears and one reverse gear. Automobiles have six forward gears and one reverse gear. The M1A2 tank has five forward gears and one reverse gear.

➤ **To put the vehicle into gear, you can either:**

- Press the right shoulder button (up) or left shoulder button (down) buttons on your game controller.
- Press the **E** key (up) or **Q** key (down).

7.2.3 Controlling the Throttle

If you are using a racing wheel controller, use the gas and brake pedals.

➤ **To move the vehicle, do one of the following:**

- On a game controller, pull the right trigger.
- On a racing wheel controller, depress the Throttle.
- Press the **W** key.

➤ **To brake the vehicle, do one of the following:**

- On a game controller, pull the left trigger.
- On a racing wheel controller, depress the Brake.
- Press the **S** key.

ⓘ Game controller triggers are axes and therefore can respond to the amount of pressure that you apply. For example, pull the right trigger slightly to accelerate gently or fully depress it to accelerate rapidly.

7.2.4 Steering a Vehicle

➤ **To steer the vehicle, do one of the following:**

- Use the left thumbstick on a game controller
- Use the steering wheel of a racing wheel controller.
- Press the **A** key (left) or **D** key (right).

7.2.5 Looking Around

In some vehicles, ou can look around when you are in the Driver role.

➤ **To look around:**

- Press and hold the right mouse button and then move the mouse.
- Move the right thumbstick on a game controller.

7.2.6 Using the Backup Camera View

There are times when you want to see behind your vehicle, such as when you are driving in reverse. Vehicles that are configured with a backup camera view allow you to change the perspective to see behind you.

➤ **To switch to (or from) the backup camera view, you can:**

- Press the **B** key.
- Cycle through the available views until the backup view displays (**V** on the keyboard or Y button on a game controller).

7.2.7 Releasing and Engaging the Parking Brake

➤ **To set the parking brake on or off:**

- On the Action menu, toggle Parking Brake: On/Off.
- On a game controller, use left thumbstick (press).
- On a racing wheel Controller, press the Parking Brake button.
- Press the **P** key.

7.2.8 Locking and Unlocking the Differential

A differential allows the wheels on an axle to rotate at different speeds. Locking the differential may improve traction under certain terrain conditions. Players controlling a ground vehicle configured with a differential—such as the LAV, MRAP M-ATV, and M1278 L-ATV-HGC (JLTV)—can enable or disable it.

➤ **To lock or unlock the differential, do one of the following:**

- On the Action menu, toggle Differential Lock: On/Off.
- On a game controller, press the X button.
- On a racing wheel controller, press the Differential Gear button.
- Press the **C** key.

7.2.9 Turning the Headlights On and Off

➤ **To turn on headlights, do one of the following:**

- On the Action menu, toggle Headlights: On/Off.
- On a game controller, press the A button.
- On a racing wheel controller, press the Lights button.
- Press the **L** key.

7.3. The Gunner Role

The gunner can:

- Fire the gun.
- Select a munition to fire.
- Rotate and elevate the gun.
- Zoom in and out.
- Change the sensor. (For details, see "7.6. Changing the Sensor for a Vehicle Crew Member" on page 194.)

> The controls for the Driver/Gunner role are different. See "12.1.5 Ground Vehicle Controls" on page 245 and "12.5. Game Controller Mappings for Vehicles with Weapons" on page 251.

7.3.1 Firing a Vehicle's Weapon

As a gunner, you can fire the weapon. Some vehicle's weapons have different munitions and you can cycle through those to select the munition you want to fire, as described in "7.3.2 Choosing a Weapon" below.

Sabot or heat rounds fired from the M1A1 SA tank travel in an arc, not a straight line. The calculation includes the mass of the projectile, air resistance, and the drag coefficient. VR-Engage calculates for collisions in each frame, which can impact performance.

➤ **To fire a ground vehicle's gun, do one of the following:**

- Press the right trigger on your game controller.
- Press the **F** key.
- Click the left mouse button.

7.3.2 Choosing a Weapon

You can cycle through available munitions to fire the one you want.

➤ To cycle through the weapon list using a game controller, use the left trigger.

➤ To select a weapon using the keyboard, press the **R** key for the next weapon or the **V** key for the previous weapon.

➤ **To cycle weapons using the Action menu (LAV only):**

1. Open the Action menu.
2. Use an Action menu control method to use the **Cycle Weapon** option.

7.3.3 Moving a Turret or Gun

The Gunner and Driver/Gunner roles can move the turret or gun on vehicles that have these weapons.

➤ To move the weapon in the Gunner role, use the right thumbstick on your game controller. You can also use the mouse.

➤ To move the weapon in the Driver/Gunner role, use the D-pad.

➤ To move the weapon in either role, press the **H**, **K**, **U**, and **J** keys.

7.3.4 Stabilizing a Vehicle's Gun

Some ground vehicles equipped with weapons allow you to stabilize the guns. This locks the azimuth and elevation to the values they had with respect to the world when the turret was last moved.

If you have VR-Forces with the VR-Engage extensions, you can configure whether gun stabilization is enabled by default by editing the weapon system in the Simulation Object Editor. See the information on editing a system's properties in the *VR-Forces Users Guide*.

➤ **To enable or disable gun stabilization:**

1. While acting as the gunner, open the Action menu.
2. Choose **Gun Stabilization**.
3. Use an Action menu control method to toggle the setting: **Heading** stabilizes the gun and **Disabled** does not stabilize the gun.

7.3.5 The LSV MK II Gunner Roles

There are two gunner roles on the Light Strike Vehicle MK II (LSV): the primary and secondary gunners.

- The Primary Gunner can control three weapons: an automatic grenade launcher, machine gun, or Spike missile launcher. The use of each weapon is described below. You can fire the weapon only when the azimuth is between -90° and 90°.
- The Secondary Gunner has control of the machine gun. You can fire the machine gun only when the azimuth is between -45° and 45°.

ⓘ The three weapons that the Primary Gunner can use represent the weapons available on the different real-world models of the LSV. The entity model used in VR-Engage represents only one of these while being able to simulate any of them.

Arming the Weapons

By default, the weapons are unarmed. The gunner or commander must arm the weapon before the player in either gunner role can fire it.

➤ **To arm or disarm the weapon as a gunner:**

1. Open the Action menu.
2. Use an Action menu command to toggle the Armed status.

➤ **To arm or disarm the weapon as the commander:**

1. Open the Action menu.
2. Use an Action menu command to toggle the Primary Gunner Armed or Secondary Gunner Armed status.

Firing the Machine Gun or Grenade Launcher

You can move the LSV's machine gun and grenade launcher using the standard controls described in "7.3.3 Moving a Turret or Gun" on the previous page.

You fire the machine gun and grenade launcher using the standard controls described in "7.3.1 Firing a Vehicle’s Weapon" on the previous page.

Changing the Primary Gunner's Weapon

➤ To cycle weapons as the Primary Gunner, press the **B** key.

➤ To change the grenade type when using the grenade launcher, press the **R** or **V** key on the keyboard.


To cycle the grenades using a game controller, use the left trigger or D-pad up button.

Using the Spike Missile Launcher

The LSV's Primary Gunner role includes a Spike missile launcher. It can be fired in one of two trajectories: direct or top attack. You can aim and fire the weapon, or you can specify an indirect target using coordinates. When you fire the missile, the screen displays the missile's point-of-view as it flies to its ultimate destination. While the missile is flying, you can direct it using a game controller or by clicking in the window.

The missile has a short flying time, approximately 30 seconds. After that, it falls to the terrain.

Launching the Missile

➤ **To specify a missile trajectory and fire a missile manually:**

1. Open the Action menu.
2. Use an Action menu command to toggle the Missile Trajectory to either direct-attack or top-attack.
3. To aim the launcher, use one of the standard controls. Make sure that the Range displays a number: if it reads --, the weapon will not fire.
4. Fire the launcher using one of the standard weapon controls.

Specifying an Indirect Target

To specify an indirect target, use MGRS coordinates. You must separate the grid zone, square identifier, and numerical location (easting and northing) values with either spaces or hyphens. The missile targets the center of the grid you specify, therefore use the most precise resolution required to strike the target. You can specify 1 m resolution (`4Q FJ 17547 55543`), 10 m resolution (`4Q FJ 1739 5540`), and so on.

➤ **To fire a missile at an indirect target:**

1. Open the Action menu and then select **Indirect Target**.
2. Enter the target MGRS coordinates.
3. Press **Enter**.
4. Optionally, change the missile trajectory as described above.
5. Fire the launcher using one of the standard weapon controls.

For indirect fire, the launcher fires the missile at your aiming angle, and the missile then flies toward the target coordinates. If there is an obstruction in its flight path, such as a hill, it collides with the obstruction and detonates. If it cannot plot a path from the current angle, the launcher does not fire.

➤ To return to manual fire after entering coordinates, open the Action menu and then select **Clear Indirect Coordinate**.

Redirecting the Missile

After you launch the missile, you can redirect it toward another target using either the mouse or a game controller.

- With the game controller, you can steer it with simple flight controls.
- With the mouse, you can click on the terrain or an entity. You can continue clicking to change its target.

7.4. The Commander Roles

There are several commander roles in VR-Engage. Commanders can rotate the camera to look around, zoom in and out, change the sensor view, and trigger events. Commanders can also communicate with other crew members and other players using the intercom or radio. On some vehicles, the commander role is also configured to take over the Gunner role, slew the gun to a target, and designate a target with a laser. This last is known as "slew to cue", which only works when the Gunner role is being played by another user.

- To learn about taking over the Gunner role, see "7.4.3 Overriding the Gunner" on the facing page.
- To learn how to slew to cue, see "7.4.4 Slewing the Gun to Designate a Target" on the facing page.
- To learn how to use the laser designator, see "7.4.5 Using the Laser Designator" on the facing page.
- To learn more about triggering events, see "5.6. Starting VR-Forces Events" on page 147.

7.4.1 Changing Positions

The Commander role on the GTK Boxer CRV and M2A3 Bradley IFV can change positions, such as between out-the-hatch and RWS. In the out-the-hatch position, the commander can use binoculars and night vision goggles. In the RWS position, they can take control of the gun.

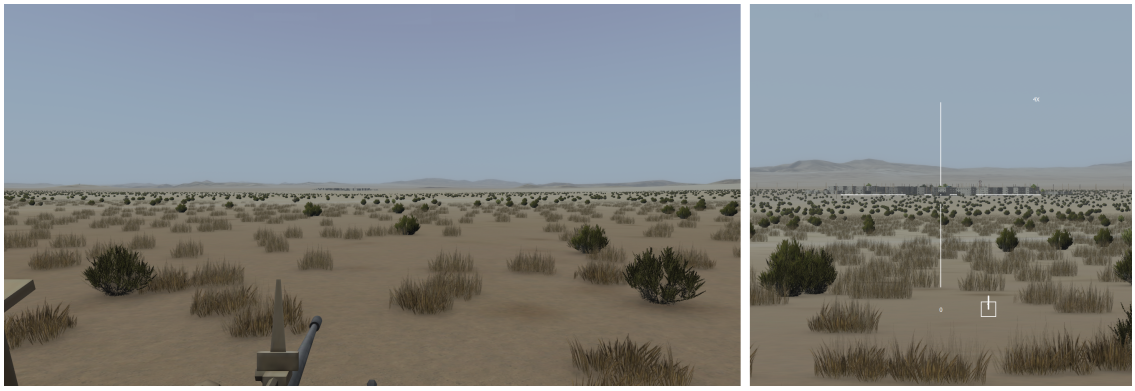
➤ To change positions, press the **H** key.

7.4.2 Switching Views in the Commander Position

There are two views for the commander when in the Commander role on a M2A3 Bradley IFV or GTK Boxer CRV: the commander hatch and commander sight.

➤ To toggle the view in the commander role, press the **H** key.

Figure 74. Hatch and Sight views on M2A3 Bradley IFV



7.4.3 Overriding the Gunner

On ground vehicles that are configured for it, the Commander role can take over the Gunner role's controls to point the gunner at a target. You might need to be in the commander sight view to override the gunner.

➤ To enter or exit Gunner Override, press the **G** key.

7.4.4 Slewing the Gun to Designate a Target

The Commander role on some vehicles can designate a target for the user in the Gunner role (slew to cue). This requires two players on separate workstations: one acting as the commander and another acting as the gunner.

➤ **To slew the gunner's gun to a specific point:**

1. Use the controls to focus your eyepoint on the target.
2. Press **C** on the keyboard. You take temporary control of the gun, slewing it to point at the target.
3. Release the key to return control to the gunner.

7.4.5 Using the Laser Designator

On the GTK Boxer CRV, the Commander role can use the laser designator to lase a target when in the RWS position. (Lasing requires line of sight.) The default when embarking on the vehicle is the out-the-hatch position. See "7.4.1 Changing Positions" on the previous page to learn about switching between these two positions.

To move the gun, you can use the right thumbstick on a game controller, mouse, or keyboard. See "7.3.3 Moving a Turret or Gun" on page 189.

➤ **To use the laser designator:**

1. Point the gun at what you want to target.
2. Press the **L** key or open the Action menu and then toggle Lasing to On.

➤ **To specify a laser code:**

1. Open the Action menu and then choose **Laser Code**.
2. Use the keyboard to enter a number and then press **Enter**.

➤ To stop lasing, open the Action menu and then toggle Lasing to Off.

7.5. Using the Intercom

Players who are embarked on a ground vehicle can communicate using the intercom. The intercom is voice activated and transmits any sound that is greater than the VOX threshold.

➤ **To use the intercom:**

1. Open the Action menu.
2. Choose **Radio**.
3. Cycle through the options to choose **Intercom**.
4. Start talking.

7.6. Changing the Sensor for a Vehicle Crew Member

The Gunner and Commander roles can change their sensor mode using the keyboard or game controller.

➤ To change the sensor for a gunner or commander, press the **M** key to cycle through the available sensors.

➤ To change the sensor for a gunner, use right thumbstick (press) until you see the sensor that you want. The gunner can also choose the sensor from the Action menu.

➤ To toggle the sensor for a commander, press the D-pad down button.

7.7. Pivoting a Tank

Tanks can pivot in place.

➤ **To pivot a tank:**

1. Start the engine.
2. Press the X button on the game controller or **C** on the keyboard. The Gear setting changes to Pivot in the HUD overlay.
3. Put the throttle forward to engage the engine.
4. Use the right thumbstick to pivot left or right.



You can also enable or disable Pivot by toggling the setting in the Action menu.

7.8. Towing Other Ground Entities

The driver of some ground vehicles can tow other ground vehicles and some ground entities. When you are positioning the vehicle to tow the entity, you might find it helpful to see behind you. See "7.2.6 Using the Backup Camera View" on page 187.

When creating a scenario in VR-Forces, you can view whether an entity is capable of towing or being towed by checking the tags in the Create Object Palette.

➤ **To tow a ground entity while in the driver role:**

1. Position your vehicle in front of the entity you want to tow. When you are in range to hook up and tow the entity, an icon displays () and the Action menu opens to the Towing entry.
2. Press **H** on the keyboard or use an Action menu command to select the option. The icon changes to show that you have hooked the other entity for towing ().
3. Drive the vehicle and the towed entity as desired.

7.9. Emergency Lights and Sirens

The police cruiser and ambulance entities have emergency lights and sirens.

➤ **To turn on emergency lights and sirens:**

1. Open the Action menu.
2. Choose **Emergency Lights** or a specific type of siren.
3. Toggle the option on.



8. Controlling Aircraft

This chapter explains how to control aircraft.

8.1. Introduction 198
8.1.1 Aircraft Roles 198
8.1.2 The Head-Up Display 198
8.1.3 The Instrument Panel 200
8.1.4 Aircraft Display Options 202
8.2. Taking Control of an Aircraft 203
8.3. Using Autopilot 204
8.3.1 Setting the Autopilot Mode 204
8.4. Controlling Landing Gear 205
8.5. Controlling Flaps 205
8.6. Controlling the Engine on Rotary-Wing Aircraft 205
8.7. Selecting a Bullseye 206
8.8. Using the Radar Warning Receiver 207
8.9. Launching Countermeasures (Chaff and Flare) 207
8.10. Choosing a Weapon 208
8.11. Air-to-Ground Combat 209
8.11.1 Targeting an Air-to-Ground Weapon Using MGRS Coordinates 209
8.11.2 Targeting an Air-to-Ground Weapon Using a Laser Code 209
8.12. Air-to-Air Combat 210
8.12.1 Selecting Targets 211
8.12.2 Firing an Air-to-Air Weapon 213
8.12.3 Focusing the Sensor on the Radar Target 214
8.13. Using the Common Operating Picture 214
8.13.1 Displaying COP Information for an Entity 214
8.13.2 Toggling the COP Focus Area 215
8.13.3 Selecting an Entity for the COP 215

8.1. Introduction

VR-Engage allows you to control fixed-wing aircraft and rotary-wing aircraft. VR-Engage uses the RTDynamics RotorLib and FixedWingLib libraries to provide realistic flight dynamics and aircraft controls. VR-Engage includes mappings for various controllers; see "12.8. Flight Controllers" on page 255 for details. (You can configure VR-Engage to work with other controllers. See "12.12. Configuring Game Controllers and Keyboard Control" on page 272.) You can also use the keyboard to control an aircraft in a very simple manner; see "12.1.6 Aircraft Controls" on page 247.

When you are flying an aircraft, you have an out-the-window head-up display (HUD) and an instrument panel. VR-Engage also supports a 3D cockpit. For more information about display options, see "8.1.4 Aircraft Display Options" on page 202. In models that are configured with articulated parts, flaps and control services move.

For a list of aircraft included with VR-Engage, see "Chapter 13. Controllable Entities" on page 291. You can add more aircraft. For information, see "Chapter 14. Adding and Editing VR-Engage Entity Models" on page 335.

8.1.1 Aircraft Roles

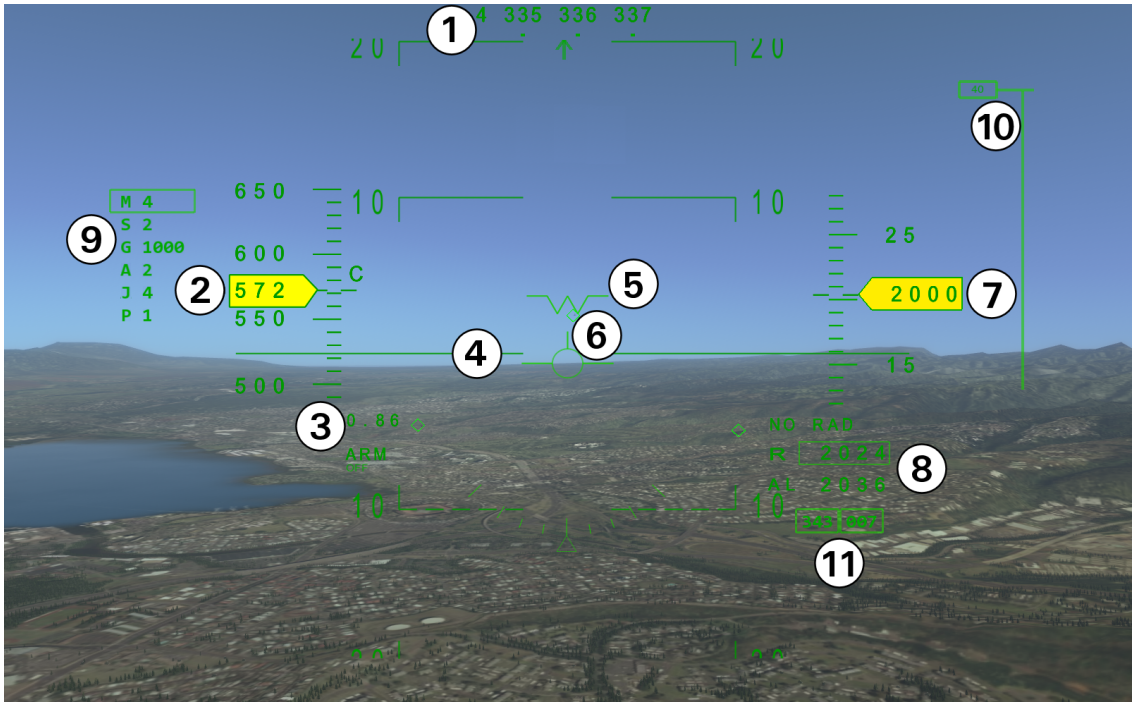
All piloted aircraft support the Pilot role, while UAVs support the Sensor Operator role. Many piloted aircraft also support the Sensor Operator role. Some aircraft support other roles that you can select from the Choose Role Panel.

To learn what roles are available for each aircraft, see "Chapter 13. Controllable Entities" on page 291.

8.1.2 The Head-Up Display

The head-up display (HUD) provides an out-the-window view with an overlay showing important flight instrument information.

Figure 75. Fighter fixed-wing aircraft HUD



The information shown includes:

- 1 – Heading Indicator
- 2 – Airspeed Indicator
- 3 – Mach Number (fixed wing only)
- 4 – Attitude Indicator (Artificial Horizon)
- 5 – Flight Path Indicator - direction aircraft is facing
- 6 – Flight Path Indicator - direction of aircraft's movement (fixed wing only)
- 7 – Altimeter
- 8 – Altitude (R = above ground level; AL = above mean sea level)

Aircraft entities with weapons display additional information:

- 9 – Weapon System Counts
- 10 – Launch Success Zone (for selected weapon system)
- 11 – Bearing and Distance to Bullseye

The weapon system counts display only the weapons configured for the entity. The list can include up to seven weapon systems.

The full acronyms or names of the weapon systems are shown in the Weapons and Stores multi-function display in the instrument panel (Figure 76 below) and the Action menu.

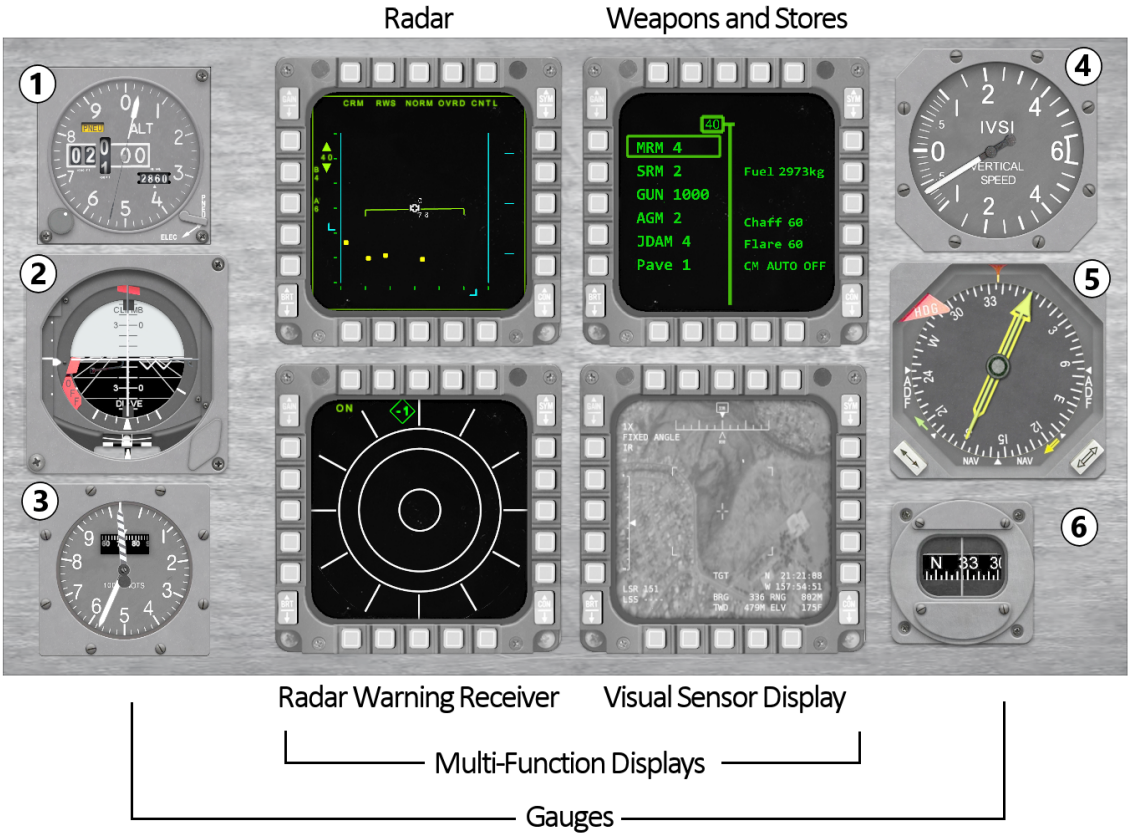
The simple Launch Success Zone shows only the weapon's minimum range, maximum range, and range to the currently selected target. The number indicates the weapon's maximum range in nautical miles.

8.1.3 The Instrument Panel

The Instruments Only display shows the instrument panel (Figure 76), which has six gauges and four multi-function displays (MFD). (The 1 Screen Horizontal display shows six gauges and two MFDs on the instrument panel.)

In the 2 Screen display, the instrument panel is an independent window. You can minimize and maximize it. Sometimes it is hidden behind the HUD. If that happens, press **Alt**+**Tab** to bring it to the front.

Figure 76. Instrument panel



The gauges are:

- 1 – Altimeter
- 2 – Attitude Indicator (Artificial Horizon)
- 3 – Airspeed Indicator
- 4 – Vertical Speed Indicator
- 5 – Heading Indicator
- 6 – Compass

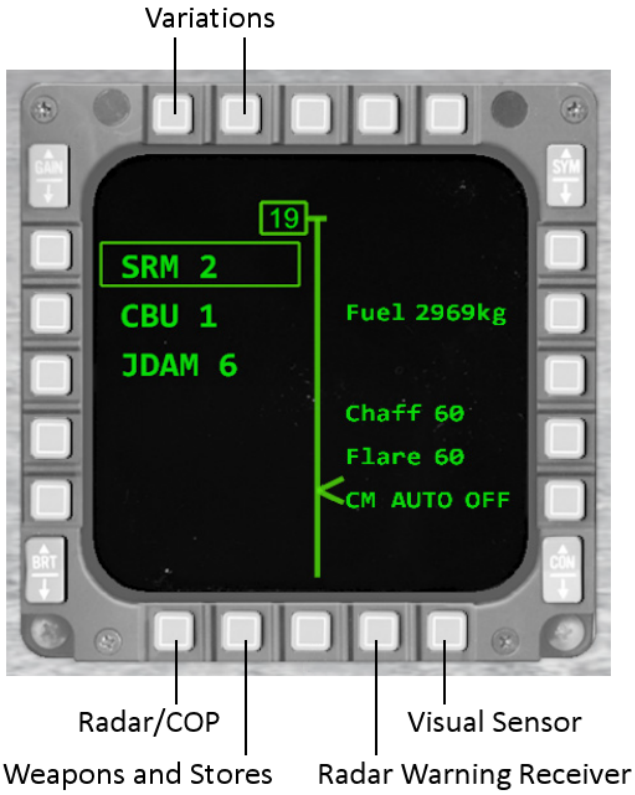
The MFDs are:

- Tactical Situation Display (radar or radar system).
- Weapons/Stores Control Panel.
- Radar Warning Receiver. Shows entities that are detecting you on their radar. The rings show the effective threat range.
- Visual Sensor Display. Displays the view from the gimbaled sensor or a plan view display of the scenario.

Changing the Multi-Function Display

Each MFD window can display any of the possible display types.

Figure 77. Multi-function display



➤ To select a display, click one of the buttons along the bottom of the display.

➤ To view a variation of the display (radar, radar warning receiver, and visual sensor), use either of the first two buttons at the top of the display.

- If the MFD display does not change when you click the buttons, the instrument panel might not have focus. Maximize it or click it at least once to ensure that it has focus.
- You cannot change the multi-function display in a 3D cockpit.

8.1.4 Aircraft Display Options

VR-Engage has options for displaying the head-up display (HUD) and instruments panel:

- 1 Screen Horizontal. HUD and instrument panel are integrated in one window, optimized for a single monitor in horizontal (landscape) orientation.
- 2 Screen. The HUD and instrument panel are in separate windows. If you have two monitors you can move the instrument panel into the second monitor to see both displays at the same time. If you choose this option and are using one monitor, the HUD is full screen and the instrument panel is in a separate window (which might be hidden behind the HUD).
- 2 MFD Panel. Displays the HUD and two MFDs.
- HUD Only. Displays the HUD only, optimized for a monitor in horizontal (landscape) mode.
- Instruments Only. Only the instrument panel displays.
- 3D Cockpit. A 3D cockpit that has the HUD and instruments (Figure 78). In this display option, you cannot change the multi-function display. The 3D cockpit is not available for all aircraft. (In VR-Engage 2.2, it is available for the Beechcraft MC-12 King Air, Beechcraft MC-12 King Air Single Seater, and T38-C Talon jet.)
- VR Headset. Select this option if you are using a head-mounted display. (Available only if an HMD is connected.)
- MR Headset. Select this option if you are using a mixed reality head-mounted display. That is, an HMD with a camera attached so that you can see both the virtual world and real world objects such as your hands and joysticks. (Available only if an HMD is connected.)

Figure 78. 3D cockpit



➤ **To choose an aircraft display option:**

1. In the Choose Role Panel, select the aircraft you want to control.
2. In the Display(s) list, select the display option.

> If you experience problems with a display mode, make sure that vrEngage.exe is included in the BlackApps.dat file. See "A.1. Performance and Display Issues" on page 394.

8.2. Taking Control of an Aircraft

To take control of an aircraft, you select it in the Choose Role Panel, just as you would for a human character or a vehicle.

It is beyond the scope of this manual to teach you how to fly a fixed-wing or rotary-wing aircraft. For a brief tutorial on flying the aircraft in VR-Engage, see the section for fixed-wing aircraft in the chapter *Controlling Entities in VR-Engage* in the *VR-Engage First Experience Guide*.

You can use an HMD while controlling an aircraft. Hand controllers are not supported for flying aircraft. You must still control them using a game controller or flight controller.

> If you take control of an aircraft that is in the air, you must start flying the aircraft as soon as you click Engage. Even though it may take several seconds or longer to display the HUD and instrument panel, the aircraft is already flying. If you take control of an aircraft that is close to the ground, such as a low flying helicopter, there is a risk that it could crash before the HUD is available if you do not immediately maintain its altitude.

8.3. Using Autopilot

You can set an aircraft to use autopilot while still otherwise controlling it. Autopilot has the modes:

- ALT. Set the altitude and use the current heading and speed. (Equivalent to VR-Forces Fly Altitude task.)
- HDG. Set the heading and use the current altitude and speed. (Equivalent to VR-Forces Fly Heading task.)
- SPD. Set the speed and use the current heading and altitude.
- ALT/HDG/SPD. Fly at the specified altitude, heading, and speed. If no altitude, heading, speed, or any of those are specified, use the aircraft's current settings.
- PATH. Fly along a route that you select. (Equivalent to VR-Forces Move Along Route task.)

If a helicopter is on the ground and you set it to ALT/HDG/SPD without specifying an altitude, it takes off, moves to an altitude of 100 feet and flies at its current heading. If a fixed-wing aircraft is on the ground, it does not take off. For more information about how aircraft respond to ALT/HDG/SPD, please see the Fly Heading and Altitude task in the *VR-Forces User Guide*.

If you are using a Thrustmaster HOTAS Warthog, pressing the Autopilot button when the toggle is set to PATH button causes the plane to fly a pattern hold around a waypoint called PATH, if it exists. Otherwise it flies a pattern hold around the current location.

8.3.1 Setting the Autopilot Mode

➤ **To set the autopilot mode or turn off autopilot:**

1. On the Action menu, choose **Movement > Autopilot**. The menu changes to the Autopilot menu.
2. Choose a setting. The Autopilot menu changes based on the autopilot setting:
   - If you choose **ALT**, the **Set Hold Altitude** option is added to the menu.
   - If you choose **HDG**, the **Set Hold Heading** option is added to the menu.
   - If you choose **SPD**, the **Set Hold Speed** option is added to the menu.
   - If you choose **ALT/HDG/SPD**, **Set Hold Altitude**, **Set Hold Heading** and the **Set Hold Speed** options are added to the menu.
   - If you choose **PATH**, you can select from a route in the scenario and the **Set Hold Speed** option is added to the Autopilot menu.
3. Choose the altitude, heading, or speed option that you want to set. An input dialog box opens.
4. Type the value that you want to set.
5. Press **Enter**.

8.4. Controlling Landing Gear

Fixed-wing entities can raise and lower landing gear.

➤ **To control landing gear:**

1. Open the Action menu.
2. Choose **Movement > Landing Gear**.
3. Use an Action menu control method to select Up or Down.


You can also toggle landing gear by pressing the **G** key.

8.5. Controlling Flaps

You can set the flaps on a fixed-wing entity up or down. In models that are configured with articulated parts, the flaps will move.

➤ **To control flaps:**

1. Open the Action menu.
2. Choose **Movement > Flaps**.
3. Use an Action menu control method to toggle the landing gear to Up or Down, MVR (Maneuver).


You can also control flaps using a joystick or game controller. See "12.8. Flight Controllers" on page 255 for details.

8.6. Controlling the Engine on Rotary-Wing Aircraft

For rotary-wing aircraft, you can control whether the engine (power plant) is on or off when you are in the pilot role. You can only turn the engine off or shut it off when it is on the ground.

➤ **To toggle the engine power:**

1. Open the Action menu.
2. Choose **Movement > Power Plant**.
3. Use an Action menu control method to select either On or Off.

- When you place a rotary-wing aircraft using either VR-Forces or the VR-Engage create entity feature, the engine defaults to being on. In VR-Forces, you can shut off the engine by choosing **Set > Appearance > Appearance** and setting the Power Plant On option to Off.
- If the aircraft is in the air, regardless of its setting in VR-Forces, the Power Plant is set to On when a player engages in the pilot role.

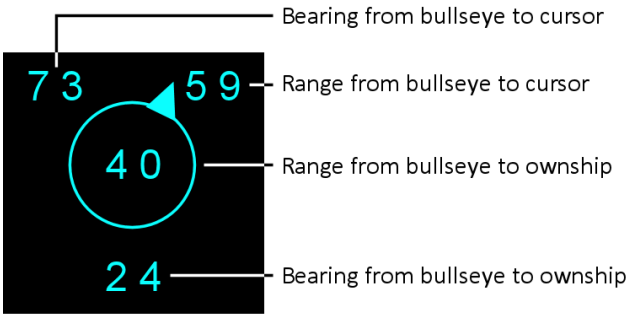
8.7. Selecting a Bullseye

Bullseyes are points that are specified before a pilot flies a mission. While the pilots are on the mission, they report their position—bearing and distance—relative to that bullseye. This allows the pilots to provide an accurate report of their location that opposing forces will not recognize.

If you do not specify a bullseye, the HUD displays the bearing and distance as 000 and 000. After you select a bullseye, the HUD displays the bearing and range of your position relative to the bullseye's coordinates (Figure 75 on page 199). The radar system displays the bearing and range from the bullseye to the cursor, and the bearing and range from the bullseye to the ownship. All ranges are measured in nautical miles, and distances are calculated using the great circle distance.

The bullseyes that you can select when engaged in a pilot role are tactical graphics added to a scenario in VR-Forces.

Figure 79. Bullseye display in radar system



➤ **To set a bullseye:**

1. On the Action menu, choose **Select Bullseye**. The Bullseyes menu opens, listing the bullseye tactical graphics defined in the scenario.
2. Use an Action menu control method to select the desired bullseye.

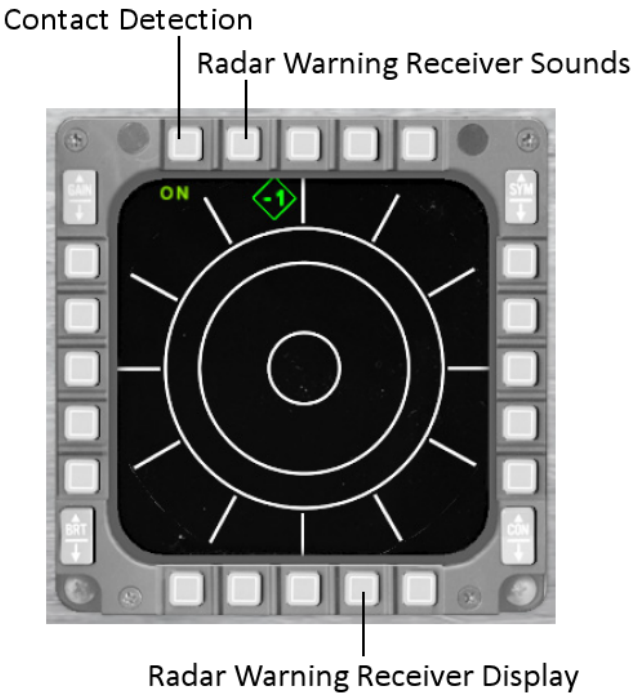
➤ **To specify which north the bullseye bearing should use:**

1. Open the Action menu.
2. Use an Action menu control method to toggle the North Reference to True, Magnetic, or Grid.

8.8. Using the Radar Warning Receiver

Rotary-wing and fixed-wing entities have a radar warning receiver (RWR). It sounds a warning if it detects radar in addition to displaying the radar contacts. You can enable or disable the RWR and the sound.

Figure 80. Radar Warning Receiver



➤ **To enable or disable the radar warning receiver:**

1. In one of the multi-function displays (MFD) of the instrument panel, click the fourth button at the bottom to show the RWR.
2. Click the first button at the top of the MFD to turn contact detection on or off.

➤ **To enable or disable the radar warning receiver sounds:**

1. Open the Action menu.
2. Choose **Sensors > RWR Sounds**.
3. Use an Action menu control method to toggle sounds on or off.



You can also toggle the radar warning receiver sounds by pressing the **M** key or clicking the second button at the top of the multi-function display.

8.9. Launching Countermeasures (Chaff and Flare)

Aircraft configured with countermeasures have the ability to automatically launch chaff and flare when they detect a threat. You can also launch chaff and flare manually.

➤ **To enable or disable countermeasures:**

1. Open the Action menu.
2. Choose **Engagement > Auto Launch CM**.
3. Use an Action menu control method to toggle automatic countermeasures on or off.

➤ To launch countermeasures manually, press the **C** key for chaff and the **F** key for flare.

8.10. Choosing a Weapon

Aircraft have specific loadouts of air-to-ground and air-to-air weapons, which display on the left side of the HUD. You can choose the weapon that you want to use. In the HUD (Figure 75 on page 199) and the Weapons and Stores display (Figure 77 on page 201), a rectangle indicates the selected weapon.

Table 8. Aircraft weapons

Weapon Type	HUD Abbreviation	Action Menu Abbreviation
Air-to-Air		
Long-Range Missiles	L	LRM
Medium-Range Missiles	M	MRM
Short-Range Missiles	S	SRM
Air-to-Ground		
Gun	G	GUN
Bombs		
Cluster	C	CBU
JDAM	J	JDAM
KAB	K	KAB
Paveway	P	Pave
Air-to-Ground Missiles	A	AGM

➤ **To select a weapon:**

1. Open the Action menu.
2. Choose **Engagement > Selected Weapon**.
3. Use an Action menu control method to cycle through the weapons available on your aircraft until the one you want to use is selected.



You can also cycle through the weapon list using the Select Weapon button on your game controller or joystick, or by pressing **R** on the keyboard.

8.11. Air-to-Ground Combat

You can fire air-to-ground bombs or missiles. By default, they try to hit wherever the sensor view is looking. You can specify a target using MGRS coordinates, or you can lase the target and look at the location being lased. Lasing can be done by a VR-Forces entity or a VR-Engage character configured with laser designator. For lased targets, you can continuously track what the laser is targeting.

8.11.1 Targeting an Air-to-Ground Weapon Using MGRS Coordinates

MGRS coordinates must specify a grid zone, square identifier, and numerical location (easting and northing). The numerical location values must have the same number of digits. For example, a code of 04QFJ190557 would be built from:

- Grid zone: 04Q
- Square identifier: FJ
- Easting: 190
- Northing: 557

In VR-Engage, if you change the coordinate system to MGRS (Display Settings dialog box, Display Units Settings page), the 3D entity label for an entity (windowed mode, 3D view) displays its location in MGRS coordinates. You can then use this value to specify the coordinates for dropping a bomb. To get the MGRS coordinates for a tactical graphic, you can view the state data in VR-Forces.

For more information about MGRS coordinates, see relevant pages available on the internet.

➤ **To fire an air-to-ground weapon using MGRS coordinates:**

1. Choose an air-to-ground weapon. See "8.10. Choosing a Weapon" on the previous page.
2. On the Action menu, choose **Sensors > Gimbaled Sensor**. The Gimbaled Sensor menu displays.
3. Choose **Target MGRS Coordinate**. The Input MGRS Coordinate dialog box opens.
4. Type the coordinates you want to target and then press **Enter**. The gimbaled sensor centers on those coordinates.
5. Select an air-to-ground weapon. See "8.10. Choosing a Weapon" on the previous page.
6. To fire the weapon, press the trigger on a joystick, the right trigger on a game controller, or **Spacebar**.

8.11.2 Targeting an Air-to-Ground Weapon Using a Laser Code

You need the laser code to target an air-to-ground weapon with that code. This might be from someone who has the information from the scenario, or another player that is lasing the target as part of the scenario.

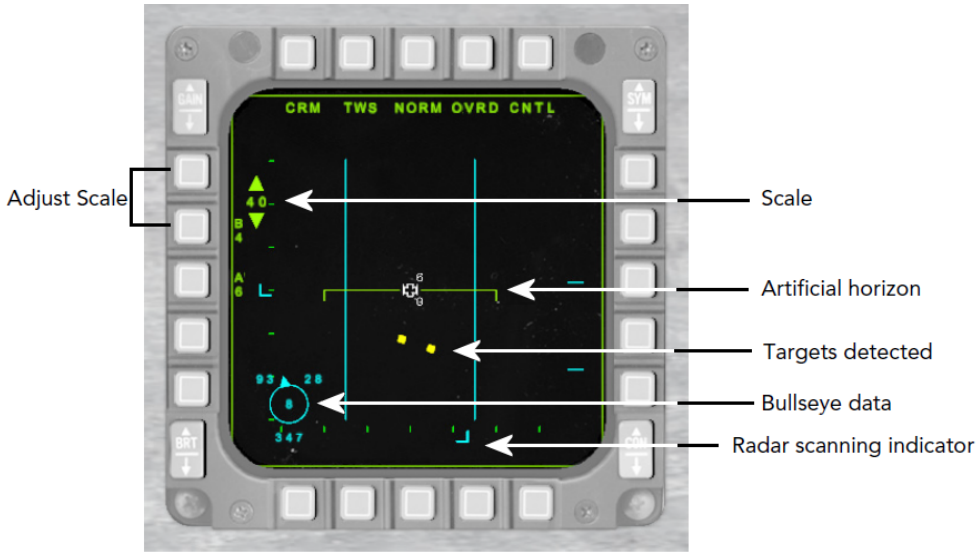
➤ **To target a laser spot:**

1. Choose an air-to-ground weapon.
2. On the Action menu, choose **Sensors > Gimbaled Sensor**. The Gimbaled Sensor menu displays.
3. Choose **Target Laser Spot**. The Input Laser Code dialog box opens.
4. Type the laser code you are looking for and then press **Enter**. The gimbaled sensor centers on the lased target or location, and the LSS code shows that code and blinks in the Visual Sensor Display (if the display is active in your instrument panel).
5. Select an air-to-ground weapon. See "8.10. Choosing a Weapon" on page 208.
6. To fire the weapon, press the trigger on a joystick, the right trigger on a game controller, or **Spacebar**.

8.12. Air-to-Air Combat

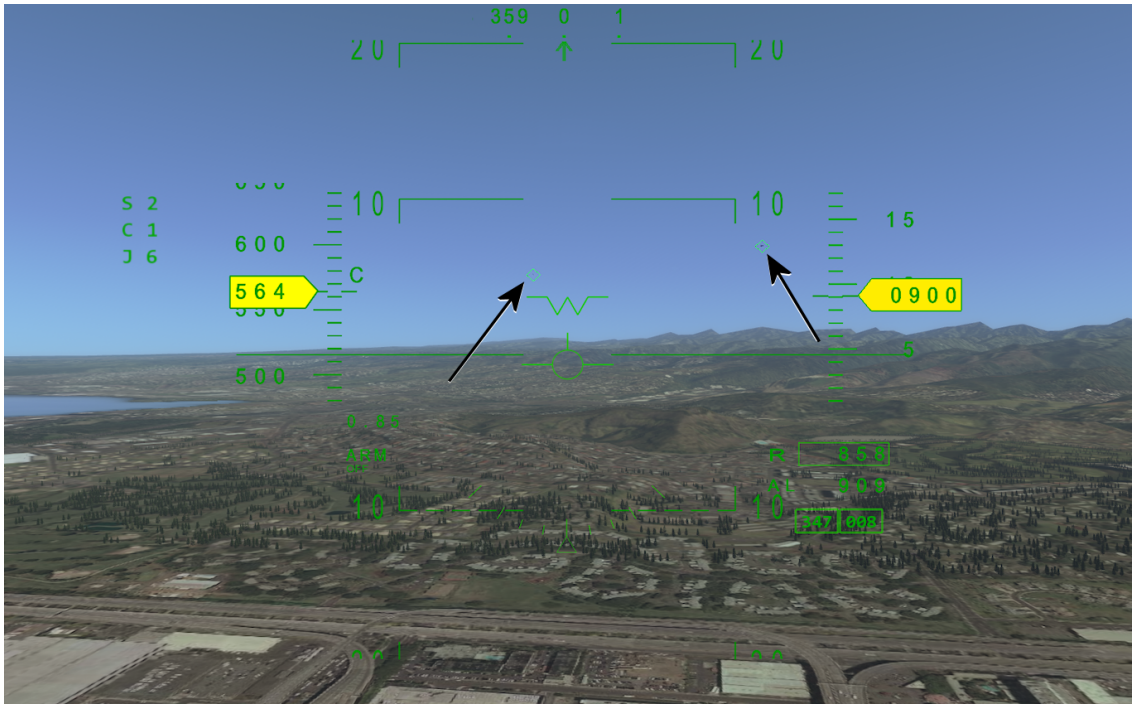
To fire a weapon at another aircraft, you must have a valid target in the Radar System (Figure 81). A blue cursor along the bottom of the Radar System shows when the radar is scanning. Yellow icons indicate detected targets. Detected contacts that are out of the selected range blink (increasing the range may shift their position and cancel the blinking).

Figure 81. Radar System



You can also see targets as green diamonds on the HUD (Figure 82).

Figure 82. Targets in HUD



8.12.1 Selecting Targets

Select a target using the cursor select controls on the joystick or keyboard. Once you have locked on a target (Figure 83), you can fire at it as long as it in range. See "Chapter 12. Mouse, Keypad, and Controller Mappings" on page 241 to learn about the cursor controls.

The Radar System has two modes: Range While Scan (RWS), which is the default setting, and Track While Scan (TWS). TWS, by default, has a narrower antenna sweep and collects more information (such as the target's velocity as well as its range). Both modes allow you to select multiple targets. To learn about targeting more than one entity, see "Selecting and Firing on Multiple Targets" on page 213.

You can see that you have locked on a target when the following things happen in the Radar System:

- The first target icon you select becomes an outline and it is circled to indicate it is the current target. Each subsequent target icon changes to an outline.
- If you select the same target a second time, the radar switches to the Single Track Target (STT) sub-mode and you cannot select another target (other targets no longer appear in the display).
- If the radar is in STT mode, the antenna tracks the designated target instead of sweeping back and forth.
- The Track ID number for the target displays.
- The Track of Interest (TOI) readout line (ROL) displays for the selected target. If you move the cursor to another target, that target's information displays in the TOI ROL. If you have not selected the target, the text is black on white. Once you select

the target, the text is white on black.

The TOI ROL includes:

- The Track ID
- Target's bearing and range from bullseye
- Target altitude
- Target heading
- Target speed (Mach number)
- Target type
- Target's bearing and range to target from your aircraft

The information for the currently selected target also displays in the HUD, and the selected target blinks (Figure 84).

Figure 83. Radar Lock in the Radar

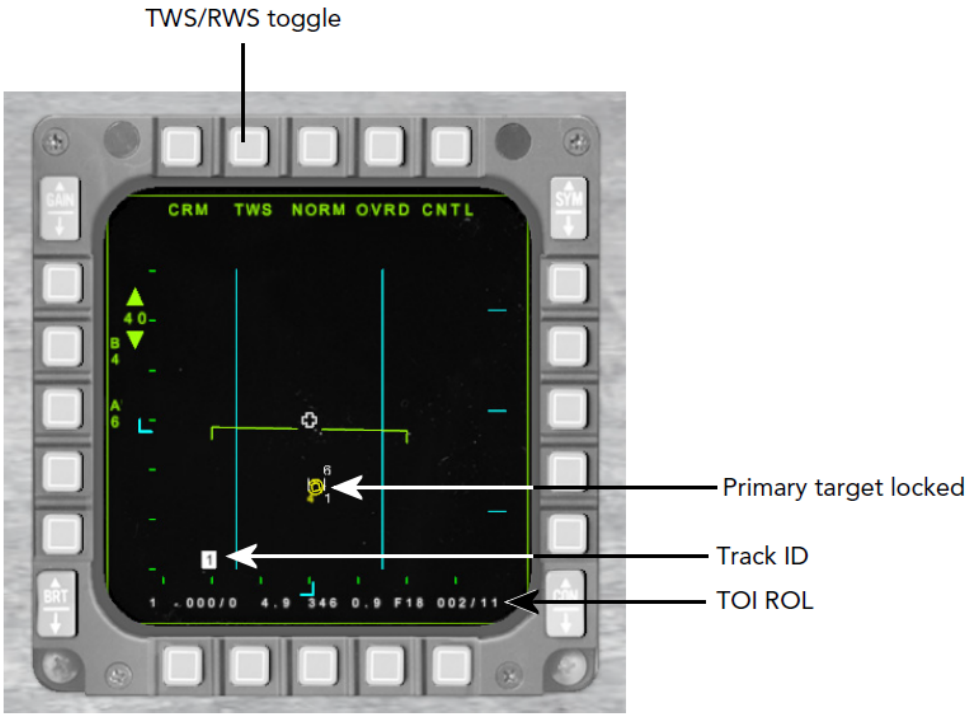
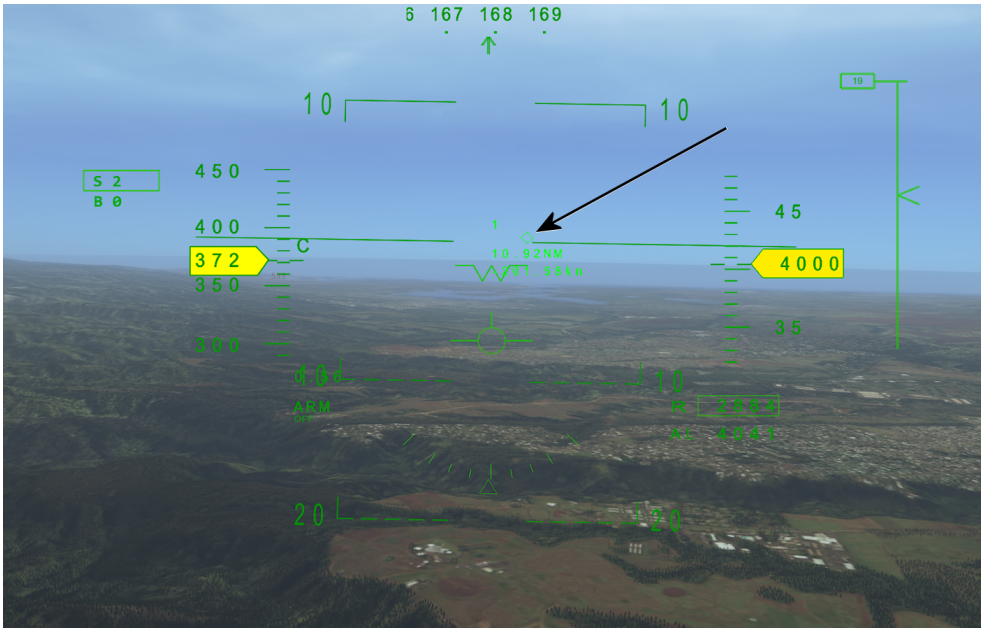


Figure 84. Radar Lock in HUD



Selecting and Firing on Multiple Targets

You can use the cursor controls in the Radar System to select multiple targets, up to six. The targets are added to an air target list in the order that you select them. This list determines the order of attack.

Each time you fire the selected air-to-air weapon, it fires at the next-to-shoot (NTS) target in the list and then cycles the list forward. The NTS target blinks in the HUD, and is circled in the radar. You can also cycle the list forward or backward using the cycling controls. For example, if you have targets 4, 5, 6, and 7, the weapon fires at the targets in order. When you fire the weapon or cycle the list forward, the order of targets becomes 5, 6, 7, and 4. If you cycle the list backward, the order is 7, 4, 5, and 6.

Targets are included in the list until they are destroyed, you unselect a target, or you clear the entire list. Additionally, changing the Radar System mode clears any selected targets and the list. You unselect a target or clear the air target list using the controls on the joystick or keyboard.

8.12.2 Firing an Air-to-Air Weapon

Air-to-air weapons use the radar for targeting.

➤ **To fire an air-to-air weapon:**

1. Choose an air-to-air weapon. See "8.10. Choosing a Weapon" on page 208.
2. Select the target. See "8.12.1 Selecting Targets" on page 211.
3. Press the trigger on a joystick, the right trigger on a game controller, or **Spacebar** to fire the weapon.

8.12.3 Focusing the Sensor on the Radar Target

You can set the gimbaled sensor to follow the current radar target.

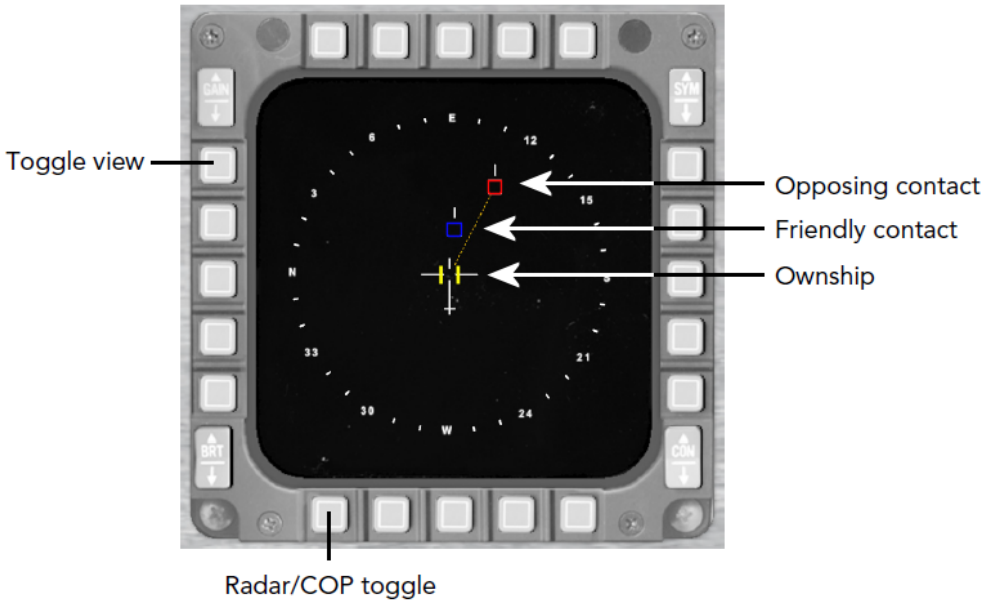
➤ **To focus the sensor on the radar target:**

1. First, target an aircraft. See "8.12.1 Selecting Targets" on page 211.
2. On the Action menu, choose **Gimbaled Sensor > Target Radar Target**. The Visual Sensor focuses on and follows the currently selected target.

8.13. Using the Common Operating Picture

The Tactical Data Link (TDL) Common Operating Picture (COP) is an alternate display to the radar system, displaying surveillance and TDL tracks. Only entities within range of your radar are displayed. In the Radar System, you can use the cursor to select which entity's engagement information to display.

Figure 85. TDL Common Operating Picture



➤ To view the TDL COP, in one of the multi-function displays (MFD) of the instrument panel, click the first button at the bottom to toggle between the Radar System and the COP.

8.13.1 Displaying COP Information for an Entity

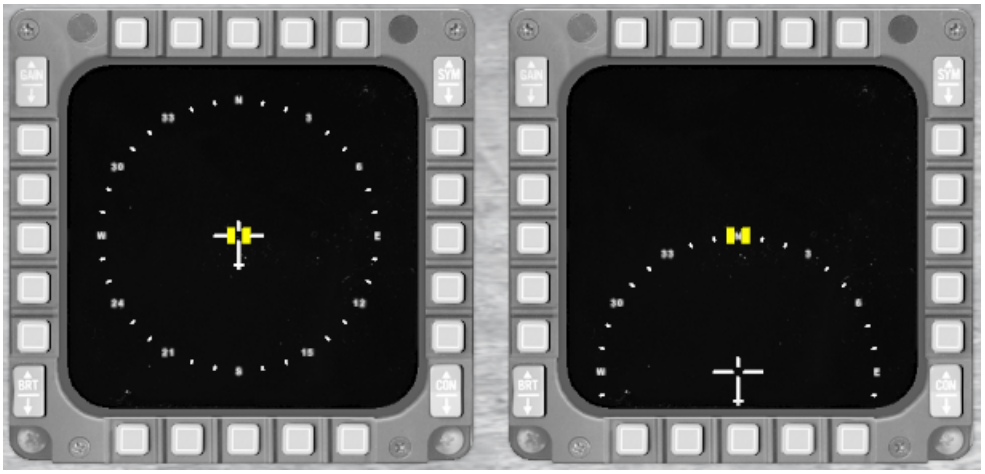
You can see the information about an entity detected by your radar system

➤ To display the information for an entity in the COP, use the controls on the joystick or keyboard to move the cursor over that entity. See "Chapter 12. Mouse, Keypad, and Controller Mappings" on page 241 to learn about the cursor controls.

8.13.2 Toggling the COP Focus Area

By default, the COP centers on the ownship entity. You can shift the focus to a point in front of the entity.

Figure 86. COP Focus Areas - Center and Forward



➤ To toggle the focus between centering on the ownship entity or a point in front of the entity, click the top button on the left side of the MFD.

8.13.3 Selecting an Entity for the COP

You can use the COP to display engagement lines between the aircraft you are flying and entities that you select with the radar.

➤ To select an entity, use the Radar System. See "8.12.1 Selecting Targets" on page 211.



9. Controlling Drones

This chapter explains how to control drones in VR-Engage. It is specific to drone entities that allow the player to fly the drone.

> VR-Engage includes other UAV entities that have only the Sensor Operator role. To learn more, see "Chapter 10. Sensor Operator Role" on page 227.

9.1. Overview 218
- 9.1.1 Drones Included with VR-Engage 218
- 9.1.2 Drone Flight Controls 219
- 9.1.3 Drone Camera Displays 219
- 9.1.4 Video and Images 221
- 9.1.5 Drone Battery Life 221

9.2. Controlling Deployable Drones 221
- 9.2.1 Deploying Drones 222
- 9.2.2 Stopping and Resuming Use of the Drone 222
- 9.2.3 Landing the Drone 223
- 9.2.4 Returning the Drone to its Starting Position 223
- 9.2.5 Recovering the Drone 223
- 9.2.6 Switching Control Between Character and Spark Quadcopter 223
- 9.2.7 Setting Control to Manual or Task 224
- 9.2.8 Changing the Visual Sensor on the LSV Quadcopter 224

9.3. Operating Independent Drones 225
- 9.3.1 Changing the Visual Sensors 225
- 9.3.2 Attacking with the Munition Drone 225

9.1. Overview

Drone operation is a feature of specific entities and roles in VR-Engage. It allows you to fly one of several quadcopter drones.

9.1.1 Drones Included with VR-Engage

Quadcopter for LSV

The Light Strike Vehicle MK II (LSV) includes a Drone Operator role. In that role, a player can deploy and control a quadcopter equipped with a camera that supports sensor modes (night vision, infrared, and so on). When players deploy the drone, the observer view changes to the drone's camera view, filling the window. Instead of piloting the drone, the player can yield control to a VR-Forces user or another player who can then give the drone tasks to perform. The Drone Operator continues to control the camera.

Figure 87. Quadcopter looking at LSV



Spark Quadcopter

The three entities that can deploy the Spark quadcopter drone are:

- Civilian Female With Drone
- Civilian Male With Drone
- Police Officer With Drone

After you deploy the drone, it appears and the perspective switches from first person to third person (Figure 88). It also displays the drone's camera view (Figure 89 on page 220). The gamepad and keyboard switch to controlling the drone by default instead of the human, but you can choose which one to control. As the drone moves, the default is for the human character to pivot and watch the drone, keeping it in the center of the main screen. You can also change this.

Figure 88. Civilian Male with Drone and drone camera view



Independent Quadcopter Drones

The other controllable drones are not deployed by a player role. Users can create them, or they can be included in a scenario and then controlled by a player or CGF. Like the LSV's quadcopter, all of these include cameras that use the full screen, using somewhat different status indicators (see Figure 90 on page 221).

One of the drones is equipped with a munition. Players can fly it into targets, at which point it explodes and deals damage.

9.1.2 Drone Flight Controls

The controls for piloting drones are covered in "12.1.4 Drone Controls" on page 245 and "12.4. Game Controller Mappings for Drones" on page 250.

9.1.3 Drone Camera Displays

The drone camera view shows what the drone can see and also indicates the status of the drone. The operator can pan the camera up or down, and also zoom in or out.

➤ **To operate the drone's camera:**

1. Add or select a character with a drone.
2. Engage the character.
3. To pan the camera up and down:
   - Use the game controller's D-pad (up/down).
   - Press the **Up arrow** and **Down arrow** keys.
4. To zoom in or out:
   - Use the game controller's right shoulder button or left shoulder button.
   - Press the **Page Up** and **Page Down** keys.

Deployable Drone View

The Spark quadcopter's default display is as a phone screen (Figure 89). The LSV's quadcopter has the same status indicators but is a first-person view (Figure 87 on page 218).

Figure 89. Drone camera view and status indicators



- 1 – Cross-hairs
- 2 – Current speed: vertical and horizontal
- 3 – Platform information: entity name, location, and height above terrain
- 4 – Camera and focus information
- 5 – Pitch
- 6 – Reticle
- 7 – Compass
- 8 – Battery charge
- 9 – Position information: horizontal distance from operator, altitude above ground level, and heading

On the Spark quadcopter, you can change the view. The default is a third-person view of the character with the drone's camera display at the top right (Figure 88 on the previous page). You can also view the drone's camera as the first-person view, like the LSV's quadcopter. Finally, you can view only the human in third-person view.

➤ To change the interface display, press the **F6** key to cycle through the three views.

Independent Drones' First Person View

Other drones use a first-person view (FPV) that fills the screen. Its status indicators are not quite the same as those for the deployable drones; they are similar to those for the sensor operator role.

Figure 90. First person drone view and status indicators



- 1 – Cross-hairs
- 2 – Current location
- 3 – Platform information: entity name, location, and height above terrain
- 4 – Camera and control information
- 5 – Pitch
- 6 – Reticle
- 7 – Compass

9.1.4 Video and Images

You can use the drone to stream video and capture images. See "Chapter 11. Video Stream and Image Capture" on page 237.

9.1.5 Drone Battery Life

The deployable drones have a limited battery life. The current charge status displays in the drone's camera view (Figure 89 on the previous page). If the battery charge falls below about 5% while it is in the air, the drone descends to the ground.

9.2. Controlling Deployable Drones

To deploy and operate a quadcopter drone, you must engage as one of the three human entities with the drone (Civilian Female With Drone, Civilian Male With Drone, or Police Officer With Drone), or embark on the Light Strike Vehicle MK II (LSV) in the Drone Operator role. The MAK One Demo scenario includes a Civilian Male With Drone entity in the neutral (green) force.

> You can create one of the drone operator entities to use if you do not want to use the MAK One Demo scenario. To learn how to create entities, see "3.2.2 Adding an Entity to a Session" on page 65.

9.2.1 Deploying Drones

Spark Quadcopter

When you deploy the drone in one of the human entity roles, you use the keyboard or game controller to control the drone. To control the human character, see "9.2.6 Switching Control Between Character and Spark Quadcopter" on the facing page.

➤ **To deploy and operate the Spark quadcopter:**

1. Add or select a character with a drone.
2. Engage the character.
3. Open the Action menu and choose **Deployable Objects > Spark > Deploy**.
4. Use the keyboard or game controller to operate the drone. See "9.1.2 Drone Flight Controls" on page 219.

LSV's Quadcopter

When you deploy the LSV's quadcopter, you can control it manually. You can also yield control to allow a VR-Forces user or another player, allowing them to assign tasks to the drone. To learn more, see "9.2.7 Setting Control to Manual or Task" on page 224.

➤ **To deploy and operate the LSV's quadcopter:**

1. Embark on the Light Strike Vehicle MK II (LSV) in the Drone Operator role.
2. Open the Action menu and choose **Deployable Objects > Quadcopter > Deploy**.
3. Use the game controller or the keyboard to operate the drone. See "9.1.2 Drone Flight Controls" on page 219.

9.2.2 Stopping and Resuming Use of the Drone

While the drone is deployed, you can stop or resume use of the drone.

➤ **To stop or restart using the drone:**

1. Open the Action menu and choose **Drone** to open the Drone submenu.
2. Choose the appropriate option:
   - **Stop Using Drone**. Closes the drone's camera view and leaves the drone in its current location. You can control the human character with the game controller.
   - **Use Drone**. Resumes using the drone. The camera view reopens.

9.2.3 Landing the Drone

You can tell the drone to land itself at its current location.

➤ To land the drone, open the Action menu and then choose **Deployable Objects > *drone name* > Land**, where *drone name* is Spark or Quadcopter. It descends to the ground.

9.2.4 Returning the Drone to its Starting Position

You can send a command to the drone to have it return from where it is currently located to the starting point where it was first deployed.

➤ To return the drone to its starting location, open the Action menu and choose **Deployable Objects > *drone name*> Return to Home**.

The drone camera view closes and the drone flies itself back to its starting position.

9.2.5 Recovering the Drone

When you are finished using the drone, you can command it to return and stow itself.

➤ To recover the drone, open the Action menu and choose **Deployable Objects > *drone name* > Recover**.

The drone camera view closes and the drone flies itself to your location from where it was. Afterward, its battery is immediately recharged to full.

ⓘ If the drone is low on battery power, it cannot fly itself to you. You can move to the drone's location on the ground and then recover it.

9.2.6 Switching Control Between Character and Spark Quadcopter

When engaged in one of the human with drone roles, you can change whether you are controlling the human character or the drone.

➤ **To change which entity you are controlling:**

1. Open the Action menu and then choose **Deployable Objects > Spark**.
2. Use an Action menu control to toggle the **Gamepad Control**:
   - Human: Use the keyboard and game controller to control the human character.
   - Drone: Use the keyboard and game controller to control the drone.

You can also choose whether the human character's focus is on the drone.

➤ **To change the character's focus:**

1. Open the Action menu and then choose **Deployable Objects > Spark**.
2. Use an Action menu control to toggle the **View Focus**:
   - Human: The human character no longer keeps the drone in the center of the screen.
   - Drone: The human character's focus is on the drone.

9.2.7 Setting Control to Manual or Task

When controlling the LSV's quadcopter, you can manage whether you control the drone or if another person can assign it tasks. For example, a player in the LSV Drone Operator role can continue to watch the quadcopter's first-person camera view while a VR-Forces user gives it tasks to perform, such as following a route.

To learn how to assign tasks using VR-Forces, refer to the *VR-Forces User Guide* or help. To learn how VR-Engage players can assign tasks to an entity, see "6.10. Tasking Other Entities" on page 176.

➤ **To change the LSV's quadcopter control setting:**

1. Open the Action menu and then choose **Deployable Objects > Quadcopter**.
2. Use an Action menu control to toggle the **Drone Control**:
   - Manual: The player in the LSV's Drone Operator role can operate the drone. No other players can assign it tasks.
   - Task: A VR-Forces user (or another player) can assign tasks to the quadcopter. The player in the LSV's Drone Operator role can no longer fly the drone. They maintain control of the camera and sensors, and can use their Action menu.

9.2.8 Changing the Visual Sensor on the LSV Quadcopter

The LSV's quadcopter camera can view the scenario in visual mode, NVG mode, or IR mode (either white hot or black hot).

➤ **To change the visual sensor on the LSV's quadcopter:**

1. Open the Action menu and then choose **Deployable Objects > Quadcopter**.
2. Use an Action menu control to cycle through the **Sensor Mode** options to the one that you want.

9.3. Operating Independent Drones

The drones that are not deployed have some features that are not shared by the Spark and LSV quadcopters.

9.3.1 Changing the Visual Sensors

Other pilotable (non-deployed) drones can view the scenario in out-the-window, NVG, or IR mode.

➤ **To change the visual sensor:**

1. Open the Action menu.
2. Use an Action menu control to cycle through the **Sensor Mode** options to the one that you want.



You can also change the sensor mode by pressing the **M** key.

9.3.2 Attacking with the Munition Drone

The DJI Mavic 3 Small Munition allows the person operating this entity to fly it into another entity and inflict damage. Depending on the targeted entity, it may take damage or be destroyed. (While you can fly other drones into targets, that only destroys the drone.)



10. Sensor Operator Role

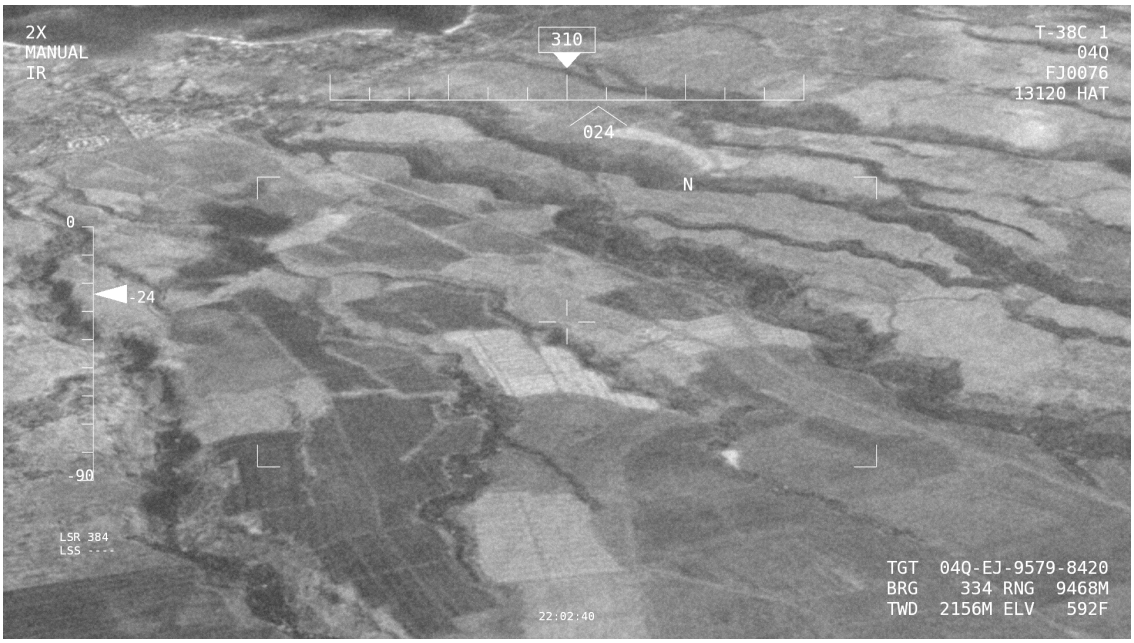
This chapter explains the sensor operator role in VR-Engage.

10.1. The Sensor Operator Role 228
10.1.1 Focusing the Sensor 229
10.1.2 Changing the Sensor Mode 230
10.1.3 Resetting the Sensor 230
10.1.4 Video and Pictures 230
10.1.5 Defining a Laser Code 230
10.1.6 Lasing Targets 231
10.1.7 Firing Weapons 231
10.2. Creating a Remote Camera Entity 231
10.3. Placing Sensors on Remote Entities 232
10.3.1 Configuring the Location of the Remote Camera 232
10.4. Configuring the Sensor Operator Overlay 234

10.1. The Sensor Operator Role

VR-Engage players can control sensors and remote cameras using the sensor operator role. The sensor operator can slew the sensor, zoom in and out, and can focus it on specific locations and entities.

Figure 91. Sensor operator view



You can add remote cameras to a scenario and set their heading and pitch. VR-Engage is also configured to automatically add cameras to remotely simulated (that is, non-VR-Forces) F-18 and UAV entities.

- You can expand the remote cameras to other entities by modifying the componentAttachmentTable.mtl in the SMS.
- Gimbaled sensors get their range from the laser controller and the sysdef file to ensure that the range is accurate.

If a sensor is mounted on an entity, the sensor operator does not control the entity. Therefore, you need another player to operate the entity, it must have a VR-Forces plan or task, or it must be mounted on a remote entity.

➤ **To operate a camera:**

1. In the Choose Role Panel, select a camera or an entity whose sensor you want to control.
2. In the Role list, choose Sensor Operator.
3. Click Engage. The window changes to 3D mode and a sensor overlay displays.

4. To zoom in and out, use the game controller's right thumbstick (forward/back), press **Page Up** or **Page Down**, scroll the mouse wheel, or use your joystick.
5. To slew the sensor, use the game controller's left thumbstick, press the arrow keys, move the mouse, or use your joystick.

10.1.1 Focusing the Sensor

You can focus the sensor on:

- The center of the screen.
- On a location specified by an MGRS coordinate or laser designation code.

  When you specify an MGRS coordinate or laser designation code, the sensor focuses on that spot. However, if a laser spot moves, the sensor does not follow it.
- On a waypoint. If a scenario has waypoints, you can focus on them.
- On a street. If a terrain has shapefiles that have street names, you can focus on a street. The sensor tracks the closest vertex. It must be within five kilometers of the X,Y coordinates of the sensor.

➤ **To set an MGRS target or laser code:**

1. Engage an aircraft as the sensor operator or engage a remote camera.
2. On the Action menu, choose **Gimbaled Sensor**. The Gimbaled Sensor menu opens.
3. Choose **Target MGRS Coordinate** or **Target Laser Spot**. A data entry dialog box opens.
4. Type the code.
5. Press **Enter**.

➤ **To track a waypoint:**

1. Engage an aircraft as the sensor operator or engage a remote camera.
2. On the Action menu, choose **Gimbaled Sensor**.
3. On the Gimbaled Sensor menu, choose **Track Waypoint**. A list of waypoints opens.
4. Use an Action menu control method to select the waypoint you want to track.

➤ **To track a street:**

1. Engage an aircraft as the sensor operator or engage a remote camera.
2. On the Action menu, choose **Gimbaled Sensor**.
3. On the Gimbaled Sensor menu, choose **Enter Street Name**. A text entry dialog box opens.
4. Type the name of the street you want to track.
5. Press **Enter**.

➤ To focus the sensor on the location currently at the center of the screen, press **L** or the A button on your game controller.

10.1.2 Changing the Sensor Mode

A sensor operator can view the scenario in visual mode, NVG mode, or IR mode.

You can set the available modes in *./data/simulationModelSets/VREngage/roles/sensorOperator.lua* by editing the **sensorModes** array. When you change the sensor mode, it cycles through the available modes in the order specified in the array.

➤ **To change the sensor mode:**

1. Engage an aircraft as the sensor operator or engage a remote camera.
2. Open the Action menu and then choose **Gimbaled Sensor**.
3. On the Gimbaled Sensor menu, use an Action menu control to cycle through the **Mode** options to the one that you want.



You can also change the sensor mode by pressing **M** or the X button on a game controller.

10.1.3 Resetting the Sensor

When you reset the sensor, it returns to a fixed angle at the neutral azimuth and elevation.

➤ **To reset the sensor:**

1. On the Action menu, choose **Gimbaled Sensor**.
2. On the Gimbaled Sensor menu, choose **Reset Sensor**.



You can also reset the sensor by pressing **R** or the Y button on a game controller.

10.1.4 Video and Pictures

You can use the sensor operator to stream video and capture images. See "Chapter 11. Video Stream and Image Capture" on page 237 for details.

10.1.5 Defining a Laser Code

You can define the laser code used when you lase a target. Laser codes can be up to four digits long. See "10.1.6 Lasing Targets" on the facing page to learn how to lase a target.

➤ **To define the laser code:**

1. On the Action menu, choose **Set Laser Code**. Input Laser Code dialog box opens.
2. Enter a laser code and then press **Enter**. The LSR code displays in the sensor view.

Your laser designator uses that code.

10.1.6 Lasing Targets

Sensor operators can lase targets if they have line of sight to the target. When you lase a target, the display shows that the laser is on. If you enable laser designators, a green circle and laser code displays on the lased location. See "6.14.3 Enabling Laser Designators" on page 181.

➤ **To lase a target or turn off lasing:**

1. Focus the sensor on a target or location. See "10.1.1 Focusing the Sensor" on page 229.
2. On the Action menu, choose **Lasing**.
3. Use an Action menu control method to toggle Lasing on or off.

10.1.7 Firing Weapons

When you engaged as a sensor operator, you can use the weapons available on the aircraft. The typical case is to fire at your own laser designator. You can also target a laser designator that another entity is currently lasing if you have line of sight on the designator spot.

➤ **To target, select, and fire a weapon:**

1. To fire at your own laser designator:
   a. Define your laser code (see "10.1.5 Defining a Laser Code" on the previous page).
   b. Focus the sensor and lase the target (see "10.1.6 Lasing Targets" above).

   To fire at another entity's laser designator, choose **Gimbaled Sensor > Target Laser Spot** and then enter that laser code. The LSS code displays in the sensor view.
2. In the Action menu, choose **Selected Weapon** and then use an Action menu control method to cycle through weapons until the one you want is displayed (Table 8 on page 208).
3. In the Action menu, choose **Fire Selected Weapon**.


You can also press **W** to cycle the weapons, and **F** or the right trigger to fire the weapon.

10.2. Creating a Remote Camera Entity

When you host a session or run an unhosted session, you can add a remote camera (or other cultural feature), and attach it to (embark on) an entity.

➤ **To create the remote camera:**

1. In the Choose Role Panel, select the Create a New Entity option. The window lists the entity types that you can create.
2. Optionally, use the Categories list to filter for Cultural Feature.

3. Optionally, switch from the 3D view to the Plan View display mode (2D) by selecting an option at the bottom of the panel.
4. Select the Remote Camera entity.
5. Optionally, in the Marking box, you can type a name for the character. The maximum length is 11 characters.
6. Optionally, specify a heading. For remote cameras, the heading is relative to the object it is attached to.
7. Optionally, specify the initial pitch.
8. Click on the terrain or entity where you want to place the camera. (You can navigate the Plan View or 3D view using your mouse and keyboard. For details about navigating the terrain, see the chapter *Moving the Observer* in the *VR-Vantage User Guide*. If you click on an entity, this attaches the camera to the entity.
9. If you simply want to observe the simulation from the perspective of the selected character but not actually control it, select the As Spectator check box.
10. If you want to use VR-Engage in full-screen mode, select the Full Screen check box.

    If you want to use VR-Engage in windowed mode, clear the Full Screen check box.
11. Click Engage. You can now control your character and participate in the scenario (assuming you did not choose to be a spectator).

10.3. Placing Sensors on Remote Entities

VR-Engage uses the VR-Forces remote component attachment feature to automatically place gimbaled sensors on remotely simulated entities. A remotely simulated entity is an entity that is part of an exercise, but is simulated by an application other than VR-Forces. The sensor is attached to the remote entity and moves with it. You can use the sensor operator role to operate the sensor. You cannot operate the remote entity.

The entities to which VR-Engage automatically attaches sensors are configured in *./data/simulationModelSets/VR-Engage/extra/componentAttachmentTable.mtl*.

The location where the sensor is embarked on the remote entity is configurable in the VR-Forces Simulation Object Editor.

If your scenario uses multiple simulation engines, then by default, the remote sensor is simulated on the first one.

10.3.1 Configuring the Location of the Remote Camera

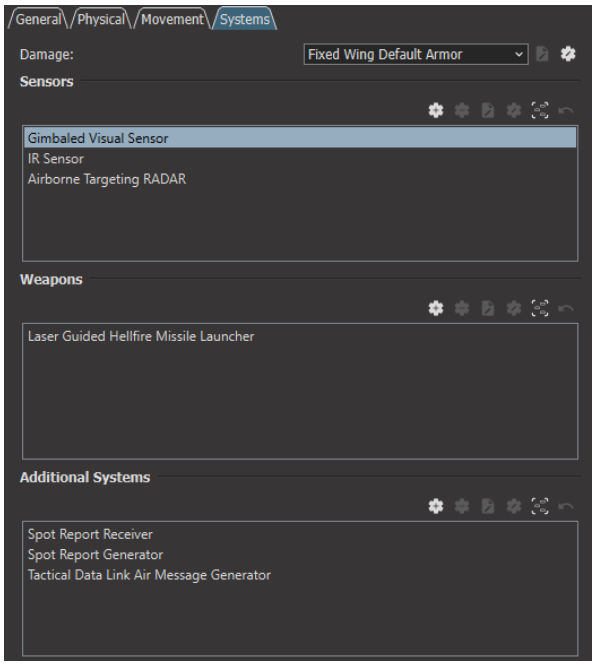
VR-Engage places a sensor on a remote entity at a location relative to the center of the entity. You can specify an offset from the default location.

➤ **To specify the offset for placing a sensor on a remote entity:**

1. In a version of VR-Forces that has the VR-Engage extensions installed, open the Simulation Object Editor.
2. Open *VR-Engage.sms*.

3. In the list of objects, select the VR-Forces entity that you want to configure to use a remote camera.
4. On the right, select the Systems tab.

Figure 92. Gimbaled Visual Sensor system



5. Select the Gimbaled Visual Sensor system.
6. Click the Properties button ( ). The Gimbaled Visual Sensor Properties dialog box opens.

Figure 93. Gimbaled Visual Sensor Properties



7. Edit the Sensor Location parameters.
8. Click OK.
9. Save the SMS.
10. Copy the updated SMS to your VR-Engage installation's *./data/simulationModelSets* directory (or you can simply copy the *./data/simulationModelSets/VR-Engage/vrfSim/Remote Sensor.entity* file).

10.4. Configuring the Sensor Operator Overlay

The information displayed on the sensor operator overlay is controlled by keywords in the following lines in *./data/simulationModelSets/VR-Engage/roles/sensorOperator.lua*:

```
...
  components = {
  ...
    ["vrvSensorOverlayUpdater"] = {
      --List of possible values for "sections":
      -- "Crosshair"         - Displays 4 corners of a box.
```

```
    -- "PlatformInfo"      - Displays information about
                             ownship entity.
    -- "VrePlatformInfo"   - Replaces "PlatformInfo". Adds Date and
                             Time displays.
    -- "TargetInfo"        - Displays ownship coordinates.
    -- "VreTargetInfo"     - Replaces "TargetInfo". Replaces
                             coordinate display with center of
                             sensor's view.
    -- "VreMgrsTargetInfo" - Replaces "VreTargetInfo". Displays
                             coordinates in MGRS.
    -- "DisplayInfo"       - Displays sensor view information,
                             including mode, tracking mode
                             and zoom level.
    -- "VreDisplayInfo"    - Replaces "DisplayInfo". Strips off
                             sensor mode "(CAMERAFX)" or
                             "(SENSORFX)" suffixes.
    -- DisplayLaser"       - Displays Tracking laser code and
                             aircraft laser code.
    -- "Pitch"             - Displays ladder showing sensor elevation
                             relative to platform.
    -- "Compass"           - Displays ladder showing sensor azimuth
                             relative to platform, and "N" indicator
                             displaying platform heading.
    -- "Reticle"           - Displays cross in the middle of
                             the screen.
    -- "DateTime"          - Displays date and time.
    -- "Time"              - Displays only time.
    -- "Rec"               - Displays static text showing "REC".
    componentType = "DtVrvSensorOverlayUpdater";
    priority = 10;
    --Defaults to VRV sections, with VRE updates for some sections.
    sections = {"Crosshair", "VrePlatformInfo", "VreTargetInfo",
      "VreDisplayInfo", "LaserInfo", "Pitch", "Reticle", "Compass"};
  };
};
includes={
  "devices/radio.pilot.lua";
};
```

The overlay to use is chosen based on the object type enumeration. You can add or remove overlay components by adding or removing keywords. The keywords are as follows, referenced to Figure 94 and Figure 95:

- 1 – Crosshair.
- 2 – VreTargetInfo. Show location using latitude/longitude coordinates.
- 2 – VreMgrsTargetInfo. Show location using MGRS coordinates.
- 3 – PlatformInfo.
- 4 – VreDisplayInfo.

- 5 – Pitch.
- 6 – Reticle.
- 7 – Compass.
- 8 – Time.
- 9 – LaserInfo. You define the Laser Code (LSR) for lasing a target by choosing **Set Laser Code** from the Action menu. You specify the Laser Spot (LSS) for the weapon by choosing **Gimbaled Sensor > Target Laser Spot** from the Action menu.
- 10 – Rec. Record. This is only text. VR-Engage does not record anything.
- 11 – DateTime.

Figure 94. Sensor operator overlay for aircraft

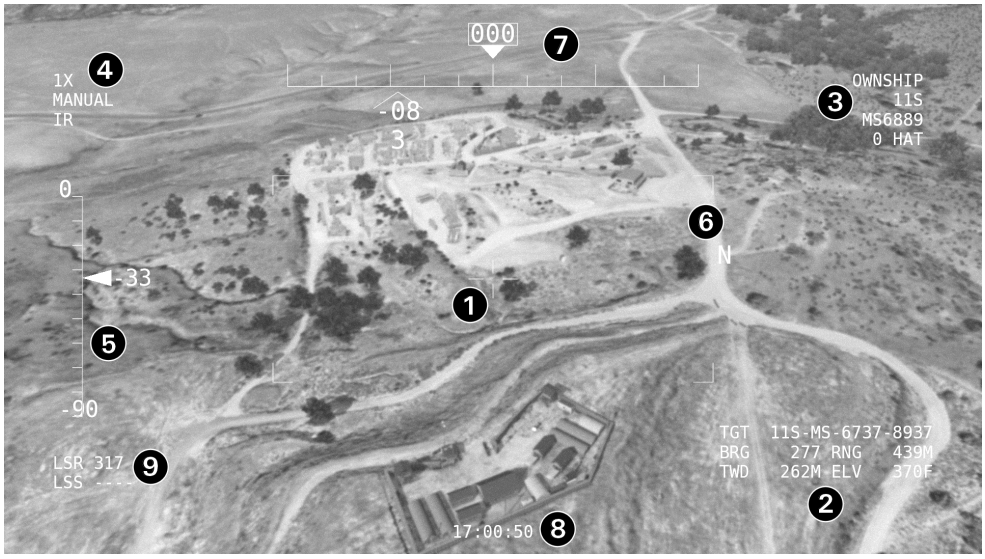
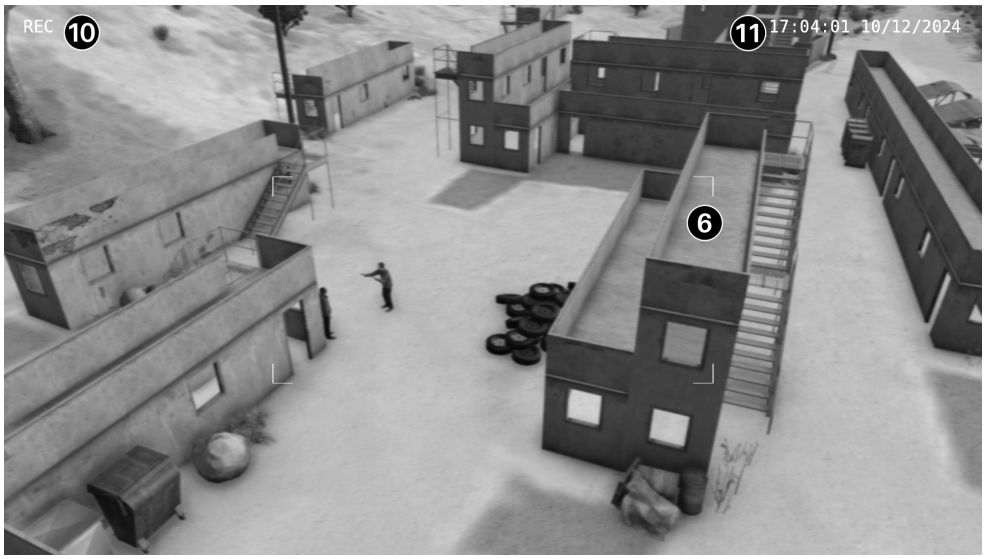


Figure 95. Sensor operator overlay for security camera



You can also write a plug-in to create new overlay elements.



11. Video Stream and Image Capture

This chapter explains how to use the Video Stream and Capture Image features for the Sensor Operator and Drone.

11.1. Introduction 238
11.2. Video Streams 238
- 11.2.1 Streaming Video as a Sensor Operator 238
- 11.2.2 Streaming Video from a Drone 238
- 11.2.3 Specifying the Video Stream File Settings 239
- 11.2.4 Viewing a Video File 239
- 11.2.5 Receiving a Video Stream 239

11.3. Capturing Images 240

11.1. Introduction

You can use the sensor operator or the drone operator roles to stream video and take pictures from their virtual sensors or cameras.

For more about those features, see "Chapter 9. Controlling Drones" on page 217 and "Chapter 10. Sensor Operator Role" on page 227.

11.2. Video Streams

Sensor operators and drones can stream video. The default is to "stream" (save) to a file, but you can also stream over a network. By default, video captures are saved to files in *./userData/videoCaptures*. See "11.2.3 Specifying the Video Stream File Settings" on the facing page for information on how to configure VR-Engage to stream video files to files in a different folder or to a network.

To play the video file or receive the video stream, you must install a media player that can process the video stream, such as VLC media player.

11.2.1 Streaming Video as a Sensor Operator

The sensor operator streams the camera's video feed.

➤ **To stream video as a sensor operator:**

1. Add or select an entity that has a sensor.
2. Engage as the Sensor Operator.
3. Open the Action menu and choose **Gimbaled Sensor**.
4. Use an Action menu control to toggle **Video Stream** to Started.

   A user can then view the file or receive the video stream. See "11.2.4 Viewing a Video File" on the facing page or "11.2.5 Receiving a Video Stream" on the facing page.

11.2.2 Streaming Video from a Drone

The deployable drones (the Spark and LSV quadcopters) can stream the full VR-Engage window. The Spark quadcopter can also stream only the drone's camera view.

After the first time you install VR-Engage, there can be a delay before video streaming begins.

➤ **To stream video from a drone:**

1. Add or select an entity with a drone.
2. Engage the character.
3. Open the Action menu and choose **Deployable Objects > Deploy Spark**.

4. From the Drone submenu, choose **Streaming Video** and use an Action menu control to open the submenu. The Streaming Video submenu opens.
5. Use an Action menu control to toggle **Video Stream** to Started.
6. Choose **Video Source** and use an Action menu control to open the submenu. The Video Source menu opens.
7. Select whether the source of the video stream should be the Main Window or the Drone View.

   A user can then view the file or receive the video stream. See "11.2.4 Viewing a Video File" below or "11.2.5 Receiving a Video Stream" below.

11.2.3 Specifying the Video Stream File Settings

VR-Engage comes with a list of pre-existing configurations for streaming video. You can access the list of available configurations either by choosing **Settings > Video Streams** in Admin mode or by looking in *./appData/videoStreams*. The default stream configuration is called *H264 HW-MPGTS-FILE*, which instructs VR-Engage to stream H264-encoded video to a file in *./userData/videoCaptures*.

You can specify which configuration VR-Engage should use by editing the **DtStreamingVideoUpdater** component in *./data/simulationModelSets/VR-Engage/roles/sensorOperator.lua* or *./data/simulationModelSets/VR-Engage/roles/devices/deployableObjects.lua* and specifying a video stream configuration.

- If you want to modify any of the pre-existing video stream configurations or create a new one, please see the chapter *Using Streaming Video* in the *VR-Vantage User Guide*.
- To access Admin mode, press **Shift+F12**. For details, see "3.1.3 Admin, User, and Lockdown Modes" on page 59.

11.2.4 Viewing a Video File

When you save a video stream to a file, you can open it in a media player such as VLC media player. By default, you can find your captured streaming video files in *./userData/videoCaptures*. Files streamed from a sensor operator begin with "sensorOperator" and files streamed from a drone begin with "drone".

11.2.5 Receiving a Video Stream

If you have configured VR-Engage to stream video over a network, you must have a viewing application installed on the machine where you want to receive and view the video.

➤ **To receive the video stream:**

1. Launch the viewing application.
2. Configure the application with the same network parameters, such as the UDP port, as those specified in the video streaming configuration that you specified VR-Engage should use.

As long as the video stream is turned on for the sensor or drone, the video stream should display in the viewing application.

11.3. Capturing Images

The sensor operator and drone have cameras that can take screen shots of the camera view and save them to your computer. When you save a screen shot, it is saved in *./userData/screenshots* as a *PNG* file.

➤ **To take pictures with the sensor operator:**

1. Add or select an entity that has a sensor.
2. Engage as the Sensor Operator.
3. Open the Action menu and choose **Gimbaled Sensor > Capture Image**.

➤ **To take pictures with the drone:**

1. Add or select a character with a drone.
2. Engage the character.
3. Open the Action menu and choose **Deployable Objects > Deploy Spark**.
4. Fly the drone to the location where you want to take a picture.
5. Open the Action menu and choose **Drone > Capture Image**.


You can also capture images using the B button on the controller.



12. Mouse, Keypad, and Controller Mappings

This chapter describes mouse and keyboard mappings and controller mappings for VR-Engage. For details about configuration additional controllers, see "12.12. Configuring Game Controllers and Keyboard Control" on page 272.

12.1. Mouse and Keyboard Mappings 242
- 12.1.1 Application Management 242
- 12.1.2 Action Menu Controls 242
- 12.1.3 Human Controls 243
- 12.1.4 Drone Controls 245
- 12.1.5 Ground Vehicle Controls 245
- 12.1.6 Aircraft Controls 247
- 12.1.7 Sensor Operator Controls 248

12.2. Game Controller Mappings for Using the Action Menu 248
12.3. Game Controller Mappings for Human Control 249
12.4. Game Controller Mappings for Drones 250
12.5. Game Controller Mappings for Vehicles with Weapons 251
12.6. Game Controller Mappings for Cars and Trucks 252
12.7. Racing Wheel Controllers 252
12.8. Flight Controllers 255
- 12.8.1 HOTAS Warthog Mappings for Fixed-Wing Entities 255
- 12.8.2 HOTAS Warthog Mappings for Rotary-Wing Entities 258
- 12.8.3 T.Flight Hotas X Mappings for Fixed-Wing Entities 260
- 12.8.4 T.Flight Hotas X Mappings for Rotary-Wing Entities 263
- 12.8.5 Game Controller Flight Control Mappings for Fixed-Wing Entities 266
- 12.8.6 Game Controller Flight Control Mappings for Rotary-Wing Entities 267
- 12.8.7 Game Controller Mappings for the Sensor Operator 267

12.9. Head-Mounted Display Controllers 268
12.10. Game Controller Button Numbering 269
12.11. Tips for Logitech Gamepad 272
12.12. Configuring Game Controllers and Keyboard Control 272

12.12.1 Windows Game Controllers 273
12.12.2 The Game Controller Utility 273
12.12.3 File Structure 274
12.12.4 Mappings 275
12.12.5 Input Types 276
12.12.6 Player Controls State 278
12.12.7 Component Mapping Groups 280
12.12.8 List of Actions 281

12.1. Mouse and Keyboard Mappings

Using the mouse and keyboard, you can control human characters, ground vehicles, and navigate the Action menu. You can also perform some aircraft behaviors.

12.1.1 Application Management

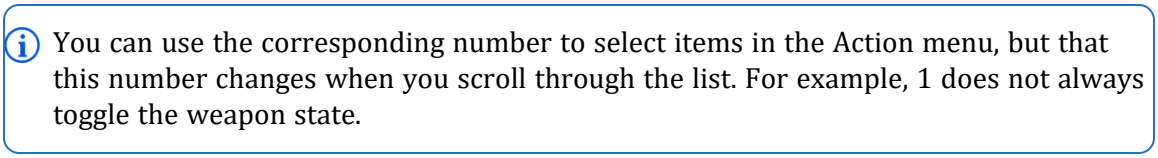
Table 9 lists mouse and keyboard controls for all roles in VR-Engage.

Table 9. Keyboard controls in all roles

Function	Key
Open display of gamepad and keyboard controls	F1
Open HTML help	Shift+F1
Toggle Action menu	~ (tilde)
Toggle full screen	Ctrl+Enter
Disable/enable mouse control	Ctrl
Exit to Application Menu / close Action menu	Esc
Toggle between Admin and User modes	Shift+F12
Reset to starting position (Admin mode only)	F5
Reset in current position (Admin mode only)	Ctrl+F5

12.1.2 Action Menu Controls

Table 10 lists keyboard controls for the Action menu.



You can use the corresponding number to select items in the Action menu, but that this number changes when you scroll through the list. For example, 1 does not always toggle the weapon state.

Table 10. Action menu mouse and keyboard mappings

Mouse Mappings	
Select menu item	Number matching menu item Enter key Mouse click (when mouse control is disabled)
Keyboard Mappings	
Toggle Action menu	~ (tilde)
Next menu item	Up arrow
Previous menu item	Down arrow
Select menu item	Right arrow or Enter
Return to previous menu	Left arrow

12.1.3 Human Controls

Table 11 lists mouse and keyboard controls for human players.

Table 11. Human player mouse and keyboard controls

Function	**Key / mouse movement**
Mouse Mappings	
Turn / move weapon	mouse move
Aim	right mouse button
Fire	left mouse button
Detect target (launchers)	T
Zoom in (binoculars)	mouse wheel forward
Zoom out (binoculars)	mouse wheel backward
Keyboard Mappings	
Movement	
Run forward	W
Sprint forward	left Shift+W
Walk forward	left Alt+W
Run backward	S
Sprint backward	left Shift+S
Walk backward	left Alt+S
Turn left	H
Turn right	K
Posture up	Q
Posture down	E

Table 11. Human player mouse and keyboard controls (continued)

Function	Key / mouse movement
Stand	C
Lay prone	Z
Toggle view mode (first- / third-person)	V
Parachuting	
Deploy parachute	Spacebar
Steer left or right	A or D
Slow rate of descent	A+D
Engagement	
Aim	**,** (comma)
Strafe left	A
Strafe right	D
Pitch weapon/look up	U
Pitch weapon/look down	J
Toggle weapon state (Stowed, Ready, Aiming)	**'** (apostrophe)
Cycle weapons	Spacebar
Select weapon	1, 2, 3, and so on (not on number pad)
Reload weapon	R
Seek mode (Javelin only)	T
Toggle Top or Direct Attack (Javelin only)	M
Wide field-of-view (WFOV, Javelin only)	scroll backward
Narrow field-of-view (NFOV, Javelin only)	scroll forward
View and Accessories	
Look (weapons are disabled while looking)	**.** (period) + either mouse or H, K, U, and J keys
Toggle night vision goggles	N
Toggle binoculars	B
Zoom in	=
Zoom out	- (dash, not on number pad)
Talk on the radio	T
Toggle flashlight	F
Toggle compass	G
Toggle minimap	M
Zoom in on minimap	+ (plus sign on number pad)

Table 11. Human player mouse and keyboard controls (continued)

Function	Key / mouse movement
Zoom out of minimap	- (minus sign on number pad)
Reset minimap zoom	0 (number pad)
Toggle HUD (weapon, health, and so on)	/

12.1.4 Drone Controls

Table 12 lists the keyboard controls for aircraft.

Table 12. Keyboard mappings for drones

Function	Key
Flight Controls	
Altitude up	Q
Altitude down	E
Fly to the right	A
Fly to the left	D
Yaw (turn) right	C
Yaw (turn) left	Z
Gimbaled Sensor Controls	
Slew camera up	Up arrow
Slew camera down	Down arrow
Zoom in	Page Up
Zoom out	Page Down
Cycle sensor modes	

12.1.5 Ground Vehicle Controls

Table 13 lists mouse and keyboard controls for ground vehicles.

Table 13. Keyboard mappings for ground vehicles

Function	Key / mouse movement
Movement and view (driver role)	
Start/stop engine	X
Throttle	W
Brake	S
Steer left	A
Steer right	D

Table 13. Keyboard mappings for ground vehicles (continued)

Function	Key / mouse movement
Shift gear up	E
Shift gear down	Q
Parking brake	P
Headlights	L
Differential lock (LAV) Pivot (tanks)	C
Cycle saved views	V
Specify saved view	1, 2, 3, 4, 5, 6 (as defined)
Cycle roles	[and]
Engagement (gunner and driver/gunner roles)	
Move turret (gunner)	mouse move
Fire weapon (gunner)	left mouse button
Cycle sensor	M
Turret left	H
Turret right	K
Turret up	U
Turret down	J
Next weapon	R
Previous weapon	V
Fire weapon	F
Commander role	
Look	mouse move
Toggle laser designator	L
Slew to cue	C
Gunner override	G
Cycle sensor	M
Reset sensor	R
Change position	H (M2A3 Bradley IFV and GTK Boxers only)
Zoom Mappings (commander and gunner only)	
Zoom in	mouse wheel forward
Zoom out	mouse wheel backward

12.1.6 Aircraft Controls

Table 14 lists the keyboard controls for aircraft. See "12.8. Flight Controllers" on page 255 for more complete controls of aircraft.

Table 14. Keyboard mappings for fixed-wing aircraft

Function	Key
Toggle radar warning receiver	M
Suppress warnings	Enter
Observer mode	V
Toggle landing gear	G
Flight Controls	
Pitch up	Down arrow
Pitch down	Up arrow
Roll right	right arrow
Roll left	left arrow
Yaw right	D
Yaw left	A
Throttle (fixed wing)	W
Air brake (fixed wing)	S
Collective up (rotary wing)	W
Collective down (rotary wing)	S
Engagement	
Cycle weapons	R
Fire selected weapon	Spacebar
Move radar cursor up	I
Move radar cursor down	K
Move radar cursor left	J
Move radar cursor right	L
Select target closest to cursor	Z
Unselect target closest to cursor	X
Cycle target list backward	Y
Cycle target list forward	H
Clear air target list	U
Replenish fuel and ammunition	O

Table 14. Keyboard mappings for fixed-wing aircraft (continued)

Function	Key
Countermeasures	
Launch chaff	C
Launch flares	F

12.1.7 Sensor Operator Controls

Table 15 lists mouse and keyboard controls for sensor operator players.

Table 15. Keyboard mappings for sensor operator

Function	Key / mouse movement
Slew sensor or camera	arrow keys mouse move
Zoom out	Page Down key mouse wheel backward
Zoom in	Page Up key mouse wheel forward
Cycle sensor	M
Track location	L
Track object	O
Reset sensor	R

12.2. Game Controller Mappings for Using the Action Menu

Figure 96 on the facing page illustrates the game controller numbering and naming conventions.

Table 16 lists the controls for using the Action menu.

Table 16. Game controller mappings - Action menu

Function	Game Controller Key / Button	Button / Axis
Toggle Action menu	Back button	Button 4
Move in menu	D-pad	Buttons 11, 12, 13, and 14
Select	A button	Button 0

12.3. Game Controller Mappings for Human Control

Figure 96 illustrates the game controller numbering and naming conventions.

Figure 96. Generic game controller button conventions

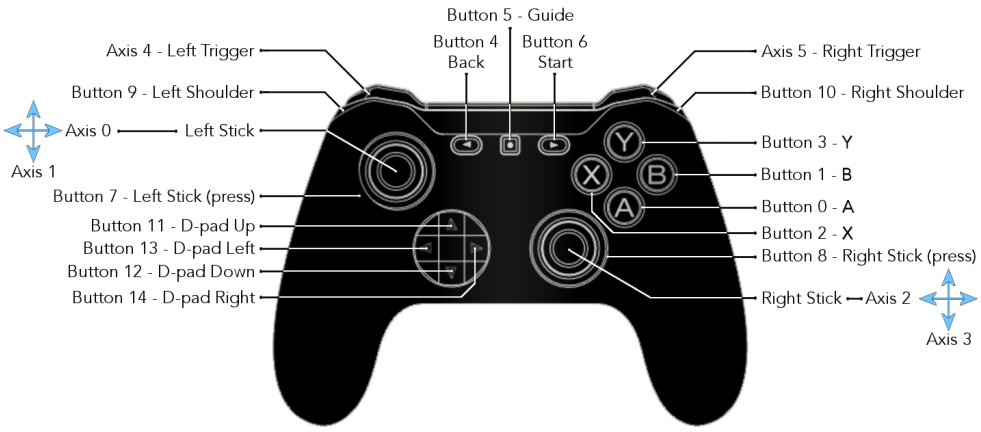


Table 17 lists the game controller buttons for human characters.

Table 17. Game controller mappings - human

Function	Game Controller Key / Button	Button / Axis
Movement and view		
Move	left thumbstick	Axis 0 and Axis 1
Sprint	left thumbstick (press)	Button 7
Turn / move weapon	right thumbstick	Axis 2 and Axis 3
Free look	right thumbstick (press)	Button 8
Posture select	B button	Button 1
Toggle binoculars	X button	Button 2
Zoom binoculars out or in	right shoulder button left shoulder button	Button 9 or Button 10
Talk on radio	A button	Button 0
Toggle night vision goggles	D-pad down	Button 12
Human Combat		
Aim	left trigger	Axis 4
Fire/lase target	right trigger	Axis 5
Cycle weapons	Y button	Button 3
Parachuting (paratroopers only)		
Deploy parachute	A button	Button 0

Table 17. Game controller mappings - human (continued)

Function	Game Controller Key / Button	Button / Axis
Steer left or right	left shoulder button right shoulder button	Button 9 or Button 10
Slow rate of descent	both shoulder buttons simultaneously	Buttons 9 and 10

12.4. Game Controller Mappings for Drones

These controls are for the drone that can be deployed by human characters: Civilian Female With Drone, Civilian Male With Drone, and Police Officer With Drone. Figure 96 on the previous page illustrates the game controller numbering and naming conventions.

Table 18 lists the controls for the drone.

Table 18. Game controller mappings - drone

Function	Game Controller Key / Button	Button / Axis
Drone Flight Controls		
Altitude	left thumbstick (forward/back)	Axis 1
Direction Facing	left thumbstick (left/right)	Axis 0
Movement	right thumbstick	Axis 2 and Axis 3
Drone Camera Controls		
Zoom out	left shoulder button	Button 9
Zoom in	right shoulder button	Button 10
Capture Image	B button	Button 1
Pan up or down	D-pad (up/down)	Buttons 11 and 12

12.5. Game Controller Mappings for Vehicles with Weapons

Figure 96 on page 249 illustrates the game controller numbering and naming conventions.

Table 19 lists the driver's controls for the vehicles with weapons (LAV, tanks, and so on).

Table 19. Driver game controller mappings - vehicles with weapons

Function	Game Controller Key / Button	Button / Axis
Vehicle Control		
Start/stop engine	Start button	Button 6
Throttle	right trigger	Axis 5
Shift gear up	right shoulder button	Button 10
Shift gear down	left shoulder button	Button 9
Brake	left trigger	Axis 4
Steering	left thumbstick	Axis 0
Differential lock (LAV) Pivot (tanks)	X button	Button 2
Headlights	A button	Button 0
Parking brake	left thumbstick (press)	Button 7
Cycle saved views	Y button	Button 3
Driver look around (LAV only)	right thumbstick	Axis 2 and Axis 3
Engagement (gunner/driver role only)		
Turret	D-pad	Buttons 11, 12, 13, and 14
Cycle weapon	left trigger	Axis 4
Fire weapon	A	Button 0

Table 20 lists the commander's controls for vehicles with weapons. (Not all tanks are configured with commander roles.)

Table 20. Commander game controller mappings - vehicles with weapons

Function	Game Controller Key / Button	Button / Axis
Rotate/elevate camera	right thumbstick	Axis 2 and Axis 3
Change sensor	D-pad down	Button 14
Zoom out	left shoulder button	Button 9
Zoom in	right shoulder button	Button 10

Table 21 lists the gunner's controls for vehicles with weapons.

Table 21. Gunner game controller mappings - vehicles with weapons

Function	Game Controller Key / Button	Button / Axis
Rotate/elevate gun	right thumbstick	Axis 2 and Axis 3
Change sensor	right thumbstick (press)	Button 8
Zoom out	left shoulder button	Button 9
Zoom in	right shoulder button	Button 10
Next weapon	left trigger	Axis 4
Previous weapon	D-pad up	Button 11
Fire weapon	right trigger	Axis 5

12.6. Game Controller Mappings for Cars and Trucks

This topic describes game controller mapping for cars and trucks. These vehicles have a driver role, but for the most part do not have weapons. Figure 96 on page 249 illustrates the game controller numbering and naming conventions.

Table 22 lists the driver's controls for cars and trucks.

Table 22. Driver game controller mappings - cars and trucks

Function	Game Controller Key / Button	Button / Axis
Vehicle Control		
Start/stop engine	Start button	Button 6
Throttle	right trigger	Axis 5
Shift gear up	right shoulder button	Button 10
Shift gear down	left shoulder button	Button 9
Parking Brake	left thumbstick (press)	Button 7
Steering	left thumbstick	Axis 0
Headlights	A button	Button 0
Drive look around	right thumbstick	Axis 2 and Axis 3

12.7. Racing Wheel Controllers

VR-Engage supports the Logitech G27 and G29 racing wheel controllers.

> If your racing wheel is not working as expected, make sure you are using the latest drivers.

Figure 97 illustrates the mappings of the G27 racing wheel.

Figure 97. G27 racing wheel controls



Table 23 lists the G27 racing wheel controls.

Table 23. G27 Racing Wheel control mappings

Function	Key/Button
Next View	Button 4 Down Toggle
Previous View	Button 3 Down Toggle
Cycle Sensor	Button 16 Down Toggle
Brake	Axis 3
Differential Lock	Button 7 Down Toggle
Engine Enable	Button 8 Down Toggle
Gear Down	Button 6 down Toggle
Gear Up	Button 5 Down Toggle
Headlights	Button 20 Down Toggle
Parking Brake	Button 22 Down Toggle
Reset	Button 23 Down Toggle
Steering	Axis 1
Throttle	Axis 2

Figure 98 illustrates the mappings of the G29 racing wheel.

Figure 98. G29 racing wheel controls

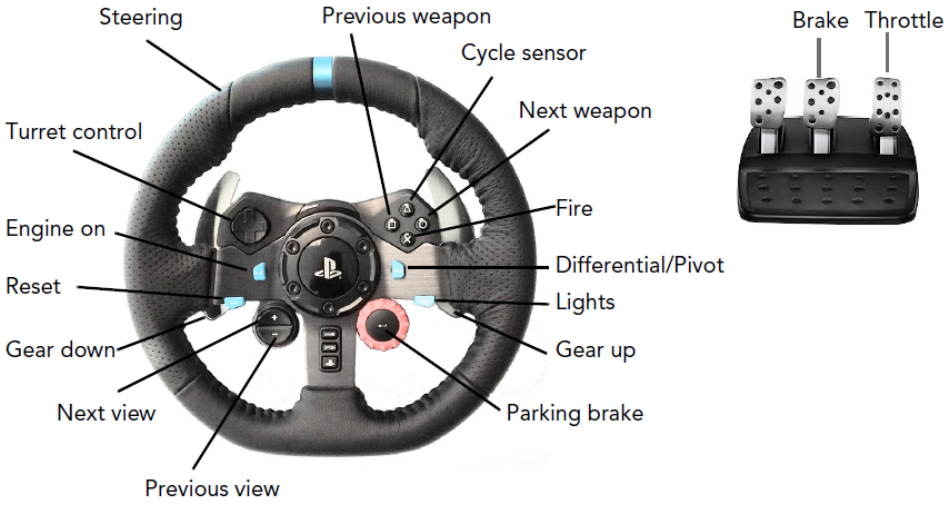


Table 24 lists the G29 racing wheel controls.

Table 24. G29 racing wheel control mappings

Function	Key/Button
Next View	Button 20 Down Toggle
Previous View	Button 21 Down Toggle
Cycle Sensor	Button 4 Down Toggle
Brake	Axis 3
Differential Lock	Button 7 Down Toggle
Engine Enable	Button 8 Down Toggle
Gear Down	Button 6 down Toggle
Gear Up	Button 5 Down Toggle
Headlights	Button 11 Down Toggle
Parking Brake	Button 24 Down Toggle
Reset	Button 12 Down Toggle
Steering	Axis 1
Throttle	Axis 2
Differential Lock	Button 7 Down Toggle

12.8. Flight Controllers

VR-Engage includes mappings for the Thrustmaster HOTAS Warthog flight (Joystick-HOTAS Warthog) controller, the T.Flight Hotas X flight controller, the Gamepad F310 and F710 (Logitech Dual Action), and Xbox controllers.

12.8.1 HOTAS Warthog Mappings for Fixed-Wing Entities

Figure 99 and Figure 100 show the mappings for the Thrustmaster Warthog flight controller for fixed-wing entities.

Figure 99. Thrustmaster HOTAS Warthog joystick mappings - fixed-wing



Figure 100. Thrustmaster HOTAS Warthog throttle mappings - fixed-wing

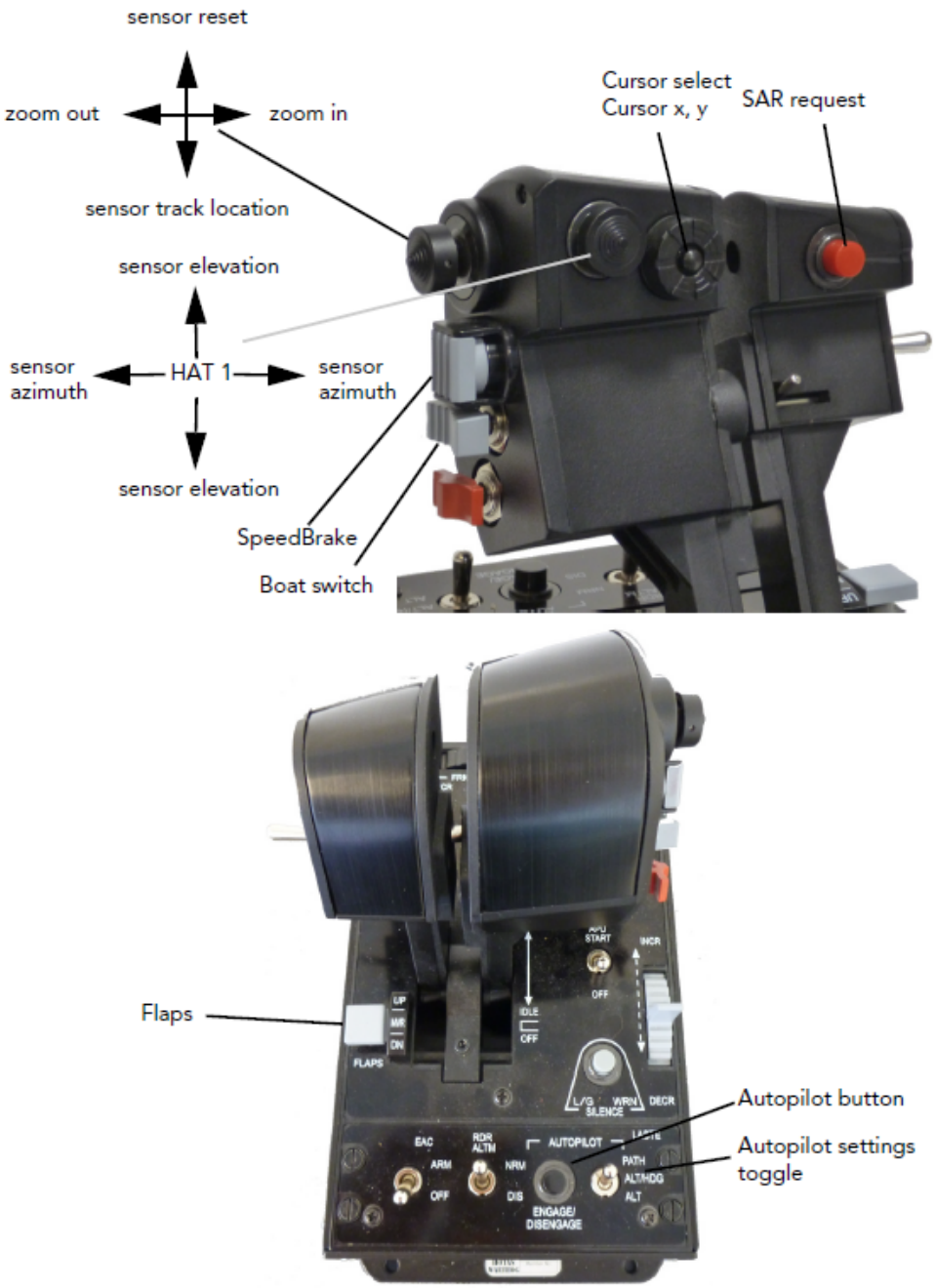


Table 25 lists the flight controls for the Thrustmaster HOTAS Warthog controller for fixed-wing entities.

Table 25. Thrustmaster HOTAS Warthog mappings - fixed-wing

Function	Key/Button
Extended Flight Controls	
AutopilotMode1	Throttle Button 27 Down Toggle
AutopilotMode2	Throttle Button 28 Down Toggle
AutopilotToggle	Throttle Button 26 Down Toggle
Flap 1	Throttle Button 22 Down Toggle. Toggle states match joystick up, maneuver, down.
Flap 2	Throttle Button 23 Down Toggle. Toggle states match joystick up, maneuver, down.
NoseWheelSteering	Button 3 Down Toggle
ToggleLandingGear	G Toggle
Engagement	
cycle weapon	Throttle Button 5 Down Toggle
fire selected weapon	Throttle Button 1 Down Toggle
replenish fuel and ordnance	Throttle Button 4 Down Toggle
FixedWingLib CGF Flight Controls	
air brake	Boat Switch (Button 10) Down. This causes excess drag and the plane's speed drops quickly.
pitch	Joystick forward and back (Axis 2)
roll	Joystick left and right (Axis 1)
throttle	Throttle lever (Axis 3)
wheel brake	SpeedBrake (Button 7) Down, button 8 Down. If you pull back and release, it re-centers. If you push forward, it locks in place. Both directions activate full brakes.
Gimbaled Sensor	
sensor-azimuth	Throttle HAT 1 Right (Up), HAT 1 Left (Down)
sensor-cycle	Joystick Button 7 Down Toggle
sensor-elevation	Throttle HAT 1 Up (Up), HAT 1 Down (Down)
sensor-reset	Throttle Button 3 Down
sensor-track-location	Throttle Button 5 Down
sensor-zoom	Throttle Button 4 (Down), Button 6 (Up)
Observer Control	
ObserverAttachMode	V (on the keyboard)
Radar Scope	
cursor-select	Button 1 (also clears the selection from the air target list)
cursor-x	Axis 1

Table 25. Thrustmaster HOTAS Warthog mappings - fixed-wing (continued)

Function	Key/Button
cursor-y	Axis 2
Clear air target list	U (on the keyboard)

12.8.2 HOTAS Warthog Mappings for Rotary-Wing Entities

Figure 101 and Figure 102 show the mappings for the Thrustmaster Warthog flight controller for rotary-wing entities.

Figure 101. Thrustmaster HOTAS Warthog joystick mappings - rotary-wing



Figure 102. Thrustmaster HOTAS Warthog throttle mappings - rotary-wing

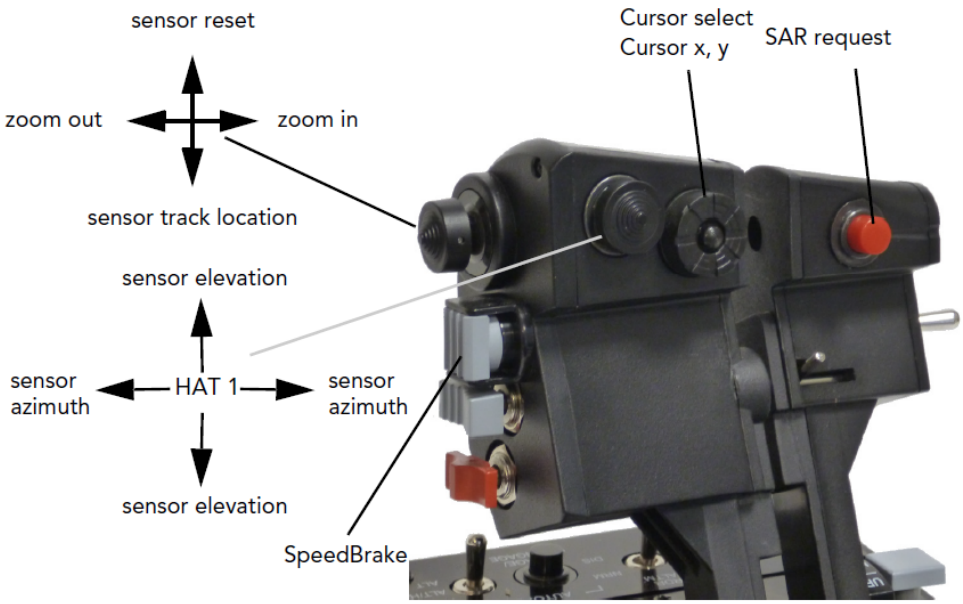


Table 26 lists the flight controls for the Thrustmaster HOTAS Warthog controller for rotary-wing entities.

Table 26. Thrustmaster HOTAS Warthog mappings - rotary-wing

Function	Key/Button
Engagement	
fire selected weapon	Button 1 Down Toggle
cycle weapon	Button 5 Down Toggle
replenish fuel and ordnance	Button 4 Down Toggle
Gimbaled Sensor	
sensor-azimuth	Throttle HAT 1 Right (Up), HAT 1 Left (Down)
sensor-cycle	Joystick Button 7 Down Toggle
sensor-elevation	Throttle HAT 1 Up (Up), HAT 1 Down (Down)
sensor-reset	Throttle Button 3 Down

Table 26. Thrustmaster HOTAS Warthog mappings - rotary-wing (continued)

Function	Key/Button
sensor-track-location	Throttle Button 5 Down
sensor-zoom	Throttle Button 4 (Down), Button 6 (Up)
Radar Scope	
cursor-select	Button 1 (also removes selection from the air target list)
cursor-x	Axis 1
cursor-y	Axis 2
Clear air target list	U (on keyboard)
RotorLib CGF Rotary Wing Controls	
collective	Axis 3, <UP>, <DOWN>
pedal	HAT 1 Up, HAT 1 Left
pitch	Axis 2, W
roll	Axis 1, D (Up), A (Down)
wheel brake	SpeedBrake (Button 7) Down, button 8 Down. If you pull back and release, it recenters. If you push forward, it locks in place. Both directions activate full brakes.

12.8.3 T.Flight Hotas X Mappings for Fixed-Wing Entities

Figure 103 shows the joystick controls for the T.Flight Hotas X flight controller for fixed-wing entities.

VR-Engage also supports the T.Flight Hotas 4 controller. The button layout is similar to the T.Flight Hotas X.

Figure 103. T.Flight Hotas X joystick - fixed wing



Figure 104 shows the throttle controls for the T.Flight Hotas X flight controller for fixed-wing entities.

Figure 104. T.Flight Hotas X throttle - fixed-wing

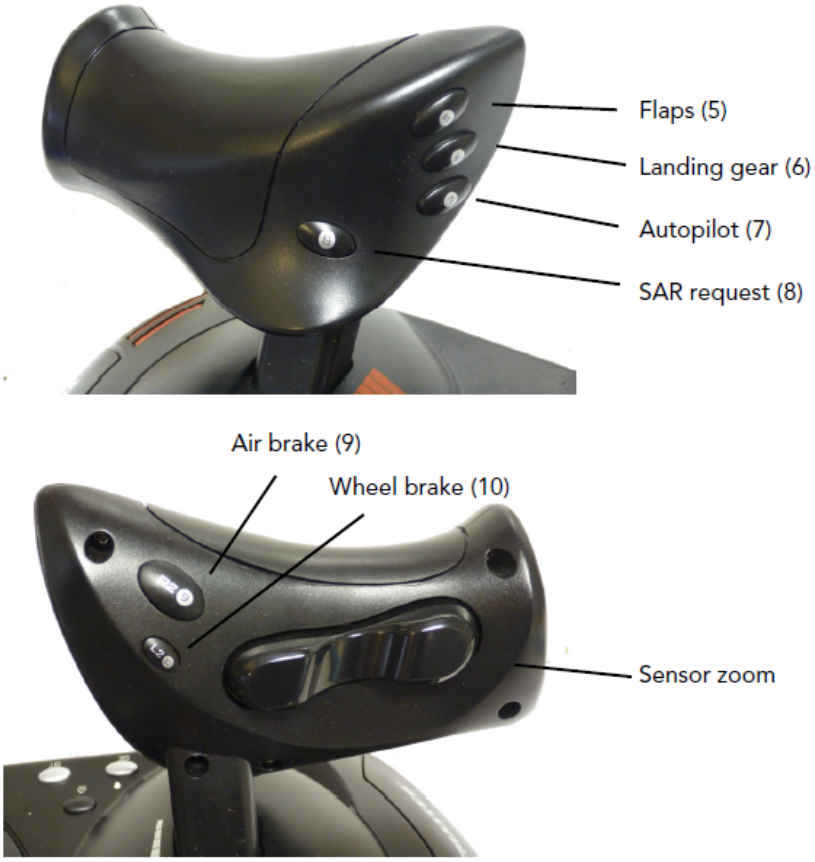


Table 27 lists the flight controls for the T.Flight Hotas X controller for fixed-wing entities.

Table 27. T.Flight Hotas X mappings - fixed-wing

Function	Key/Button
Engagement	
cycle weapon	Button 5 Down Toggle
fire selected weapon	Button 1 Down Toggle
cursor select target	Button 3 Down Toggle
FixedWingLib CGF Flight Controls	
pedal	Joystick twist left and right (Axis 4)
pitch	Joystick forward and back (Axis 2)
roll	Joystick left and right (Axis 1)
throttle	Throttle forward and back (Axis 3)
autopilot	Button 7
air brake	Button 9 Down
landing gear	Button 6

Table 27. T.Flight Hotas X mappings - fixed-wing (continued)

Function	Key/Button
flaps	Button 5
wheel brake	Button 10 Down
Menu	
Action menu	Button 4 Down
down	HAT 1 Down, <DOWN>
left	HAT 1 Left, <LEFT>
right	HAT 1 Right, <RIGHT>
up	HAT 1 Up, <UP>
Observer Control	
Attach mode	V (on keyboard)
Radar Scope	
down	HAT 1 Down, <DOWN>
left	HAT 1 Left, <LEFT>
right	HAT 1 Right, <RIGHT>
up	HAT 1 Up, <UP>
select target closest to cursor	Z (on keyboard)
clear current radar selection	X (on keyboard)
Release Bomb	
release	Button 1 Down

12.8.4 T.Flight Hotas X Mappings for Rotary-Wing Entities

Figure 105 shows the joystick controls for the T.Flight Hotas X flight controller for rotary-wing entities.

VR-Engage also supports the T.Flight Hotas 4 controller. The button layout is similar to the T.Flight Hotas X.

Figure 105. T.Flight Hotas X joystick - rotary-wing



Figure 106 shows the throttle controls for the T.Flight Hotas X flight controller for rotary-wing entities.

Figure 106. T.Flight Hotas X throttle - rotary-wing



Table 28 lists the flight controls for the T.Flight Hotas X controller for rotary-wing entities.

Table 28. T.Flight Hotas X mappings - rotary-wing

Function	Key/Button
Engagement	
cycle weapon	Button 5 Down Toggle
fire selected weapon	Button 1 Down Toggle
cursor select target	Button 3 Down Toggle
Menu	
Action menu	Button 4 Down
down	HAT 1 Down, <DOWN>
left	HAT 1 Left, <LEFT>
right	HAT 1 Right, <RIGHT>
up	HAT 1 Up, <UP>
Radar Scope	
down	HAT 1 Down, <DOWN>

Table 28. T.Flight Hotas X mappings - rotary-wing (continued)

Function	Key/Button
left	HAT 1 Left, <LEFT>
right	HAT 1 Right, <RIGHT>
up	HAT 1 Up, <UP>
select target closest to cursor	Z (on keyboard)
clear current radar selection	X (on keyboard)
RotorLib CGF Rotary Wing Controls	
collective	Throttle forward and back (Axis 3)
pedal	Joystick twist left and right (Axis 4)
pitch	Joystick forward and back (Axis 2)
roll	Joystick left and right (Axis 1)
wheel brake	Button 10 Down

12.8.5 Game Controller Flight Control Mappings for Fixed-Wing Entities

Figure 96 on page 249 illustrates the game controller numbering and naming conventions. Table 29 lists the flight controls for fixed-wing entities.

Table 29. Game controller flight control mappings - fixed-wing

Function	Game Controller Key / Button	Button / Axis
Engagement		
Cycle weapons	left trigger	Axis 4
Fire weapon	right trigger	Axis 5
Radar cursor select	left shoulder button	Button 9
Radar cursor	D-pad	Buttons 11, 12, 13, and 14
Flight Controls		
Pedal	left thumbstick (left/right)	Axis 0
Throttle	left thumbstick (forward/back)	Axis 1
Pitch	right thumbstick (forward/back)	Axis 3
Roll	right thumbstick (left/right)	Axis 2
Air brakes	A button	Button 0

Table 29. Game controller flight control mappings - fixed-wing (continued)

Function	Game Controller Key / Button	Button / Axis
Wheel brakes	X button	Button 2
Flaps	Y button	Button 3
Auto pilot	right shoulder button	Button 10

12.8.6 Game Controller Flight Control Mappings for Rotary-Wing Entities

Figure 96 on page 249 illustrates the game controller numbering and naming conventions.

Table 30 lists the flight controls for rotary-wing entities

Table 30. Game controller flight control mappings - rotary-wing

Function	Game Controller Key / Button	Button / Axis
Rotary Wing Controls		
Collective	left thumbstick (forward/back)	Axis 1
Pedal	left thumbstick (left/right)	Axis 0
Pitch	right thumbstick (forward/back)	Axis 3
Roll	right thumbstick (left/right)	Axis 2
Engagement		
Cycle weapons	left trigger	Axis 4
Fire weapon	right trigger	Axis 5

12.8.7 Game Controller Mappings for the Sensor Operator

Figure 96 on page 249 illustrates the game controller numbering and naming conventions.

Table 31 lists the controls for the sensor operator

Table 31. Game controller mappings - sensor operator

Function	Game Controller Key / Button	Button / Axis
Sensor orientation	left thumbstick	Axis 0 and Axis 1
Sensor mode	X button	Button 2
Track location	A button	Button 0
Track the entity closest to the center of the screen	B button	Button 1
Reset sensor	Y button	Button 3
Zoom	right thumbstick (forward/back)	Axis 3

12.9. Head-Mounted Display Controllers

The Oculus Rift and HTC Vive head-mounted displays support hand controllers. This section shows the mappings for these controllers, which can be used for soldier roles. If you do not have hand controllers, you can use a keyboard, Xbox controller, or Logitech Gamepad. (HMD hand controllers are not mapped in joystick settings.)

Figure 107 illustrates the controls on the Oculus Rift touch controller.

Figure 107. Oculus Rift touch controller

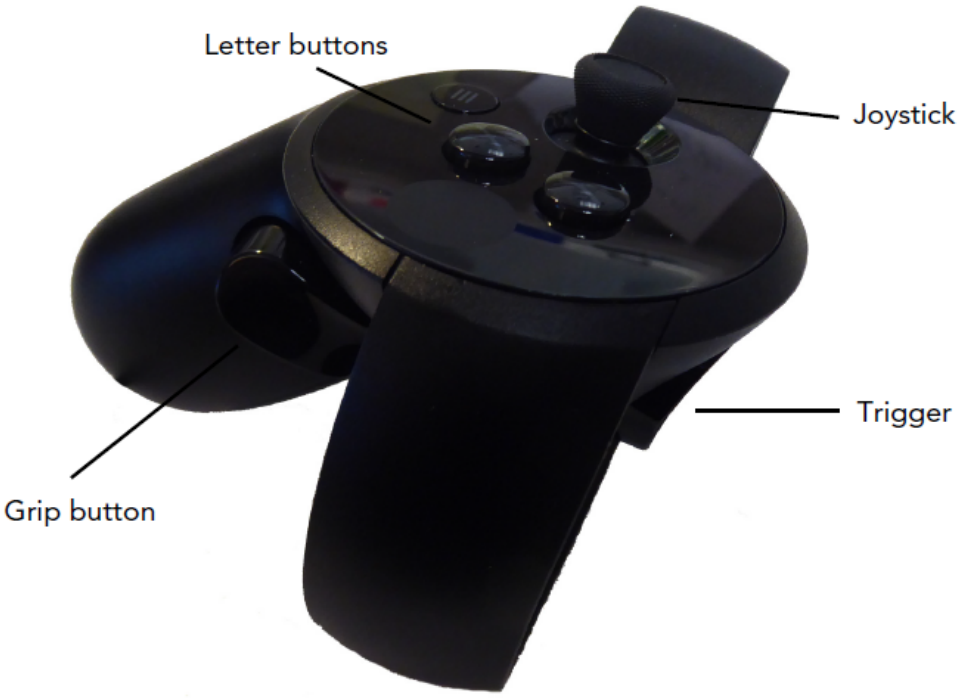
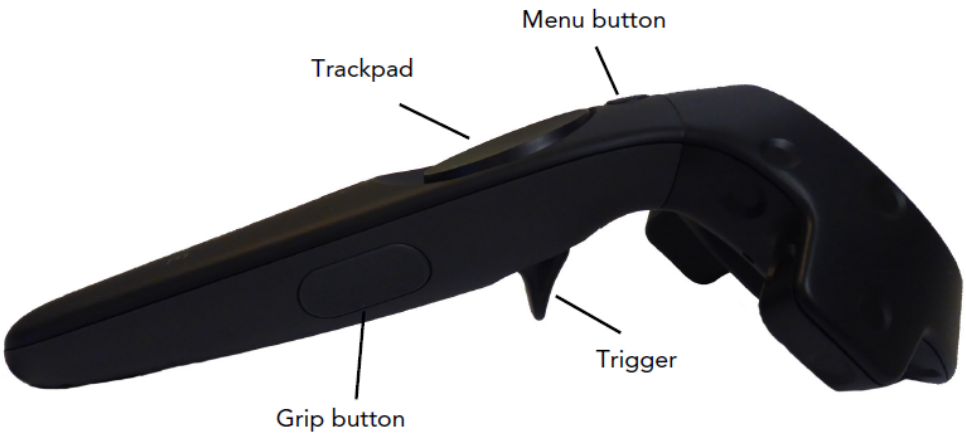


Figure 108 illustrates the controls on the Oculus Rift touch controller.

Figure 108. HTC Vive controller



The controls work as follows:

- Fire. Press the trigger on the dominant hand to fire the weapon.
- Aim. Press the grip button to raise the weapon to the aim-down-sights position.
- Move. Movement is controlled by the right hand joystick (Oculus Rift) or trackpad (HTC Vive):
  - Walk forward. Push/touch forward. Movement is based on the orientation of the entity, not the direction it is looking or aiming.
  - Walk backwards. Pull/touch backward.
  - Walk sideways. Push/touch left and right
  - Sprint. Push down while pressing forward.
- Turn. Press/touch the left hand joystick or trackpad to the left or right.
- Change posture. Press the trigger on the non-dominant hand for Posture Up. Press the grip button on the non-dominant hand for Posture Down.
- Action menu. To open, press Y on the non-dominant hand Oculus touch controller. Press the menu button on the non-dominant hand HTC Vive controller. While the menu is open, the joystick or trackpad acts as a D-pad.

12.10. Game Controller Button Numbering

Each trigger and joystick direction is considered to be an axis and can be mapped to a joystick control. Everything else is a button and can be mapped to a button control. Game controllers are mapped in *.lua* files. (For details, see "12.12. Configuring Game Controllers and Keyboard Control" on page 272.)

The mappings are identified by button and axis numbers. Figure 109 and Figure 110 identify the button and axis numbering scheme for the Xbox and Gamepad controllers.

Figure 109. Xbox controller button and axis numbering



Figure 110. Logitech Gamepad button and axis numbering

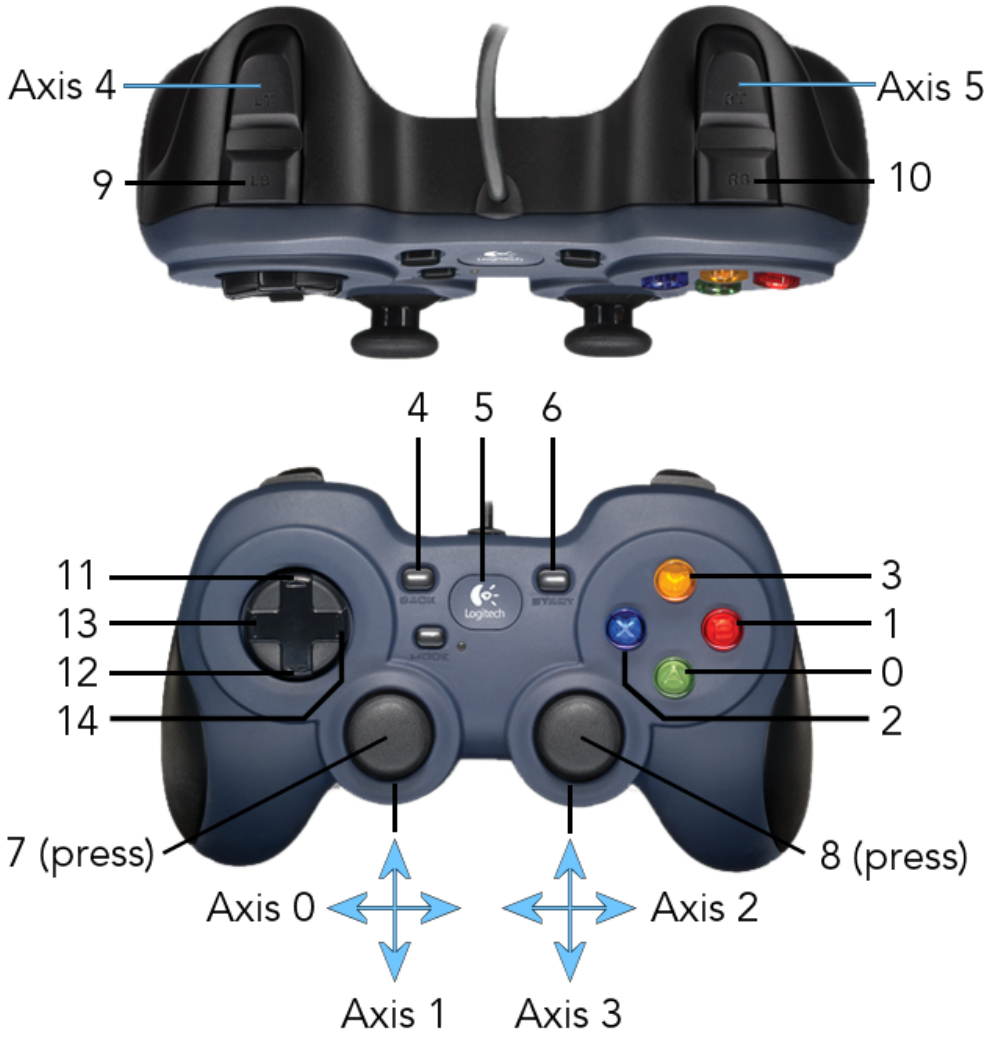


Table 32. Controller numbering

Number	Description
Button	
0	Xbox A or corresponding
1	Xbox B or corresponding
2	Xbox X or corresponding
3	Xbox Y or corresponding
4	Xbox Back or corresponding
5	Xbox Guide or corresponding
6	Xbox Start or corresponding
7	Left stick press

Table 32. Controller numbering (continued)

Number	Description
8	Right stick press
9	Left shoulder button
10	Right shoulder button
11	D-pad up
12	D-pad down
13	D-pad left
14	D-pad right
15	Miscellaneous (Xbox Series X share button, PlayStation 5 microphone button, Nintendo Switch Pro capture button, Amazon Luna microphone button)
16	Xbox Elite paddle P1
17	Xbox Elite paddle P1
18	Xbox Elite paddle P1
19	Xbox Elite paddle P1
20	PlayStation 4 / PlayStation 5 touchpad button
Axis	
0	Left stick x axis
1	Left stick Y axis
2	Right stick x axis
3	Right stick y axis
4	Left trigger
5	Right trigger

12.11. Tips for Logitech Gamepad

The Mode button switches the functions of the D-pad and the left thumbstick. Since it is easy to inadvertently press it, if the D-pad and thumbstick are not working as expected, check the Mode button. It should be off (no green light).

There is a switch on the back of the Gamepad that you can set to either D (DirectInput) or X (XInput). Either input should work to control VR-Engage entities as described in the previous sections. If you experience issues with X, set the controller to D.

12.12. Configuring Game Controllers and Keyboard Control

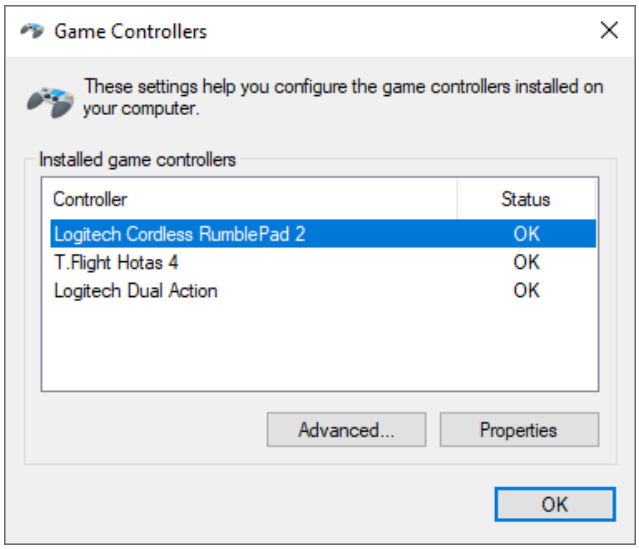
You can control entity movement and weapon fire using a game controller (such as a gamepad or joystick), a keyboard and mouse, or a combination of all of these. You can attach multiple joysticks and can use a mix of joystick and keyboard control at the same time. You configure joystick and keyboard mappings in *.lua* files.

12.12.1 Windows Game Controllers

VR-Engage requires that devices be a USB Windows Game Controller (as opposed to a USB human interface device) You can verify that a USB device is a game controller with the Game Controllers dialog. If the device is a game controller, you will see an entry for the device in that dialog when the device is connected (Figure 111).

➤ To open the Game Controllers dialog, in the Windows search bar, enter Set Up USB Game Controllers and then select the entry.

Figure 111. Game Controllers dialog box



12.12.2 The Game Controller Utility

In the *./bin64* directory, there is an executable called *gameControllerTool.exe*. When you run it from the command prompt, it discovers any joysticks connected to the machine and displays the SDL information about them. You can then use the information to configure new devices in a *.lua* input file.

This example shows the basic SDL output for an Xbox controller:

```
Found 1 joystick(s)

Joystick Name:     'Controller (Xbox One For Windows)'
Joystick Number:   0
Number of Axes:    6
Number of Buttons: 16
Number of Hats:    1
Number of Balls:   0
```

Table 33 lists the command-line options for the utility.

Table 33. gameControllerTool command-line options

Option	Description
`(--help | -h | -?)`	Displays usage information.
`(--debug | --verbose | -v)`	Run with increased verbosity to display detailed information for the connected devices.
`(--listDevices | -l)`	Lists all of the input devices connected to the machine. This is the default behavior when no command-line options are specified.
`(--deviceNumber | -n)` *number*	Returns the information for the specified device. You can get a list of the device numbers by using the `--listDevices` option (or running the utility with no command-line options). In the example above, the number is `0` (Joystick Number).

12.12.3 File Structure

Role Input Configuration Files

By default, all input configuration files are stored in *./data/simulationModelSets/VR-Engage/input*. You can change the location for these by specifying an alternate path in *./appData/scripts/playerStation.lua*:

```
  -- Path to the input configuration directory for VRE entities.
  -- Contains input mapping files for different entity types and roles.
  inputSettingsPath = "$(DATA_DIR)/simulationModelSets/VR-
Engage/input/";
```

The configuration files at the top level typically represent the component input files, pointed to from the "inputConfigFile" field of the control logic components of the role configuration files. Some files closely align to roles, such as *driverInput.lua* and *gunnerInput.lua*. The *commonInput.lua* file maps functionality common to multiple roles, for example, navigating the Action menu, saved views, and mouse control.

Every component configuration file contains a single "configFiles" field with a list of input-device specific *.lua* configuration files.

This example is from *humanInput.lua*:

```
  configFiles = {
    "human/gamepad.lua";
    "human/keyboard.lua";
    "human/mouse.lua";
  }
```

Device Configuration Files

A device configuration file contains a single "inputDevices" field with a set of device configurations. The device configuration files are:

- DeviceName. Values are mouse, keyboard, joystick, or gamepad.
- DeviceType. This applies to joysticks and game controllers. Set this to configure mappings for a very specific device instead of all devices of that type, and should contain the device name, for example, "Logitech Extreme 3D".

  If there are both generic and specific mappings for a kind of input, the specific mapping is used. Generic device types are "gamepad" and "joystick". To create a custom mapping for a specific type of device, you must create an entry for it and set the DeviceType to the name of the device as it appears in the Game Controllers dialog box (Figure 111 on page 273).
- MappingGroups. A set of mappings for the device's input (keyboard or controller) to some particular action in the software. The action could be an entity behavior, such as moving or shooting. It could also be an administrative action, such as releasing mouse control or opening and closing the Action menu.
  - Group. The mapping group name represents the context in which the mapped actions are used. For example, all actions in the "action-menu" group are only available when the user opens the Action menu.
  - Mappings. List of input-action mappings for the group.

12.12.4 Mappings

Common fields:

- Type. Describes the type of input event. Each device type supports different types of inputs:
  - Keyboard. key
  - Mouse. motion, button, double-click, scroll
  - Gamepad. axis, button (see "12.10. Game Controller Button Numbering" on page 269)
  - Joystick. axis, button, hat

  Each input type requires additional fields describing the specific input. For details on the input types, see "12.12.5 Input Types" on the next page.
- Action. The output or outcome of the input being triggered. See "12.12.8 List of Actions" on page 281.

These optional option fields manipulate the action frequency or the values:

- repeatRate (positive number). Intended to use with discrete inputs (buttons or keyboard keys). It emulates the input being repeatedly pressed and released according to the given rate in milliseconds for as long as it is being held.
- smoothing (Boolean). Used to simulate joystick movement with the mouse, for example, to make turret movement more realistic.
- valueTransforms. A list of functions for manipulating the output value. Each value transform has a type field, with parameter fields according to the type.

  When x denotes the raw input value:

Type	Relevant input type	Parameter fields	Output
dead-zone	axis	value (near 0)	$0\ if\ \lvert x\rvert \le value,\ else\ x$
gain	axis	value (between 0 and 1)	$value \times x^3 + \left(1 - value\right) \times x$
invert	all	N/A	$-x$
scale	axis	value	$value \times x$
remap	axis	inputMin, inputMax, outputMin, outputMax	$outputMin + \frac{x - inputMin}{\lvert inputMax - inputMin\rvert} \times outputMax$

You can use multiple transforms at one time, even of the same type. The transforms are processed according to their order in the configuration file. For example, with value transforms of gain and scale in this order, the output is:

$$scale \times (gain \times x)^3 + \left(1 - gain\right) \times x$$

If the order is reversed, the output is:

$$gain \times (scale \times x)^3 + \left(1 - gain\right) \times scale \times x$$

For example:

```
{type="motion"; axis="y"; action="turret-elevation";
  valueTransforms = {
    {type="scale"; value=0.8};
  };
  smoothing=true;
};
```

12.12.5 Input Types

Button

Represents a pressable button on the mouse, gamepad, or joystick. For example:

```
{type="button"; id=0; action="fire";}
```

It requires the ID of the button on the device:

- Mouse: 0 for the left button and 1 for the right button. No other mouse buttons are supported.
- Gamepad: See "12.10. Game Controller Button Numbering" on page 269. If your gamepad has additional buttons, they cannot be mapped.
- Joystick. Button IDs are device specific.

Axis

Represents stick and trigger axis movements on the gamepad or joystick. For example:

```
{type="axis"; id=0; action="steering";}
```

It requires the ID of the axis on the device:

- Gamepad: See "12.10. Game Controller Button Numbering" on page 269. Only the axis shown in the figures and listed in the table are supported.
- Joystick: Axis IDs are device specific.

Moving a stick to the left or down is taken as negative movement of the axis, while right or up is positive movement. Triggers allow only positive movement.

Motion

Represents mouse movements, mostly used for mouse-look. Mouse motion is only processed when the user has mouse control enabled.

For example:

```
{type="motion"; axis="x"; action="turret-azimuth";}
```

Requires an axis field with a value of x or y.

Mouse Control

Mouse control is disabled by default. You can only enable mouse control when the `mouseControlSupport` configuration field of the control logic component in the role configuration file is "true". The control logic components are DtDriverControlLogic, DtHumanControlLogic, DtGunnerControlLogic, and so on.

When engaging with mouse control support, it can be toggled with the `toggle-mouse-control` action (**Alt**+**Enter** or **Ctrl**+left click) or enabled with enable-mouse control (double clicking).

Mouse control is automatically disabled when the VR-Engage window loses focus and is re-enabled when focus is restored, assuming it was enabled before losing focus.

Scroll

Represents mouse wheel scrolling. VR-Engage only supports the vertical axis of the first mouse wheel. If a mouse has multiple wheels or allows side-scrolling, those functionalities cannot be mapped.

For example:

```
{type="scroll"; action="sensor-zoom";}
```

Double-click

Requires the mouse button ID: 0 for the left button and 1 for the right button.

> If "button" and "double-click" mappings share the same ID, double-clicking momentarily executes the "button" mapped action as well.

Key

Represents a key press on the keyboard. Requires a key field.

For example:

```
  {type="key"; key="<F12>"; modifiers={"<SHIFT>"}; action="toggle-
application-mode"}
```

The key values that you can use are:

- Character keys. The character with letters in uppercase, for example, `1` or `A`.
- Keypad number. `<KP0>` through `<KP9>`.
- Functional keys. The keyname surrounded by angle brackets:
  - Arrow keys. `<UP>`, `<DOWN>`, `<LEFT>`, `<RIGHT>`
  - Function keys. `<F1>` through `<F12>`
  - Non-numeric number pad keys. `<SUBSTRACT>`, `<MULTIPLY>`, `<DIVIDE>`, `<DECIMAL>`
  - Modifier keys. `<LEFT SHIFT>`, `<RIGHT SHIFT>`, `<LEFT ALT>`, `<RIGHT ALT>`, `<LEFT CTRL>`, `<RIGHT CTRL>`
  - Others. `<BACKSPACE>`, `<TAB>`, `<CLEAR>`, `<PAUSE>`, `<ESC>`, `<SPACE>`, `<PAGE UP>`, `<PAGE DOWN>`, `<END>`, `<HOME>`, `<PRINT SCREEN>`, `<INSERT>`, `<DELETE>`

  Additional keys are not supported.

Modifiers is an optional field that allows you to map a conjunctional key and a modifier press to an action. The supported modifiers values are `<SHIFT>`, `<ALT>`, and `<CTRL>`. There is no distinction between left and right modifier presses. A mapping can have more than one modifier, for example:

```
  key="A"; modifiers={"<CTRL>, <SHIFT>"};
```

maps **Ctrl**+**Shift**+**A**.

Hat

Represents use of a joystick. The ID field is optional and defaults to 0. The position value is required and represents the direction to which the hat switch is moved: up, right-up, right, right-down, down, left-down, left, left-up (when returned to the natural position).

For example:

```
  {type="hat"; id=0; position="down"; action="next-item"}
```

12.12.6 Player Controls State

Player controls are enabled when:

- No modal is open.
- The scenario is playing.
- The entity is engaged.

When player controls are disabled, only loaded mapping layers that are always enabled remain active. Currently this applies to `app` and `entity-control` groups.

Mapping Conflicts

If the same input is mapped to different actions within the same group, the last mapping defined in the file is used.

Mapping groups are loaded in layers, with each layer (the last one loaded in most cases) having priority over the ones on the bottom. If the same input is mapped to different actions between different groups, the one on the top layer counts, assuming the action is applicable (that is, it has an action handler when the input is made). For example, when opening the Action menu while engaged as a human, pressing "down" selects the next menu option instead of causing the human character to look down. This should be regarded when setting the priority of `ControlLogic` and `SimLogic` components in the role configuration files.

You can map multiple inputs to the same action.

Player controls are enabled when no modal is open, the scenario is playing, and the entity is engaged. (See `DtEngagedState::updatePlayerControlsState`).

When player controls are disabled, only loaded mapping layers that are always enabled remain active. Currently, this applies to the `app` and `entity-control` groups.

When players controls are disabled, mouse control is also turned off.

Raw Input Handling

The action input menu request requires raw input. When active (logic-wise—a raw input handler is set), it has priority over all existing mappings.

The implication is that, for example, if the user pressed A on the gamepad, the input menu processes the input and any mapping of A to an action is ignored.

Current Limitations

- Simultaneous activation of multiple inputs cannot be mapped to a single action, except when using keyboard modifiers.
- There is no distinction between left and right modifiers.
- Keyboard modifiers cannot be used with non-keyboard inputs. This means that you cannot map **Alt**+left click to a single action.
- If multiple joysticks or game controllers of the same type are connected, they must share the same mapping. You cannot have "X" activating "fire weapon" on the first connected Xbox controller and "reload" on the other.
- Non-standard keyboard keys, additional mouse buttons over the standard three, additional mouse wheels, and vertical scroll cannot be mapped.
- Controller touchpads and sensors are not supported.
- When the window is out of focus (even minimized/hidden), gamepad and joystick inputs continue to work. This does not apply to the mouse and keyboard.

- When mouse control is enabled and the Action menu is open, a user can select Action menu options with the mouse by moving the mouse fast enough. It is a bit difficult, depending on the window size, but can potentially happen by accident.

12.12.7 Component Mapping Groups

Table 34. Component mapping groups

Component name	Default config files	Group name	Loaded when
App/DtPlayerStationApp	appInput.lua (hardcoded)	App	Always
Common/DtMenu	commonInput.lua (hardcoded)	action-menu	Action/debug menu open
Common/DtEntityControlLogic		entity-control	Engaged
Common/DtSavedViewUpdater		view	Engaged
DtHumanInputLogic	humanInput.lua	general	Engaged
		combat	Engaged
		binoculars	Binoculars are used
		parachuting	Parachuting
		drone	Using deployed drone
DtCommanderControlLogic	commanderInput.lua lldrInput.lua	general	Engaged
DtDriverControlLogic	driverInput.lua driverGunnerInput.lua	driving	Engaged
DtGunnerControlLogic	gunnerInput.lua driverGunnerInput.lua	engagement	Engaged
		commander-rws	Using commander override (Commander RWS role)
DtLaserTargetDesignatorSimLogic	lldrInput.lua	laser-target-designator	Using LLDR

Table 34. Component mapping groups (continued)

Component name	Default config files	Group name	Loaded when
DtPilotSimLogic	pilotInput.lua	fixed-wing	Engaged fixed-wing
		rotary-wing	Engaged rotary-wing
		extended-flight-controls	Engaged
		instruments	Engaged
DtSensorOperatorInputLogic	sensorOperatorInput.lua	general	Engaged
DtSpectatorSimLogic	spectatorInput.lua	general	Engaged

12.12.8 List of Actions

Binary actions only take values of 0 and 1. Depending on the action, it can be triggered when on a press (1) or release (0). The former is most common. Binary continuous are actions that trigger a behavior that continues while the input is held and stops when it is released.

Axis actions are actions that can receive a value between -1 and 1. Axis- is -1 to 0, and Axis+ is 0 to 1.

Two-sided axis inputs, such as the joystick or gamepad axis and mouse motion, are less suited to binary actions because one side would be disabled and most would require strong movement to register as a binary output.

Key and button inputs can be mapped to different areas of axis actions using value transforms. For example, you could map a button to "yaw" (human) with an "invert" value transform and a scale of "0.5" to slowly turn left.

Table 35. Actions

Group	Action	Type	Notes
action-menu			
	close-menu	Binary	
	next-item	Binary	
	page-down	Binary	
	page-up	Binary	
	previous-item	Binary	
	return	Binary	
	select	Binary	

Table 35. Actions (continued)

Group	Action	Type	Notes
app			
	escape	Binary	
	help	Binary	
	resetView	Binary	Non-standard action name
	toggle-application-mode	Binary	
	toggle-full-screen	Binary	
	user-guide	Binary	
binoculars			
	zoom	Axis	
	zoom-in	Axis+	
	zoom-out	Axis-	
combat			
	aim	Binary continuous	
	cycle-weapon-state	Binary	
	fire	Binary continuous	
	next-weapon	Binary	
	reload	Binary	
	select-weapon-1	Binary	
	select-weapon-2	Binary	
	select-weapon-3	Binary	
	select-weapon-4	Binary	
	select-weapon-5	Binary	
	select-weapon-6	Binary	
	select-weapon-7	Binary	
	select-weapon-8	Binary	
	select-weapon-9	Binary	
	select-weapon-10	Binary	
	toggle-weapon-down	Binary	
	toggle-weapon-stowed	Binary	
	weapon-cycle	Binary	

Table 35. Actions (continued)

Group	Action	Type	Notes
parachute	parachute-left-toggle	Binary	
	parachute-right-toggle	Binary	
commander-rws			
	commander-override	Binary	
	slew-to-cue	Binary	
driving			
	brake	Binary continuous	
	differential-lock	Binary	Only enabled when `hasDifferentialLock=true`
	engine-enable	Binary	
	gear-down	Binary	
	gear-up	Binary	
	headlights	Binary	
	parking-brake	Binary	
	pivot	Binary	Only enabled when `canPivot=true`
	steering	Axis	
	steer-left	Axis-	
	steer-right	Axis+	
	throttle	Axis+	
drone			
	camera-elevation	Binary continuous	
	capture-image	Binary	
	drone-collective	Axis	
	drone-pedal	Axis	
	drone-pitch	Axis	
	drone-roll	Axis	
	drone-wheel-brake	Binary	
	toggle-focus-on-drone	Binary	
	toggle-pid-menu	Binary	

Table 35. Actions (continued)

Group	Action	Type	Notes
	toggle-remote-view-viewport	Binary	
engagement			
	commander-override	Binary	
	fire	Binary continuous	
	next-sensor	Binary	
	next-weapon	Binary	
	previous-weapon	Binary	
	slew-to-cue	Binary	
	turret-elevation	Axis	
	turret-azimuth	Axis	
	turret-down	Axis-	
	turret-left	Axis-	
	turret-right	Axis+	
	turret-up	Axis+	
	zoom	Axis	
	zoom-in	Axis+	
	zoom-out	Axis-	
entity-control			
	toggle-action-menu	Binary	
	toggle-debug-menu	Binary	
	reset-vr-head-position	Binary	
	reset	Binary	
	reset-in-place	Binary	
	push-to-talk	Binary continuous	
	toggle-mouse-control	Binary	Only when "mouseControlSupport" is true and the player is not a spectator.
	enable-mouse-control	Binary	

Table 35. Actions (continued)

Group	Action	Type	Notes
extended-flight-controls			
	autopilot-mode-1	Binary	
	autopilot-mode-2	Binary	
	autopilot-toggle	Binary	
	cycle-weapon	Binary	
	flap1	Binary	
	flap2	Binary	
	flap-toggle	Binary	
	launch-chaff	Binary	
	launch-flares	Binary	
	nose-wheel-steering	Binary continuous	
	replenish-fuel-and-ammo	Binary	
	suppress-warnings	Binary	
	toggle-ar-camera-menu	Binary	
	toggle-countermeasures	Binary	
	toggle-landing-gear	Binary	
	toggle-observer-attach-mode	Binary	
	trigger	Binary continuous	
fixed-wing			
	air-brake	Axis	RT Dynamics only
	flap-state	Axis	RT Dynamics only
	landing-gear	Axis	RT Dynamics only
	pedal	Axis	
	pitch	Axis	
	roll	Axis	
	throttle	Axis	
	wheel-brake	Axis	RT Dynamics only

Table 35. Actions (continued)

Group	Action	Type	Notes
general (commander)			
	sensor-azimuth	Axis	
	sensor-elevation	Axis	
	sensor-zoom-in	Axis+	
	sensor-zoom-in	Axis	
	sensor-zoom-out	Axis-	
general (human)			
	first-person-mode	Binary	
	free-look	Binary continuous	
	minimap-cycle	Binary	
	minimap-reset-zoom	Binary	
	minimap-zoom-in	Axis+	
	minimap-zoom-out	Axis-	
	mouse-pitch	Axis	Mouse only
	mouse-yaw	Axis	Mouse only
	move	Axis	
	move-backward	Axis-	
	move-forward	Axis+	
	parachute-deploy	Binary	Loads parachute group
	pitch	Axis	
	pitch-down	Axis-	
	pitch-up	Axis+	
	posture-down	Binary	
	posture-prone	Binary	
	posture-select	Binary	
	posture-stand	Binary	
	posture-up	Binary	
	sprint	Binary continuous	
	strafe	Axis	
	strafe-left	Axis-	
	strafe-right	Axis+	

Table 35. Actions (continued)

Group	Action	Type	Notes
	third-person-mode	Binary	
	toggle-ar-camera-menu	Binary	
	toggle-binoculars	Binary	
	toggle-compass	Binary	
	toggle-compass-units	Binary	
	toggle-flashlight	Binary	
	toggle-nvg	Binary	
	toggle-view-mode	Binary	
	walk	Binary continuous	
	yaw	Axis	
	yaw-left	Axis-	
	yaw-right	Axis+	
general (sensor operator)			
	capture-image	Binary	
	cycle-sensor	Binary	
	cycle-weapon	Binary	
	fire-weapon	Binary continuous	
	sensor-azimuth	Axis	
	sensor-elevation	Axis	
	sensor-reset	Binary	
	sensor-track-location	Binary	
	sensor-track-object	Binary	
	sensor-zoom	Axis	
instruments			
	clear-targets	Binary	
	cop-cursor-x	Axis	
	cop-cursor-y	Axis	
	cycle-targets-backward	Binary	

Table 35. Actions (continued)

Group	Action	Type	Notes
	cycle-targets-forward	Binary	
	dms-down	Binary	
	dms-left	Binary	
	dms-right	Binary	
	dms-up	Binary	
	radar-cursor-deselect	Binary	
	radar-cursor-down	Axis-	
	radar-cursor-left	Axis-	
	radar-cursor-right	Axis+	
	radar-cursor-select	Binary	
	radar-cursor-up	Axis+	
	radar-cursor-x	Axis	
	radar-cursor-y	Axis	
	sensor-azimuth	Axis	
	sensor-cycle	Binary	
	sensor-elevation	Axis	
	sensor-reset	Binary	
	sensor-track-location	Binary	
	sensor-track-object	Binary	
	sensor-zoom	Axis	
	toggle-rwr-sounds	Binary	
laser-target-designator			
	press-and-hold-to-lase	Binary continuous	
	toggle-laser	Binary	
rotary-wing			
	collective	Axis	
	pedal	Axis	RT Dynamics only
	pitch	Axis	
	roll	Axis	
	wheel-brake	Axis	RT Dynamics only

Table 35. Actions (continued)

Group	Action	Type	Notes
	yaw	Axis	non-RT Dynamics only
spectator			
	azimuth	Axis	
	elevation	Axis	
	reset-view	Binary	
	zoom	Axis	
	zoom-in	Axis+	
	zoom-out	Axis-	
view			
	next-view	Binary	
	previous-view	Binary	
	view-1	Binary	
	view-2	Binary	
	view-3	Binary	
	view-4	Binary	
	view-5	Binary	
	view-6	Binary	
	view-7	Binary	
	view-8	Binary	
	view-9	Binary	
	view-10	Binary	



13. Controllable Entities

This chapter lists the VR-Forces entity types that are controllable from VR-Engage out of the box. The VR-Engage SMS includes these entities in VR-Forces that you can control from VR-Engage if you are running a VR-Forces scenario built with the VR-Engage SMS. Each entity lists the roles and sub-roles that are available for that entity.

You can configure other entities to be controllable, as described in "Chapter 14. Adding and Editing VR-Engage Entity Models" on page 335.

13.1. Human Characters 292
13.2. Ground Vehicles 312
13.3. Fixed-Wing Aircraft 324
13.4. Rotary-Wing Aircraft 328
13.5. Other Entities 331

13.1. Human Characters

- **Afghan Police Officer**: Human Character



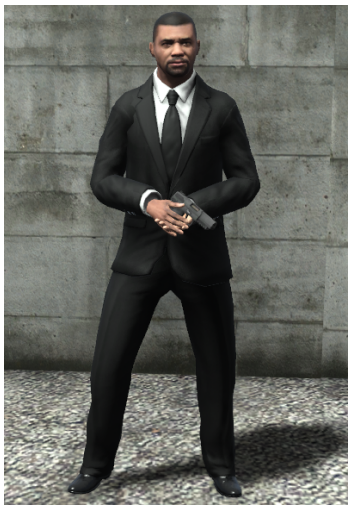
- **Afghan Soldier**: Human Character



- **African Insurgent**: Human Character



- **Agent M9**: Human Character



- **Australian Soldier EF88**: Human Character



- **Australian Soldier XM107**: Human Character



- **Brazilian Soldier**: Human Character



- **Chilean M16**: Human Character



- **Chilean M16-M203**: Human Character



- **Chilean M4**: Human Character



- **Civilian Female With Drone**: Human Character



- **Civilian Male With Drone**: Human Character



- **Colombian Police Officer**: Human Character



- **Colombian Soldier**: Human Character



- **DI AK-47**: Human Character



- **DI Lasing (CIS)**: Human Character



- **DI Lasing (US)**: Human Character



- **Farmer M9**: Human Character



- **ISIS Fighter**: Human Character



- **Insurgent**: Human Character



- **Insurgent M9**: Human Character



- **Iranian Soldier**: Human Character



- **Iraqi Police Officer**: Human Character



- **Iraqi Soldier**: Human Character



- **Israeli Soldier**: Human Character



- **Israeli Soldier M4**: Human Character



- **Kurdish Fighter**: Human Character



- **Libyan Police Officer**: Human Character



- **Libyan Soldier**: Human Character



- **Mideast Combatant**: Human Character



- **Pakistani Police Officer**: Human Character



- **Pakistani Soldier**: Human Character



- **Paratrooper**: Human Character



- **Paratrooper M9**: Human Character



- **Police Officer With Drone**: Human Character



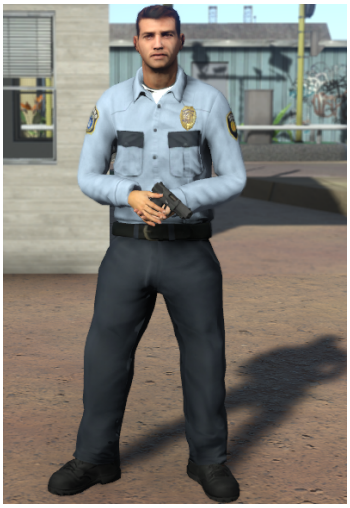
- **Policeman M9**: Human Character



- **Russian Rebels**: Human Character



- **Security Guard M9**: Human Character



- **Singaporean Soldier**: Human Character



- **Somali Police Officer**: Human Character



- **Somali Soldier**: Human Character



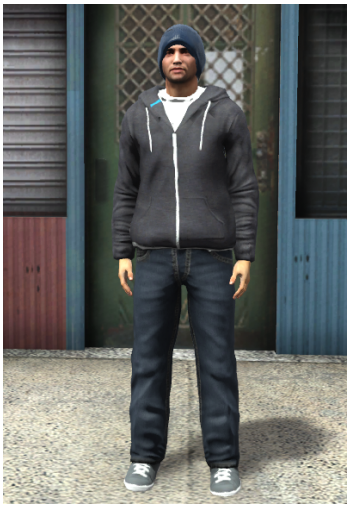
- **Sudanese Police Officer**: Human Character



- **Sudanese Soldier**: Human Character



- **Suspect**: Human Character



- **Suspect M9**: Human Character



- **Syrian Police Officer**: Human Character



- **Syrian Soldier**: Human Character



- **Taliban**: Human Character



- **US Army Javelin**: Human Character



- **US Army Javelin (Tripod)**: Human Character



- **US Army M16**: Human Character



- **US Army M16-M203**: Human Character



- **US Army M4**: Human Character



- **US Army M9**: Human Character



- **US Border Patrol M9**: Human Character



- **USMC M16**: Human Character



- **USMC M16-M203**: Human Character



- **USMC M4**: Human Character



- **Ugandan Police Officer**: Human Character



- **Ugandan Soldier**: Human Character



13.2. Ground Vehicles

- **AN PED-1 LLDR**: Operator Lasing



- **ASLAV-25**: Commander Hatch, Driver, Gunner
  Tow Hook (can tow) , Towable (can also be towed)



- **Ambulance**: Driver
  Towable (can be towed)



- **Amphibious Combat Vehicle (ACV)**: Driver, Gunner, Passenger
  Towable (can be towed)



- **BMP-3 AFV**: Driver



- **Bushmaster PMV**: Driver, Gunner, Passenger
  Tow Hook (can tow) , Towable (can also be towed)



- **FV 510 Warrior**: Commander Hatch, Driver, Gunner



- **FV4034 Challenger 2 MBT**: Commander Hatch, Driver, Gunner, Machine Gun



- **Fiat 124**: Driver



- **GTK Boxer APC**: Driver, Gunner, Passenger
  Tow Hook (can tow) , Towable (can also be towed)



- **GTK Boxer CRV**: Commander, Commander Gunsight, Commander Hatch, Commander RWS, Commander Sight, Driver, Driver/Gunner, Gunner, Passenger
  Tow Hook (can tow)



- **HMMWV Utility Vehicle**: Driver



- **LAV 6 ISC**: Commander, Driver, Driver/Gunner, Gunner, Passenger



- **LAV III APC**: Commander, Driver, Driver/Gunner, Gunner, Passenger
  Tow Hook (can tow) , Towable (can also be towed)



- **Leopard 2A4**: Commander, Driver, Gunner, Loader



- **Light Strike Vehicle MK II (LSV)**: Commander, Commander Laser, Commander Lookaround, Driver, Drone Operator, Primary Gunner, Primary Gunner - AGL, Primary Gunner - MG, Primary Gunner - Spike, Secondary Gunner



- **M1083 MTV Cargo Truck**: Driver



- **M1120A4 HEMTT LHS**: Driver



- **M1126 Stryker ICV with M2**: Driver, Gunner
  Tow Hook (can tow)



- **M113 APC**: Driver



- **M1151 Expanded Capacity Armament Carrier (ECAC)**: Driver



- **M1151 HMMWV ECAC with M2**: Driver, Gunner



- **M1151 HMMWV ECAC with M240**: Driver, Gunner



- **M1151 HMMWV ECAC with M240B**: Driver, Gunner, Seats 3-5 Passenger



- **M1151 HMMWV ECAC with MK 19**: Driver, Gunner, Seats 3-5 Passenger



- **M1278 L-ATV-HGC (JLTV)**: Driver, Gunner
  Tow Hook (can tow) , Towable (can also be towed)



- **M1A1 AIM (AUS)**: Commander, Driver, Driver/Gunner, Gunner, Loader



- **M1A1 AIM (AUS) Mineplow**: Commander, Driver, Driver/Gunner, Gunner, Loader



- **M1A1 SA Abrams**: Commander Hatch, Driver, Driver/Gunner, Gunner, Loader



- **M1A2 Abrams MBT**: Commander, Driver, Driver/Gunner, Gunner, Loader



- **M1A2 SEP Abrams**: Commander, Driver, Driver/Gunner, Gunner, Loader



- **M270 MLRS**: Driver



- **M2A3 Bradley IFV**: Commander, Commander Gunsight, Commander Hatch, Commander Independent Viewer, Commander Sight, Driver, Gunner



- **M915 Tractor Truck**: Driver



- **M977 HEMTT Cargo Truck**: Driver



- **M983 HEMTT Patriot Tractor**: Driver



- **MRAP M-ATV**: Driver, Gunner
  Tow Hook (can tow) , Towable (can also be towed)



- **MSVS SMP Gun Tractor**: Driver



- **Police Cruiser**: Driver



- **Sport Hatchback Coupe**: Driver



- **Sport StandardSize Sedan**: Driver



- **Standard MidSize Sedan**: Driver



- **Technical Truck**: Driver, Gunner
  Tow Hook (can tow) , Towable (can also be towed)



- **Ural 4320**: Driver
  Tow Hook (can tow) , Towable (can also be towed)



13.3. Fixed-Wing Aircraft

- **Beechcraft MC-12 King Air**: Pilot, Sensor Operator



- **Eurofighter Typhoon**: Pilot, Sensor Operator



- **F A-18 Hornet**: Pilot, Sensor Operator



- **F-16A Fighting Falcon**: Pilot, Sensor Operator



- **F-35A Lightning II**: Pilot, Sensor Operator



- **F-35B Lightning II**: Pilot, Sensor Operator



- **F-35C Lightning II**: Pilot, Sensor Operator



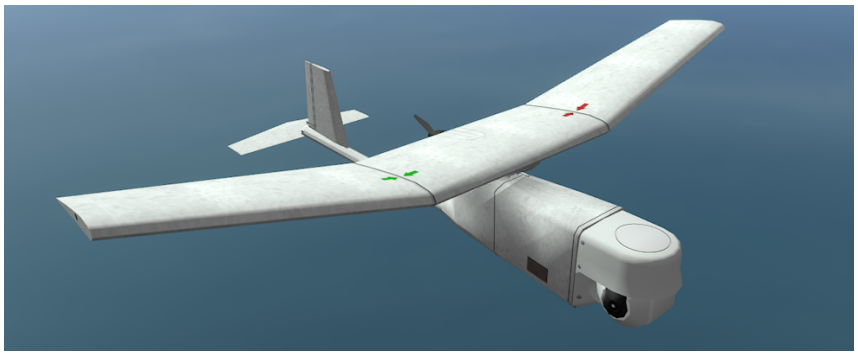
- **MQ-9 Reaper UAV**: Sensor Operator



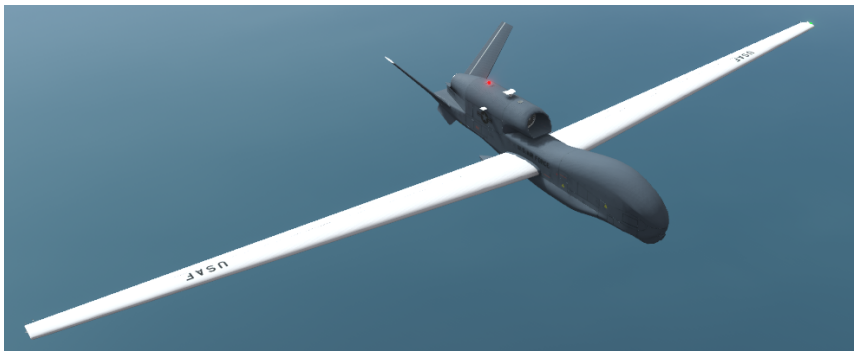
- **MiG-29 Fulcrum**: Pilot, Sensor Operator



- **RQ-11 Raven UAV**: Sensor Operator



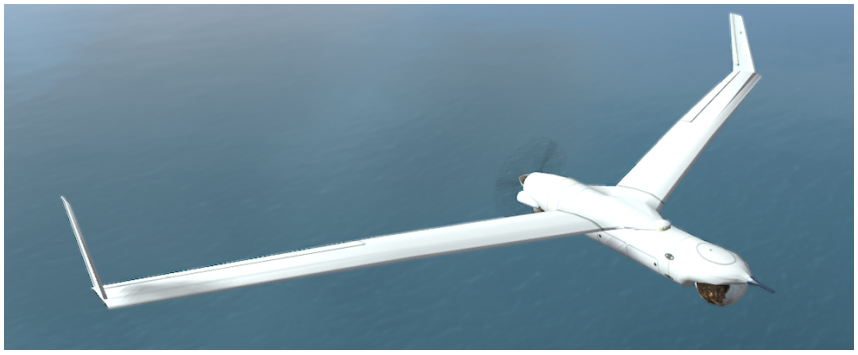
- **RQ-4 Global Hawk UAV**: Sensor Operator



- **RQ-7 Shadow UAV**: Sensor Operator



- **ScanEagle UAV**: Sensor Operator



- **T-38C Talon**: Pilot, Sensor Operator



13.4. Rotary-Wing Aircraft

- **AH-64D Apache Longbow**: Pilot, Sensor Operator



- **AH-64E Apache Guardian with FCR**: Pilot, Sensor Operator



- **CH-149 Cormorant**: Pilot, Sensor Operator



- **CH-47 Chinook**: Pilot, Sling Load Operator



- **DJI M200 Quadcopter**: Pilot



- **DJI Mavic 3 Small Munition**: Pilot



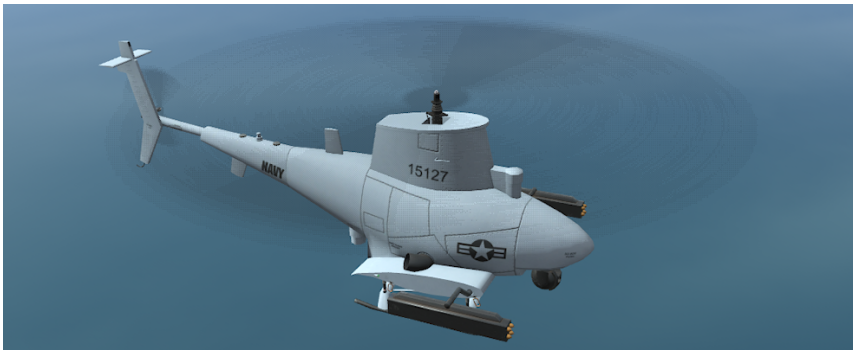
- **Flare Rapider 4 Quadcopter**: Pilot



- **MH-60 Black Hawk**: Pilot, Sensor Operator



- **MQ-8B Fire Scout**: Sensor Operator



- **Parrot Anafi Quadcopter**: Pilot



- **Quadcopter**: Pilot

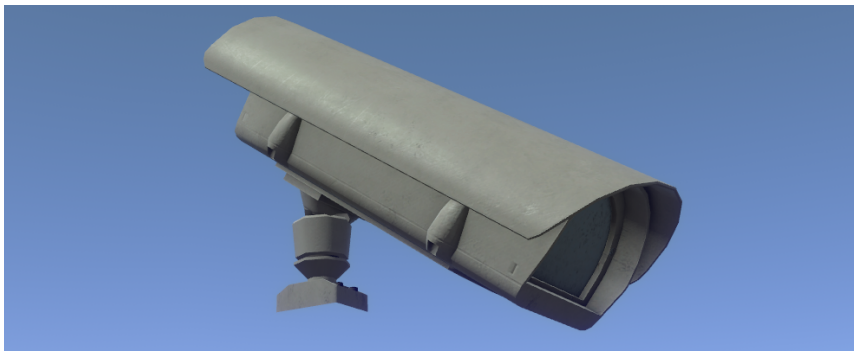


- **UH-60M Blackhawk**: Pilot

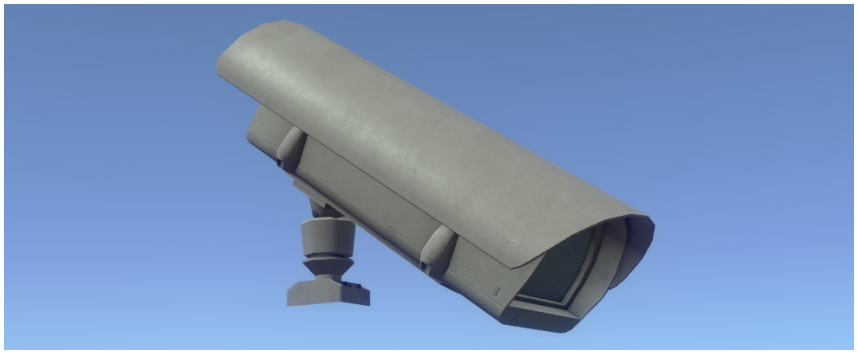


13.5. Other Entities

- **Remote Camera**: Sensor Operator



- **Security Camera**: Sensor Operator





III. Configuration and Scenario Creation

MAK ONE applications allow you to add and customize models and overlays for VR-Engage as required. You can also create VR-Forces scenarios for players to engage with and add VR-Forces scripts to player menus.



14. Adding and Editing VR-Engage Entity Models

This chapter explains how to edit the VR-Engage simulation model set to add or edit models that you can control from VR-Engage. You can also read "Chapter 15. Model Customization Tutorial" on page 361 for an example.

14.1. The VR-Engage Simulation Model Set 336
14.2. Extend the VR-Engage SMS 336
14.2.1 Creating a New Simulation Model Set 337
14.2.2 Creating a New VR-Engage-Controllable Entity 337
14.2.3 Editing a VR-Engage-Controllable Entity 338
14.2.4 Deploying the SMS to VR-Engage 339
14.3. VR-Engage Extensions to .entity Files 339
14.3.1 Structure of Role Data in .entity Files 339
14.3.2 The vrEngageRoles and role Entries 339
14.3.3 The parameterOverride Entry 341
14.3.4 Role-Slot Association 342
14.3.5 Roles Included with VR-Engage 342
14.4. Ground Vehicles and Vortex 347
14.4.1 Prerequisites 347
14.4.2 Configuring VR-Engage Ground Vehicles 352

14.1. The VR-Engage Simulation Model Set

The entity models that VR-Engage simulates are defined in a set of files called a simulation model set (SMS). VR-Engage includes an SMS called *VR-Engage.sms*. The example scenarios included with VR-Engage all use this SMS. If you create an unhosted session, the list of entities available to you is drawn from *VR-Engage.sms*. If you install the VR-Engage extensions package on a compatible VR-Forces installation, *VR-Engage.sms* is added to VR-Forces.

VR-Engage.sms includes all of the simulation objects in the default VR-Forces SMS, *EntityLevel.sms*. A subset of human, ground, and air entities are modified so that they can be controlled by a VR-Engage player. When you create a scenario in VR-Forces for use with VR-Engage, you can use any of the simulation objects in the SMS. The extended *VR-Engage.sms* entities function in the scenario as they normally would (that is, as they do in *EntityLevel.sms*) when they are controlled by VR-Forces. However they can also be controlled at any time by a VR-Engage user.

VR-Engage ships with a set of entities that you can control, listed in "Chapter 13. Controllable Entities" on page 291. You can edit the entity models in *VR-Engage.sms* and add new ones. You add and edit models using the VR-Forces Simulation Object Editor, an application included with VR-Forces. You can then use your SMS in scenarios that you create with VR-Forces, as described in "Chapter 16. Creating VR-Engage Scenarios in VR-Forces" on page 371. If you do not have VR-Forces, contact your MAK Technologies salesperson for information about obtaining VR-Forces.

> This chapter assumes that you understand the concepts of SMSs and how to use the Simulation Object Editor. For details, please see the relevant chapters in Section XII, “Creating and Editing Simulation Models,” in the *VR-Forces User Guide*.

14.2. Extend the VR-Engage SMS

You may want to edit simulation models or add new ones when you need to:

- Create an entity similar to an existing one (for example, a new type of car or aircraft).
- Change the performance characteristics of an existing entity.
- Replace the 3D model associated with an entity.
- Replace or modify the first person model.

The first three modifications can typically be done using the Simulation Object Editor. Replacing or changing the first person model requires different tools and a different skill set. Contact MAK Technologies at https://www.mak.com/mak-one/support/portal for more information about changing the first person model.

The best practice is to leave the original VR-Engage SMS unmodified and create an extended one. It makes maintaining your changes easier in future version and makes it less likely that you will inadvertently modify an existing entity or behavior.

The overall approach to extending *VR-Engage.sms* is:

1. Create a new SMS that includes *VR-Engage.sms*.
2. Make the desired changes to the SMS.
3. Copy the new SMS to your VR-Engage installation.
4. Configure VR-Engage to use the new SMS by default when starting an unhosted session or creating a new scenario in a hosted session.

This chapter covers some of the basics of working in the SMS, and also a tutorial. For more details, such as how to change the 3D model, refer to the *VR-Forces User Guide*.

14.2.1 Creating a New Simulation Model Set

If you want to modify or extend the VR-Engage simulation model set, we recommend that you create a new simulation model set that includes *VR-Engage.sms*. This makes the differences between your customizations and the installed SMS clear. It also makes it easier to revert to the installed SMS if your new SMS has problems.

For information about other ways to create an SMS, please see the *VR-Forces User Guide*.

➤ **To create a new simulation model set:**

1. Open the Simulation Object Editor.
2. Choose **File > New Simulation Model Set**. The New Simulation Model Set dialog box opens.
3. Select New Simulation Model Set. (This is the recommended option.)
4. Click Next. The Simulation Model Set Type page displays.
5. Select Extend Existing Simulation Model Set.
6. Click Next. The Extend Existing Simulation Model Set page displays.
7. Select *VR-Engage.sms*.
8. Click Next. The New Model Set Name page displays.
9. In the File Name box, enter a name for the new model set.
10. Click Finish. The Simulation Object Editor loads systems information from the included model sets and displays the new SMS. This can take a few minutes.

    Since all of the simulation objects are included from other SMSs, none of them are currently editable. They are listed in the Objects List using gray text.
11. Choose **File > Save Simulation Model Set**.

14.2.2 Creating a New VR-Engage-Controllable Entity

The recommended way to create a new entity that can be controlled by a VR-Engage player is to base it on an existing entity of the type you want to create. To control an entity in VR-Engage, it must be one of the entities that was included from *VR-Engage.sms*. Entities that are included from *EntityLevel.sms* cannot be controlled from VR-Engage.

When creating a new entity based on an existing one, you can either Inherit or Copy. If you choose Copy, there is no connection to the original entity. Any changes to the original will not affect your new entity. If you choose Inherit, there is a parent-child relationship between your entity and the original. Use Inherit if your model is similar to the parent because it is easier when you upgrade the SMS for the next software version if there are changes to the parent model. Use Copy only when you intend to change everything about the new entity: model, systems, embarkation, and so on.

➤ **To create a new entity that can be controlled by a VR-Engage player:**

1. In the Simulation Object Editor, load your new SMS.
2. In the Tags list, select VR-Engage. The Objects List is filtered to show only models that are included from *VR-Engage.sms*.
3. Select an entity similar to the one you want to create. (A yellow bar is added to the window. It displays the SMS where the selected entity is originally included.)
4. Choose **Model > New Model from Existing (Inherit)** (or **(Copy)**). The Create New Model dialog box opens.
5. Enter a name for the new model.
6. Optionally, edit the Entity Type.
7. Click OK.
8. Edit the new entity.
9. Save the SMS.

14.2.3 Editing a VR-Engage-Controllable Entity

You can only edit models that are part of the loaded SMS. You cannot edit models that are included from another SMS. Therefore, if you want to edit one of the models included from *VR-Engage.sms*, you must promote it to your SMS. (You can always edit any of the models you created using the procedure in "14.2.2 Creating a New VR-Engage-Controllable Entity" on the previous page.)

1. In the Simulation Object Editor, load your SMS.
2. In the Tags list, select VR-Engage. The Objects List is filtered to show only models that are included from *VR-Engage.sms*.
3. Select the entity you want to edit. If the entity is not part of the main SMS, a yellow bar is added to the window. It displays the SMS where the selected entity is currently included.
4. If the entity is not part of the main SMS, choose **Model > Promote to Main Simulation Model Set** or click the Promote button in the yellow bar. The model is promoted. It is now shown in dark text at the top of the Objects List.
5. Edit the model as desired.
6. Save the SMS.

14.2.4 Deploying the SMS to VR-Engage

After you create the new SMS, you must copy it to VR-Engage. This allows you to run scenarios that were created in VR-Forces using the new SMS. You can also specify that the new SMS be used by default when you create an unhosted session or create a new scenario in a hosted session in VR-Engage.

1. Copy the SMS file and associated data folder to your VR-Engage installation, preserving the path. The path to your SMS will look similar to:

   ./data/simulationModelSets/<your SMS name>.sms

   ./data/simulationModelSets/<your SMS name>

2. Configure VR-Engage to use the new SMS by default:
   a. In a text editor, open *./appData/scripts/playerStation.lua*.
   b. Set the variable **defaultSimulationModelSets** to point to your new SMS, for example:

```
  -- Default simulation model set(s) to use when launching
the simulation engine.
  -- Specify list of SMSs separated by commas.
  defaultSimulationModelSets = {
    "$(DATA_DIR)/simulationModelSets/myVR-Engage.sms"
  };
```

14.3. VR-Engage Extensions to *.entity* Files

14.3.1 Structure of Role Data in *.entity* Files

Roles are defined in Lua role definition files and VR-Engage includes a number of role configuration files, which are listed in Table 38 on page 343. Those roles are then referenced in *.entity* files in the `vrEngageRoles` entry, which is placed near the end of the *.entity* file before the `</simObject>` close tag.

```
<simObjects>
  <simObject>
    ...
    <vrEngageRoles>
      <role> ... </role>
      <role> ... </role>
    </vrEngageRoles>
  </simObject>
</simObjects>
```

14.3.2 The vrEngageRoles and role Entries

The `<vrEngageRoles>` entry in *.entity* files contains a list of one or more `<role>` entry for the entity. The <role> entry has two required parameters: paramName and defaultDisplayLayout. There are other, optional parameters.

```
  <role paramName="Driver" slotName="Driver" unlisted="false"
defaultDisplayLayout="1 Screen Horizontal">@(DATA_
DIR)\simulationModelSets\VR-Engage\roles\<role file>.lua
    <parameterOverride> ... </parameterOverride>
      ...
    <parameterOverride> ... </parameterOverride>
  </role>
```

Table 36. role parameters

Type	Name	Default	Required	Description
Attribute	defaultDisplayLayout	1 Screen Horizontal	Yes	The default visual layout for this entity when in that role. A role can include one or more display layouts, depending on the hardware and visual appearance desired. You can refer to an existing role file for the list of available layouts. The role definition file must include at least one display layout set for the default. All roles included in VR-Engage have a `"1 Screen Horizontal"` layout, at the minimum. Each role file includes the `displayLayouts` section at the top. That section defines the components and parameters needed to control the viewpoint and any HUD overlays.
Attribute	paramName		Yes	Name of the role as it will display in the Choose Role panel's Role list.

Table 36. role parameters (continued)

Type	Name	Default	Required	Description
Element	parameterOverride		No	You can specify zero or more parameterOverride child elements. A parameterOverride contains Lua code snippets to override specific table elements in the role configuration file.
Element Text			Yes	Path to the Lua script that defines the role configuration.
Attribute	slotName		No	Name of the Embarkation Slot associated with this role. This role is automatically engaged when the player's entity is embarked in this slot. If this attribute is not provided, this role will not be accessible via embarkation and can only be engaged from the Choose Role panel. For more information about roles and slots, see "14.3.4 Role-Slot Association" on the next page.
Attribute	unlisted	false	No	Boolean indicating whether the role should be available for selection in the Choose Role panel.

14.3.3 The parameterOverride Entry

You can override any parameters from the role configuration file using overrides in the *.entity* file. This allows you to use a single role configuration file in multiple entities with entity-specific customizations made in the appropriate *.entity* files. The parameters you can override are the entries in the role's Lua `components` table. The syntax for the overrides is: <`parameterOverride paramName="`*`key in components table.lua table entry`*`">`*`value`*`</parameterOverride>`. For example:

```
  <parameterOverride paramName="savedViewUpdater.initialView">"4 -
Driver View"</parameterOverride>
```

Table 37. parameterOverride parameters

Type	Name	Default	Required	Description
Attribute	paramName		Yes	Name of a Lua component table item to override in the role file. The format of this string is <componentName>.<Lua table entry>. In the above example, the `initialView` table member of the `savedViewUpdater` component is the item being replaced.
Element Text			Yes	A Lua code snippet to replace the corresponding entry in the role configuration file.



For an example of editing these values to configure your own customized entity, see "Chapter 15. Model Customization Tutorial" on page 361.

14.3.4 Role-Slot Association

Roles are associated with slots by the `slotName` parameter of the VR-Engage `<role>` entry. The role is then associated with the embarkation slot with a matching `slotName` parameter. A role can only be associated with one slot, and a slot should only be associated with one role. It is not required that every role be associated with an embarkation slot, nor is it required that every slot have an associated role.

If a slot has an associated role, that vehicle role is automatically engaged whenever a player's entity is embarked into that slot, or when a player engages with an entity already embarked in the slot. If a player entity is embarked in a slot that does not have an associated role, the player remains engaged in the embarked human entity.

If a role does not define a valid slot association, there is no way to engage in that vehicle role via embarkation. To play this role, a player must engage directly with the vehicle from the Choose Role panel, or by command from the VR-Engage Manager Panel in VR-Forces.

To learn more about roles and slots, see "Appendix B. Embarkation Slot Configuration in VR-Engage" on page 397.

14.3.5 Roles Included with VR-Engage

Table 38 lists many of the out-of-box roles configured on entities in *VR-Engage.sms*. These files are found in *./data/simulationModelSets/VR-Engage/roles*.

Some role files are derived from other, more generalized roles. Many entities in the VR-Engage SMS are configured with more than one role. To see how roles are derived, see "Role Inheritance Hierarchy" on page 345.

Meta-roles enable a group of roles to be combined into a single logical role, which VR-Engage users can select from the Choose Role panel or VR-Forces instructors can select in the VR-Engage Manager Panel. Once engaged in a meta-role, players can quickly switch between sub-roles using keyboard shortcuts without returning to the Choose Role panel. For example, the M2A3 Commander role contains both the Commander Sight and Commander Hatch sub-roles, and players can toggle between them with a single key press.

Table 38. Available roles

Role Configuration File	Description
Base Roles (Foundation)	
baseRole.lua	Foundation role configuration inherited by all other roles
platform.lua	Base platform role inherited by vehicle roles
humanBase.lua	Base configuration for all human/dismounted roles
default.lua	Default role configuration fallback
Human Dismounted Roles	
human.lua	Standard dismounted infantry/soldier role
humanWithoutVr.lua	Dismounted soldier without VR hardware support
militaryHuman.lua	Military personnel with standard equipment
diLasingHuman.lua	Dismounted infantry with laser designation
lldr.lua	Lightweight Laser Designator Rangefinder (LLDR) operator
notionalPassenger.lua	Notional passenger role (non-interactive)
Generic Vehicle Roles	
driver.lua	Generic ground vehicle driver
driverGunner.lua	Combined driver/gunner for light vehicles
tankDriver.lua	Main battle tank driver
tankDriverWith3D.lua	Tank driver with 3D audio support
gunner.lua	Generic ground vehicle gunner
tankGunner.lua	Main battle tank gunner
tankMachineGunner.lua	Tank coaxial machine gunner
machineGunner.lua	Generic vehicle machine gunner
emergencyVehicleDriver.lua	Emergency vehicle driver
remoteWeaponSystem.lua	Remote weapon station operator
Commander Roles (meta-roles)	
commander.lua	Ground vehicle commander (meta-role)
commanderHatch.lua	Commander out-the-hatch viewpoint (sub-role)

Table 38. Available roles (continued)

Role Configuration File	Description
commanderHatch2.lua	Alternative commander hatch (sub-role)
commanderWithViews.lua	Commander with predefined saved views
Vehicle-Specific Roles	
Boxer GTK	
boxerDriver.lua	GTK Boxer driver
boxerGunner.lua	GTK Boxer gunner
HMMWV	
hmmwvGunner.lua	HMMWV gunner
LAV-25	
lavGunner.lua	LAV-25 gunner
Light Strike Vehicle (LSV) Roles	
lsvAgl.lua	LSV automatic grenade launcher operator
lsvCommanderLaser.lua	LSV commander with laser designation
lsvCommanderLookaround.lua	LSV commander 360° lookaround
lsvDroneOperator.lua	LSV drone operator
lsvGunner.lua	LSV gunner
lsvLookaround.lua	LSV 360° observer
lsvMissileLauncher.lua	LSV missile launcher operator
lsvPrimaryMg.lua	LSV primary machine gun operator
lsvSecondaryGunner.lua	LSV secondary gunner
M2A3 Bradley IFV	
M2A3Commander.lua	M2A3 commander (meta-role)
Commander Sight	Inside vehicle view (sub-role, unlisted)
Commander Independent Viewer	(sub-sub-role)
Commander Gunsight	(sub-sub-role)
M2A3Gunner.lua	M2A3 gunner
Stryker	
strykerGunner.lua	Stryker gunner
Aircraft Roles	
pilot.lua	Generic aircraft pilot
fixedWingPilot.lua	Fixed-wing aircraft pilot
rotaryWingPilot.lua	Helicopter pilot
t38Pilot.lua	T-38 trainer aircraft pilot

Table 38. Available roles (continued)

Role Configuration File	Description
slingLoadOperator.lua	Helicopter sling load operator
Drone/UAV Roles	
fpvDronePilot.lua	First-person view drone pilot
Sensor Operator Roles	
sensorOperator.lua	Generic sensor operator
airSensorOperator.lua	Air-based sensor operator
cameraSensorOperator.lua	Camera-based ground sensor operator
remoteCameraSensorOperator.lua	Remote camera operator
Special Role Types	
metaRole.lua	Meta-role framework for grouping multiple sub-roles

Role Inheritance Hierarchy

This list presents the inheritance hierarchy for the roles included in VR-Engage.

Child roles inherit components and configuration from parent roles. Parameters in child roles can override the parameters in the parent role, which allows for entity-specific customization.

Custom configurations are defined in subdirectories of the SMS's roles directory (for example, *./data/simulationModelSets/VR-Engage/roles*). Device configurations (radio, minimap, deployable objects, and so on) are defined per role category in *devices*. Menu configurations are defined in *menus*. Saved camera views are stored in *savedViews*.

In the list below:

- Indentation shows parent-child relationships in meta-role structures.
- Base roles are inherited by other roles but not directly selectable.
- Sub-roles are indicated with an asterisk (*). Sub-roles are also not directly selectable; they are part of the role that they are a part of.

baseRole.lua

- **humanBase.lua**
  - human.lua
    - militaryHuman.lua
    - diLasingHuman.lua

- humanWithoutVr.lua
- lldr.lua
- notionalPassenger.lua

- **platform.lua**
  - driver.lua
    - boxerDriver.lua
    - emergencyVehicleDriver.lua
    - tankDriver.lua
      - tankDriverWith3d.lua
    - driverGunner.lua
    - commander.lua
      - commanderHatch.lua *
      - commanderHatch2.lua *
      - commanderWithViews.lua
    - gunner.lua
      - boxerGunner.lua
      - hmmwvGunner.lua
      - lavGunner.lua
      - strykerGunner.lua
      - M2A3Gunner.lua
      - machineGunner.lua
      - tankGunner.lua
      - tankMachineGunner.lua
      - remoteWeaponSystem.lua
      - lsvGunner.lua
      - lsvAgl.lua
      - lsvPrimaryMg.lua
      - lsvMissileLauncher.lua
      - lsvSecondaryGunner.lua
    - M2A3Commander.lua
      - Commander Sight *
        - Commander Independent Viewer *
        - Commander Gunsight *
      - Commander Hatch *

- lsvCommanderLaser.lua
- lsvCommanderLookaround.lua
- lsvLookaround.lua
- lsvDroneOperator.lua
- pilot.lua
  - fixedWingPilot.lua
    - t38Pilot.lua
  - rotaryWingPilot.lua
  - fpvDronePilot.lua
- sensorOperator.lua
  - airSensorOperator.lua
  - cameraSensorOperator.lua
  - remoteCameraSensorOperator.lua
- slingLoadOperator.lua

- metaRole.lua
- default.lua

14.4. Ground Vehicles and Vortex

VR-Engage ground vehicles use a higher-fidelity physics-based model for movement under player control, as compared to the kinematics-based model used by vehicles under CGF control. The physics-based vehicle model is based on Vortex® Studio from CM Labs. It models the behavior of ground vehicles—vehicle dynamics, tire-ground interaction, weight distribution, and so on—all of which provide a more realistic driving experience for role players. VR-Engage ground vehicles are configured with both the high-fidelity Vortex-based and CGF kinematics models. An entity switches between these models at runtime, depending on whether there is a player actively engaged as a driver on the vehicle.

The following sections explain how to configure the high-fidelity Vortex-based movement model used by VR-Engage ground vehicle entities.

14.4.1 Prerequisites

Before you can configure Vortex ground vehicles for use with VR-Engage, you must:

- Consult relevant documentation, including that provided by MAK Technologies and CM Labs.
- License the tools for creating and customizing content, including the VR-Forces Simulation Object Editor (SOE). Vortex Studio Editor is optional and requires a separate Vortex Studio license from CM Labs. Most people can use one of the movement system files included with VR-Engage to create a new entity. You need that tool only for advanced configuration, such as creating new Vortex models or

modifying aspects of existing Vortex models.

- Use the design overview to determine how you will extend and configure entities.

Overview of Vortex Ground Vehicle Configuration

This documentation assumes that you are familiar with the concepts and tools for modifying VR-Forces entity models. Please see the chapters in the *Creating and Editing Simulation Models* section of the *VR-Forces User Guide* or help, particularly *The Simulation Object Editor and SMSs*, *Editing Entities for Entity-Level Scenarios*, and *Configuring Embarkation*.

For concepts specific to VR-Engage that are associated with creating and configuring VR-Engage-controllable entities, see "Chapter 14. Adding and Editing VR-Engage Entity Models" on page 335 and "Chapter 15. Model Customization Tutorial" on page 361. In many cases, this may provide enough background for you to start configuring the movement model in VR-Engage entities, where the existing Vortex models and configuration parameters available out-of-box are sufficient for your fidelity needs.

For more advanced configuration, you must understand the underlying Vortex Vehicle model. You might need advanced configuration when:

- Your simulation requires detailed geometry of the hull or other aspects of the model. In that case, you must configure Vortex vehicle parameters available only through the Vortex Studio Editor.
- None of the Vortex vehicle mechanisms included with VR-Engage are suitable for the entity you need to simulate. You would then create a completely new Vortex vehicle mechanism (the underlying Vortex movement model), rather than reusing an existing one.
- You need to customize the towing parameters.

Customizing a model at this level requires a deeper understanding of how the Vortex physics model works and how to use the CM Labs tools to create and configure those models. Please read the CM Labs Vortex Studio documentation, which you can access from their website (https://www.cm-labs.com/vortex-studio).

Additional VR-Engage-specific conventions, requirements, and tools for integrating your custom Vortex model in VR-Engage are discussed in the *VR-Engage Developers Guide*.

Tools

To configure Vortex ground vehicles, use the VR-Forces Simulation Object Editor and possibly a text editor. For more advanced configuration, use the Vortex Studio Editor.

- Simulation Object Editor (SOE). This is the primary tool you would use to configure VR-Engage vehicles. In many cases, this is the only tool necessary for configuring the high-fidelity movement model.
- Text Editor. If you need to edit parameters not available through the SOE, use a text editor to edit the *.entity* file.
- Vortex Studio Editor. Use this tool for more advanced configuration, such as creating new Vortex models from scratch or modifying aspects of existing Vortex

models that require access to the underlying Vortex vehicle mechanism file. Note that Vortex Studio Editor requires a separate Vortex Studio license from CM Labs.

Designing a Configuration

Configuration of Vortex entities does not require coding. However, we have provided a high-level overview of design to help you understand the content available for modification and which tool(s) are best suited for making changes.

There are two primary elements that a VR-Engage *.entity* file configured for high-fidelity vehicle movement requires: the *.entity* file must be configured with Vortex-specific platform (*.ope*) and movement system (*.sysdef*) files. The movement system includes the Vortex mechanism file.

Platform File

All entities in MAK ONE applications are configured with a platform file, which defines the state repository used for the entity and a number of sim components, such as the reactive task script controller and set data controller. Typically, users do not need to modify a platform file. They simply need to ensure that they choose the specific platform appropriate for the type of vehicle.

For VR-Forces entity-level ground vehicles, there is a single platform file, *Ground_Vehicle.ope*. All entity-level vehicles configured for CGF ground vehicle movement use this platform file. VR-Engage replaces the entity-level platform file with one that includes elements required by VR-Engage. For VR-Engage ground vehicles, the platform file *Vortex_Ground_Vehicle.ope* is configured for all Vortex entities..

Movement System File

To configure a ground vehicle entity for high-fidelity movement, you must also have a Vortex-based movement system. The movement system definition file (*.sysdef*) must contain a DtVortexMovementActuator, which models movement of the vehicle as a Vortex vehicle while under player control. It also includes a kinematics actuator for movement while under CGF control.

The various movement *.sysdef* files used in VR-Engage vehicles with high-fidelity movement models inherit from the template file, *vortex.template_sysdef*. The template file includes the Vortex-based movement actuator, as well as all of the controllers and actuators for CGF movement. Table 39 on the next page lists the movement *.sysdef* files available for configuration on a VR-Engage ground vehicle.

The names of the movement *.sysdef* files describe the types of vehicles for which they are most appropriate. Use the *.sysdef* file that most closely matches the type of vehicle you want to configure. You should also consider the system type:

- Archetype systems. The *.sysdef* files marked as "Archetype" in the table have file names that begin with *ground-tracked*, *ground-truck*, or *ground-trailer*. They represent generic vehicle classes rather than specific real-world vehicles. These archetypes offer the greatest degree of configurability through the SOE.
- Vehicle-specific systems. Some *.sysdef* files—*ground-lav-vortex.sysdef*, *ground-hmmwv-vortex.sysdef*, *ground-ambulance-vortex.sysdef*, and so on—are tailored to specific vehicles and may have fewer configurable parameters.

The Vortex-based movement actuator defines several parameters that are made available for configuration in those *.sysdef* files that include the template. The parameters include various aspects of ground vehicle behavior. A key parameter is the Vortex Mechanism File, which defines the vehicle structure and behavior. The variation in *.sysdef* files is driven largely by the associated Vortex mechanism file configured in the *.sysdef* file. For more information, see "Vortex Mechanism Files" on the facing page.

Table 39. Movement system definition files

Movement system file	Vortex mechanism file	Description
Wheeled Vehicles		
ground-ambulance-vortex.sysdef	Ambulance.vxmechanism	Ambulance-specific dynamics
ground-hmmwv-vortex.sysdef	HMMWV.vxmechanism	HMMWV-specific model
ground-policecar-vortex.sysdef	Police_Car.vxmechanism	Law enforcement vehicle
ground-sportscar-vortex.sysdef	Sports_Car.vxmechanism	High-performance civilian vehicle
ground-truck-vortex.sysdef	Generic truck mechanism	Base truck configuration
ground-truck-4-wheels-vortex.sysdef	Truck4Wheels.vxmechanism	Light truck archetype
ground-truck-6-wheels-vortex.sysdef	Truck6Wheels.vxmechanism	Medium truck archetype
ground-truck-8-wheels-vortex.sysdef	Truck8Wheels.vxmechanism	Heavy truck archetype
ground-truck-APC-8-wheels-vortex.sysdef	TruckAPC8Wheels.vxmechanism	Armored personnel carrier (unarmed)
ground-truck-APC-wMainGun-8-wheels-vortex.sysdef	TruckAPC8Wheels_wMainGun.vxmechanism	Armored personnel carrier with weapon
Tracked Vehicles		
ground-tracked-vortex.sysdef	M1A2.vxmechanism	M1A2-specific with main gun
ground-tracked-vortex-no-gun.sysdef	M1A2.vxmechanism	Unarmed tank chassis
ground-tracked-10-wheels-vortex.sysdef	Track10Wheels.vxmechanism	Light tracked vehicle archetype (10-wheel suspension)
ground-tracked-12-wheels-vortex.sysdef	Track12Wheels.vxmechanism	Medium tracked vehicle archetype (12-wheel suspension)

Table 39. Movement system definition files (continued)

Movement system file	Vortex mechanism file	Description
ground-tracked-14-wheels-vortex.sysdef	Track14Wheels.vxmechanism	Heavy tracked vehicle (14-wheel suspension, unarmed)
ground-tracked-14-wheels-wMainGun-vortex.sysdef	Track14Wheels_ wMainGun.vxmechanism	Heavy tank (14-wheel suspension with weapon)
Light Armored Vehicles		
ground-lav-vortex.sysdef	LAV_III.vxmechanism	LAV III-specific with turret
ground-lav-vortex-no-gun.sysdef	LAV_III.vxmechanism	Unarmed LAV configuration
Trailers		
ground-trailer-2-wheels-vortex.sysdef	Trailer2Wheels.vxmechanism	Small towed trailer archetype
ground-trailer-4-wheels-vortex.sysdef	Trailer4Wheels.vxmechanism	Large towed trailer archetype

Vortex Mechanism Files

A Vortex mechanism file contains the vehicle's physical setup, allowing it to interact realistically with terrain, respond to user input, and simulate motion and forces in real time. This file is created using the Vortex Studio editor. Several Vortex mechanism files are included with VR-Engage, modeling a diverse set of vehicle types. The Vortex mechanism files included with VR-Engage range from vehicle-specific models, such as for the LAV III vehicle, to more generic "archetype" vehicle models. The movement systems table (Table 39 on the previous page) shows the Vortex mechanism files associated with each movement system.

The process for creating a Vortex mechanism file is beyond the scope of this document. See the CM Labs Vortex Studio documentation for instructions on building vehicle mechanism files.

VR-Engage includes scripts for adapting a new vehicle mechanism file for use in the application. Note that there are a few required rules and conventions for creating Vortex vehicle mechanism files suitable for use within VR-Engage. To learn about the scripts, rules and conventions for incorporating new vehicle mechanisms, see the *VR-Engage Developers Guide*.

Archetype Mechanisms

The archetype vehicle systems do not represent any specific, real-world vehicle, but rather a class of vehicles, such as tracked vehicles with 5 wheels or trucks with 6 wheels. These generic mechanisms allow you to configure a variety of vehicles without constructing a new Vortex mechanism for each.

14.4.2 Configuring VR-Engage Ground Vehicles

The workflow for configuring a ground vehicle for role players to control in VR-Engage is:

1. Create a new entity.
2. Set the platform for the entity.
3. Specify the entity's mass.
4. Edit the movement system parameters.
5. Test the new entity.

Creating a New Entity

If the vehicle entity you want to configure does not yet exist as a VR-Engage entity, create a new one. Use the VR-Forces SOE to create the new entity. See "14.2.2 Creating a New VR-Engage-Controllable Entity" on page 337. We recommend following the Create from Existing guidelines, choosing an existing VR-Engage-level entity as a starting point. Select one that is similar to the vehicle you want to simulate. When you create the new entity, it has a new entity file (*.entity*).

This workflow configures modifications to the VR-Engage SMS in the compatible VR-Forces installation, where you can use the SOE to modify or extend entities in the SMS. Be sure to copy the SMS modifications or extensions to all VR-Engage installations that will participate in a session that uses your changes.

Setting the Platform for the New Entity

In your new *.entity* file, check the platform file (*.ope*) specified in the file. Depending on the original entity you used as a template, it may already have an appropriate platform. If it is not, however, then edit the *.entity* file using a text editor and specify the most appropriate platform file from Table 39 on page 350. For an example, see "15.4.1 Specify a VR-Engage-Capable Platform" on page 365. The most appropriate platform file is the one that most closely matches the characteristics of the vehicle you are modeling.

Setting the Entity's Mass

The vehicle's mass affects the simulation in the way that it would in a physics simulation. It affects the vehicle's weight, inertia and momentum. For more information on how Vortex uses the vehicle's mass, refer to the CM Labs documentation.

Using the SOE, open the SMS that contains your new entity file. Select the Movement tab and then edit the vehicle's Mass parameter. Note that the Mass parameter applies to both the CGF and Vortex models.

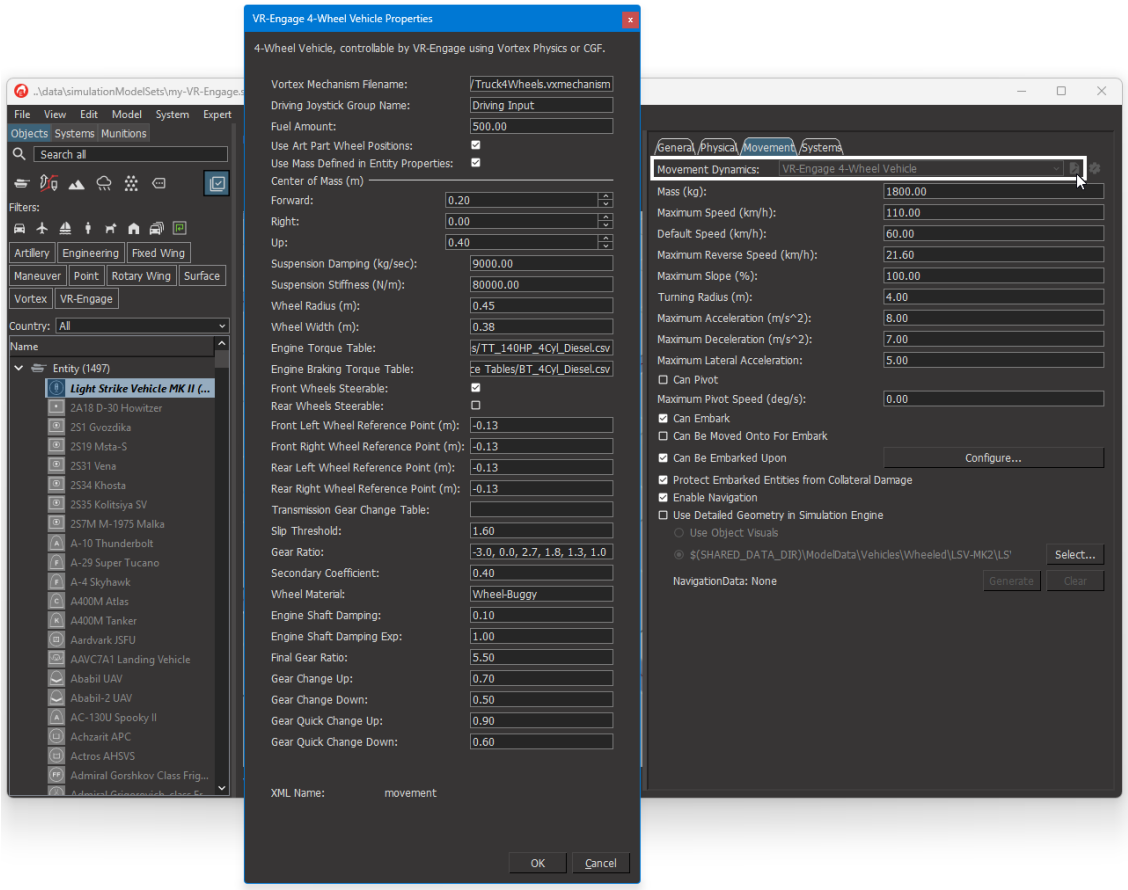
Editing the Movement System Parameters

Once you have either created a new entity for your vehicle or chosen an existing entity to modify, and you have configured it with the appropriate platform file, you can use the SOE for the remaining configuration. Refer to the *Editing Entities for Entity-Level Scenarios* chapter of the *VR-Forces User Guide* or help for information on editing a simulation object's systems using the SOE.

Select the movement system most appropriate for your vehicle. As described in "Movement System File" on page 349, there are a variety of systems included with VR-Engage. Vehicle-specific systems are tailored to specific vehicles, such as the LAV III (*ground-lav-vortex.sysdef*) or HMMWV (*ground-hmmwv-vortex.sysdef*). Archetype systems (those beginning with *ground-tracked*, *ground-truck*, or *ground-trailer*) provide a generic representation of a vehicle class. Both types of systems can be customized through parameters, however, archetype systems offer the greatest degree of configurability through the SOE. Refer to Table 39 on page 350 for a list of the available systems.

To access the configuration parameters for the movement system in the SOE, select the entity you want to edit and then select the Movement tab (on the right-hand side of the SOE). Next, click the Edit Properties button () for the Movement Dynamics entry. The Properties dialog box for configuring the movement system opens. Figure 112 shows an example of editing the movement properties of a vehicle that uses the *ground-truck-4-wheels-vortex.sysdef* (VR-Engage 4-Wheel Vehicle) archetype system.

Figure 112. Movement dynamics system properties in SOE



You can edit the properties of the movement system using the Properties dialog box. Any modification to the movement system applies to the entity and is made as an override to the *.entity* file. The system, meaning the *.sysdef* file, is unchanged. Other entities that use the same movement system continue to behave as they did before.

The available properties for archetype systems such as the *ground-truck-4-wheels-vortex.sysdef* are specific to the corresponding Vortex mechanism file configured in the *.sysdef*. Vehicle-specific systems have a different set of properties appropriate to their corresponding Vortex mechanism. Archetype systems generally offer more configurable parameters than vehicle-specific systems. You can find more details on the available parameters in the CM Labs Vortex documentation, particularly the *Vortex Tuning Guide*.

Additional Notes on Editing Entity Parameters

Because the standard VR-Forces CGF movement system for a ground vehicle is replaced by a higher-fidelity Vortex-based movement system when an entity is player-controlled, certain parameters that you can see and edit in the SOE do not apply when the entity is player-controlled. For example, when the CGF movement system is active, the dimensions of the entity's bounding box dictate when an entity collides with a wall or other obstacle. When an entity is player-controlled in VR-Engage, however, the simple bounding box parameters are ignored when computing ownship collisions. Instead, the Vortex-based model computes collisions based on higher-fidelity geometry information that is configured in the Vortex mechanism – including wheel sizes and locations, as well as runtime knowledge about the positions of any articulated parts, shock absorber compressions, gear ratios, automatic transmission parameters, and so on.

The bounding box dimensions are used for purposes other than ownship collisions, for example, VR-Forces uses them to compute whether the VR-Engage-controlled entity blocks line of sight between VR-Forces CGF entities, or whether a VR-Forces entity's path is blocked by the VR-Engage controlled entity. Therefore, it is important to accurately configure the CGF bounding box information in the SOE even for entities types that are usually player-controlled.

Testing and Debugging the New Entity

VR-Engage includes tools for testing and tuning Vortex vehicles at runtime:

- Vortex Debugging Tool. Use this to view and adjust Vortex-specific vehicle parameters.
- VR-Engage Debug Menu. Use this to visualize a variety of VR-Engage model parameters, including Vortex-specific details.

Vortex Debugging Tool

VR-Engage includes a runtime tool for viewing and adjusting Vortex vehicle parameters on-the-fly. This tool is based on CM Labs' remote debugging capability. When it is enabled and you run VR-Engage and engage as a driver in a vehicle, it launches a second window through which you can view and modify Vortex model parameters in real time while actively engaged as a driver in VR-Engage. Use this to experiment with and adjust the model as you drive the vehicle around in VR-Engage. Be aware that this tool requires a working knowledge of Vortex Studio. Refer to the CM Labs documentation for more on remote debugging with Vortex Studio.

Any changes to parameters that you make through the remote debugger are in effect only for the duration of the session. You must then make the changes to the vehicle either by modifying the movement system file in the SOE (for those parameters that are available

for configuration at the movement system level), or in the Vortex mechanism itself using Vortex Studio (for any other model parameters not exposed for configuration at the movement system level).

Enabling the Vortex Debugging Tool

The Vortex remote debugging tool is disabled by default. To enable it, edit the Vortex configuration file, *./appData/scripts/vortexConfig.lua*. Enable this setting for the sim engine where you will simulate the entity you are debugging. For example, if you are hosting the scenario with VR-Forces, you must enable it in the VR-Forces installation directory. If you are running VR-Engage and either self-hosting a session or creating an unhosted session, enable it in the VR-Engage directory.

In the appropriate file, change the `configFilename` value to `"default-debug.vxc"`:

```
vortexTerrain={
  -- Config file to load to the Vortex Application
  -- To enable the Vortex runtime debugging tool,
  -- set configFilename = "default-debug.vxc".
  configFilename = "default-debug.vxc";
  ...
}
```

After you change that setting, when you launch the host application, the Vortex runtime debug tool launches as well. If at any point you want to stop using the tool whenever you run VR-Engage, reset the value for `configFilename` to `"default.vxc"` in *vortexConfig.lua*. The Vortex debugging tool degrades runtime performance, therefore you might want to disable it when you are not actively configuring and tuning Vortex vehicles.

When enabled, this additional window opens when you launch VR-Forces or VR-Engage and either create or join a session.

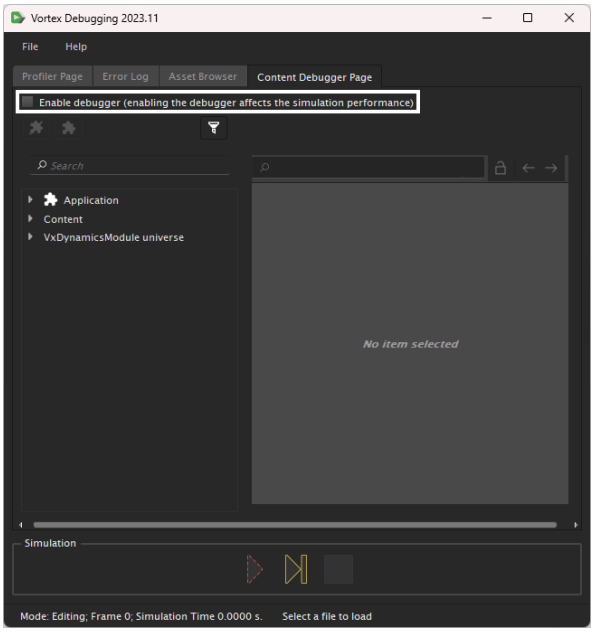
Launching the Vortex Debugging Tool

Once you have enabled the Vortex remote debugging tool in the configuration file, it is available any time you run VR-Engage or VR-Forces. The following procedure is written for VR-Engage, but it also applies to VR-Forces.

➤ **To run the Vortex remote debugging tool:**

1. Launch VR-Engage (as described in "3.1. Starting VR-Engage and Joining a Session" on page 56).
2. Create a session. The Vortex Debugging tool window opens.
3. To enable the debugger, select the Content Debugger Page tab and then select the Enable Debugger check box (Figure 113).

Figure 113. Vortex Debugging tool



The tool is now ready to display details about any Vortex-based VR-Engage ground vehicle in the session.

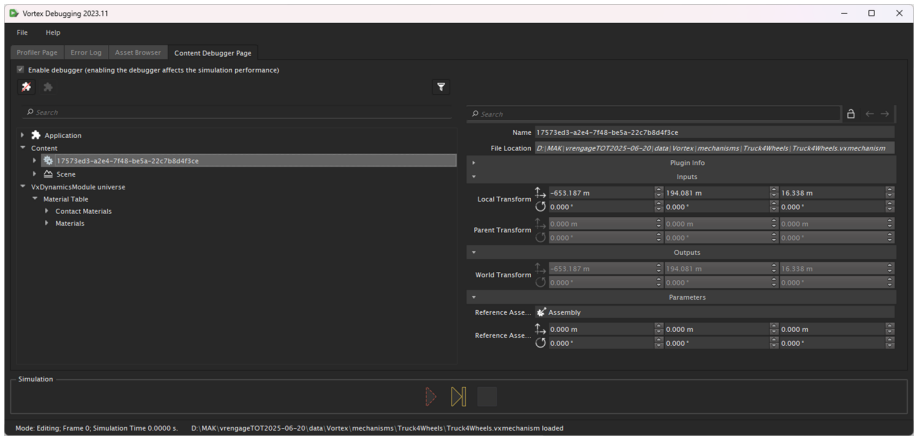
Viewing and Modifying Vortex-Related Parameters

➤ **To view the parameters and make temporary modifications:**

1. Start by selecting the vehicle you want to examine in VR-Engage. Create or select an existing vehicle and engage it in the driver role.
2. In the Vortex Debugging window, select the Content Debugger Page tab and expand the Content list in the left-hand pane. It displays a list of UUIDs, each corresponding to a Vortex-based vehicle currently running in the session. You can locate the entity you want to examine (that is, the vehicle currently engaged) by looking up its UUID in the VR-Engage Debug Menu and matching it to the entry in the Vortex Debugging window.
3. Click to expand the desired entry.

   When you initially select a given Vortex vehicle entry, the right-hand pane in the window expands to display information including the name (UUID), Vortex mechanism file, and the position and orientation of the entity.

Figure 114. Vortex Debugging tool, entity selected



4. Expand the entity entry in the left-hand panel to expose the structure of the underlying Vortex mechanism. From there, you can select various elements to view and modify in the model at runtime. (For a detailed explanation of the various elements in a Vortex mechanism, consult the CM Labs documentation.)
5. Select the elements in the entity list. The right-hand pane displays the editable details of the model as it runs. Elements you most likely want to observe and adjust are described in Table 40.

Using the Vortex Debugging Tool effectively requires a working knowledge of Vortex mechanisms. See the CM Labs Vortex Studio documentation to learn more. You can modify some parameters in the debugging tool at runtime, but not all of them. Note that when you modify a value in the debugger, the new value persists only for the duration of the session. To make the changes permanent, you must edit the vehicle configuration using the SOE.

Table 40 lists the most commonly used sections.

Table 40. Key debugging interface sections

Section	Description
Assembly/Chassis	View and modify fundamental vehicle properties including mass and center of mass parameters.
Vehicle/Dynamics Component	Access vehicle drivetrain attributes through the Parameters tab.
Configuration Interface	This section contains nearly all movement system parameters that are configurable through system configuration in the SOE. Changes to parameters are felt immediately while driving. This real-time adjustment capability is particularly valuable for tuning wheel-related parameters and visualizing their effects.
VR-Engage Output	Displays the raw output data from the Vortex model being passed to VR-Engage. Use this section to verify that vehicle behavior and HUD data visualization accurately reflects the underlying Vortex model output.

Using the VR-Engage Debug Menu for Vortex Vehicles

The VR-Engage Debug Menu has a tab dedicated to Vortex-specific details, which the user can examine at runtime for the current ownship. It provides runtime information on:

- Vortex terrain-related data about paging requests, collision geometry, dynamic terrain
- Vehicle interaction with terrain
- Articulated parts

Data is presented both as text and as graphical 3D overlays in the main view.

- Debug menus display over the interface, and clicks in the debug menu also register on the interface. Be sure to position the debug menu so that it is not over any buttons.
- To read more about the VR-Engage Debug Menu, see "4.7. Using the VR-Engage Debug Menu" on page 130.

➤ **To view Vortex-specific data for an engaged entity:**

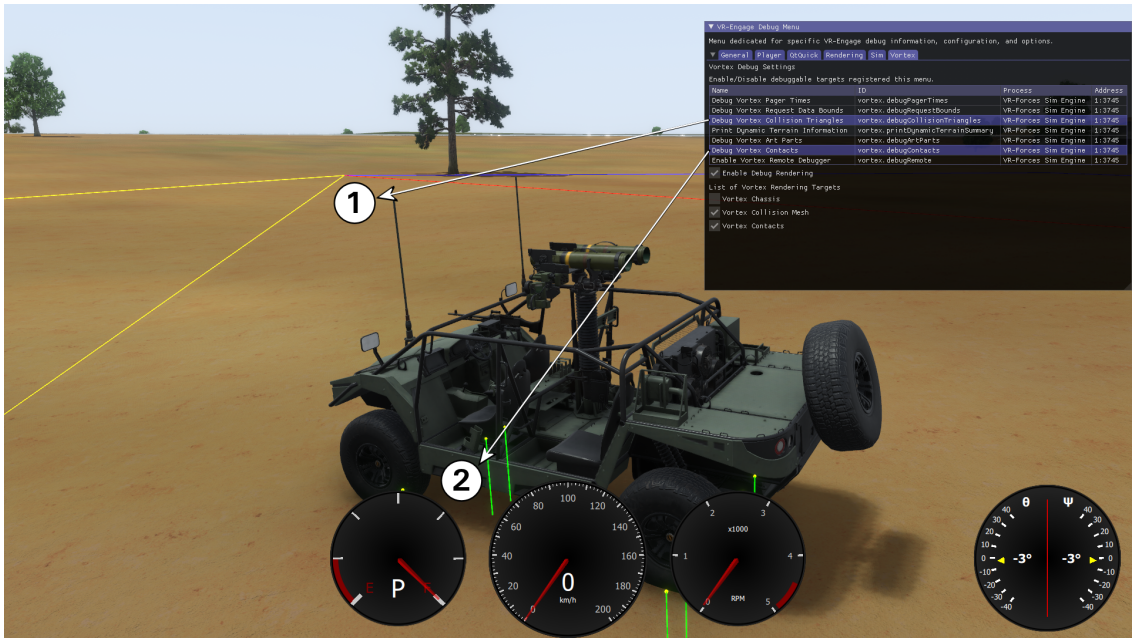
1. Launch VR-Engage and load a scenario (either hosted or unhosted).
2. Select or place the entity that you want to test and engage in the driver role.
3. Press **Shift+F4**. The VR-Engage Debug Menu opens.
4. In the General tab, select the Enable Debug Messages check box. (You might need to press **Ctrl** to regain access to the cursor.)
5. Select the Vortex tab.
6. In the Enable/Disable Debuggable Targets table, select one or more targets that you want to debug.
7. Select the Enable Debug Rendering check box.
8. Depending on the debuggable targets that you have selected, one or more tendering targets display in the List of Rendering Targets. Select the check boxes that you want to visualize.
9. Toggle the debuggable targets and rendering targets as needed.

In Figure 115, the Vortex collision mesh and contact points are enabled:

- 1 - The terrain collision mesh displays as colored triangles.
- 2 - Contact points/normal vectors display as yellow points with green line segments.

The debugger's visualizations are useful for debugging issues related to the terrain.

Figure 115. Vortex debugging information





15. Model Customization Tutorial

There are a number of tasks involved in the process of creating or customizing a entity for your use. This tutorial shows you how to take an existing VR-Forces model and customize it.

15.1. Overview of Customizing a Model 362
15.2. Extending VR-Engage.sms 362
15.3. Editing the Entity in the SOE 363
15.3.1 Promote the Entity to your SMS 363
15.3.2 Review the Entity's Systems 364
15.4. Editing the Tank's .entity File 365
15.4.1 Specify a VR-Engage-Capable Platform 365
15.4.2 Add the VR-Engage Tag 366
15.4.3 Add the Driving Role Component 366
15.5. Test Driving Your Tank 366
15.5.1 Copy the SMS Files 367
15.5.2 Launch VR-Engage with Your SMS 367
15.5.3 Drive Your Tank 368
15.6. Creating the Views for Your Tank 368
15.6.1 Create a New Driver Views File 368
15.6.2 Override Tank Driver View with Your View 369

15.1. Overview of Customizing a Model

This tutorial demonstrates how to add a VR-Engage driver role to a VR-Forces CGF entity. Specifically, we will edit the existing T-72 MBT tank to be player controllable in that role. This T-72 MBT tank is a CGF entity located in the VR-Engage SMS, which is an extension of VR-Forces's EntityLevel SMS. Out of the box, it does not have any VR-Engage role player capabilities.

To complete the tutorial, you must have VR-Engage and a compatible version of VR-Forces with the VR-Engage extensions. See "Installing the VR-Forces Extensions" on page 39.

In this tutorial, you will:

1. Create a custom SMS that extends the VR-Engage SMS.
2. Promote an entity to your custom SMS.
3. Make the entity drivable by players and test drive it.
4. Add custom views for the driver role.

The following are beyond the scope of the tutorial:

- Configuring a new Vortex model tailored to this vehicle. Instead, we take advantage of the existing Vortex mechanism file defined for tanks because it is a close enough approximation for the T-72 MBT tank. For more about configuring Vortex models, see "14.4. Ground Vehicles and Vortex" on page 347.
- Mapping the wheel and sprocket articulated parts on the 3D model to the simulation model. The T-72 MBT tank we use in this tutorial has already been configured by an artist to include the necessary articulated parts.
- Configuring pop-up menus that allow human characters to walk up to the vehicle and choose a role.

15.2. Extending *VR-Engage.sms*

First, we will create a new SMS that extends *VR-Engage.sms*. The new SMS will include all of the existing VR-Engage-controllable entities, and then we'll start configuring the T-72 MBT tank. We start by extending the SMS in our VR-Forces installation.

➤ **To create an extension of the existing SMS:**

1. Open the Simulation Object Editor, which is part of your VR-Forces installation. In Windows, open the Start menu and then choose **MAK Technologies > VR-Forces 5.2d Simulation Object Editor**.
2. Choose **File > New Simulation Model Set**. The New Simulation Model Set dialog box opens.
3. Select New Simulation Model Set. (This is the recommended option.)
4. Click Next. The Simulation Model Set Type page displays.
5. Select Extend Existing Simulation Model Set.
6. Click Next. The Extend Existing Simulation Model Set page displays.

7. Select *VR-Engage.sms*.
8. Click Next. The New Model Set Name page displays.
9. In the File Name box, enter `TankTutorial`.
10. Click Finish. The Simulation Object Editor loads systems information from the included model sets and displays the new SMS. This can take a few minutes.

    Since all of the simulation objects are included from other SMSs, none of them are currently editable. They are listed in the Objects List using gray text.

15.3. Editing the Entity in the SOE

In the Simulation Object Editor (SOE), you must make changes to your custom SMS to enable you to make edits to the T-72 MBT tank.

We'll promote the entity and then change the movement system to the VR-Engage Tracked Vehicle system. That system has the same capabilities of the Tracks system, which MAK ONE applications use when the vehicle is under CGF control. It also has the higher-fidelity Vortex-based movement model when the tank is under player control.

- In the SOE, you see other parameters that are available for configuration in the Movement tab such as mass, max-speed, max-slope, and so on. These only apply to the lower-fidelity VR-Forces CGF movement model. The comparable model parameters for the higher-fidelity Vortex model are in the Vortex mechanism file, separate from the SOE settings.
- In the General tab for theT-72 MBT tank, you can see that the Platform is set to Ground Vehicle. This is the platform file set for most ground vehicles and contains the components and parameters common to ground vehicle platforms. Like the Movement system we need to change it to a version that extends the entity with additional VR-Engage capabilities. You cannot edit that using the SOE, however, so we will edit it in a text editor later in this tutorial.

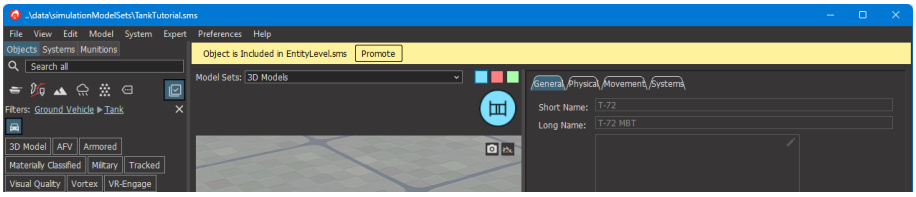
15.3.1 Promote the Entity to your SMS

To make an existing VR-Forces entity a configurable part of your custom VR-Engage SMS, you promote it.

➤ **To promote the tank entity:**

1. In the Objects list, select T-72 MBT. Optionally, you can use the filters to narrow the list.
2. In the yellow bar near the top of the window, click Promote (Figure 116). The entity is now available to edit in your SMS.

Figure 116. Promote Simulation Object



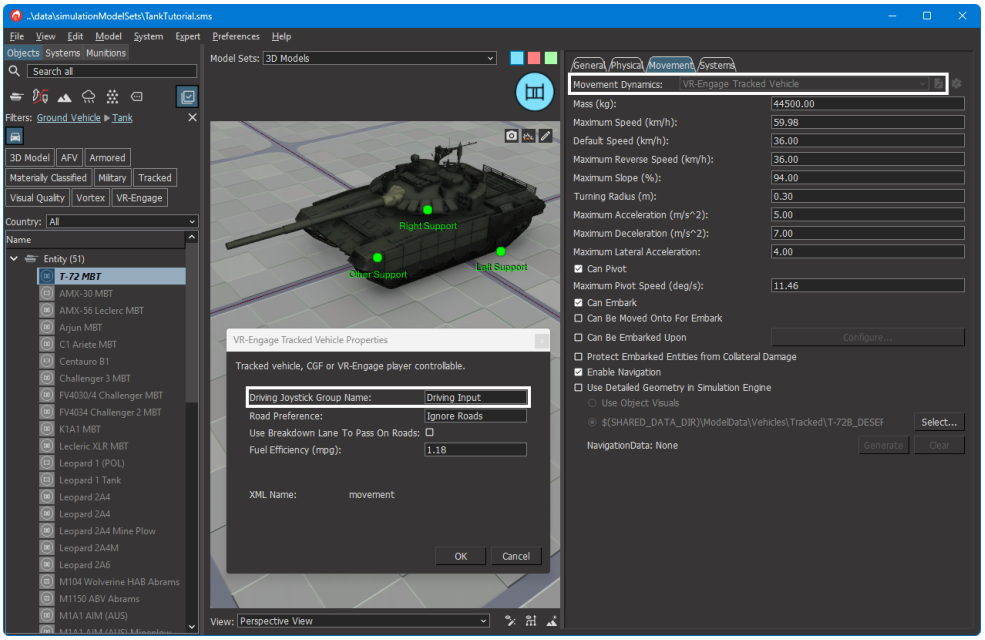
15.3.2 Review the Entity's Systems

To make the T-72 MBT tank entity drivable as a player-controlled entity, replace the CGF-only movement system with the corresponding VR-Engage movement system. It provides all of the CGF capabilities for autonomous movement, allowing it to be controlled by the CGF in a scenario, and also adds the ability for players to assume control and drive the vehicle themselves.

➤ **To configure the movement system:**

1. Select the Movement tab.
2. From the Movement Dynamics list, choose VR-Engage Tracked Vehicle.
3. Click Edit Properties ( ). The VR-Engage Tracked Vehicle Properties dialog box opens (Figure 117).
4. Note the value for the Driving Joystick Group Name, which is Driving Input.
5. Choose **File > Save Simulation Model Set**.
6. Close the SOE. If you are prompted to do so, click OK to save your changes again.

Figure 117. T-72 MBT tank in SOE



15.4. Editing the Tank's *.entity* File

To configure our tank as an entity that can be controlled and driven by VR-Engage players, we need to edit the *T-72 MBT.entity* file. We can edit the entity file in a text editor and then open it again in the SOE without losing the changes we made.



You can look at another *.entity* file, such as *./data/simulationModelSets/VR-Engage/vrfSim/vreM1A2_Abrams_MBT.entity*, to see its configuration.

15.4.1 Specify a VR-Engage-Capable Platform

All VR-Engage entities must be configured with a VR-Engage-specific platform file. Platform files contain components and parameters common to all entities of a given category: human, ground vehicle, air vehicle, and so on.

The VR-Engage versions of these platform files extend the corresponding entity-level SMS versions with various overrides related to embarkation, weapon firing, state repository extensions, and more.

For our tank, we'll replace `Ground_Vehicle.ope` with `Vortex_Ground_Vehicle.ope`. We do this by editing the tank's *.entity* file.

➤ **To open the entity file and edit the platform:**

1. In your VR-Forces installation, navigate to *./data/simulationModelSets /TankTutorial/vrfSim* and then use a text editor to open the *T-72 MBT.entity* file.
2. Near the top of the file, edit the simObject entry to change the platform. When you are finished, the line should read:

```
  <simObject objectType="1:1:222:1:2:1:0"
matchType="1:1:222:1:2:1:0" platform="@(platforms-dir)/Vortex_
Ground_Vehicle.ope">
```

3. Save the file.

15.4.2 Add the VR-Engage Tag

The `gui-categories` parameter determines the filters available in the Create Object Palette that allow users to filter the list of entities when they are making their selection. Adding the `VR-Engage` tag to this parameter helps VR-Forces users choose VR-Engage-controllable entities when they create a scenario.

➤ **To add the VR-Engage tag:**

1. In the *T-72 MBT.entity* file, locate the `gui-categories` parameter. It's near the bottom of the file.
2. Add "`VR-Engage`" to the list. It should now look similar to:

```
  <string paramName="gui-categories">"3D Model" "Entity" "Armored"
"Ground Vehicle" "Military" "Tank" "Tracked" "VR-Engage"</string>
```

3. Save the file.

15.4.3 Add the Driving Role Component

Finally, we add the tank driver role to the *.entity* file. This connects our tank to the driving system (the `Vortex_Ground_Vehicle.ope` platform) that we specified earlier. The role itself is defined in a role definition file, in this case *tankDriver.lua*. The `<role>` entry in *.entity* files has two parameters, as described in "14.3.2 The vrEngageRoles and role Entries" on page 339.

➤ **To add the driving role:**

1. Locate the `<vrEngageRoles/>` entry, near the bottom of the *T-72 MBT.entity* file and immediately before the closing tag of `</simObject>`.
2. Replace the entire `<vrEngageRoles/>` entry with:

```
<vrEngageRoles>
  <role paramName="Driver" defaultDisplayLayout="1 Screen
      Horizontal">@(DATA_DIR)\simulationModelSets\VR-Engage
      \roles\tankDriver.lua</role>
</vrEngageRoles>
```

3. Save the file.

15.5. Test Driving Your Tank

To test drive your tank nd make sure that it is VR-Engage-controllable, first we need to copy the SMS files from the VR-Forces directory to the VR-Engage directory. We also need to update the *playerStation.lua* file to use the TankTutorial SMS. After we complete those tasks, we can use an unhosted session of VR-Engage to create our entity and drive it.

15.5.1 Copy the SMS Files

➤ **To copy the SMS files:**

1. Navigate to your VR-Forces installation and the *./data/simulationModelSets* directory.
2. Select the *TankTutorial.sms* file and the *TankTutorial* directory.
3. Copy these files.
4. Navigate to your VR-Engage installation and the *./data/simulationModelSets* directory.
5. Paste the files.

15.5.2 Launch VR-Engage with Your SMS

While VR-Engage supports multiple SMSs, we do not need to use more than the one you created earlier. It includes everything from the default *VR-Engage.sms file*, plus your T-72 MBT tank.

➤ **To launch VR-Engage with your custom SMS:**

1. Open a command prompt or terminal window.
2. Navigate to your VR-Engage *./bin64* directory, for example:

```
cd "C:\MAK\vrEngage2.2\bin64"
```

3. Launch VR-Engage with the `--setting` command-line argument to override the default SMS, for example:

```
vrEngage.exe --setting
"playerStationApp.defaultSimulationModelSets
= {'$(DATA_DIR)/simulationModelSets/TankTutorial.sms'};"
```

Alternatively, you can create a custom settings file that overrides the default SMS. Use this approach when you have multiple settings that you want to override in *playerStation.lua*.

➤ **To load your SMS using a custom settings file:**

1. Navigate to *./appData/scripts* in your VR-Engage installation
2. Using a text editor, create a new file named *tankTutorialSettings.lua* with the following content:

```
-- Custom settings for Tank Tutorial
-- This file overrides settings from playerStation.lua.
playerStationApp.defaultSimulationModelSets = {
  "$(DATA_DIR)/simulationModelSets/TankTutorial.sms"
};
```

3. Save the file.
4. Launch VR-Engage using the custom settings file, for example:

```
vrEngage.exe --customSettingsFile tankTutorialSettings.lua
```

> The `$(DATA_DIR)` variable points to the *./data* directory in your VR-Engage installation.

15.5.3 Drive Your Tank

We will start an unhosted scenario in VR-Engage that uses your SMS. (You cannot use any of the scenarios included with VR-Engage to place and drive your tank because they all use the *VR-Engage.sms*.)

➤ **To start an unhosted session and then drive your tank:**

1. Launch VR-Engage using one of the methods described in the previous section.
2. In the Choose Connection screen, select the connection that you want to use and then click Connect. DIS (7) Localhost is sufficient for a simple test.
3. In the Choose Session screen, click Start Unhosted Session.
4. In the Choose Terrain list, select Ground_DB II.mtf.
5. Click Load Terrain. VR-Engage starts the sim engine and loads the terrain.
6. In the Choose Role panel, select Create a New Entity.
7. Select your T-72 MBT tank and then click on the terrain to place it.
8. Click Engage.
9. Click the Play button () to start the scenario.
10. Drive the tank. See "7.2. Driving Ground Vehicles" on page 186 to learn more about operating tanks.
11. You can change the viewpoint as described in "5.5. Changing the View Mode" on page 147.

    These viewpoints were defined for another vehicle. We'll customize the viewpoints in the next sections of this tutorial.

15.6. Creating the Views for Your Tank

If you changed your viewpoint in the previous procedure, you may have noticed that one of the views is close to the driver's perspective but was partially obscured by the vehicle. That viewpoint was configured for a different vehicle, the M1A2 Abrams. The geometry of the T-72 MBT tank's 3D model is different and we want to create a new driver view for our tank, which we do by creating an observer views file.

15.6.1 Create a New Driver Views File

This procedure assumes that you have completed the tasks in "15.5. Test Driving Your Tank" on page 366 and currently have the T-72 MBT tank in an unhosted session.

➤ **To create the observer views file:**

1. If you are in the driver role of T-72 MBT tank, press **Esc** and then click Choose Role.
2. Press **Shift+F12** to switch to Admin mode.

3. Choose **View > Attachments Panel**.
4. Choose **View > Observer Views Panel**.
5. Optionally, undock these two panels and move them somewhere that you can see them clearly.
6. Press the Spacebar to position the observer directly behind the entity. Your tank's marking text now displays in the Attached Objects list in the Attachments Panel.
7. In the Observer Views panel, click Clear Observer Views (✕). You want to start with a blank slate of viewpoints.
8. Click Yes in the confirmation message.
9. In the Attachment Panel, change the attachment mode from Follow to Mimic.
10. Using the standard keyboard controls for the observer, position the viewpoint where you want it to be.
11. In the Observer Views panel, click Add Observer View (+). It is added to the list with a default name.
12. Optionally, you can edit the name. Double-click to edit the name and then press **Tab** when you are finished, or click on the VR-Engage title bar. For this tutorial, we refer to this viewpoint as "Driver View".
13. In the Observer Views Panel, click Export All Views to a File ().
14. In the confirmation message, click Save As. The Save Observer File As dialog box opens.
15. In the File Name box, enter `T-72MBTSavedViews`.
16. Click Save. The file is saved in *./appData/settings/vrEngage*.

15.6.2 Override Tank Driver View with Your View

For this tutorial, we will not create a new role or edit the existing *tankDriver.lua* file. Instead, we'll edit the entity file, referencing our saved views file and setting our view as the initial view.

Use overrides in the *.entity* file to override the saved views. To learn more about overrides, see "14.3.3 The parameterOverride Entry" on page 341. For example, if you look at the *./data/simulationModelSets/VR-Engage/roles/tankDriver.lua* file, you can see the saved views and initial views settings that we are going to override. They are in the DtSavedViewUpdater component (lines 24 and 25).

```
21    ["savedViewUpdater"] = {
22      componentType = "DtSavedViewUpdater";
23      priority = 10;
24      savedViews="m1DriverViews.osrx";
25      initialView="1 - Driver";
26      allowChangingViews = true;
27      viewChannelKeywordMap={
28      };
29      clippingPlaneSettings = {
```

```
30        ["default"] = {
31          { mode = "default"; nearClipDistance = 0.25; clipPolicy =
    0; dynamicNearClipAttachedObjectPolicy = 0; }
32        };
33      };
34    };
```

➤ **To add your viewpoint to the *.entity* file and override the role settings:**

1. In your VR-Engage installation, open the *./data/simulationModelSets /TankTutorial/vrfSim/T-72 MBT.entity* file in a text editor.
2. Edit the driver role entry to add parameter overrides to use your views file and driver view:

```
  <vrEngageRoles>
    <role paramName="Driver" defaultDisplayLayout="1 Screen
Horizontal">@(DATA_DIR)\simulationModelSets\VR-
Engage\roles\tankDriver.lua
      <parameterOverride
paramName="savedViewUpdater.savedViews">"T-
72MBTSavedViews.osrx"</parameterOverride>
      <parameterOverride
paramName="savedViewUpdater.initialView">"Driver
View"</parameterOverride>
    </role>
  </vrEngageRoles>
```

3. Save the file.

To test and see your changes, you must close and restart VR-Engage to allow it to read your parameter overrides. (Remember to use one of the methods in "15.5.2 Launch VR-Engage with Your SMS" on page 367.)



16. Creating VR-Engage Scenarios in VR-Forces

This chapter describes how you can create your own scenarios that include VR-Engage controllable entities. While VR-Engage can load or join a session of any entity-level VR-Forces scenario, players can only choose an entity that is configured as a VR-Engage controllable entity.

VR-Engage does not allow you to create and save scenarios. You must use VR-Forces with the VR-Engage add-ons to create scenarios that include VR-Engage controllable entities.

16.1. Installing the VR-Forces/VR-Engage Extensions 372
16.2. Creating a VR-Engage Scenario in VR-Forces 372
16.2.1 Creating a Load-Balanced Scenario 374
16.2.2 Configuring Ground Vehicles on Geocentric Terrains 375
16.2.3 Vortex While Untasked 375
16.3. Running Your Scenario 376
16.3.1 Hosting a Scenario in VR-Forces 376
16.3.2 Hosting a Scenario in VR-Engage 377

16.1. Installing the VR-Forces/VR-Engage Extensions

Before you can create your own VR-Engage scenarios, you must install the *vrEngage#.#-win64-vc##-YYYYMMDD.VRF#.#-extensions* plug-in on a version of VR-Forces that is compatible with your copy of VR-Engage. In addition to other features, the installer extended VR-Forces with the *VR-Engage.SMS*, which includes entities that a VR-Engage user can control. These entities also have all of the built-in capabilities of the corresponding entity from the VR-Forces *EntityLevel.SMS*.

For more information, see "Installing the VR-Forces Extensions" on page 39.

16.2. Creating a VR-Engage Scenario in VR-Forces

You can use VR-Forces to create scenarios. If you want to include entities that can be controlled by VR-Engage players, you must use the *VR-Engage.sms* (or an SMS that is an extension of the VR-Engage SMS) and include entities that have been configured as controllable entities. "Chapter 13. Controllable Entities" on page 291 lists the entities that are included with VR-Engage. "Chapter 14. Adding and Editing VR-Engage Entity Models" on page 335 explains how to configure other models to be controlled by VR-Engage players. When you select the VR-Engage tag in the Simulation Objects Palette (Figure 119), you see a list of VR-Engage controllable entities.

Your scenario can include other entities that are not VR-Engage controllable and players can interact with them. The VR-Engage SMS is an extension of VR-Forces's Entity Level SMS, including all of the simulation objects, which allows you to add them to scenarios for VR-Engage. (Non-controllable entities are not listed in the VR-Engage Choose Role panel.)

You can assign tasks and plans to all entities in the scenario. If a player engages as a controllable entity, its plan or task is suspended while the player is engaged. That entity then resumes its assigned behaviors when the player disengages. You can also use plan triggers to give entities responses to players' actions.

Before creating the scenario, ensure that you are using a version of VR-Forces that is compatible with the version of VR-Engage that you want to use and also has the extensions plug-in for that version of VR-Engage. To check for compatibility, see the *VR-Engage Release Notes* or the Build Compatibility page on the MAK website at https://www.mak.com/build-compatibility.

➤ **To create a scenario that is compatible with VR-Engage:**

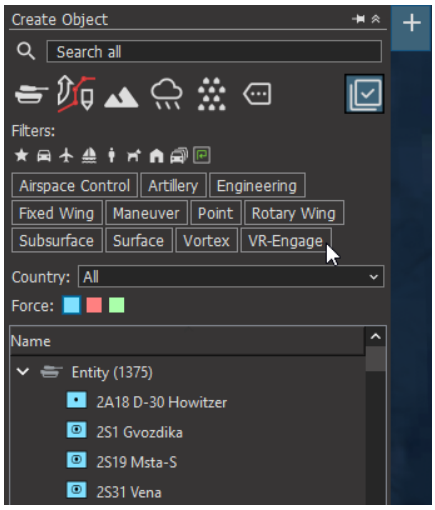
1. Open VR-Forces.
2. In the Scenario Startup dialog box, specify VR-Engage as your Simulation Model Set (Figure 118), or select an SMS that you created.
3. Select a terrain database.

Figure 118. New scenario in VR-Forces Scenario Startup



4. Click OK.
5. By default, VR-Forces displays the Create Object Palette. You can add any simulation objects or tactical graphics included in the palettes, not only VR-Engage controllable entities.
6. To filter the list of simulation objects to those that can be controlled by VR-Engage players, use the VR-Engage tag (Figure 119). You can also enter VR-Engage in the Search box.

Figure 119. VR-Engage filter in Simulation Objects palette



7. Add entities, tasks, plans, and so on, as described in the *Scenarios* section of the *VR-Forces Users Guide* or help.
8. Save your scenario.

16.2.1 Creating a Load-Balanced Scenario

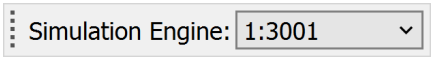
In VR-Forces, you can create a scenario where the simulation objects are distributed across different sim engines as part of the saved scenario. When you load the scenario, the simulation objects are automatically distributed across those sim engines. These can be VR-Forces and VR-Engage sim engines.

If your use case includes having multiple VR-Engage stations that run sim engines to host the player entities (as described in "3.1.4 Hosting an Entity in VR-Engage While Participating in a VR-Forces Session" on page 60), you can take advantage of load balancing the scenario when create it.

➤ **To create a load-balanced scenario:**

1. Launch the VR-Forces and VR-Engage stations. Be sure that each station that will participate in the scenario are set to run a sim engine as described in "Configuring VR-Engage to Start a Sim Engine" on page 61.
2. In VR-Forces, choose **View > Toolbars > Selected Simulation Engine Toolbar**. The toolbar is added to the GUI (Figure 120).

Figure 120. Selected Simulation Engine toolbar



3. When placing the entities that VR-Forces will control, select the VR-Forces simulation engine from the toolbar and then add the entities.
4. To place a VR-Engage entity that a player will control:
    a. Select the appropriate sim engine from the toolbar.
    b. Select the entity from the Create Object palette and then place it on the terrain.
    c. Repeat for each VR-Engage controllable entity. You cannot place two entities on the same VR-Engage sim engine.
5. Save the scenario.

Load Balancing for Ground Vehicles

When deciding how to load balance ground vehicles in a scenario, consider the following for player-controlled Vortex entities:

- Vortex entities require a stable frame rate. The Vortex simulation model is sensitive to long frame times, therefore try to avoid situations that result in occasional long frames. As a rule of thumb, maintain a frame range greater than 20 FPS on simulation (sim) engines simulating Vortex entities.
- High-fidelity physics interactions between Vortex vehicles (specifically colliding or towing) require that all Vortex entities involved in the interactions be simulated on the same sim engine. If this type of interaction is important in your scenario, group Vortex entities together on the same sim engine, provided it can still maintain an adequate minimum frame rate.

If high-fidelity collisions are not a priority, you can configure one Vortex vehicle per player station's sim engine. This provides improved performance for that vehicle and is a more scalable approach when there are multiple Vortex entities in the scenario.

16.2.2 Configuring Ground Vehicles on Geocentric Terrains

VR-Engage ground vehicles use the Vortex physics libraries for their movement systems. When a vehicle moves on a geocentric terrain, VR-Engage must convert the geocentric coordinates to a flat earth coordinate system. The conversion is based on a user-defined origin that, ideally, represents the location where the majority of simulation activity will take place. As vehicles move further away from the origin, projection errors can occur. When distances from the origin exceed several thousand kilometers, the projection error may be noticeable and vehicles may drift from their expected line of movement.

Default Origin

By default, VR-Engage sets the origin point based on the position of Vortex entities. When you load an existing scenario that includes Vortex vehicle entities, the origin is at the average position of all Vortex entities. When you create a new scenario or load one that does not include any Vortex entities, the origin is set at the position where you initially place the first Vortex entity.

Specify an Origin

If you want, you can specify an origin that is close to the location where you the simulated vehicles will be most active. To set the origin, specify values for the **coordinateConverterOrigin** parameter in *./appData/scripts/vortexConfig.lua*. The values are in decimal degrees for latitude and longitude and meters for elevation. If the parameter is not specified, the default values are 0,0,0.

This parameter is used only for geocentric terrains. For projected terrains, the terrain data is used directly and no conversion is needed.

16.2.3 Vortex While Untasked

By default, ground vehicles in VR-Forces scenarios for VR-Engage do not have Vortex enabled, which means they do not have high-fidelity vehicle dynamics. This improves scenario performance, however, you might need untasked vehicles to have that physics in your scenario. If you want a ground vehicle to use the Vortex physics model even when it is not occupied by a player or otherwise tasked in the scenario, you can enable it using the Vortex While Untasked set data request.

➤ **To enable or disable Vortex on a ground vehicle:**

1. In VR-Forces, select the ground vehicle entity.
2. Choose **Set > Action > Vortex while untasked**. The Vortex While Untasked parameters display.
3. Select or clear the Enable check box and then click OK.

16.3. Running Your Scenario

When hosting a scenario, you can host it in either VR-Forces or VR-Engage. Use the application that best suits your needs.

VR-Engage performs better when it is hosting only the ownship in a scenario. If you have multiple users connecting different instances of VR-Engage to a scenario hosted in VR-Engage, or even if you have one user running a large scenario, performance and frame rate can deteriorate. Some entities, such as Vortex-based ground vehicles, require a minimum frame rate (approximately 20 FPS) to remain stable. You should use VR-Engage to host only simple scenarios or for basic testing.

VR-Forces is the preferred choice when hosting a session where there will be multiple players or there are a number of entities in the scenario. Depending on the complexity of your scenario, you may want to host the simulation on one or more dedicated simulation engines. You can host these engines on one or more machines as needed to distribute the workload. (Typically, multiple simulation engines are not necessary. See the chapter on *Optimizing Performance* in the *VR-Forces User Guide*.)

Regardless of where the scenario is hosted, VR-Engage instances can join the scenario and the players can choose a role for a controllable entity; see "3.2.1 Choosing an Existing Role" on page 64. If the scenario is hosted by VR-Forces, you can also assign roles to players as described in "3.2.4 Assigning Roles from VR-Forces" on page 67.

16.3.1 Hosting a Scenario in VR-Forces

When you host a scenario using one or more instances of VR-Forces, you would load the scenario and then have the VR-Engage stations connect to it. This works only when the connection information is the same in all instances of VR-Forces and VR-Engage.

For more information, see "3.3. Connecting to VR-Forces" on page 69.

> While VR-Engage can join any entity-level scenario, players can only participate in scenarios that include VR-Engage controllable entities.

Hosting a Scenario with Multiple Sim Engines

When you load-balance a scenario, it saves the mapping between objects and simulation engines in the scenario's object mapping file. If it does not find the required simulation engines, the Remap/Start Simulation Engine dialog box displays. In that case, click Cancel and then do one of the following:

- Start each associated station and ensure that the VR-Engage ownship simulation engines are started and ready.
- Load the scenario as described in "Manually Load Balancing the Scenario in VR-Forces" on page 62.

16.3.2 Hosting a Scenario in VR-Engage

If you want to host your scenario in VR-Engage, such as a simple scenario for one player or for testing purposes, you simply need to copy your scenario to the *./userData/scenarios* directory. When you launch VR-Engage and choose to Host VR-Forces Session, your scenario is included in the list of available scenarios.

For more information, see "3.1.1 Hosting a VR-Forces Session" on page 57.



17. Customizing Overlays

This chapter provides examples of how you can customize the overlay for an entity.

17.1. Overview of Custom Overlays 380
17.1.1 Types of Overlays 380
17.1.2 Examples of Each Type of Overlay 380
17.2. Adding a QML Overlay 381
17.2.1 Testing Your QML Overlay 382
17.3. Adding a Custom Sensor Overlay 383
17.4. Adding a GL Studio Overlay 383
17.4.1 Creating a GL Studio RSO 384
17.4.2 Creating VR-Vantage GL Studio Updaters 384
17.4.3 Configuring the VR-Vantage GL Studio Model Definition 384
17.4.4 Modifying Role Configuration Scripts 385

17.1. Overview of Custom Overlays

There are several ways to customize the overlays for an entity's role. Depending on the complexity of the overlay, you may need to modify a configuration script, write code, or use GL Studio.

This chapter provides typical examples of customizing overlays using data that are stored as player station state attributes. It does not cover overlays that involve extensive development, or overlays that require additional data that is not currently available out of the box. If you need a complex overlay that requires C++ development, such as those used in 3D cockpits or those used with VR, AR, or MR displays, please visit https://www.mak.com/mak-one/support/portal and request assistance.

17.1.1 Types of Overlays

The types of overlays covered in this chapter are:

- QML overlays, which use QtQuick to create an overlay that is then referenced in the role configuration scripts. You can create complex and dynamic 2D overlays with relative ease and no additional tools. It does require some familiarity with JavaScript, the Qt Modeling Language (QML), and Lua script. Please see https://doc.qt.io/qt-5/qmlreference.html and https://www.lua.org/docs.html.
- Sensor overlays, which use the built-in capabilities of VR-Vantage and are further customized by allowing you to declare which elements you want to make visible. This also requires you to edit the Lua configuration files.
- GL Studio overlays, which are dynamic overlays that require some C++ development. This requires using the GL Studio editor from DiSTI to create an RSO. To use your own RSO, you must have a GL Studio runtime license from DiSTI. Please visit https://www.mak.com/mak-one/support/portal for assistance with licenses.

17.1.2 Examples of Each Type of Overlay

QML and GL Studio overlays can be dynamic and include:

- Text on the screen to reflect the state attribute of the entity.
- Rich graphics.
- User interaction (GL Studio only).

Sensor overlays allow you to have a dynamic UAV-style overlay that display:

- Target information.
- Location data.
- Reticle or cross-hairs.
- Heading, pitch, or roll information.
- Compass elements.

Detailed examples are covered in the following topics.

17.2. Adding a QML Overlay

You can add an overlay using the `DtQtQuickOverlay` component in the role definition. This component defines the QML (Qt Modeling Language) file to load and the connections between player station state attributes that are fed into QML for display in the overlay and their corresponding QML data structure names.

Player station state attributes are specified in the DtQtQuickOverlay component in the role file. You can find the player station attributes that are currently available to display in existing role files. The available attributes vary by role. Below is an example of the basic driver role using the `DtQtQuickOverlay`, found in the *./data/simulationModelSets/VR-Engage/roles/driver.lua* definition file:

```
  components = {
    ["hudMapper"] = {
      componentType = "DtGlStudioHudMapperLogic";
      priority = 10;
      modelDefinitions = {"off"};
    };
    ["quickOverlay"] = {
      componentType = "DtQtQuickOverlay";
      priority = 10;
      qmlFilename = "HUDS/groundVehicle/dashboard.qml";
      qmlDataItemObjectName = "valueSource";
      qmlDataItemBindings = {
        { stateAttribute = "speed" };
        { stateAttribute = "rpm" };
        { stateAttribute = "set-gear"; qmlProperty = "gear" };
        { stateAttribute = "resource:movement|fuel"; qmlProperty =
"fuel" };
      };
    };
 ...
  }
```

Figure 121. Car driver overlay



First, we have a hudMapper component to disable any GL Studio HUD that is currently mapped to the vehicle, `DtGlStudioHudMapperLogic` with the line `modelDefinitions = {"off"};`.

Second, we have quickOverlay component, `DtQtQuickOverlay`, to specify the QML overlay. The parameters are:

- qmlFilename. The path to the QML overlay file.
- qmlDataItemObjectName. Defines the QML item where the properties are set. This requires a corresponding `Item` in QML identified by a matching `objectName`. You can see an example of the car's `Item` in *./data/UI/HUDS/groundVehicle/dashboard.qml*.
- qmlDataBindingItems. Maps the state attributes, set in C++, to the QML item properties. The stateAttribute provides the value to use and the qmlProperty is the value to set. In the example above, the "set-gear" stateAttribute maps to the "gear" qmlProperty in the QML valueSource `Item`.

You can find the QML documentation at https://doc.qt.io/qt-5/qmlreference.html.

17.2.1 Testing Your QML Overlay

You can quickly test your QML file using the small QML test application, *qmlTestApp.exe*, which is included with VR-Engage. This app quickly loads and visualizes your QML file. To use it, navigate to *./bin64/qmlTestApp.exe* and run it with the relative path to your QML file. For example:

```
qmlTestApp.exe -d ../data/UI/HUDS/groundVehicle -f gunsight.qml
```

You can press F5 at any time to reload the file. This allows you to iteratively check whether your edits give you the look that you want.

17.3. Adding a Custom Sensor Overlay

If you need an overlay that is similar to the VR-Vantage sensor overlay (Figure 122), you can configure it by adding a `DtVrvSensorOverlayUpdater` component to a sensor operator's role definition Lua script and indicating which sections of the built-in overlay should be seen in your version.

Figure 122. Sensor overlay



For details on customizing the overlay for sensor operator roles, see "10.4. Configuring the Sensor Operator Overlay" on page 234.

> Sensor overlays pull data from the current observer. If you need sensor overlay sections in addition to those in the built-in overlay, you must use the VR-Vantage API to build a plug-in and add the section.

17.4. Adding a GL Studio Overlay

VR-Engage supports overlays created with DiSTI's GL Studio. VR-Engage uses VR-Vantage's GL Studio integration to configure and update any loaded RSOs. You must have a GL Studio license to develop a GL Studio RSO.

To add a new GL Studio overlay, you need to complete a series of procedures:

1. Create the new RSO using DiSTI's GL Studio editor.
2. Create VR-Vantage GL Studio updaters as needed.
3. Create a new VR-Vantage GL Studio model definition.
4. Modify the role configuration scripts to use the new model definition.

Please refer to DiSTI's documentation on GL Studio for details on creating an RSO.

Figure 123. GL Studio overlay



17.4.1 Creating a GL Studio RSO

To create a new GL Studio RSO, use DiSTI's GL Studio application. You can create the overlay and its properties. The properties provide a way to update values seen on the overlay or drive the dynamic components in the overlay.

The output from GL Studio is source code that is compiled into a DLL file. Save the DLL file in *./data/huds64*.

17.4.2 Creating VR-Vantage GL Studio Updaters

To update the RSO with data, you must configure the RSO's model definition with the list of VR-Vantage GL Studio updaters to provide data to the RSO.

For a list of available updaters, please see the section on cockpits in the *Wakes, Cockpits, 2D Icons, and Other Models* chapter in the *Adding Content to MAK ONE Applications Reference Manual*.

17.4.3 Configuring the VR-Vantage GL Studio Model Definition

VR-Engage includes the Model Definition Editor in the Visual Model Editors dialog, which allows you to configure model definitions for GL Studio RSOs.

For more information, please see the *Model Definitions* chapter in the *Adding Content to MAK ONE Applications Reference Manual*.

17.4.4 Modifying Role Configuration Scripts

VR-Engage entity roles are defined with Lua scripts in *./data/simulationModelSets/VR-Engage/roles*. To use your new GL Studio model definition when a role is activated, you need to add a `DtGlStudioHudMapperLogic` component to the role definition. In this example, we are using our `vreHUDPlusMFDs` model:

```
{ componentType = "DtGlStudioHudMapperLogic";
    priority = 10;
    modelDefinitions = {"vreHUDPlusMFDs"};
}
```

This component overrides the entity type mapping of VR-Vantage with the declared mapping.



18. Activating Scripts in VR-Forces

Characters in VR-Engage can activate scripts in the VR-Forces simulation engine.

18.1. Activating Lua Scripts in VR-Forces 388
18.2. Adding a Script to the Action Menu 388
18.3. Activating a Script 389

18.1. Activating Lua Scripts in VR-Forces

VR-Forces lets users add tasks and set data requests by writing Lua scripts. Lua is a scripting language that does not require compilers and software development tools like those required by the VR-Forces C++ toolkit. Also, you do not need a VR-Forces developers license to add Lua scripts.

VR-Engage characters can trigger scripts or activate them in the VR-Forces simulation engine. Once a script is triggered, there is no further involvement of the character you are playing. The script must have previously been written in VR-Forces. Then it must be added to the Action menu to be accessible to characters.

For information about how to write a script, please see the *VR-Forces User Guide*. This chapter explains how to add a script to the Action menu and how to execute it.

18.2. Adding a Script to the Action Menu

The Action menu is configured in the file *./data/simulationModelSets/VR-Engage/roles/baseRole.lua*.

The Action menu and the Scripts submenu are configured in the following lines of the file (comments have been removed):

```
updaters={
  DtActionMenuUpdater={
    menus={
      ["Action"] = {
        parent="";
        size={x = 400, y = 300};
        position={ x= 100, y = 300};
        elements={
          {label = "Flashlight:", action = "toggleFlashlight",
decorator="$STATE", stateAttribute="flashlight", stateList={"OFF",
"ON"} };
          {label = "Night Vision:", action = "toggleNvg",
decorator="$STATE", stateAttribute="NVG", stateList={"OFF", "ON"} };
          {label = "Radio", action = "cycleRadio", decorator="$STATE",
stateAttribute="currentRadio"};
          {label = "Task entity", action = "selectMenu", param =
"Select Entity"};
          {label = "Execute Script", action = "selectMenu", param =
"Scripts"};
            ...
          };
        };
        ...
      ["Scripts"] ={
        parent="Action";
        elements={
```

```
          {label = "Script 1", action="scriptedTask", param="Fire_For_
Effect_on_Location", cursor="selectLocationCursor"};
        };
      };
...
```

> The `label = "Execute Script"` line might be commented out. You need to uncomment it for the Execute Script menu command to be enabled.

The Scripts menu option is added as an element of the Action menu. It has an element for each script to be listed on the submenu. In this case, there is one script with these parameters:

- `label`. The text shown on the menu.
- `action`. Must be scriptedTask.
- `param`. The name of the Lua script.
- `cursor`. A location selected by the player. This is the only input to the script.

The only input that a VR-Engage character can provide to a script is a cursor position.

18.3. Activating a Script

You can tell VR-Forces to activate a script.

➤ **To activate a script:**

1. Open the Action menu.
2. Select Execute Script.
3. Press the **Right arrow** key or the right button on the D-pad to show the submenu. The Scripts menu opens.
4. Select the script that you want to activate.
5. Press the **Right arrow** key or right button on the D-pad.



IV. Appendices

The appendices contain troubleshooting and supplemental information.



A. Troubleshooting

This appendix describes solutions to common issues. If your problem is not covered here, check the MAK ONE Product Support Knowledge Base: https://www.mak.com/knowledge.

If you require assistance, visit MAK's Support Portal and provide any information about your issue. See "Reporting an Issue Using the .callstack.log and .dmp Files" on page 41.

A.1. Performance and Display Issues 394
A.2. Using the Tracy Client 394
A.3. The Fault Tolerant Heap 395
A.4. Graphics Card Binary Compatibility 395
A.5. Issues with VRAM 395
A.6. Remote Camera Placement 396
A.7. Other Limitations 396

A.1. Performance and Display Issues

There are a few issues that require you to add the application name to the *BlackApps.dat* file.

- Nahimic can severely impact performance and frame rates because it interacts with certain hardware capabilities, vantage leverages, in an extremely detrimental way.
- Under certain circumstances, opening VR-Engage on a second display prevents the program from starting.
- In addition, launching from the main display and then moving the mouse cursor to the other monitor causes the program to freeze and not respond.
- Nahimic can sometimes cause VR-Engage to crash.
- You might also experience crashes when trying to engage rotary- or fixed-wing aircraft.

If you experience any of these issues, edit the Nahimic *BlackApps.dat* file and add the application name (for example, vrengage.exe) to the list. You can search *C:\ProgramData* for *BlackApps.dat* to locate it. This file might be found in *C:\ProgramData\A-Volute\A-Volute.Nahimic\Modules\Scheduled\Configurator*, *C:\ProgramData\A-Volute\AVolute.28054DF1F58B4\Modules\ScheduledModules\Configurator*, or something similar.

A.2. Using the Tracy Client

VR-Engage includes the Tracy Client, which you can use to evaluate performance or save a file to send to MAK Support to help debug performance issues. To learn more about using the Tracy Client, read the documentation at https://bitbucket.org/wolfpld/tracy/downloads.

> ⚠ VR-Engage is compatible with the Tracy Client version that was included in the installation package. If you update Tracy Client, it might no longer be compatible.

➤ **To run the Tracy Client and save a file:**

1. Ensure that you are in Admin mode (**Shift+F12**).
2. Choose **Settings > Display**. The Display Settings dialog box opens. Select the Render Settings page.
3. Select the Performance Statistics Overlay check box. A statistics display is added to the window.
4. Next to Collect Stats, select the Tracy check box.
5. To run Tracy Client, from the Start menu, choose **MAK Technologies > VR-Engage 2.2 Tools**. In the Tools folder, double-click the Tracy shortcut.
6. Verify the client address and then click Connect. The Tracy Client displays the information from VR-Engage.
7. To save the file, click Connection (). The Connection dialog opens.

8. Click Save Trace.
9. Save the file.



You can also open the Performance Statistics Overlay by pressing **F2** twice and then selecting the Tracy check box next to Collect Stats. (Note that, if you do this while the Choose Role Panel is visible, it might be hidden behind that panel. Simply move the overlay.)

A.3. The Fault Tolerant Heap

The Fault Tolerant Heap (FTH) is a subsystem on Windows that can affect software developers and testers. If a software application crashes many times—for example, in the process of programming and debugging that application—Windows adds it to the FTH. This is significantly slower than the normal heap and can have a negative impact on the performance of the application as a result.

For more information and to learn possible solutions, see https://docs.microsoft.com/en-us/windows/win32/win7appqual/fault-tolerant-heap.

A.4. Graphics Card Binary Compatibility

If your graphics card driver is updated or rolled back, you can see an error:glsl program linking error: Program binary could not be loaded. Binary is not compatible with current driver/hardware combination. Driver build date *Mmm DD YYYY*. Please check build information of source that generated the binary. To resolve this, completely delete the *./appData/cache* directory. The cached shader binaries are invalid with the updated or rolled back graphics card driver.

A.5. Issues with VRAM

You can experience issues with hangs or crashes that are caused by high VRAM. If that is the case, use the `--limitEntityTextures` or `--limitTerrainTextures` command-line options to use only mipmap levels at or below the resolution for entities or terrain. Use two integers for the width and height. For example:

```
  vrEngage.exe --limitEntityTextures "1024 1024" --limitTerrainTextures
"1024 1024"
```

If you continue to experience crashes, use `--limitGpuMemoryPercentage` in addition to these command-line options. Typical values are between 85 and 100, for example: `--limitGpuMemoryPercentage 85`. VR-Engage does not make new textures or buffers as a percentage of the available video RAM beyond the specified percentage. This applies only to VR-Engage-side textures: terrain insets, entities, indirected osgEarth layers (feature_geom and buildings). It does not apply to SilverLining, Triton, or osgEarth tiles. The latter are controlled by the *.earth* file options.

A.6. Remote Camera Placement

It is sometimes difficult to precisely position a remote camera on an entity. The camera may jump to a nearby spot, rather than attaching to the desired location on the entity. If this starts happening, the workaround is to place a different entity (human or other entity) in an arbitrary location on the terrain. Then return to the process of creating a new remote camera on the desired entity.

A.7. Other Limitations

There are some other limitations when using VR-Engage:

- If you task the fire_team unit to perform a Move to Location task, the unit spreads out into a widely spaced formation. Doing this in a constricted area such as the warehouse area in Hawaii leads to unexpected results, as the entities struggle to get into formation.
- SensorFX works with virtual reality devices but has a significant impact on performance.



B. Embarkation Slot Configuration in VR-Engage

VR-Engage supports the entity-level embarkation slot configuration behavior. This appendix describes how embarkation slots are configured for player roles.

B.1. Overview of Embarkation Slots 398
B.2. Door Point-type Slots for Scripted Doors on Walkable Vehicles 398
B.2.1 Ingress vs. Egress Door Points 398
B.2.2 Example Door Point Slot Configuration 398
B.2.3 Door Point Parameters 399
B.3. Embark Point-type Slots 400
B.3.1 Example Embark Point Slot Configuration 400
B.3.2 Embark Point Parameters 401

B.1. Overview of Embarkation Slots

VR-Engage is designed to be compatible with EntityLevel SMS Embarkation Slot configuration and CGF embarkation behavior. See the chapter on Configuring Embarkation in the *VR-Forces User Guide* for instructions on configuring Embarkation Slots. In some cases, no additional Embarkation Slot configuration may be required.

When a VR-Engage player approaches an unoccupied slot position, VR-Engage presents the player with a menu showing the slot name or the name of the role associated with that slot, if there is one. If the player selects an option from the menu, they are embarked in the chosen slot and automatically engaged in the associated role. If the slot has been configured with a 0,0,0 relative position, then the slot is presented in the menu as long as the player is in proximityto the vehicle.

If this behavior is sufficient, no further slot configuration is required. However, VR-Engage vehicles can be configured with two additional special slot types, "DoorPoint" and "EmbarkPoint", for greater embarkation control.

B.2. Door Point-type Slots for Scripted Doors on Walkable Vehicles

For higher fidelity vehicles with articulated doors and walkable interior geometry, you might want to configure a special Door Point slot to act as a menu action point, giving players the ability to open or close a vehicle door. See the chapter *MAK Specification Notes* in the *Adding Content to MAK ONE Applications Reference Manual* for instructions on setting up vehicles with walkable interior geometry and functional doors. A script to toggle the door is also required.

A Door Point is defined by an Embarkation Slot with the special type "DoorPoint" and an Ingress Technique, Egress Technique, or both, which corresponds to the name of the script that toggles a given door. The Door Point slot should be positioned where the player is expected to open the door from, typically at the location of the door itself. When the player approaches this point, a menu option is presented to control the door. When the player selects this option, the associated script is called to open or close the door.

B.2.1 Ingress vs. Egress Door Points

You can define a Door Point for use from inside or outside of the vehicle. If the script name appears in the Ingress Technique field, the Door Point is presented on the menu when the player is outside the vehicle. If you use the Egress Technique field, the Door Point is presented when the user is embarked on the vehicle. Defining the script name in both Ingress and Egress fields in the same Door Point slot makes that Door Point usable for both cases.

B.2.2 Example Door Point Slot Configuration

In the example, the entries in lines 2, 3, and 7 are inherited from the standard slot configuration, but are ignored on a VR-Engage `DoorPoint` type slot.

Note that this example uses both `ingressTechnique` and `egressTechnique`. This makes this Door Point functional for players both inside and outside of the vehicle.

```
 <embarkationSlot embarkableObject="-1:3:1:-1:-1:-1:-1:-1"
capacity="0"
  jumpToLocation="True" turnToHeading="True" orientOnIngress="False"
invisibleWhenEmbarked="False" embarkedAppearance=" 0XA0000"
embarkedAnimation="stand_ready">
  <embarkedOrientation yaw="0" pitch="0" roll="0"/>
  <embarkedPosition x="-4.2" y="0" z="0"/>
  <string paramName="slotName">Rear Door</string>
  <string paramName="slotType">DoorPoint</string>
  <string paramName="stationName">16</string>
  <string paramName="ingressTechnique">Toggle_Cargo_Ramp</string>
  <string paramName="egressTechnique">Toggle_Cargo_Ramp</string>
 </embarkationSlot>
```

B.2.3 Door Point Parameters

Table 41. Parameters for Door Points

Type	Name	Required	Description
Attribute	capacity	Yes	This field should always be set to 0. This prevents this special slot from being presented as a regular embarkation slot in the VR-Forces GUI embarkation dialog.
Attribute	embarkableObject	Yes	This field functions exactly as it does for any VR-Forces slot. The Door Point is only available for entities which match the type filter.
Element	embarkedPosition	Yes	This field defines the position of the Door Point on the vehicle, which controls where the player will be presented with the menu option.
Element Text	egressTechnique	No	Defines the name of the script used to control the door. When egressTechnique field is not blank, this Door Point will be presented to players approaching the door from inside the vehicle. Ingress and egressTechnique may be used together, or separately.
Element Text	ingressTechnique	No	Defines the name of the script used to control the door. When ingressTechnique field is not blank, this Door Point will be presented to players approaching the door from outside of the vehicle.

Table 41. Parameters for Door Points (continued)

Type	Name	Required	Description
Element Text	slotName	Yes	The slotName field on a Door Point slot defines the text which will appear on the menu when the player approaches the Door Point position.
Element Text	slotType	Yes	This field must be "DoorPoint". This is what defines this slot as a special VR-Engage Door Point slot.

B.3. Embark Point-type Slots

The second special slot type that VR-Engage supports is the Embark Point slot. Embark Points are used to give greater control over the position on a vehicle where the menu option to embark in a given slot is presented to the player. For embarkation slots with no associated Embark Point, the position of the embarkation slot itself is used. In many cases, this may be sufficient. A car door, for example, is near enough to the seat associated with it that a special Embark Point is probably not necessary.

On the other hand, the driver's seat in a main battle tank may not be near enough to the perimeter of the vehicle to be accessible in this manner. In that case, you can create an Embark Point and place it outside of the tank at the position where a player might expect to climb onto the tank to gain access to the driver's hatch. Embark Points are associated with slots by slotName, and multiple Embark Points can be associated with the same slot. This may be used, for example, to give a driver access from points on either side of a vehicle.

A position of 0,0,0 works for Embark Points just as it does for all embarkation slots, providing the menu option to embark in the associated slot whenever the player is in proximity to the vehicle.

B.3.1 Example Embark Point Slot Configuration

In the example, the entries in lines 2, 3, 5, 7, and 9 are inherited from the standard slot configuration, but are ignored on a VR-Engage EmbarkPoint type slot.

Unlike a Door Point, the `slotName` of the Embark Point is not used. Instead, the menu for this point shows the name of the associated embarkation slo, or the name of the role associated with that slot.

```
1   <embarkationSlot embarkableObject="-1:3:1:-1:-1:-1:-1:-1"
    capacity="0"
2     jumpToLocation="True" turnToHeading="True"
    orientOnIngress="False" invisibleWhenEmbarked="False"
    embarkedAppearance=" 0XA0000" embarkedAnimation="stand_ready">
3     <embarkedOrientation yaw="0" pitch="0" roll="0"/>
4     <embarkedPosition x="0" y="0" z="0"/>
5     <string paramName="slotName">Gunner</string>
6     <string paramName="slotType">EmbarkPoint</string>
```

```
    <string paramName="stationName">16</string>
    <string paramName="ingressTechnique">Embark:Gunner</string>
    <string paramName="egressTechnique"/>
  </embarkationSlot>
```

B.3.2 Embark Point Parameters

Table 42. Embark Point parameters

Type	Name	Required	Description
Attribute	capacity	Yes	This field should always be set to 0. This prevents this special slot from being presented as a regular embarkation slot in the VR-Forces GUI embarkation dialog.
Attribute	embarkableObject	Yes	This field functions exactly as it does for any VR-Forces slot. That is, the Embark Point is only available for entities which match the type filter. In most cases, this field should be set to match the associated embarkation slot. If the player entity type does not match the filter on both the Embark Point and the associated embarkation slot, the Embark Point will not work.
Element	embarkedPosition	Yes	This field defines the position of the Embark Point on the vehicle, which controls where the player will be presented with the menu option to embark. A value of 0,0,0 means that the menu option will be available anywhere in proximity to the vehicle.
Element Text	ingressTechnique	Yes	The ingressTechnique string associates the Embark Point with an embarkation slot. This string should be in the form of "Embark:*slotName*", where *slotName* matches the slotName parameter of a regular embarkation slot.
Element Text	slotType	Yes	This field must be "EmbarkPoint". This is what defines this slot as a special VR-Engage Embark Point slot.



C. VR-Engage Support for CIGI

VR-Engage has limited support for the Common Image Generator Interface (CIGI). It can act as a CIGI host for another IG application. This chapter explains the basics of configuration. If you need more information, use the Support Portal to request assistance: https://www.mak.com/mak-one/support/portal.

C.1. Introduction to CIGI in VR-Engage 404
C.2. Tutorial for Using VR-Engage as a CIGI Host 404

C.1. Introduction to CIGI in VR-Engage

VR-Engage can act as a CIGI host, allowing another IG application to display a player-controlled entity. This was implemented for integrators who are required to use a different IG.

CIGI is not enabled by default. It requires a host configuration to send the observer view to the IG application and a display layout configuration for the role to display that view. You define these configurations in Lua files. Players then select the CIGI display layout when they engage in that role.

VR-Engage includes example CIGI display configurations for the ground vehicle driver and gunner roles. These are used in the the following section, which walks you through a simple tutorial using VR-Vantage as an IG for VR-Engage.

> Only CIGI 4.0 is supported at this time. For more information on CIGI and MAK ONE, see *Section IX. VR-Vantage Support for CIGI* in the *VR-Vantage User Guide*.

C.2. Tutorial for Using VR-Engage as a CIGI Host

This tutorial shows you how to use VR-Vantage as an IG. In this example, you run both VR-Engage and VR-Vantage on the same machine. Ensure that they are both installed before beginning. (A real-world use case would be to connect to an IG from another software provider; this is simply an example.)

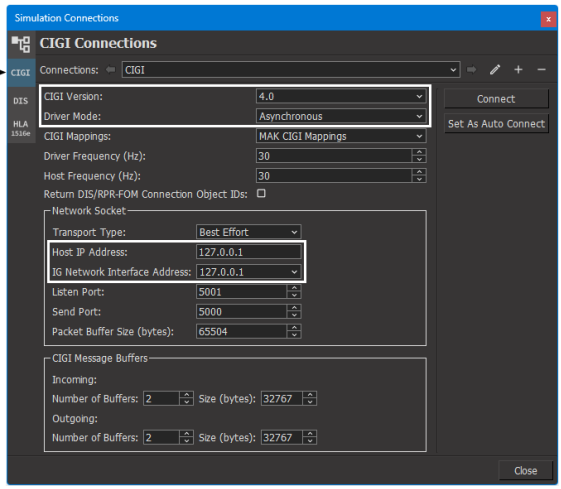
➤ **To try out using VR-Engage as a CIGI host for VR-Vantage:**

1. Navigate to *./data/simulationModelSets/VR-Engage/roles*.
2. Using a text editor, edit *driver.lua*.
3. In the `includes` section near the end of the file, uncomment the line for the CIGI display layout file. It should now read:

   ```
   includes = {
   "@(vre-roles-dir)/devices/radio.vehicle.lua";
   "@(vre-roles-dir)/displayLayouts/driver.cigi.lua";
   };
   ```

   The display layout file uses the *./appData/settings/vrfSim/cigiHost/toVantageIg.lua* file to set up the CIGI network, entity and item mappings, and terrain IDs.
4. Launch VR-Vantage.
5. In the Load Terrain dialog box, select Ala Moana.mtf and then click OK.
6. Choose **Settings > Connection**. The Simulation Connections dialog box opens.
7. Select the CIGI Connections page.
8. For the CIGI Version, select 4.0.
9. For the Driver Mode, select Asynchronous.
10. Set both the Host IP Address and IG Network Interface Address to 127.0.0.1.

Figure 124. CIGI simulation connection in VR-Vantage



11. Click Connect.
12. Close the Simulation Connections dialog box.
13. Launch VR-Engage as described in "3.1.1 Hosting a VR-Forces Session" on page 57.
14. Load a scenario. For this tutorial, we use the VR-Engage Test Drive scenario.
15. In the Choose Role panel, select LAV 1.
16. From the Role list, select Driver.
17. From the Display(s) list, select CIGI To Vantage IG.

Figure 125. Driver role in Test Drive Scenario, configured for CIGI display



18. Click Engage.

    VR-Engage becomes minimized. You can use keyboard controls to operate the vehicle or gun. Its actions are reflected in VR-Vantage. (The application must have focus. Do not select another application.)

19. When you are finished, press **Esc** to open the VR-Engage Application Menu. You can then exit VR-Engage.
20. Close VR-Vantage.

If you have any questions about using CIGI to connect VR-Engage to an IG, please contact MAK or your distributor.



MAK Products Glossary

This glossary defines terms used in MAK ONE product documentation.

A

absolute dead reckoning

A method of dead-reckoning in which an application uses the timestamps contained within state updates if the sender is using absolute timestamping. Contrast with relative dead reckoning.

aggregate

The combination of individual entities, aggregates, or both into a single object — for example, an organizational group such as a platoon.

API

Application Programming Interface.

articulated part

A part of an entity that is capable of movement relative to the entity, such as a wheel, turret, or gun.

attached part

An object that is attached to another object, such as a rocket attached to a jet.

B

body coordinate frame

A coordinate framework centered on an entity. To represent the orientation of an entity, an application uses Euler angles or a rotation matrix that rotates from a reference frame to the body coordinate frame.

C

channel

A channel renders the view of an observer in a 3D visualization application.

clipping

A feature that prevents the display of terrain and objects or parts of objects, that are closer than or farther than specific distances from the observer.

clipping planes

The range of distances from the observer (near clipping and far clipping) in which an application clips (does not display) objects.

culling

The process of discarding database objects which are not within view, and therefore need not be rendered and displayed.

Cartesian coordinates

A system for indicating location by means of three planes intersecting at right angles to one another at the origin.

D

Data Distribution Management

The set of High-Level Architecture services used to specify the routing of data between publishers and subscribers. DDM adds greater control over the Declaration Management (DM) services that only specify filtering based on the class of the data (e.g., ground vehicle versus aircraft). DDM uses the data content to express regions of interest and regions of affect whose overlap establish a communication channel. An example is geographic filtering where a publisher indicates a region around the entity or effect and a subscriber indicates a region expressing its sensor range.

dead-reckoning

A process by which an application calculates the expected location of an entity during periods between state updates, based on velocity, acceleration, and rotational velocity. See also absolute dead reckoning and relative dead reckoning.

DDM

See Data Distribution Management.

Defense Modeling and Simulation Office

The former name of the executive secretariat for the Executive Council on Modeling and Simulation (EXCIMS), which provided a full-time focal point for information concerning Department of Defense modeling and simulation (M&S) activities. Renamed Modeling and Simulation Coordination Office. Currently the MSCO promulgates M&S policy, initiatives, and guidance to promote cooperation among DoD components to maximize efficiency and effectiveness.

DIS

See Distributed Interactive Simulation.

DIS data packets

See Protocol Data Unit.

display engine

An object in an application that can render 3D data.

Distributed Interactive Simulation

The DIS protocol is a set of standards that govern how participating applications share information about the virtual world. The protocol specifies a set of packets, called protocol data units (PDUs), that communicate this information. Each PDU identifies the sender and contains other information depending on the PDU type. The DIS protocol also specifies when and how frequently PDUs are sent.

The DIS protocol has been superseded by the High-Level Architecture for use by the Department of Defense.

DMSO

See Defense Modeling and Simulation Office.

E

element definition

In VR-Forces and VR-Vantage, an element definition specifies all of the visualizers that control the various aspects of visualizing a scene element. For example, an element definition includes the main model or military symbology, trailing effects, cockpit displays, and so on.

emitter

A simulated device on an entity that emits electromagnetic radiation, for example, radar.

entity

An element in a simulation, such as a vehicle or a person, that is represented in the simulation through the issuance of state messages.

entity maintenance

A process by which the MAK Data Logger compensates for discontinuities following a time jump. The Logger sends out interim state messages for entities that were present before the time jump so that they do not time out before the next update message arrives.

entity-type

A list of seven components as specified by the DIS protocol and the HLA RPR FOM. If none of the seven components contains a wildcard (-1) value, then the entity type refers to a specific, narrowly-defined entity. If some components contain a wildcard, then the entity type refers to a class of entities. A larger number of wildcards indicates a broader, more general class.

The components of an entity-type are:

- Entity kind
- Domain
- Country
- Category
- Subcategory
- Specific
- Extra.

environmental process

According to the IEEE 1278.1a specification (DIS), environmental process PDUs communicate simple environment variables, small scale environmental updates, and embedded processes.

Euler angles

A set of three angles used to describe the orientation of an entity as a set of three successive rotations about three different orthogonal axes (x, y, and z). These angles specify successive rotations needed to transform from a reference coordinate system to the entity's body coordinate system.

event

An interaction between objects or between an object and the terrain, such as firing of a munition, or a collision of entities.

exercise

The DIS term for a one or more interacting simulation applications. Compare to federation execution in HLA.

exercise connection

The object (DtExerciseConn) through which a VR-Link application connects to the network. State messages are sent through the exercise connection and information about remote entities is received through the exercise connection. (All MAK ONE products are VR-Link applications.)

exercise ID

A numeric identification for a DIS simulation exercise.

F

FED file

Federation Execution Data file. The Federation Execution Data is the subset of FOM data needed and read by the MAK Products Glossary. VR-Link applications also read the FED file.

federate

A connection to the MAK Products Glossary. Typically a single simulation application can be thought of as a federate.

federate protocol

A message-based RTI API utilizing Protocol Buffers encoding that enables federates to invoke RTI services using a simple TCP connection.

federation

A group of HLA federates capable of playing in the same federation execution.

Federation Object Model

Defines the data content of a federation execution.

federation execution

The federation execution represents the actual operation, over time, of a subset of the federates and the RTI initialization data taken from a particular federation. A federation execution is the logical equivalent of a DIS simulation exercise.

field of view

Controls the perspective of the observer.

A wide field of view creates an effect like that of a wide-angle camera lens. Objects appear smaller and farther away from the observer, since the observer coverage spans a wider area. Depth become exaggerated.

A narrow field of view creates an effect like that of a telephoto lens. Objects appear larger and closer to the observer, and the overall scene depth appears flattened. The distances between objects appears compressed.

filter range

A setting that prevents distant entities from being processed while allowing distant terrain to appear normally.

FOM

See Federation Object Model.

FOM Module

In HLA Evolved and HLA 4, defines additional modular data content for a federation execution, usually extensions to an existing Federation Object Model. (Also supported by the HLA 1.3 and HLA 1516 versions of the MAK RTI. Per the HLA Evolved interface specification:

"A partial FOM (containing some or all OMT tables) that specifies a modular component of a FOM. A FOM module may contain classes not inherent to it but upon which the FOM module depends, i.e., superclasses to the modular components. These superclasses will be included in the FOM module either as complete or scaffolding definitions."

frame rate

The rate at which the application displays updated images.

G

ground clamping

A process by which the a 3D visualization application keeps an entity anchored to the surface of its terrain database, regardless of the altitude data contained in its entity state message.

geocentric coordinates

A coordinate system calculated with respect to the earth's center. The origin of the geocentric coordinate system is the center of the earth. The positive X-axis passes through the prime meridian at the equator; the positive Y-axis passes through 90 degrees east longitude at the equator; and the positive Z-axis passes though the north pole.

geodetic coordinates

A coordinate system in which position is determined relative to a reference ellipsoid, such as the surface of the earth at sea level. In MAK ONE applications, geodetic coordinates consist of latitude and longitude in radians, and altitude in meters above the reference ellipsoid.

gridded data

Data that has been processed in a rectangular array of points, in X, Y or latitude/longitude, at which single data values define a two dimensional function. According to the IEEE 1278.1a specification, gridded data transmits information about large-scale or high-fidelity spatially and temporally varying ambient fields and about environmental processes and features.

guise

An alternative entity type used to display an object depending on the force ID. For example, a tank could look like an M1A1 to friendly forces and a T72 to hostile forces.

H

head-up display

A set of indicators and readouts superimposed onto a graphics display. Also called an overlay.

heartbeat

In DIS, the frequency with which current Protocol Data Units are sent to the network regardless of whether the entity's state has changed.

High-Level Architecture

The High Level Architecture (HLA) for simulations is a U.S. Department of Defense (DOD)-wide initiative to provide an architecture to support interoperability and reuse of simulations. The HLA supersedes DIS for the DOD.

HLA

See High-Level Architecture.

I

IG

image generator

interaction

A message describing a simulation event. An interaction describes an event, it does not update an objects state.

L

Logger control PDUs

A set of DIS PDUs or HLA interactions that let you control the MAK Data Logger remotely.

logical range

A suite of TENA Resources, sharing a common object model, that work together for a given range event. Similar in concept to the HLA Federation.

Local RTI Component

The libraries on a computer that implement a Run-Time Infrastructure. A federate communicates with the HLA network through the LRC.

LRC

See Local RTI Component.

M

MÄK Technologies Lisp

An adaptation of the Lisp language used in configuration files for MAK ONE products.

message

A general term used to refer to DIS PDUs and HLA interactions and state updates. In TENA, a message is similar to an HLA interaction.

MIM

The MOM & Initialization Module (MIM) allows for extensions to be made to the High-Level Architecture standard MOM. It is a subset of the Federation Object Model that contains OMT tables that describe the HLA MOM. The MIM also contains additional predefined HLA constructs such as object and interaction roots, data types, transportation types and dimensions. HLA specifies a standard MIM that is incorporated into all FDDs automatically by the Run-Time Infrastructure. The standard MIM can be replaced with a user-supplied MIM containing the standard MIM plus extensions.

model definition

In VR-Forces and VR-Vantage, a model definition specifies the 3D model and other basic attributes of a model. It is referenced by an element definition.

Modeling and Simulation Coordination Office

The executive secretariat for the Executive Council on Modeling and Simulation (EXCIMS), which provided a full-time focal point for information concerning Department of Defense modeling and simulation (M&S) activities. The MSCO promulgates M&S policy, initiatives, and guidance to promote cooperation among DoD components to maximize efficiency and effectiveness. (Formerly DMSO)

MSCO

Modeling and Simulation Coordination Office. (Formerly DMSO.)

MTL

See MÄK Technologies Lisp.

N

NETN FOM

The NETN FOM (NATO Education and Training Network Distributed Simulation Federation Object Model) is an extension of the RPR FOM 2.0, allowing other federates using pure RPR FOM to interoperate with applications using the NETN FOM.

O

orientation clamping

Adjusts the pitch and roll of an entity so that it appears properly seated when the terrain is inclined. For example, if a tank is moving horizontally across the face of a hill, orientation clamping prevents the tank from appearing level, and therefore, partially embedded into the hillside. Used with ground clamping.

object

An element in a simulation that has persistence, as opposed to an interaction, which is a transient element.

object handle

An integer that an application uses to identify a particular object in Run-Time Infrastructure service calls. An object handle is meaningful only to a particular federate. The same object can be known to different federates by different object handles.

object name

A character string that can be used to identify an object. In HLA, the object name is known to the Run-Time Infrastructure, and the RTI provides functions to find out an object's name, given its handle, and vice versa. Object names can be chosen by applications that register the objects with the RTI, however if you do not want to choose names for objects, the RTI will assign names for you.

observer

In MAK ONE 3D applications, the point of view into the scene. (Called the eyepoint in MAK Stealth 6.x and earlier.)

P

PDU

See Protocol Data Unit.

Protocol Data Unit

A unit of data message (packet) that is passed on a network between DIS simulation applications.

R

radio transmitter

A simulated device on an entity, capable of transmitting radio communications.

radio receiver

A simulated device on an entity, capable of receiving radio communications.

Real-time Platform Reference FOM

An HLA reference Federation Object Model based on the DIS protocol.

recording

A MAK Data Logger file that stores a history of the interactions in a simulation for playback.

reflected entity list

A list of entities simulated by remote applications (federates in HLA) and about which the local simulation has received information over the network.

reference FOM

A Federation Object Model designed to be used as a whole, or with modification, by a wide variety of similar federation executions.

relative dead reckoning

A method of dead reckoning in which an application approximates the location of an object based on the local time of receipt of the last state update message. Contrast with absolute dead reckoning.

remote display engine

A display engine running in an application that is separate from the master application, and which is controlled by the master application.

RID file

See RTI Initialization Data.

RTI Initialization Data

Data required by a Run-Time Infrastructure during initialization, independent of the FOM being used. RID data is usually dependent on a specific implementation of the RTI.

RPR FOM

See Real-time Platform Reference FOM.

RTI

See Run-Time Infrastructure.

Run-Time Infrastructure

A library and other supporting software that implements the HLA interface specification. All federates communicate with one another in an HLA environment through RTI functions.

S

scene element

In VR-Vantage and VR-Forces, any object that is rendered in the scene, such as an entity model, interaction, or tactical graphic.

Simulation Interoperability Standards Organization

A group that seeks to promote modeling and simulation interoperability and reuse for the benefit of diverse M&S communities, including developers, procurers, and users, world-wide.

simulation time

In VR-Forces, simulation time is used in dead-reckoning of remote entities and thresholding of local entities. Typically, simulation time is set once during each iteration of the application's main simulation loop so that all entities are dead-reckoned based on the same value of current time.

SISO

See Simulation Interoperability Standards Organization.

smooth period

The period of time over which trajectory smoothing takes place.

smoothing

A method of ensuring that transitions from an entity's dead-reckoned position to its actual position are not so abrupt as to be visually disconcerting.

state

The current status of an object, including location, direction of movement, extent of damage, and so on.

T

tape

An alternative term for a Logger recording. Not a physical magnetic tape.

terrain following

In a 3D visualization application, causes the observer (eyepoint) to maintain a constant distance above the terrain surface. The observer's height changes in tandem with the peaks and valleys as it passes over the geography.

timeout

The period of time in which an application continues to display an entity after that entity's update messages have stopped appearing on the network. Time-outs are usually not used in HLA because there is no set frequency (no heartbeat) for transmitting messages.

timescale

A factor by which time is accelerated or slowed during playback of a Data Logger recording.

topographic coordinates

A right-handed Cartesian coordinate system whose X-Y plane is tangent to the earth's surface at the origin, with the positive X axis pointing north, the positive Y axis pointing east, and the positive Z axis pointing down. There are an infinite number of topographic coordinate systems – one for each point on the earth's surface.

topographic coordinate frame

A coordinate frame in the context of the terrain.

trajectory smoothing

A method used by to smooth positional discontinuities that could occur when new state updates arrive.

U

UDP port

A network channel through which an application sends and receives data for DIS exercises.

Universal Transverse Mercator

In general, a non-cartesian coordinate system in which the X, Y, and Z correspond to easting (nearly east), northing (nearly north), and height above an ellipsoid that approximates the surface of the earth.

UTM

See Universal Transverse Mercator.

V

view control messages

A set of programmatic messages that let you control the view in VR-Vantage applications remotely.



Index

#

3D cockpit, 202

A

AAR, 34
absolute timestamps, 76-77
accuracy, reticle or sight, 163
action
  list, 281
Action menu, 134
  game controller mapping, 248
  mouse and keyboard mappings, 242
  opening, 134
  selecting options, 134
add-on, VR-Engage for VR-Forces, 39
address
  host, 106
  target, 76
advisory, ignoring, 100, 111
Afghan Police Officer, 292
Afghan Soldier, 292
African Insurgent, 293
after-action review, 34
Agent M9, 293
AH-64D Apache Longbow, 328
AH-64E Apache Guardian with FCR, 328
aiming
  grenade, 164
  rifle and handgun, 163
  turret, 189
  weapon, 163
Air mode, running, 87
aircraft, 198
  controlling, 203
  countermeasures, 207
  game controller mapping, 266-267
  gunner, 198
  HMD, 203
  keyboard mappings, 247
  roles, 198
air-to-air
  combat, 210
  target
    firing at, 213
    radar, 211
    sensor view, 214
air-to-ground
  combat, 209
  target
    laser, 209
    MGRS coordinate, 209
alpha to coverage, 122
  VRV_ALPHA_TO_COVERAGE_RENDERBIN environment variable, 123
altitude
  setting for autopilot, 204
ambient occlusion
  enabling, 124
  settings, 125
Ambulance, 313
Amphibious Combat Vehicle (ACV), 313
AN PED-1 LLDR, 312
AN/PED-1 LLDR
  lasing, 171
animation
  DI-Guy, 105

anti-aliasing, 87, 122
appearance
  DI-Guy, 105
application
  data, specifying location of, 87-88, 103
  ID
    DIS, 76
    HLA, 75
  number, 100-101, 103
    range, 103
  priority, 93
application ID, 101
  range, 101, 111
Application Number parameter, 75-76
application, MAK ONE
  build information, 38
application, restrictions, 88
AR, See augmented reality
ASLAV-25, 312
assigning task, 176
asynchronous
  I/O
    enable, 102, 109
  IO
    DIS, 77
Asynchronous IO parameter, 77
attach parameter, 85
audio, 148
  global settings, 148
  MAK radio settings, 141
augmented reality, 78, 82
  configuring, 81
  system requirements, 78
Australian Soldier EF88, 293
Australian Soldier XM107, 294
auto connect period, 88
auto reconnect period, 88
autopilot mode, 204
  setting, 204

B

back-end
  multiple, 86
batch file
  command-line option, 104
batch mode
  command line, 104
battery, drone, 221
bearing
  measurement unit, 206
Beechcraft MC-12 King Air, 324
bin, render, 123
binoculars, 178
bit depth, 89
  and stencil, 122
BMP-3 AFV, 313
brake
  ground vehicle, slowing, 187
  parking, 188
Brazilian Soldier, 294
buffer
  depth, 89
  size, netsocket, 76
buffer size, 102, 107
  netsocket, 76
build information, 38
bullseye
  north reference, 206
  selecting, 206
Bushmaster PMV, 314

C

cache
  specifying directory, 87
callstack.log file, 40
camera
  backup, ground vehicle, 187
  drone view, 219
  remote, 228, 231
  Vive, testing, 83
capturing images, 240
car, See also ground vehicle
car, game controller mapping, 252
carry person, 174
CDB
  performance, 89
CH-47 Chinook, 329
CH-149 Cormorant, 329
chaff, 207
channel
  output, enable, 105
  text chat, 145
character
  DI-Guy, 105
chat window, 142
  channels, configuring, 145
  commands, 143
  VR-Forces, 144
Chilean M4, 295
Chilean M16, 294
Chilean M16-M203, 295
Choose Role Panel, 70
choosing
  role, 63

weapon
aircraft, 208
gunner, ground vehicle, 189
soldier, 162
CIGI
host, 404
tutorial, 404
Civilian Female With Drone, 295
Civilian Male With Drone, 296
clock mode, 104
cloud
synchronization, 93
CLU, 168
cockpit, 3D, 202
Colombian Police Officer, 296
Colombian Soldier, 296
combat
air-to-air, 210
air-to-ground, 209
ground vehicle, 188
soldier, 162
command launch unit, 168
command line
loading scene, 99
loading terrain, 99
options, 87
commander
ground vehicle, 192
position, 192
view, 192
command-line option
--, 99, 110
-a, 100-101, 103
-A, 101, 105
--additionalSystemScriptsDirectory, 103
--airMode, 87
--antiAliasing, 87, 122
--appCacheDir, 87
--appDataDir, 87, 103
--appIdRange, 101, 111
--appIdRangeMax, 103
--appMode, 88
--appNumber, 100-101, 103
--attributePduStrategy, 104
--autoConnect, 88
--autoReconnectPeriod, 88
-B, 88, 104
--bindObservers, 88
-C, 88
-c, 98
--clockMode, 104
--connectionless, 104
--countryCodesMappingFile, 104
-d, 107
--dataDir, 89, 105
--defaultObjectTimeout, 101, 111
--depthBuffer, 89
--destAddrString, 101, 105
--deviceAddress, 101, 105
--diGuyAnimationsFile, 105
--diGuyCharacterDataFile, 105
--dis, 89
--dis5Timestamps, 105
--disableDynamicTerrain, 89
--disableErrorDialog, 89
--disableEscapeKey, 89
--disableGlBufferTracking, 89
--disableIndirectEntities, 89
--disableIndirectExternalRenderer, 90
--disableKDTrees, 90
--disableModelSets, 90
--disableParallelPublisherTick, 105
--disableParallelTick, 105
--disableRemoteDiscovery, 105
--disableReverseZBuffer, 90
--disableTerrainObserverViews, 90
--discoveredModelPreloadFileName, 90
--displayEnginePort, 90
--disPort, 101, 112
--disVersion, 101, 112
--doNotLoadVrfPlugins, 105
--doNotUseConsole, 105, 108
-E, 91
-e, 92
--enableChannel, 105
--enableDofConstraints, 90
--enableMultipleBackends, 86
--enablePduProfiling, 105
--enablePersistentShadowTechnique, 91
--enableSensorFX, 91
--enableTracyProfiling, 91
--enableVSync, 91
--entDispSetting, 91
--entityDefsDir, 91
--entityGridCellSize, 91
--entTypeMap, 91
--envSetting, 91
--exConnConfigFile, 92, 106
--execname, 100, 110
--exerciseId, 101, 112
--exitOnTestResultCommand, 106
--extraArgs, 92
-F, 92
-f, 100, 110
--factoryRootDir, 92
--federateName, 100, 110

--federateType, 100, 110
--fedFileName, 100, 110
--fileNotifyLevel, 106
--fileTransporterReceivePort (DIS), 100-101
--fomMapperInitData, 100, 110
--fomMapperLib, 100, 110
--fomModules, 100, 111
--forceTextShaping, 92
--frontEndPID, 106
--fullScreen, 92
--fullyLoadTerrain, 106
--groundMode, 92
--guiWindowRectangle, 92
-h, 100, 110
-H, 100-101
--help, 100, 110
--highestPriority, 93
--hla1516e, 93
--hostAddressString, 100-101, 106
-i, 101-102, 107
--ignore_rest, 99, 110
--ignoreAdvisories, 100, 111
--ignoreZoomForSpeedtreeLod, 93
--initSettingsFile, 85, 89
-K, 90
-L, 95, 107
--limitEarthSplattingTextures, 93
--limitEarthTerrainTextures, 93
--limitEarthVegetationTextures, 94
--limitEntityTextures, 94, 118
--limitEntityTexturesExclusions, 94
--limitGpuMemoryPercentage, 94
--limitTerrainTextures, 94, 118
--loadObserverViews, 95
--locale, 53, 95
--localSettingsDesignator, 100, 111
--logFileName, 95, 106
-m, 97
--mcastTtl, 102, 107
--mergeEntityTypeMap, 95
--mergeHierarchyEntityDef, 95
--mergeHierarchyTacticalGraphicsDef, 95
--mergeHierarchyUnitDef, 96
--mergeLeafEntityDef, 96
--mergeLeafTacticalGraphicsDef, 96
--mergeLeafUnitDef, 96
--mergeModelDefs, 96
--mergeTacticalGraphicsTypeMap, 96
--mergeUnitTypeMap, 96
--mimModule, 111
--modelPreloadFileName, 96
--msdlSIDCMappingFile, 107
--multicastAddresses, 102, 112
-n, 97, 107
-N, 100, 110
--netnFomRevision, 111
--netnFomVersion, 111
--noAppDataEntityDefs, 97
--noAppDataEntityTypeMap, 97
--noAppDataModelDefs, 97
--noAppDataTacticalGraphicsDefs, 97
--noAppDataTacticalGraphicsTypeMap, 97
--noAppDataUnitDefs, 97
--noAppDataUnitTypeMap, 97
--noGui, 97
--nonVrfObjectsUUIDScheme, 107
--noRtiCompilerCheck, 111
--notifyLevel, 97, 107
--numCallbackThreads, 107
-O, 93
--outputInterval, 107
--outputLogFile, 107
--overwriteMappingsOnMerge, 96-97
-p, 100, 110
-P, 101, 112
player station, 112
--plugin, 97
-q, 105
-r, 108
--recvBufferSize, 102, 107
--reportedHostAddressString (DIS), 101-102
--rprFomRevision, 101, 111
--rprFomVersion, 101, 111
-s, 98, 101-102, 108
-S, 98, 100, 111
--scenarioFileName, 107
--sendBufferSize, 102, 107
--sensorMode, 98
-sessionId, 101-102, 107
--sessionId, 107
--setFomModuleList, 111
--setMainThreadToHighPriority, 108
--setting, 98, 367
--settingsFile, 107
--sharedDataDir, 98
--sharedDataRoot, 98
--sharedDataStem, 98
--sharedSettingsDirectory, 108
--showConsole, 98
--simulationModelSet, 108
--siteId, 101-102, 108
--slave, 98
--startMinimized, 108
--stationName, 98
--subnetMask, 102, 109
--suppressDataAvailableException, 109

--suppressHatIntersectOnAttach, 98
--suppressSelfReflect, 112
-T, 109
--tacticalGraphicsDefsDir, 98
--targetFrameRate, 109
--tcpSendTimeout, 98
--testOutputFilename, 109
--timeManagement, 111
--tracyPort, 109
--unitDefsDir, 99
--unplug, 99
--useAbsoluteTimeStamps, 101-102, 109
--useAsyncIO, 102, 109
--useGeographicDdm, 101
--useIpv6, 102, 109
--userDataDir, 99, 109
--useUserSettingsDirectory, 110
-v, 99, 110
--version, 99, 110
--waitQueue, 110
--windowedMode, 99
--withDisplayEngine, 99
-x, 100-101, 110, 112
-z, 90
-Z, 93
common operating picture, See COP
compass
display units, 159
enabling, 159
using, 158
component mapping, 280
configuration
entity display, 91
environment, 91
MAK radio, 138
MAK RTI file, 49
plug-in
adding, 115
deleting, 116
world cell size, 91
configuring
joystick, 272
keyboard entity control, 272
connection
DIS, device address, 76
to network, automatic, 88
console mode, 97
controller
flight, 255
head-mounted display, 268
input types, 276
Logitech gamepad, 271-272
mapping, 275
supported, 50
T.Flight Hotas 4, 260, 263
utility, 273
Xbox, 270
controlling
aircraft, 203
drone, civilian, 221
ground vehicle, 184
coordinateConverterOrigin parameter, 375
COP, 214
focus area, 215
information, 214
selection, 215
copilot role, 198
countermeasures, aircraft, 207
crash, application, 41, 394-395
creating
entity, 65
simulation model set, 337
VR-Engage scenario, 372
criss-cross applesauce, 168
crouching, 157
cryptography, radio, 135
custom overlays, 380
customization, 113
cycle weapon
aicraft, 208
gunner, 189
soldier, 162

D

damage
human, 160
indicator, 161
sustaining, 161
weapon accuracy, 163
data
directory, 46, 87, 103
sim. engine, 105
specifying location of, 89, 105
install, 38
options, 44
shared
directory, 98
root, 98
stem, 98
user directory, 99, 109
debug
VR-Engage overlay, 130, 358
debug overlay, 128
debugging
callstack.log file, 40

dmp file, 40
file location, 42
VR-Engage, 130
decibel, configuration, 150
definition
entity, directory, 91
deployment
drone, 222
weapon, 165
depth
bits, 89
buffer, 122
designate target
commander, 193
sensor operator, 231
soldier, 179
destination address, 101, 105
DIS, 76
Destination Address parameter, 76
detonation
event, sound effect, 148, 151
developer debug overlay, 128
developer documentation, 40
Developer's Guide, installing, 40
device address, 76, 101, 105
DI AK-47, 297
DI Lasing (CIS), 297
DI Lasing (US), 297
diagnostic level, 97, 107
differential, locking and unlocking, 188
DI-Guy
character data file, 105
configuring
animations and appearances, 105
instancing, 127
performance, 127
directional pad, See D-pad
directory
data, 46
installed, 45
sim. engine, 105
Windows default install, 39
DIS
application ID, 76
asynchronous IO, 77
connection settings, 76
destination address, 76
exercise ID, 76, 101, 112
multicast TTL, 76
port, 76, 101, 112
receive buffer size, 76
self reflection, 112
send buffer size, 76
starting in, 89
timeout, 111
timeout interval, 77
version, 101, 112
DIS Version parameter, 76
Disembark Entity task, 176
disembarking
aircraft, 175
vehicle, 174
display
aircraft, 202
ground vehicle, 186
display engine
connecting to
command line, 99
port, 90
timeout, 98
Display menu, 59
Display Settings, 59
DJI M200 Quadcopter, 329, See also drone
DJI Mavic 3 Small Munition, 225, 329, See also drone
DLL, 112
plug-in
adding, 114
removing, 115
specifying for plug-in, 114
dmp file
debugging, 40
location, 42
documentation
help, 18
installation, Windows, 40
documentation, location of, 46
door
building, opening and closing, 181
Stryker, opening and closing, 173
D-pad
Action menu controls, 134
aircraft controls, 266-267
drone controls, 250
ground vehicle controls, 251-252
human controls, 249
sensor operator controls, 267
drag person, 174
driver
ground vehicle, 186
looking around, 187
role, 189
drone
battery, 221
camera view, 219

control
LSV quadcopter, 224
Spark, 223
deploy, 222
game controller mapping, 250
image capture, 240
infrared, 224-225
landing, 223
LSV
pilot control, 224
video stream, 238
munition, 225
night vision, 224-225
operator
civilian, 218
light strike vehicle, 218
LSV, 218
munition, 219
standalone, 219
pictures, taking, 240
recovery, 223
restarting use, 222
returning home, 223
Spark
control, human or drone, 223
video stream, 238
starting, 222
stopping use, 222
using, 221
video, 238
visual sensor, 224-225
drone operator, role, 218
dump file, See dmp file
dynamic linked library, 112
dynamic terrain, 89
effect on players, 147

E

earth file
limit splatting textures, 93
limit terrain textures, 93
limit vegetation textures, 94
embarkation, 398
point-type slots, 400
point-type slots, door, 398
role-specific, 342
embarking
Stryker, 173
vehicle, 70, 172
emergency light, 195
Enable Timeout Processing parameter, 77
enableMultipleBackends parameter, 86
enabling
autopilot mode, 204
HLA advisories, 75
laser designator, 181
mouse control, 156
sound effects, 148-149
time management
in sim. engine, 111
ending a scenario, 72
engaging LAV gear, 186
entity
adding to scenario, 65
controllable
adding, 337
Afghan Police Officer, 292
Afghan Soldier, 292
African Insurgent, 293
Agent M9, 293
AH-64D Apache Longbow, 328
AH-64E Apache Guardian with FCR, 328
Ambulance, 313
Amphibious Combat Vehicle (ACV), 313
AN PED-1 LLDR, 312
AN/PED-1 LLDR, 171
ASLAV-25, 312
Australian Soldier EF88, 293
Australian Soldier XM107, 294
Beechcraft MC-12 King Air, 324
BMP-3 AFV, 313
Brazilian Soldier, 294
Bushmaster PMV, 314
CH-47 Chinook, 329
CH-149 Cormorant, 329
Chilean M4, 295
Chilean M16, 294
Chilean M16-M203, 295
Civilian Female With Drone, 295
Civilian Male With Drone, 296
Colombian Police Officer, 296
Colombian Soldier, 296
DI AK-47, 297
DI Lasing (CIS), 297
DI Lasing (US), 297
DJI M200 Quadcopter, 329
DJI Mavic 3 Small Munition, 329
editing, 338
Eurofighter Typhoon, 325
F A-18 Hornet, 325
F-16A Fighting Falcon, 325
F-35 Lightning II, 326
F-35A Lightning II, 325
F-35C Lightning II, 326
Farmer M9, 298

Fiat 124, 314
Flare Rapider 4 Quadcopter, 330
FV 510 Warrior, 314
FV4034 Challenger 2 MBT, 314
GTK Boxer APC, 315
GTK Boxer CRV, 315
HMMWV Utility Vehicle, 315
Insurgent, 298
Insurgent M9, 299
Iranian Soldier, 299
Iraqi Police Officer, 299
Iraqi Soldier, 300
ISIS Fighter, 298
Israeli Soldier, 300
Israeli Soldier M4, 300
Kurdish Fighter, 301
LAV 6 ISC, 316
LAV III APC, 316
Leopard 2A4, 316
Libyan Police Officer, 301
Libyan Soldier, 301
Light Strike Vehicle MK II (LSV), 317
M1A1 AIM (AUS), 320
M1A1 AIM (AUS) Mineplow, 320
M1A1 SA, 320
M1A2 Abrams MBT, 320
M1A2 SEP Abrams, 321
M2A3 Bradley IFV, 321
M113 APC, 318
M270 MLRS, 321
M915 Tractor Truck, 321
M977 HEMTT Cargo Truck, 322
M983 HEMTT Patriot Tractor, 322
M1083 MTV Cargo Truck, 317
M1120A4 HEMTT LHS, 317
M1126 Stryker ICV with M2, 318
M1151 Expanded Capacity Armament Carrier (ECAC), 318
M1151 HMMWV ECAC with M2, 318
M1151 HMMWV ECAC with M240, 319
M1151 HMMWV ECAC with M240B, 319
M1151 HMMWV ECAC with MK 19, 319
M1278 L-ATV-HGC (JLTV), 319
MH-60 Black Hawk, 330
Mideast Combatant, 302
MiG-29 Fulcrum, 326
MQ-8B Fire Scout, 330
MQ-9 Reaper UAV, 326
MRAP M-ATV, 322
MSVS SMP Gun Tractor, 322
Pakistani Police Officer, 302
Pakistani Soldier, 302
Paratrooper, 303
Paratrooper M9, 303
Parrot Anafi Quadcopter, 330
Police Cruiser, 323
Police Officer With Drone, 303
Policeman M9, 304
Quadcopter, 331
Remote Camera, 331
RQ-4 Global Hawk UAV, 327
RQ-7 Shadow UAV, 327
RQ-11 Raven UAV, 327
Russian Rebels, 304
ScanEagle UAV, 327
Security Camera, 331
Security Guard M9, 304
Singaporean Soldier, 305
Somali Police Officer, 305
Somali Soldier, 305
Sport Hatchback Coupe, 323
Sport StandardSize Sedan, 323
Standard MidSize Sedan, 323
Sudanese Police Officer, 306
Sudanese Soldier, 306
Suspect, 306
Suspect M9, 307
Syrian Police Officer, 307
Syrian Soldier, 307
T-38C Talon, 328
Taliban, 308
Technical Truck, 324
Ugandan Police Officer, 311
Ugandan Soldier, 312
UH-60M Blackhawk, 331
Ural 4320, 324
US Army Javelin, 308
US Army Javelin (Tripod), 308
US Army M4, 309
US Army M9, 310
US Army M16, 309
US Army M16-M203, 309
US Border Patrol M9, 310
USMC M4, 311
USMC M16, 310
USMC M16-M203, 311
creating remote camera, 231
culling, 91
custom, 362-363, 365-366
definition directory, 91
display settings, 91
element definition
prevent load, 97
exclude textures, 94
existing, 64
extending, 339

indirect rendering
disable, 89
leaf file
merge, 96
limit textures, 94, 118
mapping sound effects to, 151
new, 337
remote, placing sensor on, 232
role, adding, 339
sound effects, 148
task, assigning, 176
tracking with sensor, 229
type mapping file, 91
merge, 95
prevent load, 97
VR-Forces, 69
entity parameter, 85
environment
loading configuration, 91
environment variable
MAK_INSTALL_DIR, 39
MAK_LOG_DIR, 42
MAKLMGRD_LICENSE_FILE, 48
RTI_CONFIG, 49
VRV_ALPHA_TO_COVERAGE_RENDERBIN, 123
error message, disable, 89
Esc key, disable, 89
escaping a role, 71
Eurofighter Typhoon, 325
event, scenario, 147
exampleStartupConfig.lua, 85
executable
installation location
Windows, 39
installation locations, 45
executing a script, 389
exercise
site ID, 101-102, 108
exercise ID
DIS, 76, 101, 112
Exercise ID parameter, 76
extensions, 113
external renderer
indirect, disable, 90
eyepoint, See observer

F

F A-18 Hornet, 325
F-16A Fighting Falcon, 325
F-35 Lightning II, 326
F-35A Lightning II, 325
F-35C Lightning II, 326
factory settings
directory, default, 92
Farmer M9, 298
fatigue, 160
experiencing, 161
stamina indicator, 161
weapon accuracy, 163
fault tolerant heap, disabling, 395
FDD file, 49
FED file, 49
name, 100, 110
FED File Name parameter, 74
federate
name, 100, 110
network interoperability, 34
type, 100, 110
Federate Name parameter, 75
Federate Type parameter, 74
federation execution
name
specifying, 74
Federation Name parameter, 74
Fiat 124, 314
file
FDD, 49
FED, 49
name, 100, 110
installed, 45
RID, 49
structure, 274
transporter (DIS), 100-101
video, 239
File menu, 59
fire
event
mapping sound effects to, 151
sound effects, 148
firing a weapon
accuracy, 163
ground vehicle, 189
sensor operator, 231
soldier, 164
using HMD, 164
first person view mode, 147
flaps, raising and lowering, 205
flare, 207
Flare Rapider 4 Quadcopter, 330, See also drone
flashlight, 178
FLEXlm, 46
flight controller, 255
Follow Me task, 176
FOM, 48
module, 100, 111

FOM Mapper
initialization data, 75, 100, 110
library, 75, 100, 110
FOM Mapper Initialize String parameter, 75
FOM Mapper Library parameter, 75
FOM Version parameter, 75
frame rate, 50, 109
setting, 92
frequency
hopping, 135
radio, 135
front-end
process ID, 106
FTH, 395
full screen, 97
mode, 92
full-screen mode, 77
function
profiler, 120
FV 510 Warrior, 314
FV4034 Challenger 2 MBT, 314

G

G27 racing wheel mappings, 252
G29 racing wheel mappings, 252
game controller
Action menu, 248
cars and trucks, 252
drone, 250
fixed-wing flight control, 266
generic numbering, 269
human, 249
rotary-wing flight control, 267
sensor operator, 267
vehicle, 251
Gamepad, See also game controller
generic numbering, 269
gear, engaging on ground vehicle, 186
geocentric terrain
Vortex origin, 375
geographic DDM
enable, 101
gesturing, 157
GL Studio
custom overlays, 383
model definition, 384
RSO, 384
updater, 384
GPU performance, 94
graphic
buffer, depth and stencil, 122
depth, 89
green screen, 82
grenade
choosing, 162
throwing, 164
Ground mode, running, 92
ground vehicle
backup camera, 187
braking, 187
controlling, 184
display options, 186
driving, 186
gear, shifting, 186
gunner, 188
mouse and keyboard mappings, 245
rear view, 187
roles, 184
roles on, 70
starting, 186
steering, 187
throttle, 187
towing, 195
Vortex, 347
while untasked, 375
weapon
firing, 189
selecting, 189
stabilizing, 189
GTK Boxer APC, 315
GTK Boxer CRV, 315
GUI
localizing, 51
player, 32
starting without, 97
window size, 92
guiTerrain parameter, 85
gunner
aircraft, 198
commander override, 192-193
ground vehicle, 188
LSV MK II, 190
role, 188-189

H

hand controller
firing, 164
handedness, 164
mappings, 268
handgun
aiming, 163
choosing, 162
HBAO
enabling, 124

settings, 125
heading
display unit
human role, 159
pilot role, 206
setting for autopilot, 204
headlights
turning on and off, 188
head-mounted display, See HMD
head-up display, 198
display options, 202
health
human, 160
level indicator, 161
sustaining damage, 161
weapon accuracy, 163
heap, fault tolerant, 395
helicopter, See also rotary-wing entity
power plant, 205
help
HTML5, 18
help, command-line options, 100
help, HTML5, 46
hiding
weapon, 165
hierarchy
file
merge, 95
HLA
advisory message
enabling and disabling, 75
starting in, 93
HMD
aircraft, 203
anchoring, 79
controller, 268
enabling and disabling, 80
entity control, 66
eye tracking, 79
firing weapon, 164
optimizing performance, 84
player controls, 152
setup, 82
HMMWV Utility Vehicle, 315
horizon-based ambient occlusion
enabling, 124
settings, 125
host address, 100-101
hosting a VR-Forces session, 57
hostname
license server, 47
HOTAS Warthog, 255
HTC Vive, 78, 164, 268
HUD, 198, 202
custom, 380
See also overlay, 380
human, 156
carrying, 174
combat, 162
dragging, 174
embarkation, 172
game controller mapping, 249
gestures, 157
health and stamina, 160
lasing, 179
looking around, 157
mouse and keyboard mappings, 243
movement, 157
navigation, 158
placing in vehicle, 174
resetting, 151
tasking other entities, 176

I

ID
application, 101
session, 101-102, 107
site, 101-102, 108
Ignore Advisories parameter, 75
image
capture, 240
image generator, alternative, 404
ImGui overlay
debug, 128
implicit DDM configuration, 75
Implicit Geographic DDM parameter, 75
indicator
flight instruments, 198
health, 161
posture, 157
radar, 210
stamina, 161
indirect rendering
external, disable, 90
infrared sensor, 179
initialization data, FOM Mapper, 75
initialization settings, 85, 89
Initialize String parameter, 75
injury
carry or drag wounded person, 174
place person in vehicle, 174
sustaining damage, 161
installation, 38
data
options, 44

default directory, Windows, 39
documentation, Windows, 40
RTI, 49
Windows, 38
installing
VR-Forces plug-in, 39
instancing, DI-Guy, 127
instrument panel, 200
display options, 202
Insurgent, 298
Insurgent M9, 299
intercom, 135
using, 194
intersection, KD trees, 90
interval, timeout, 101
IP address, 101-102
for outgoing data, 76
IPV6, 102, 109
IR mode, 179
sensor, 230
Iranian Soldier, 299
Iraqi Police Officer, 299
Iraqi Soldier, 300
ISIS Fighter, 298
Israeli Soldier, 300
Israeli Soldier M4, 300

J

jamming
MAK radio frequency, 142
Javelin
configuring, 166
equipping, 167
shoulder-mounted, 168
tripod, 169
joining a VR-Forces session, 69
joystick
configuration, 272

K

KD tree, 90
keyboard
controlling Action menu, 134
entity control, configuration, 272
mapping, 275
Action menu, 242
aircraft, 247
mappings, 242-243, 245
Kurdish Fighter, 301

L

ladder, climbing, 158
landing
drone, civilian, 223
gear, aircraft, 205
language, specifying, 53, 95
laser code
defining
sensor operator, 230
soldier, 181
tracking with sensor, 229
laser designator
AN/PED-1 LLDR, 171
defining laser code
sensor operator, 230
soldier, 181
enabling, 181, 193
lasing
AN/PED-1 LLDR, 171
defining laser code
sensor operator, 230
soldier, 181
displaying a gimbaled sensor view, 181
targets, 178
commander, 193
sensor operator, 231
soldier, 179
launcher
configuring, 166
equipping, 167
Javelin, shoulder-mounted, 168
Javelin, tripod, 169
RPG, 171
LAV 6 ISC, 316
LAV III APC, 316
Leopard 2A4, 316
library
dynamic, 112
FOM Mapper, 75, 100, 110
Libyan Police Officer, 301
Libyan Soldier, 301
license, 35
RTDynamics, 35
vortex, 35
VR-Forces, 36
license manager, 46
license server
hostname, 47
light
emergency, 195
ground vehicle, 188

Light Strike Vehicle MK II (LSV), 317
  arm weapons, 190
  gunner, 190
  Spike missile, 191
Listen for View Controls parameter, 75, 77
LLDR, AN/PED-1, See AN/PED-1 LLDR
load balancing, 86
loading
  entity display settings, 91
  environment configuration, 91
  object type mapping, 91
  plug-ins, 97, 106, 113
  scenario, 107
  scene from command line, 99
  terrain
    from command line, 99
    paged, 106
    unhosted session, 58
Local Settings Designator parameter, 75
locale, specifying, 53, 95
localization
  specifying, 53, 95
  VR-Engage GUI, 51
location
  bullseye, selecting, 206
locking a differential, 188
log file, 105
  callstack, 40
  location, 42
  notification level, 97, 107
  notify level, 106
  specifying, 95, 106
  vrfSim, 107
LSR, 230
LSS, 231
Lua
  script, 342
lying prone, 157

M

M1A1 AIM (AUS), 320
M1A1 AIM (AUS) Mineplow, 320
M1A1 SA, 320
M1A2 Abrams MBT, 320
M1A2 SEP Abrams, 321
M2A3 Bradley IFV, 321
M113 APC, 318
M270 MLRS, 321
M915 Tractor Truck, 321
M977 HEMTT Cargo Truck, 322
M983 HEMTT Patriot Tractor, 322
M1083 MTV Cargo Truck, 317
M1120A4 HEMTT LHS, 317
M1126 Stryker ICV with M2, 318
M1151 Expanded Capacity Armament Carrier (ECAC), 318
M1151 HMMWV ECAC with M2, 318
M1151 HMMWV ECAC with M240, 319
M1151 HMMWV ECAC with M240B, 319
M1151 HMMWV ECAC with MK 19, 319
M1278 L-ATV-HGC (JLTV), 319
MAK data
  install
    options, 44
MAK Data Logger, 34
MAK ONE integration, 33-34
MAK radio
  configuring, 138
  intercom, 138
  jamming, 142
  VOX, 138
  VR-Forces, 136
MAK RTI configuration, 49
MAK_INSTALL_DIR environment variable, 39
MAKLMGRD_LICENSE_FILE environment variable, 48
makRadio, 136
managing plug-ins, 112
manual, location of, 46
mapping
  keyboard, 242-243, 245
  mouse, 242-243, 245, 248
  overwrite on merge, 96-97
  sound effects, 151
mappings
  G27 racing wheel, 252
  G29 racing wheel, 252
  game controller
    Action menu, 248
    drone, 250
    fixed wing, 266
    human, 249
    rotary wing, 267
    sensor operator, 267
    vehicle, 251
material classification, 26
MaxBackends parameter, 86
memory, 50
  GPU usage, 94
menu
  Action, 134
    script, 388
  application, 72
message
  notification level, 97, 107

outgoing, 76
MFD, 201
MGRS code, sensor, 229
MGRS coordinates, 191, 209
MH-60 Black Hawk, 330
Mideast Combatant, 302
MiG-29 Fulcrum, 326
minimap
realize model set, 97
minimap, using, 159
minion parameter, 85
missile
aircraft, 208
Javelin, shoulder-mounted, 168
Javelin, tripod, 169
Spike, LSV, 191
mixed reality, 78, 82
configuring, 81
system requirements, 78
mixed reality headset, Windows, 78
mode
autopilot, 204
setting, 204
full screen, 77
sensor, changing, 230
view, 147
VR-Engage, 59
windowed, 77
mode parameter, 85
model
custom, 362-363, 365-366
definition, prevent load, 97
preloading
file, 96
list, 90
model definition
merging, 96
model set, disable, 90
modulation
detail, 135
major, 135
system, 135
monitor
display options for aircraft, 202
motioning, 157
mouse
control, enabling and disabling, 156
controlling Action menu, 134
mappings, 242-243, 245, 248
Action menu, 242
Move to Location task, 177
Move to My Location task, 177
movement
dynamic terrain, 147
moving
on ladders, 158
on terrain, 157
turret, 189
weapon accuracy, 163
MQ-8B Fire Scout, 330
MQ-9 Reaper UAV, 326
MR, See mixed reality
MRAP M-ATV, 322
MSVS SMP Gun Tractor, 322
multicast
address
specifying, 102, 112
multicast TTL
DIS, 76
setting, 102, 107
Multicast TTL parameter, 76
multi-function display, 200-201
multiple
back-end, 86
simulation engines, 86

N

navigation
compass, 158
minimap, 159
NETN FOM
revision, 111
version, 111
netsocket, 76
network
connecting to
automatically, 88
interoperability, 34
network card, 76
night vision goggles, 179
noise
level, 148
sound effects, 148
notification level
setting, 97, 107
number key, controlling Action menu, 134
NVG, 179
sensor mode, 230

O

object
count, 50

timeout interval, 101
object type mapping
loading, 91
observer, 368
binding to view, 88
publishing, 75, 77
view
disable, 90
file, loading, 95
Observer menu, 59
observer views, 152
occlusion
horizon-based ambient
enabling, 124
settings, 125
ocean
synchronization, 93
Oculus Rift, 78, 164, 268
opening
Action menu, 134
doors and windows, 181
Stryker door, 173
operator
drone, 218
sensor, 228
optimizing
performance, 50
virtual reality, 84
option
command-line, 87
outgoing PDU, 76
overlay
custom, 380
GL Studio, 383
scripted, 381
sensor, 383
examples, 380
sensor operator, 234
types, 380
Override Global View Control Setting parameter, 75, 77

P

paged terrain
loading, 106
Pakistani Police Officer, 302
Pakistani Soldier, 302
panel
choose role, 63
instrument, 200
parachuting, 175
parameter
Application Number, 75-76
Asynchronous IO, 77
attach, 85
coordinateConverterOrigin, 375
Destination Address, 76
DIS Version, 76
Enable Timeout Processing, 77
enableMultipleBackends, 86
entity, 85
Exercise ID, 76
FED File Name, 74
Federate Name, 75
Federate Type, 74
Federation Name, 74
FOM Mapper Initialize String, 75
FOM Mapper Library, 75
FOM Version, 75
guiTerrain, 85
Ignore Advisories, 75
Implicit Geographic DDM, 75
Initialize String, 75
Listen for View Controls, 75, 77
Local Settings Designator, 75
MaxBackends, 86
minion, 85
mode, 85
Multicast TTL, 76
Override Global View Control Setting, 75, 77
Port, 76
Publish Observers, 75, 77
Receive Buffer Size, 76
scenario, 85
Send Buffer Size, 76
Show Self Published Objects, 77
Site ID, 76
Site Number, 75-76
terrain, 85
Timeout Interval, 77
Paratrooper, 303
Paratrooper M9, 303
paratrooper, parachuting, 175
parking brake, releasing and engaging, 188
Parrot Anafi Quadcopter, 330, See also drone
path
default Windows install, 39
MAK RTI, 49
paused, 72
pausing a scenario, 71-72
PDB files
Linux, 41
Windows, 41

PDU
  profiling, enable, 105
performance, 122
  alpha to coverage, 122
  anti-aliasing, 87
  developer debug overlay, 128
  DI-Guy, 127
  fault tolerant heap, 395
  frame rate, 109
  function profiler, 120
  graphics, depth, 89
  headset, 81
  load balancing, 86
  main thread, 108
  optimizing, 50, 394
  player, 118
  setting application priority, 93
  simulation, 119
  statistics
    displaying, 120
    VR-Engage, 120
  stencil bits, 98
  terrain, 89, 128
  Tracy Client, 121, 394
  virtual reality, 81
persistent shadow technique, 91
perspective, commander, 192
pivoting tank, 194
plan
  test result, 106
player station
  command-line options, 112
playerStation.lua, 86
plug-in
  adding, 114
  configuration
    adding, 115
    deleting, 116
  DLL
    adding, 114
    removing, 115
    specifying, 114
  initializer, 92
  loading, 97, 105-106, 113
  managing, 112
  prevent loading, 99
  VR-Engage for VR-Forces, 39
  XML file, 114
    deleting, 116
Police Cruiser, 323
Police Officer With Drone, 303
Policeman M9, 304
port
  DIS, 76
  Tracy Client, 109
Port parameter, 76
position, commander, 192
posture
  changing, 157
  indicator, 157
  weapon accuracy, 163
power plant, controlling, 205
priority
  application, 93
process
  ID, front-end, 106
product installation, 38
  build information, 38
  Windows, 38
profiler function, 120
prone posture, 157
pseudo noise, 135
Publish Observers parameter, 75, 77
puck, 79

Q

Qt Linguist, 51
Qt menus, 59
Quadcopter, 331, See also drone
quiet mode, 105

R

radar
  Range While Scan, 211
  select multiple targets, 213
  Single Track Target, 211
  system, 210
  target, air-to-air, 211
  Track While Scan, 211
  warning receiver, 207
radio
  application, 136
  enabling on startup, 142
  settings, 135
  using, 135
  VR-Forces, 136
Range While Scan, radar, 211
RDE, 26, 33
rearview camera, 187
receive buffer size, 76, 102, 107
  DIS, 76
Receive Buffer Size parameter, 76

reloading, 166
Remote Camera, 331
remote camera, sensor operator, 228
remote discovery, disable, 105
remote display engine
  connecting, 33
  multiple instances, 87
  support, 33
remote entity, placing sensor on, 232
remote sensor, configuring offset, 232
render bin, 123
resetting
  entity, 151
  sensor, 230
restoring
  entities, 151
restrict application access, 88
reticle, accuracy indicator, 163
reverse Z buffer
  disable, 90
revision, RPR FOM, 101, 111
RID file, 49
rid.mtl file, 49
rifle
  aiming, 163
  choosing, 162
rocket launcher, See RPG
rocket-propelled grenade launcher, See RPG
role
  adding entity to scenario, 65
  adding to entity file, 339
  aircraft, 198
  assigning in VR-Forces, 67
  changing, 70
    by embarking, 70
    by menu, 71
  choosing, 63
    existing, 64
  commander, 192
    gunner override, 193
    laser designator, 193
    slew to cue, 193
  configuration, 274
  copilot, 198
  data structure, 339
  driver, 189
  embarkation slots, 342
  escaping, 71
  ground vehicle, 70, 184
  gunner, 188-189
  inheritance, 345
  Lua file, 342, 345
  overview, 27
  parameters, 339
  sensor operator, 228
  sling load operator, 198
rotary-wing entity
  engine power, 205
  power plant, 205
RPG
  configuring, 166
  equipping, 167
  using, 171
RPR FOM
  revision, 101, 111
  specifying version, 75
  version, 101, 111
RQ-4 Global Hawk UAV, 327
RQ-7 Shadow UAV, 327
RQ-11 Raven UAV, 327
RTDynamics, 35
RTI, 48
  prevent check, 111
RTI_CONFIG environment variable, 49
run mode
  sim. engine, 108
Russian Rebels, 304
RWS, radar, 211

S

ScanEagle UAV, 327
scenario
  creating, 372
  ending, 72
  hosting
    in VR-Engage, 376
    in VR-Forces, 376
  loading, 107
    at startup, 107
  pausing, 71-72
  state, 71
scenario event, 147
scenario parameter, 85
scene
  loading from command line, 99
screen
  multiple, aircraft, 202
  number, specifying, 95
script
  activating, 388-389
  adding to menu, 388
  role configuration, 385
  startup, 85
  system, directory, 103
scripted overlays, 381

SDK toolkit, 42
  documentation, 40
Security Camera, 331
Security Guard M9, 304
selecting
  bullseye, 206
  targets, 213
self reflection, DIS, 112
send buffer size, 76, 102, 107
  DIS, 76
Send Buffer Size parameter, 76
sensor
  capturing images, 240
  centering on screen, 229
  display
    changing, 230
    gimbaled sensor view, 181
  focusing, 214, 229
  infrared, 179
  IR mode, 230
  mode
    crew member, 194
    sensor operator, 230
  night vision, 179
  NVG mode, 230
  operator, 228
    firing weapons, 231
    lasing, 231
  placing on remote entity, 232
  range, 228
  remote, configuring offset, 232
  resetting, 230
  streaming video, 238
  street, tracking, 229
  taking pictures, 240
  tracking, 229
    laser code, 229
    MGRS, 229
  tracking, reset, 78
  visual mode, 230
sensor operator, 228
  game controller mapping, 267
  mouse mappings, 248
sensor overlay, 383
SensorFX, 26
  enable, 91
session
  hosting VR-Forces, 57
  ID, 101-102, 107
    setting at startup, 107
  scenario
    creating, 372
    hosting, 376
  starting, 73
  unhosted, 58
  VR-Forces, joining, 69
set data request
  Vortex while untasked, 375
settings
  environment configuration, 91
  initialization, 85, 89
  local, RTI, 100, 111
  shared directory, 108
  user
    directory, 110
shadow
  persistent, enable, 91
shared data
  directory, 98
  root, 98
  stem, 98
shared settings directory, 108
Show Function Profiler Overlay, 120
Show Self Published Objects parameter, 77
SIDC
  map to DIS, 107
sight
  weapon, 163
sight, weapon, 163
signaling, 157
sim engine
  enabling time management, 111
simulation engine
  multiple, 86
simulation model set
  extending, 336, 362
  in use, 58
  new, 337, 339
  SOE, 363
  specifying on command-line, 108
  upgrading, 45
simulation network
  connecting to, 73
simulation object, 50
Simulation Object Editor, 336
Singaporean Soldier, 305
Single Track Target, radar, 211
siren, emergency, 195
site ID, 101-102, 108
Site ID parameter, 76
Site Number parameter, 75-76
slew to cue, 193
sling load operator role, 198
SO file, 112
socket
  send and receive buffer size, 76

soldier, See human
Somali Police Officer, 305
Somali Soldier, 305
sound effects, 148
  decibel, 150
  enabling, 148-149
  mapping to entities, 151
sound, radar warning receiver, 207
Spark quadcopter, See drone
specifying
  federation execution name, 74
  language, 53, 95
  notification level, 97, 107
  RPR FOM version, 75
  simulation model set, 108
SpeedTree
  LOD, zoom, 93
  synchronization, 93
Spike missile launcher, 191
  indirect target, 191
  point and shoot, 191
  re-aim in flight, 192
  simple fire, 191
Sport Hatchback Coupe, 323
Sport StandardSize Sedan, 323
stabilizing a turret, 189
stamina, 160
  indicator, 161
  losing, 161
Standard MidSize Sedan, 323
standing, 157
starting
  session, 73
  VR-Engage, 56
startup
  script, 85
state
  scenario, 71
station name, 98
statistics
  displaying, 120
  VR-Engage performance, 120
SteamVR, 78
  setup, 82
  testing camera, 83
stencil bits, 98
stencil buffer, 122
stowing a weapon, 165
streaming video
  drone, 238
  receiving, 239
  sensor, 238
street, tracking with sensor, 229
Stryker, embarking on, 173
STT, radar, 211
subnet mask, 102, 109
Sudanese Police Officer, 306
Sudanese Soldier, 306
Suspect, 306
Suspect M9, 307
switching
  commander position, 192
  sensor mode, 230
Syrian Police Officer, 307
Syrian Soldier, 307
system script directory, 103

T

T.Flight Hotas 4 controller, 260, 263
T.Flight Hotas X, 255
  fixed-wing mappings, 260
  rotary-wing mappings, 263
T-38C Talon, 328
tactical graphic
  element definition
    prevent load, 97
  hier file
    directory, 98
    merge, 95
  leaf file
    directory, 98
    merge, 96
  type map
    merge, 96
    prevent load, 97
taking pictures, 240
Taliban, 308
tank, See also ground vehicle
  commander, 192
  pivoting, 194
target
  air-to-air, radar, 211
  air-to-ground
    laser, 209
    MGRS coordinate, 209
  drone strike, 225
  following with sensor view, 214
  lasing, 178
    sensor operator, 231
    soldier, 179
  select multiple, 213
target address, 76
task
  assigning to entity, 176
  Disembark Entity, 176

Follow Me, 176
Move to Location, 177
Move to My Location, 177
Technical Truck, 324
terrain, 33
dynamic, 89, 147
intersection, 128
limit textures, 94, 118
loading
from command line, 99
unhosted scenario, 58
moving on, 157
paged
loading, 106
terrain database
loading
at startup, 109
terrain parameter, 85
terrain performance, 128
text
left-to-right, 92
texture
earth file
limit splatting, 93
limit terrain, 93
limit vegetation, 94
entity
limit exclusions, 94
limit mipmap, 94, 118
third-person view mode, 147
thread
callback, 107
disable for parallel algorithm, 110
network callback, 107
priority, 108
throttle
ground vehicle, 187
throwing a grenade, 164
Thrustmaster Warthog
fixed-wing mappings, 255
rotary-wing mappings, 258
Thunderbolt, 78
tick
disable parallel, 105
publishers, 109
time management
enabling
for sim.engine, 111
time stamp, See timestamp
time-based events, synchronize, 93
timeout
DIS, 111
interval, 101
interval, DIS, 77
Timeout Interval parameter, 77
timestamp
absolute, 76-77
enable, 101-102, 109
toolkit
API, 42
SDK, 42
torch, See flashlight
touch controller
firing, 164
handedness, 164
mappings, 268
tow a vehicle, 195
Track While Scan, radar, 211
tracking
entity, 229
waypoint, 229
tracking sensor, resetting, 78
Tracy Client, 121, 394
enable, 91
translation, VR-Engage GUI text, 51
transparency
alpha to coverage, 122
triggering
event, VR-Forces, 147
script, 389
troubleshooting, 394-396
truck, See also ground vehicle
game controller mapping, 252
turret
aiming, 189
moving, 189
stabilization, 189
TWS, radar, 211

U

UAV, See also drone
piloting, 218
sensor, 228
Ugandan Police Officer, 311
Ugandan Soldier, 312
UH-60M Blackhawk, 331
unhosted session, 58
unit
element definition, prevent load, 97
hier file
directory, 99
merge, 96
leaf file
directory, 99
merge, 96

type map
merge, 96
type mapping file
prevent load, 97
unlocking a differential, 188
Ural 4320, 324
US Army Javelin, 308
US Army Javelin (Tripod), 308
US Army M4, 309
US Army M9, 310
US Army M16, 309
US Army M16-M203, 309
US Border Patrol M9, 310
user
data, specifying directory, 99, 109
USMC M4, 311
USMC M16, 310
USMC M16-M203, 311
UUID
scheme, 107

V

Valve Index, 78
variable, environment, See environment variable
Varjo, 78
vehicle
changing role, 71
disembarking from, 174
embarking on, 70, 172
game controller mapping, 251
intercom, 194
resetting, 151
version
displaying, 99, 110
product, 38
RPR FOM, 101, 111
video
RAM crash, 395
receiving stream, 239
video card, processor speed, 50
video stream, 238
drone, 238
file, 239
file settings, 239
sensor, 238
view
commander, 192
mode, 147
observer, 152
observer, load file, 95
viewpoint, See observer
virtual reality, 78, 82
configuring, 81
enabling and disabling, 80
optimizing, 84
performance, 81
system requirements, 78
tracking sensor, resetting, 78
Vive tracker, 79
visual sensor
mode, 230
vehicle crew, 194
Vive, 164
setup, 82
testing camera, 83
Vortex
configuring
for geocentric terrain, 375
ground vehicles, 347
game controller mapping, 252
Vortex While Untasked set data request, 375
vortexTerrainConfig.lua, 375
VR, See virtual reality
VRAM issues, 395
VR-Engage, 26
add-on for VR-Forces, 39
extending, 34
plug-in for VR-Forces, 372
starting, 56
user modes, 59
VR-Engage Air, 87
VR-Engage Ground, 92
VR-Engage Sensor Operator, 98
VR-Forces
add-ons installer, 372
create VR-Engage scenario, 372
entity, 69
integration, 33
radio, 136
scenario, 371, 376-377
session, hosting, 57
VR-Engage add-on, installing, 39
VR-Engage manager panel, 67
VR-Engage plug-in, 372
VR-Engage plug-in, installing, 39
VR-Forces plug-in, 39, 372
vrfSim
log file, 107
VRV_ALPHA_TO_COVERAGE_RENDERBIN environment variable, 123
VSync, 91

W

waypoint
  tracking with sensor, 229
weapon
  accuracy, 163
  aiming, 163
  aircraft
    choosing, 208
    firing, 209, 213
  deploying or stowing, 165
  firing
    ground vehicle, 189
    sensor operator, 231
    soldier, 164
    with HMD, 164
  gunner role, selecting, 189
  LSV MK II, arm and disarm, 190
  reloading, 166
  sight, 163
  soldier role, 162
window
  opening and closing, 181
  specifying, 92
windowed mode, 77, 99
Windows
  default install directory, 39
  product installation, 38

X

Xbox controller, See also game controller
  generic numbering, 269
XML
  file, plug-in, 114
    deleting, 116
  RTI configuration file, 49
XR, See also augmented reality
XTAL, 78
  setup, 82

Z

Z-fighting, 90
zoom
  binoculars, 178
  SpeedTree LOD, 93



VRN-2.2-02-260107