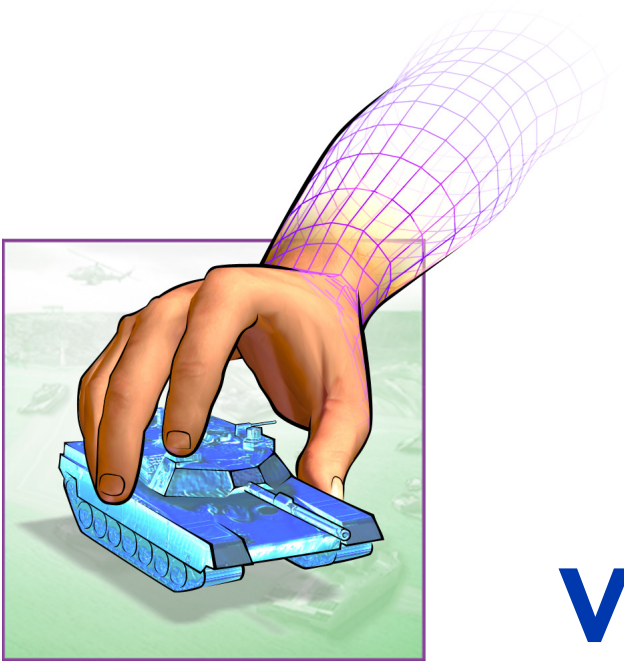
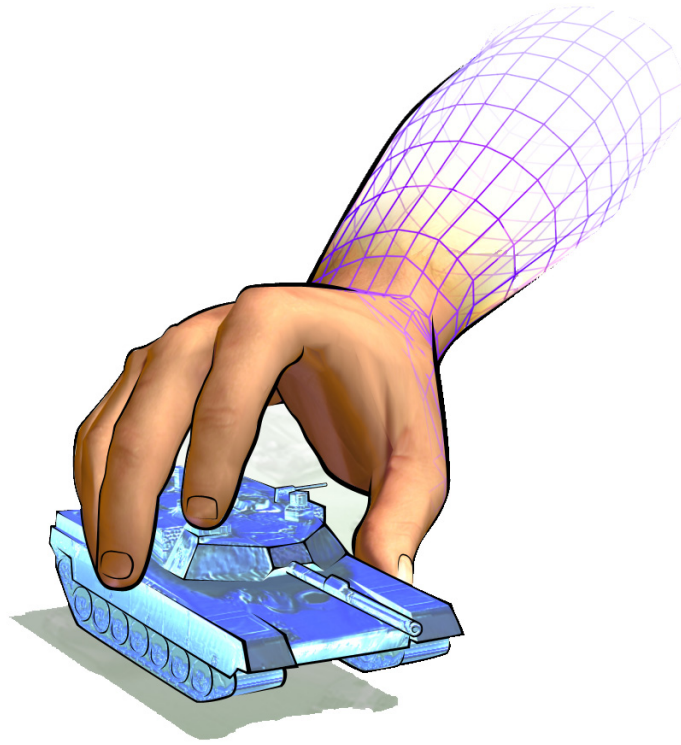




VR-Forces

Users Guide



VR-Forces

Users Guide

Copyright © 2011 VT MÄK
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 VT MÄK.

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

All other trademarks are owned by their respective companies.

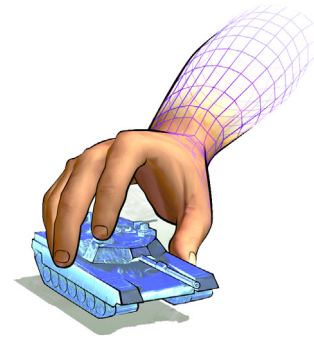
For third-party license information, please see “Third Party Licenses,” on page xx.

VT MÄK
68 Moulton St.
Cambridge, MA 02138 USA

Voice: 617-876-8085
Fax: 617-876-9208

info@mak.com
www.mak.com

Revision VRF-4.0-1-110114



Contents

Preface

How This Manual is Organized	xiii
VR-Forces Documentation.....	xv
MÄK Products	xvi
How to Contact Us	xviii
Document Conventions	xix
Hardware and Operating System Naming Conventions	xix
Mouse Button Naming Conventions.....	xx
Third Party Licenses	xx
Boost License.....	xx
libXML and libICONV	xxi
pThreads Library.....	xxi
EHS, PCRE, and PME	xxi

Chapter 1 Introduction to VR-Forces

1.1. Overview	1-3
1.1.1. Realistic Display of Vehicles and Terrain (3D View)	1-4
1.1.2. Create Complex Scenarios	1-5
1.1.3. Entity Types Supported	1-5
1.1.4. Mission Planning	1-7
1.1.5. Entity Tasks	1-8
1.1.6. Tactical Graphics	1-8
1.1.7. Terrain Agility and Composability	1-9
1.1.8. Flexible, Intuitive Graphical User Interface	1-9
1.1.9. Tactical Graphics Overlays	1-10
1.1.10. Behaviors	1-10
1.1.11. Multiple Attach Modes	1-10
1.1.12. Special Effects	1-11
1.1.13. Accurate Vehicle Positioning	1-11

1.1.14. Batch Mode Operation	1-11
1.1.15. Remote Control	1-11
1.2. The VR-Forces Toolkit	1-12
1.2.1. Plug-in Architecture	1-12
1.3. The B-HAVE Module for VR-Forces	1-13
1.4. Support for External Communications Servers	1-13
1.5. Helpful Utilities	1-14
1.6. Third-Party Software and Content	1-14
1.6.1. SilverLining	1-15
1.6.2. GL Studio	1-15
1.6.3. DI-Guy	1-16
1.6.4. SpeedTree	1-16
1.6.5. 3D Models, Terrain, and Graphical Content	1-17
1.6.6. OpenSceneGraph	1-17
1.6.7. osgEarth	1-17
1.7. Distributed Simulation Standards Supported	1-18

Chapter 2 Installing VR-Forces

2.1. Installing VR-Forces	2-2
2.1.1. Installing VR-Forces on Windows	2-2
2.1.2. Installing VR-Forces on a Linux System	2-3
2.2. Uninstalling VR-Forces	2-4
2.3. The VR-Forces Directory Structure	2-5
2.4. Installing and Setting Up the License Manager	2-6
2.4.1. Specifying the License Server	2-7
2.5. Installing an RTI	2-9
2.5.1. Installing the MÄK RTI	2-10
2.6. Localizing the Graphical User Interface	2-11
2.6.1. Translating Other Interface Files	2-14
2.6.2. Applying the Language Files	2-14
2.6.3. Merging Translation Files	2-14

Chapter 3 VR-Forces Simulation Concepts

3.1. The VR-Forces Program	3-3
3.2. Front-end and Back-end Concepts	3-3
3.2.1. How Front-ends and Back-ends Work Together	3-4
3.2.2. How VR-Forces Back-ends are Identified	3-5
3.2.3. VR-Forces Sessions	3-5
3.2.4. Coordinating Multiple Front-ends	3-7
3.2.5. Working with Multiple Back-ends	3-8

3.3. Scenarios	3-9
3.3.1. The Scenario File	3-9
3.3.2. Terrain Databases	3-9
3.3.3. The Order of Battle File	3-10
3.3.4. The Plan Files	3-10
3.3.5. The Object Map File	3-10
3.3.6. The Hostility Relationships File	3-10
3.3.7. The Batch Mode File	3-11
3.4. Objects	3-11
3.4.1. The Object Parameter Database	3-11
3.4.2. Local Objects and Remote Objects	3-12
3.5. Entities	3-12
3.5.1. How Entities are Identified	3-13
3.5.2. How Entities are Organized	3-15
3.5.3. How Entities Communicate	3-16
3.5.4. VR-Forces Uses a Component Architecture	3-17
3.5.5. Entity Behaviors and Tasks	3-19
3.6. Aggregate Entities	3-20
3.6.1. How Aggregates Are Organized	3-21
3.6.2. How Aggregates Move	3-22
3.6.3. How Aggregates Carry Out Tasks	3-23
3.7. Tactical Graphics	3-24
3.7.1. Points	3-25
3.7.2. Phase Lines	3-25
3.7.3. Routes	3-26
3.7.4. Areas	3-26
3.7.5. Obstacles	3-26
3.8. Tasks	3-26
3.9. Plans	3-27
3.10. Set Data Requests	3-28
3.11. Terrain Databases	3-28
3.11.1. Composing Terrains (and Creating MTF Files)	3-29
3.11.2. Coordinate Systems	3-30
3.11.3. MTD Files	3-31
3.11.4. Loading Supported Databases	3-31
3.11.5. Using Large Terrain Databases	3-32
3.12. Intervisibility (Line-of-Sight)	3-34
3.13. Representing and Managing Time in VR-Forces	3-35
3.13.1. Simulation Time and Exercise Time	3-35
3.13.2. Exercise Clock Modes	3-36
3.13.3. Using HLA Time Management	3-38
3.14. Interactions	3-38

Chapter 4 VR-Forces Visualization Concepts

4.1. The Display Engine	4-2
4.1.1. Window Types	4-2
4.2. The Observer	4-2
4.2.1. Observer Modes	4-3
4.3. Introduction to VR-Forces Object Modeling	4-3
4.3.1. Entity, Object, and Effects Model Formats	4-6
4.3.2. Model Sets	4-6
4.3.3. Mapping Models to Entities and Objects	4-7

Chapter 5 Starting VR-Forces

5.1. Starting VR-Forces	5-3
5.1.1. Starting VR-Forces from the VR-Forces Launcher	5-3
5.1.2. Starting Independent VR-Forces Executables	5-6
5.1.3. Specifying a Session ID	5-7
5.1.4. VR-Forces Startup Tutorial	5-7
5.2. The VR-Forces Window	5-9
5.2.1. Switching Between the 2D and 3D Views	5-11
5.2.2. Opening New Windows	5-11
5.3. Managing a Front-End's Session Connection	5-12
5.3.1. Joining a Session	5-13
5.3.2. Resigning from a Session	5-13
5.3.3. Configuring Session Messages and Join at Startup	5-14
5.4. Configuring Time Management for HLA Exercises	5-15
5.4.1. RTI Requirements for Time Management	5-15
5.4.2. Enabling Time Management for the Back-end	5-16
5.5. Opening a Terrain Database	5-17
5.5.1. Loading a Terrain Database at Startup	5-18
5.5.2. Saving a Terrain Database in GDB Format	5-18
5.5.3. Closing a Terrain	5-18
5.6. Managing VR-Forces Settings	5-19
5.6.1. Synchronizing Settings Among VR-Forces Installations	5-21
5.6.2. Global Settings and Observer-Specific Settings	5-21
5.7. Exiting VR-Forces	5-21
5.8. Configuring Simulation Connections	5-22
5.8.1. Opening the Simulation Connections Configuration Dialog Box	5-22
5.8.2. Adding a Simulation Connection	5-23
5.8.3. Editing a Simulation Connection	5-24
5.8.4. Deleting a Simulation Connection	5-24
5.8.5. Configuring Auto Connect	5-24

5.9. Command Line Options	5-25
5.9.1. Command-Line Options for vrfGui	5-25
5.9.2. Command Line Options for vrfSim	5-28
5.9.3. Command Line Options for vrfLauncher	5-32
5.9.4. Protocol-Independent Command Line Options	5-33
5.9.5. Command Line Options for HLA Federations	5-35
5.9.6. Command Line Options for DIS Exercises	5-37
5.10. Managing Plug-ins	5-39
5.10.1. Loading a Plug-in	5-39
5.10.2. Adding a Plug-in	5-40
5.10.3. Specifying the DLLs for a Plugin	5-41
5.10.4. Adding a Plug-in Configuration	5-42
5.10.5. Deleting a Plug-in Configuration	5-42
5.10.6. Deleting a Plug-in	5-43
5.10.7. Viewing a List of Loaded Plug-ins	5-43
5.11. The VR-Forces Log Files	5-44

Chapter 6 Managing Objects

6.1. The Objects List Panel	6-2
6.2. Selecting Entities, Tactical Graphics, and Props	6-4
6.2.1. Selecting Objects in the Window	6-5
6.2.2. Selecting Objects in the Objects List Panel	6-6
6.2.3. Unselecting Objects	6-6
6.3. Showing and Hiding Tactical Graphics	6-7
6.3.1. Showing and Hiding Tactical Graphics	6-8
6.3.2. Showing and Hiding Vertex Labels	6-9
6.3.3. Displaying Height-Above-Terrain Lines for Vertices	6-9
6.4. Displaying Information About an Object	6-10

Chapter 7 Viewing Entities and Entity Information

7.1. Displaying Entity Labels	7-2
7.1.1. Displaying Text Labels for 3D Models	7-4
7.1.2. Turning Entity Labels On and Off	7-5
7.1.3. Pinning 3D Text Labels to the Window	7-5
7.1.4. Displaying Entity Labels for 2D Icons	7-6
7.2. Entity Icons	7-8
7.2.1. Changing the Size of 2D Icons	7-9
7.3. Model Sets	7-10
7.3.1. Changing the Model Set	7-10
7.4. Exaggerating the Scale of 3D Entity Models	7-10
7.5. Changing Force Colors	7-13
7.6. Displaying Height-Above-Terrain Lines for Entities	7-14
7.7. Displaying Entity Resources	7-15
7.7.1. Viewing Numerical Resource Data	7-15

7.8. Configuring Entity Console Messages	7-16
7.8.1. Setting the Notification Level for Console Messages	7-16
7.8.2. Logging Console Messages to a File	7-17
7.8.3. Clearing the Object Console	7-17

Chapter 8 Managing Entity Effects

8.1. Using Ground Clamping	8-2
8.1.1. Configuring Ground Clamping	8-3
8.2. Trajectory Smoothing	8-4
8.2.1. Configuring Trajectory Smoothing	8-5
8.3. Inset Views	8-6
8.4. Displaying Trailing and Decal Effects	8-7
8.4.1. Specifying Models for Trailing Effects and Decal Effects	8-8
8.5. Displaying Track Histories	8-9
8.6. Viewing Fire and Detonation Graphics	8-10
8.7. Displaying Sensor Volumes	8-11
8.8. Displaying Cockpit Displays	8-12
8.9. Displaying Shadows	8-13
8.9.1. Configuring Shadows	8-14
8.10. Disabling Lighting and Clouds	8-14

Chapter 9 Aggregate Entities

9.1. Selecting Aggregates	9-2
9.2. Expanding and Collapsing Aggregates	9-2
9.2.1. Expanding and Collapsing Aggregates in the Echelon View	9-4
9.2.2. Expanding and Collapsing the Echelon View by Level of Aggregation	9-4
9.3. Displaying Ghosted Icons	9-5
9.4. Displaying Aggregate Icons and Bounding Volumes	9-8
9.4.1. Specifying the Color Scheme for Aggregate Bounding Volumes	9-8

Chapter 10 The Observer and Observer Modes

10.1. Changing the Observer Mode	10-2
10.2. Creating and Editing Observer Modes	10-2
10.2.1. Creating an Observer Mode	10-2
10.2.2. Deleting an Observer Mode	10-4
10.2.3. Editing an Observer Mode	10-4
10.2.4. Renaming an Observer Mode	10-4
10.3. Adding and Removing Observers	10-5
10.4. Communicating Information About the Observer	10-6
10.4.1. Making the Observer Visible to Other Applications	10-6
10.4.2. Viewing and Attaching to Other Observers	10-6
10.5. Displaying Models for Remote Observers	10-7

Chapter 11 Moving the Observer

11.1. Moving the Observer	11-3
11.1.1. Using the Compass to View the Observer's Heading	11-4
11.1.2. Editing the Compass's Attributes	11-5
11.2. Observer Movement Functions and Keyboard Mappings	11-5
11.2.1. 3D Observer Movement Coordinate Systems	11-6
11.2.2. 2D Observer Coordinate System	11-6
11.2.3. The Attached Context and Observer Movement (3D Projection Only)	11-7
11.3. Moving the Observer from the Keyboard in the 3D View	11-9
11.3.1. Navigation Options	11-11
11.3.2. Orbiting	11-12
11.3.3. Moving the Camera	11-12
11.4. Moving the Observer Using the Mouse (3D View)	11-13
11.4.1. Dragging the Terrain	11-13
11.4.2. Changing the Observer's Orientation with the Mouse	11-14
11.4.3. Dragging the View	11-14
11.4.4. Orbiting the Terrain with the Mouse	11-14
11.4.5. Teleporting the Observer	11-14
11.5. Moving the Observer in the 2D View	11-14
11.5.1. Moving the Observer from the Keyboard in the 2D View	11-15
11.5.2. Moving the Observer Using the Mouse in the 2D Projection	11-15
11.6. Forcing Orientation North	11-16
11.7. Enabling and Disabling View Constraints (3D Only)	11-17
11.8. Controlling Navigation Speed	11-18
11.8.1. Enabling or Disabling Speed Scaling	11-18
11.8.2. Changing the Observer's Movement Speed	11-19
11.9. Saving and Recalling Views	11-20
11.9.1. Recalling a View	11-21
11.9.2. Deleting Views	11-21
11.9.3. Saving Views to a File	11-21
11.9.4. Loading Views	11-22
11.10. Controlling the Observer from Other Applications	11-22
11.10.1. Enabling the Processing of View Control Messages	11-22

Chapter 12 Attaching the Observer

12.1. Attaching the Observer to an Entity or Prop	12-2
12.1.1. Attaching to an Articulated Part	12-3
12.1.2. Attaching the Observer in Mimic Track or Tether Track Mode	12-4
12.1.3. Filtering the Object Attachment List	12-4
12.1.4. Detaching from an Entity	12-4
12.1.5. Hiding the Attached Model (3D Projection Only)	12-4
12.1.6. Attaching to Props	12-5
12.2. Attach Modes in the 3D View	12-5
12.2.1. Absolute Mode	12-5
12.2.2. Follow Mode	12-6
12.2.3. Mimic Mode	12-7
12.2.4. Mimic Track Mode	12-8
12.2.5. Tether Mode	12-8
12.2.6. Track Mode	12-9
12.2.7. Tether Track Mode	12-10
12.3. Attach Modes in the 2D View	12-10
12.4. Attaching the Observer at Startup	12-11

Chapter 13 Intervisibility

13.1. Types of Intervisibility	13-2
13.1.1. Point-to-Point Intervisibility	13-2
13.1.2. Intervisibility Fan	13-3
13.1.3. Entity Intervisibility	13-3
13.2. Intervisibility Objects can be Transient or Permanent	13-4
13.2.1. Creating a Transient Intervisibility Object	13-4
13.2.2. Creating a Permanent Intervisibility Object	13-5
13.3. Editing an Intervisibility Object	13-5
13.4. Configuring Intervisibility Lines and Fans	13-6
13.4.1. Editing an Intervisibility Object's Properties	13-7
13.5. Pinning an Intervisibility Object to an Entity	13-8
13.6. Deleting Intervisibility Objects	13-8

Chapter 14 Changing How Terrain is Displayed

14.1. Introduction	14-2
14.2. Configuring Visible Surfaces (GDB Soil Types)	14-2
14.3. Coloring Terrain Based on Elevation	14-4
14.3.1. Configuring Elevation Color Settings	14-5
14.4. Displaying Grid Lines	14-6
14.4.1. Configuring Grid Lines	14-7
14.5. Displaying Contour Lines	14-8
14.5.1. Configuring Contour Lines	14-9
14.6. Displaying Models and Terrain in Wireframe Mode	14-9

14.7. Displaying Feature Data	14-10
14.7.1. Changing the Colors Used for Feature Data	14-11
14.8. Terrain Databases Provided with VR-Forces	14-13

Chapter 15 Remote Graphics

15.1. Displaying Remote Graphics	15-2
15.1.1. Viewing a List of Remote Graphics	15-3

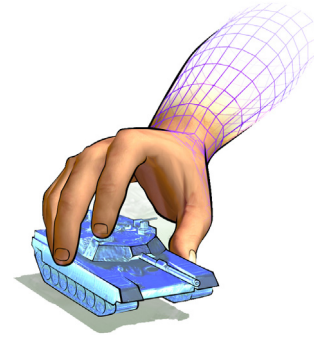
Appendix A Common GUI Controls and Behaviors

A.1. Dockable Panels and Toolbars	A-2
A.1.1. Docking and Undocking a Panel	A-2
A.1.2. Docking and Undocking a Toolbar	A-3
A.1.3. Displaying and Hiding Panels and Toolbars	A-3
A.1.4. Restoring the Layout of Toolbars and Panels	A-4
A.2. Using Tear-Off Menus	A-5
A.3. Viewing the Window in Full Screen Mode	A-5
A.4. Using Context-Sensitive Menus	A-6
A.5. Common VR-Forces Icons	A-6

MÄK Products Glossary

Index

VR-Forces Quick Reference Card



Preface

This manual is written for persons who will install VR-Forces® or use the VR-Forces application. The manual assumes that you are familiar with your operating system and that you know how to work in your computer's graphical window environment.

Please see release documentation for information about changes and updates to VR-Forces since this manual went to press.

How This Manual is Organized

The chapters are organized as follows:

Chapter 1, *Introduction to VR-Forces*, briefly describes VR-Forces and its features.

Chapter 2, *Installing VR-Forces*, explains how to install VR-Forces, set up the FLEXlm license manager, and localize the GUI.

Chapter 3, *VR-Forces Simulation Concepts*, provides the conceptual background for the simulation process. It will help put some of the discussion in this manual into context. It is even more relevant to *VR-Forces Scenario Management Guide*.

Chapter 4, *VR-Forces Visualization Concepts*, discusses the 2D and 3D interfaces and the concepts that underly their use and configuration.

Chapter 5, *Starting VR-Forces*, describes how to start VR-Forces and load terrain databases, and configure HLA time management. It also describes command line options.

Chapter 6, *Managing Objects* describes procedures common to entities, control objects, and overlay objects.

Chapter 7, *Viewing Entities and Entity Information* describes how to view entities.

Chapter 8, *Managing Entity Effects* describes how to display a variety of entity effect such as trailing effects, shadows, and sensor volumes.

Chapter 9, *Aggregate Entities* describes how to view aggregate entities.

Chapter 10, *The Observer and Observer Modes* explains what an observer is and the different ways observers can be configured.

Chapter 11, *Moving the Observer* explains how to navigate the 3D and 2D views by moving the observer.

Chapter 12, *Attaching the Observer* explains how to attach the observer to entities and objects. This is an alternative to manually moving the observer through the terrain.

Chapter 13, *Intervisibility* explains how to determine if entities can see other entities or areas of the terrain.

Chapter 14, *Changing How Terrain is Displayed* describes ways to change how terrain is displayed. These options provide different types of information about the terrain.

Chapter 15, *Remote Graphics* explains how to enable display of graphics generated by remote applications.

Appendix A, *Common GUI Controls and Behaviors* explains how to manage toolbars and other aspects of the VR-Forces GUI.

MÄK Products Glossary defines terms common to DIS, HLA, and MÄK products.

VR-Forces Quick Reference Card lists command line options, keyboard shortcuts, and navigation and view options.

VR-Forces Documentation

VR-Forces documentation is provided as manuals in PDF format, online help, and HTML class documentation. The PDF files are in the *.doc* directory. The VR-Forces documentation set is as follows:

- ♦ *VR-Forces Getting Started Guide* is a quick introduction to VR-Forces. It covers the basics of installing VR-Forces, running a scenario, and creating a scenario. It focuses on helping new users avoid common mistakes.
- ♦ *VR-Forces Documentation Center* is your central starting point for accessing the VR-Forces manual set. It has a documentation roadmap, tables of contents for all manuals, and a master index, including the TDB Tool, to help you find references to subjects that are covered in two or more manuals or locate the manual in which a particular topic is discussed. All table of contents entries and index entries are live links to the manuals.
- ♦ *VR-Forces Users Guide* describes how to install VR-Forces and configure license management. It explains how to use the VR-Forces graphical user interface to view simulations and how to manage aspects of VR-Forces that are not directly related to creating and running scenarios.
- ♦ *VR-Forces Scenario Management Guide* explains how to create and run scenarios.
- ♦ *VR-Forces Configuration Guide* explains advanced features for configuring performance and how to edit VR-Forces configuration files. It includes documentation of the Entity Editor, OPD Editor, and Scenario Merge tools.
- ♦ *VR-Forces Developers Guide* explains how to use the VR-Forces simulation and remote control APIs. It focuses primarily on the simulation engine.
- ♦ *VR-Forces Front-End Developers Guide* explains how to extend or modify the VR-Forces graphical user interface (GUI).
- ♦ *VR-Forces Migration Guide* collates API migration information for each release in the VR-Forces 3.x series.
- ♦ Online help. The VR-Forces front-end, the OPD Editor, the Entity Editor, and the TDB Tool have online help accessible from the Help menu.
- ♦ Class documentation. Classes are documented in linked HTML pages.
- ♦ *VR-Forces Release Notes*.

MÄK Products

VR-Forces is a member of the VT MÄK line of software products designed to streamline the process of developing and using networked simulated environments. The VT MÄK product line includes the following:

- ♦ **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 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.

- ♦ **MÄK RTI.** An RTI (Run-Time Infrastructure) is required to run applications using the High Level Architecture (HLA). The MÄK RTI is optimized for high performance. It has an API, RTIsby®, 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 a 2D and 3D views of a simulated environment.

You can create and view local entities, aggregate them into hierarchical units, assign tasks, set state parameters, and create plans that have tasks, set statements, and conditional statements. VR-Forces also functions as a plan view display for viewing remote entities 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 three end-user applications — VR-Vantage Stealth, VR-Vantage XR, and VR-Vantage IG — and the VR-Vantage Toolkit. VR-Vantage Stealth displays a realistic, 3D view of your virtual world. You can view this world from the inside of a simulated moving vehicle, or place the eyepoint at another moving or stationary location. The Stealth lets you switch rapidly among several predefined viewpoints while the simulation is underway.

VR-Vantage IG is a configurable desktop image generator (IG) for out the window (OTW) scenes and remote camera views. It has most of the features of the Stealth, but is optimized for its IG function.

VR-Vantage XR provides a common operational picture using the same 3D views as VR-Vantage Stealth, 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.

The VR-Vantage Toolkit is a 3D visual application development toolkit. Use it to customize or extend MÄK'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.

- ♦ **MÄK 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.
- ♦ **MÄK Plan View Display.** The Plan View Display (PVD) provides a 2D, “big picture” of a virtual environment. It shows simulated entities, such as vehicles, soldiers, and tanks moving over a 2D-map display.
- ♦ **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™ Server.** VR-TheWorld Server 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.

How to Contact Us

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

Telephone

Call or fax us at:	Voice:	617-876-8085 (extension 3 for support)
	Fax:	617-876-9208

E-mail

Sales and upgrade information:	info@mak.com
Technical support:	support@mak.com
VR-Vantage support:	vrv-support@mak.com

Internet

MÄK web site home page:	www.mak.com
License key requests:	www.mak.com/support/licenses.php
Product version and platform information:	www.mak.com/support/productversions.php
For the free, unlicensed MÄK RTI:	www.mak.com/products/rti.php
Support FAQ:	www.mak.com/support/faq.php

Post

Send postal correspondence to:	VT MÄK 68 Moulton St. Cambridge, MA, USA 02138

When requesting support, please tell us the product you are using, the version, and the platform on which you are running.

Document Conventions

This manual uses the following typographic conventions:

<code>Monospaced</code>	Indicates commands or values you enter.
Monospaced Bold	Indicates a key on the keyboard.
<i>Monospaced Italic</i>	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, (-h --help).
/	Indicates a directory. Since MÄK products run on both UNIX and Windows PC platforms, we use the / (slash) for generic discussions of pathnames. If you are running on a PC, substitute a \ (backslash) when you type pathnames.
<i>Italic</i>	Indicates a file name, pathname, or a class name.
sans Serif	Indicates a parameter or argument.
➤	Indicates a one-step 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 .
i	Indicates supplemental or clarifying information.
!	Indicates additional information that you must observe to ensure the success of a procedure or other task.

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

Hardware and Operating System Naming Conventions

Unless otherwise specified, references to UNIX include Linux, regardless of the type of computer it is running on. References to PCs refer to Intel processor-compatible computers running a version of Microsoft Windows.

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 popup 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

MÄK software products may use code from third parties. This section contains the license documentation required by these third parties.

Boost License

VR-Link, and all MÄK software that uses VR-Link uses some code which is distributed under the Boost License. All header files that contain Boost code are properly attributed. The boost web site is: www.boost.org.

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the “Software”) to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

libXML and libICONV

VR-Link and all MÄK software that uses VR-Link, links in libXML and libICONV. On some platforms the compiled libraries and header files are distributed with MÄK Products. MÄK has made no modifications to these libraries. For more information about these libraries please see the following web sites:

- ♦ The LGPL license is available at: <http://www.gnu.org/licenses/lgpl.html>
- ♦ Information about IconV is at: <http://www.gnu.org/software/libiconv/>
- ♦ Information about LibXML is at: <http://xmlsoft.org/>

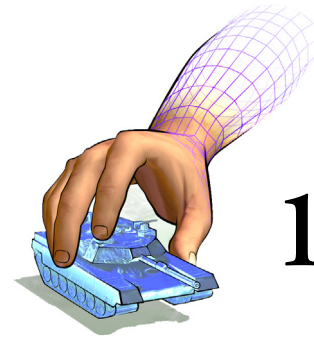
pThreads Library

VR-Exchange links with the pThreads win32 library. The library is distributed under the GNU Lesser General Public License (LGPL). MÄK has made no modification to this library. For information about the pThreads win32 library please see: <http://sourceware.org/pthreads-win32/>

EHS, PCRE, and PME

The MÄK RTI links with the EHS, PCRE, and PME libraries. These libraries are distributed under the GNU Lesser General Public License (LGPL). MÄK has made modification to these libraries only to allow them to compile on the supported platforms. For more information about these libraries please see the following web sites:

- ♦ EHS: <http://xaxxon.slackworks.com/ehs/>
- ♦ PCRE: <http://www.pcre.org/>
- ♦ PME: <http://xaxxon.slackworks.com/pme/index.html>



Introduction to VR-Forces

This chapter provides a high-level description of VR-Forces features.

Overview	1-3
Realistic Display of Vehicles and Terrain (3D View)	1-4
Create Complex Scenarios	1-5
Entity Types Supported	1-5
Mission Planning	1-7
Entity Tasks	1-8
Tactical Graphics	1-8
Terrain Agility and Composability	1-9
Flexible, Intuitive Graphical User Interface	1-9
Tactical Graphics Overlays	1-10
Behaviors	1-10
Multiple Attach Modes	1-10
Special Effects	1-11
Accurate Vehicle Positioning	1-11
Batch Mode Operation	1-11
Remote Control	1-11
The VR-Forces Toolkit	1-12
Plug-in Architecture	1-12
The B-HAVE Module for VR-Forces	1-13
Support for External Communications Servers	1-13
Helpful Utilities	1-14
Third-Party Software and Content	1-14
SilverLining	1-15
GL Studio	1-15
DI-Guy	1-16

SpeedTree.....	1-16
3D Models, Terrain, and Graphical Content.....	1-17
OpenSceneGraph.....	1-17
osgEarth.....	1-17
Distributed Simulation Standards Supported.....	1-18

1.1. Overview

VR-Forces provides both a set of APIs (the VR-Forces toolkit) for creating computer generated forces (CGF) applications, and an implementation of those APIs. VR-Forces is implemented using separate front-end (graphical user interface) and back-end (simulation engine) applications that allow tremendous flexibility for managing your simulation needs. (For information about how this works, please see [“The VR-Forces Program,”](#) on page 3-3.)

VR-Forces provides 3D and 2D views of your simulated world, integrated into one graphical user interface (GUI). The 2D view provides you with insights into a simulated battle by overlaying entities and information onto 2D views of tactical, strategic, and visual databases. The 2D view answers typical questions about the placement of forces and how the terrain might affect the engagement more easily than do three-dimensional displays. It is ideally suited for viewing simulations over large areas of open terrain.

The 3D view provides three-dimensional, perspective views of the terrain and entity models. It is ideal for scenarios that require situational awareness, simulation debugging, or after action review. It is particularly useful for placing entities with precise location and orientation and for working in terrains in which entities can be at multiple altitudes at a given X, Y coordinate, such as urban terrains.

The 3D view includes exaggerated reality (XR) features. Model scaling and additional model sets help to provide a common operational picture of the simulated world.

VR-Forces is compatible with both the Distributed Interactive Simulation (DIS) and High Level Architecture (HLA) simulation standards. It supports several database formats. VR-Forces is interoperable with other MÄK products, such as VR-Vantage Stealth and MÄK Data Logger.

The VR-Forces front-end is built with the VR-Vantage Toolkit. The VR-Vantage Toolkit is based on OpenSceneGraph¹, so you can leverage value-added plug-ins built by the OSG community and MÄK partners.

1. “OpenSceneGraph is an Open Source, cross-platform graphics toolkit for the development of high-performance graphics applications such as flight simulators, games, virtual reality and scientific visualization. It is based on the concept of a SceneGraph, providing an object-oriented framework on top of OpenGL.” (<http://www.openscenegraph.org/projects/osg/wiki/About/Introduction>)

[Table 1-1](#) lists VR-Forces features supported in the 3D and 2D views.

Table 1-1: VR-Forces features by projection

Feature	3D	2D
Terrain agility and composability	X	X
Attach modes	X	X
SpeedTrees	X	
GL-Studio cockpit displays	X	
DI-Guy characters	X	
SilverLining environment effects	X	
Track histories	X	X
View controls	X	X
Remote draw API	X	X (2D graphics only)
Trailing effects	X	
Detonation and muzzle flashes	X	
osgEarth streaming terrain	X	X
Observer movement modes		
Observer frame	X	
Level observer frame	X	
Laptop	X	
2D level observer frame		X
Model sets		
3D models	X	X
3D colorized models	X	X
2D icons	X	X

1.1.1. Realistic Display of Vehicles and Terrain (3D View)

VR-Forces provides a realistic display using terrain and moving-model geometry in the OpenFlight format. It includes many 3D vehicle models, some of which display movement of articulated parts, such as turrets and rotors. These models can change to show a damaged or destroyed state. You can provide your own models and map them to entity types. For details about terrain, please see Chapter 11, [Composing Terrains](#), in *VR-Forces Configuration Guide*. For details about models, please see Chapter 12, [Mapping Models and Effects](#), in *VR-Forces Configuration Guide*.

1.1.2. Create Complex Scenarios

VR-Forces lets you create complex scenarios to meet your simulation requirements. You can:

- ♦ Create and delete entities
- ♦ Create, delete, and edit tactical graphics
- ♦ Assign a sequence of tasks and set data requests to an entity as a plan
- ♦ Assign tasks and set data requests independently of plans
- ♦ Create global plans that run independently of any entity
- ♦ Observe the behavior of entities in response to their environment
- ♦ Create, save, and load scenarios that consist of:
 - Terrain database information
 - An order of battle
 - tactical graphics
 - A plan for each entity in the order of battle
- ♦ Manage hostility relationships
- ♦ Manage spot reports
- ♦ Determine line-of-sight between entities
- ♦ Aggregate and disaggregate, and expand and collapse aggregate entities
- ♦ Create overlays and add graphics to them.

1.1.3. Entity Types Supported

Using VR-Forces, you can interactively add individual entities to a simulation and you can aggregate them into higher echelon units. Entities provided include:

- ♦ Ground entities, such as:
 - M1A2 Tank
 - T-80 Tank
 - HMMWV Utility Vehicle
 - BTR-80 APC
 - M2A2 Bradley IFV
 - IEDs
 - Civilian vehicles, such as automobiles, police cars, and fire engines

- ♦ Air entities, such as:
 - F-16 Falcon
 - F/A-18 Hornet
 - A-10 Thunderbolt
 - SU-29 Frog
 - VTUAV Cypher
 - MiG-29 Fulcrum
 - AH-64 Apache
- ♦ Surface and subsurface entities, such as:
 - AAVC7A1 Landing craft
 - LCAC
 - Aircraft carrier
 - Guided missile frigate
 - SSBN
- ♦ Life forms, such as:
 - Dismounted infantry
 - Civilian
- ♦ Aggregate entities, such as platoons and companies.

Entity States

You can interactively modify aspects of an entity's state, including:

- ♦ Speed
- ♦ On-board stores such as fuel and ammunition
- ♦ Location
- ♦ Heading.

For more information about changing an entity's state, please see Chapter 7, *Issuing Set Data Requests*, in *VR-Forces Scenario Management Guide*.

You can modify entity parameters by editing the object parameter database and other configuration files. For more information, please see *VR-Forces Configuration Guide*.

Embarkation

Entities can embark on and disembark from other entities, for example, dismounted infantry can embark on a truck, HMMWVs can embark on an LCAC, and fixed-wing aircraft can land on and take off from aircraft carriers.

i

To use embarkation in HLA federations, you must use RPR FOM 2.0, draft 17 or later. For information about specifying the correct FOM, please see [“Starting VR-Forces,”](#) on page 5-3 and [“Specifying the RPR FOM Version,”](#) on page 5-35.

1.1.4. Mission Planning

You can create plans for entities that contain tasks, set data requests, and conditional statements. You can save plans and edit them.

Conditional statements specify that tasks will be executed only if a specific condition is true or false. Some of the conditions tested are:

- ♦ Is an entity in a specific area?
- ♦ Is an entity destroyed?
- ♦ Is an entity to the left or right of a phase line?

In addition to conditional statements that are checked only once (**If** statements), VR-Forces supports **When** statements and **While** statements.

Global Plans

Global plans exist independently of any entity. They allow you to assign tasks and set data requests to entities and execute other commands. In essence, they allow you to script the sorts of ad hoc actions that a VR-Forces could take through the GUI. This provides a great amount of flexibility to automatically modify the order of battle and tactical graphics in response to events.

For more information, please see Chapter 8, [Writing Plans](#), in *VR-Forces Scenario Management Guide*.

1.1.5. Entity Tasks

You can assign tasks to entities as part of plans or independently. The tasks supported include:

- ♦ Move to object
- ♦ Patrol along a route
- ♦ Patrol between two objects
- ♦ Follow an entity
- ♦ Fire for effect
- ♦ Take off and land.

Independently assigned tasks override the tasks in a plan, allowing you to respond to the course of a simulation as events transpire, without stopping to edit an entity's plan. For more information, please see Section 6.1, [“Assigning Tasks to Entities”](#), in *VR-Forces Scenario Management Guide*.

1.1.6. Tactical Graphics

Using VR-Forces, you can interactively create and name the following kinds of tactical graphics:

- ♦ Points
- ♦ Phase lines
- ♦ Lines
- ♦ Areas
- ♦ Obstacles.

Entities are aware of tactical graphics and can be assigned tasks such as following a route, or moving between points. VR-Forces can also determine if specified entities are in an area and this information can be used as part of mission planning. For more information, please see Chapter 5, [Overlays and Tactical Graphics](#), in *VR-Forces Scenario Management Guide*.

1.1.7. Terrain Agility and Composability

VR-Forces allows you to build your terrain at runtime using a variety of database and vector formats. It can load OpenFlight DTED, CTDB, MetaFlight, and GDB (MÄK terrain format) databases. It can load feature data from shapefiles and GDB files. You can also superimpose geospatial images such as GeoTiffs or other raster image files on the terrain. You can save these composed terrains in the MTF file format. For details, please see Chapter 11, *Composing Terrains*, in *VR-Forces Configuration Guide*.

VR-Forces supports the following database formats:

- ♦ The UTM projection and the Lambert conical conformal projection in CTDB C4B, C7B, and C7L databases
- ♦ OpenFlight UTM and flat earth databases
- ♦ MÄK Terrain Format (GDB)
- ♦ Digital terrain elevation data (DTED) databases
- ♦ Shapefiles
- ♦ Flat earth
- ♦ VMAP
- ♦ DFAD and DFD files.

VR-Forces supports the following coordinate systems:

- ♦ UTM
- ♦ Geocentric
- ♦ MGRS
- ♦ Latitude, longitude
- ♦ Database.

You can overlay raster images, such as CADRG, BMP, and geotiff files on terrain databases.

1.1.8. Flexible, Intuitive Graphical User Interface

VR-Forces's graphical user interface (GUI) allows you to manipulate VR-Forces from menus, dockable control panels, and the keyboard.

The VR-Forces display includes the following features:

- ♦ Manipulate the simulation map by zooming and panning
- ♦ Select and track entities
- ♦ Turn display features on and off
- ♦ Display a hierarchical, echelon view of entities and an embarkation view
- ♦ Display entity labels that include several types of information
- ♦ Display line-of-sight (intervisibility) for entities.

You can hide and display the control panels, keep them docked to the screen, or undock them to make more space available for the display window. The GUI controls can be hidden for full-screen demonstrations.

The GUI follows standard conventions for modern windowing systems. For additional details, please see Appendix A, *Common GUI Controls and Behaviors*. For descriptions of individual windows and dialog boxes, please see online help.

1.1.9. Tactical Graphics Overlays

VR-Forces supports tactical graphics overlays. Overlays mimic clear plastic overlays that a planner might use to draw graphics on top of a terrain map.

1.1.10. Behaviors

Entities understand how to interact with their environment. They can:

- ♦ Respond appropriately to other entities and objects, for example, avoiding entities, waiting for orders, and so on.
- ♦ Respond to detected targets, for example, deciding which of multiple targets to acquire, which weapon to fire, and what ammunition to use.
- ♦ Follow a route.

Path Planning

If a terrain database has vector data, such as roads, when you task an entity to move to a point or location, you can specify that it try to plan its path to follow any roads that lead to the destination.

1.1.11. Multiple Attach Modes

VR-Forces offers attach modes that let you:

- ♦ Manually control the observer's position and orientation. (The observer represents the user's view, or eyepoint, in the simulated world.)
- ♦ Follow an entity, maintaining a consistent positional offset, with a matching heading.
- ♦ Mimic a vehicle's position and orientation. You can place the observer inside the vehicle for a driver's-eye view, or fly outside of it, moving in tandem with the vehicle.
- ♦ Tether the observer to an entity, maintaining a consistent positional offset without changing the observer's heading.
- ♦ Automatically track a vehicle's movement from a fixed viewpoint, as if watching from a control tower.

For more information, please see Chapter 11, *Moving the Observer*.

1.1.12. Special Effects

VR-Forces includes features that help you understand the interaction between entities in a simulation and others that add a realistic tone to interactions. These features include:

- ♦ Fire and detonation lines
- ♦ Track histories
- ♦ Trailing effects such as footprints, tire tracks, and wakes.

1.1.13. Accurate Vehicle Positioning

VR-Forces can smooth the trajectories of moving vehicles to compensate for discontinuous positional data. The ground clamping feature can ensure that all ground entities are placed correctly on the terrain surface. For details, please see [“Trajectory Smoothing,”](#) on page 8-4 and [“Using Ground Clamping,”](#) on page 8-2.

1.1.14. Batch Mode Operation

Batch mode allows you to run a scenario multiple times or run several different scenarios, without direct action through the graphical user interface. This is useful for running unattended testing or to test the results of a scenario under differing conditions. You can record the results of batch mode simulations using the MÄK Logger, for later review. For more information, please see Section 1.5, [“Running VR-Forces in Batch Mode,”](#) in *VR-Forces Scenario Management Guide*.

1.1.15. Remote Control

You can control the VR-Forces observer remotely through view control messages, which can be generated by MÄK applications such as MÄK Data Logger, and by programs that use the Stealth Control Toolkit or the VR-Vantage Control Toolkit. For details, please see [“Controlling the Observer from Other Applications,”](#) on page 11-22.

1.2. The VR-Forces Toolkit

The VR-Forces toolkit lets you develop CGF applications. You can enhance the capabilities of the VR-Forces simulation engine and GUI, or integrate the simulation engine into your own user interface. The toolkit has several APIs. The Simulation API gives you control over:

- ♦ Behaviors
- ♦ Components
- ♦ Entity types
- ♦ Parameters
- ♦ Messages
- ♦ Resources
- ♦ Tactical graphics
- ♦ Plans
- ♦ Tasks.

The Remote Control API allows you to control a VR-Forces simulation engine from a remote application, such as a graphical user interface (GUI). The Terrain API lets you manipulate terrain databases.

The VR-Forces front-end is built with the VR-Vantage Toolkit. The toolkit is included with VR-Forces, allowing you to modify or extend the front-end. For details please see *VR-Forces Front-End Developers Guide*. For a complete discussion of the other APIs, please see *VR-Forces Developers Guide*.

1.2.1. Plug-in Architecture

You can modify the VR-Forces front-end and back-end by rebuilding the applications or by writing plug-ins. In fact, much of the basic functionality of the front-end is implemented using plug-ins rather than as part of the main executable. Plug-ins are dynamically linked libraries that VR-Forces can load at startup. The rationale for choosing which method to use is described in Section 1.1, “[Extending VR-Forces: Plug-ins or Stand-Alone Applications](#)”, in *VR-Forces Developers Guide*. You can manage the use of plug-ins in the front-end. For details, please see “[Managing Plug-ins](#),” on page 5-39.

1.3. The B-HAVE Module for VR-Forces

The B-HAVE (Brains for Human Activity in Virtual Environments) Module for VR-Forces is a VR-Forces plug-in that extends the movement models for lifeforms and ground platforms, as follows:

- ♦ Full 3D path finding
- ♦ Simplicity of use of basic agents (GoTo)
- ♦ Complexity and realism of behaviors (Hide, Flee, Wander)
- ♦ Complexity and realism of scenarios.

The B-HAVE Module is powered by Kynapse, the world's leading AI technology solution, from Autodesk, Inc. It is based on Kynapse's fundamental concepts of Topological Modeling for Path Finding, and Spatial Reasoning for Advanced Behaviors, all integrated within VR-Forces as advanced planning capabilities.

The B-HAVE Module offers VR-Forces users the ability to:

- ♦ Make entities move autonomously without specifying routes
- ♦ Make entities navigate through complex environments, indoors and outdoors, while avoiding collisions with obstacles and other entities
- ♦ Quickly populate a 3D environment with dozens of entities wandering in buildings, on roads, and on pavement
- ♦ Develop complex scenarios in which entities hide from each other and flee from threats.

The B-HAVE Module is a separately licensed product. It is not included with VR-Forces. Please see your MÄK salesperson for details.

1.4. Support for External Communications Servers

You can configure VR-Forces to use an external communications effects server to control the transmission of radio messages between VR-Forces entities. The communications effects server determines which entities are able to communicate with each other, and how long it takes a message to be transmitted from the originator of the message to the receiver of the message.

Any communications effects server that meets the defined interface can be used. For information about the interface, please contact support@mak.com. For more information, please see Chapter 9, *Using VR-Forces with an External Comms Effects Server*, in *VR-Forces Configuration Guide*.

1.5. Helpful Utilities

VR-Forces includes the following utilities for configuring and troubleshooting VR-Forces:

- The Entity Editor is a graphical user interface that lets you add and remove component systems from entities and specify common parameters. You can create new entities in the editor and can edit the Create menu. You can also create new simulation model sets.
- The Object Parameter Database (OPD) Editor is a graphical user interface for editing the object parameter database. Using the OPD Editor can speed up editing of the object parameter database and prevent syntactical errors that can occur when you use a text editor to edit parameters.
- The Scenario Merge tool lets you merge scenarios. This allows development teams to compose portions of a large scenario individually and easily merge them into one combined scenario.
- The MÄK Terrain Database Tool (TDB Tool) allows you to convert supported terrain databases to GDB format, optimize them for use with VR-Forces, and import vector data. It has a command line interface and a GUI. The GUI version lets you manually add vector features to databases.
- The vrfMsgDump utility displays the contents of VR-Forces messages discovered on the network.

1.6. Third-Party Software and Content

The VR-Forces front-end includes software and content licensed from third parties, including:

- SilverLining™: real-time sky and 3D cloud rendering from Sundog Software
- GL Studio®: interactive cockpit instrumentation and HMI from DiSTI®
- DI-Guy™: realistic human character animation from Boston Dynamics
- SpeedTree®: animated, 3D foliage from Interactive Data Visualization (IDV)
- 3D models, terrain, and graphical content from SimthetiQ, RealDB, TurboSquid, Terrex (now Presagis), and TerraSim
- OpenSceneGraph: an open source 3D graphics toolkit hosted at <http://www.openscenegraph.org>
- osgEarth: an open source streaming terrain plug-in by Pelican Mapping, at <http://osgEarth.org>.



The run-time and developers' rights for VR-Forces customers vary from vendor to vendor. Please read this section carefully to understand which rights are included with VR-Forces licenses, and which rights require additional third-party licenses.

1.6.1. SilverLining

VR-Forces uses SilverLining software and content to compute lighting and to render the sky, clouds, sun, and moon. SilverLining is developed by Sundog Software (<http://www.sundog-soft.com>).

Most VR-Forces customers do not need to buy a SilverLining license of any kind. In general, a VR-Forces customer has the right to use the SilverLining technology that is built into VR-Forces in any VR-Forces-based application (including custom applications built using the Toolkit). However, there are two exceptions:

- ♦ If you want to extend or change the way that MÄK has integrated SilverLining into VR-Forces by writing directly to the SilverLining API, then you need a SilverLining Developer's License.
- ♦ If you are building a commercial video game or other “mass-market” (consumer-level) commercial product on top of the VR-Forces API, and the resulting product includes the SilverLining functionality, then you need a SilverLining Developer's License.

SilverLining libraries, textures and other graphics resources are distributed in the VR-Forces package solely so that VR-Forces can use them. Use of SilverLining software or content outside of VR-Forces-based applications is not permitted by the VR-Forces license.

1.6.2. GL Studio

VR-Forces uses GL Studio software and content to render interactive cockpit instrumentation displays. GL Studio is developed by DiSTI (<http://www.disti.com>).

VR-Forces is delivered with several generic cockpit displays. You can use the built-in cockpit displays in any VR-Forces-based application (including custom applications built using the Toolkit), without a GL Studio license of any kind.

If you want to build custom cockpits using the GL Studio editor or GL Studio API, you need a GL Studio Developer's license. If you want to execute custom cockpits in VR-Forces-based applications (regardless of whether the custom cockpits are built by you or a third party), you need GL Studio Run-Time licenses.

GL Studio libraries and graphics resources are distributed in the VR-Forces package solely so that VR-Forces can use them. Use of GL Studio software or content outside of VR-Forces-based applications is not permitted by the VR-Forces license.

1.6.3. DI-Guy

VR-Forces uses DI-Guy software and content for human character animation. DI-Guy is developed by Boston Dynamics (<http://www.bostondynamics.com>).

VR-Forces comes with DI-Guy functionality built-in, and with a large set of DI-Guy characters, appearances, and animations. A VR-Forces customer does not need a DI-Guy license of any kind to use DI-Guy animated characters in a VR-Forces-based application (including custom applications built using the VR-Forces Toolkit).

However, you need a DI-Guy Developer's license if you want to:

- Develop new character types, or alter the geometry, appearances, textures, or motions of DI-Guy characters.
- Directly access the DI-Guy API to modify or extend the way that VR-Forces uses the DI-Guy API or access DI-Guy functionality that MÄK has not already integrated into VR-Forces.

DI-Guy libraries, models, textures, motion files, and other graphics resources are distributed in the VR-Forces package solely so that VR-Forces can use them. Use of DI-Guy software or content outside of VR-Forces-based applications is not permitted by the VR-Forces license.

For more information, please see Section 6.6.4, “[DI-Guy Animation](#)” and Section 7.1.1, “[Set DI-Guy Appearance](#)”, in *VR-Forces Scenario Management Guide*, and Section 2.3, “[Configuring DI-Guy Integration](#)” and Section 5.4.7, “[Editing a Lifeform's DI-Guy Character, Appearance, and Animation](#)”, in *VR-Forces Configuration Guide*.

1.6.4. SpeedTree

VR-Forces uses SpeedTree software and content for animated real-time 3D foliage and vegetation. SpeedTree is developed by Interactive Data Visualization (IDV) (<http://www.speedtree.com>).

A VR-Forces customer does not need a SpeedTree license of any kind to use the SpeedTree functionality that is built into the VR-Forces applications that MÄK delivers. You may build and run plug-ins to our out-of-the-box applications without a SpeedTree license, as long as your plug-ins do not need to directly access the SpeedTree API.

However, you need a SpeedTree Developer's license if you want to:

- Write a plug-in that uses the SpeedTree API directly to extend or modify the way that SpeedTree is integrated into VR-Forces.
- Use the VR-Forces Toolkit to develop new applications, and you want your applications to include SpeedTree functionality.

SpeedTree libraries, models, textures, and other graphics resources are distributed in the VR-Forces package solely so that VR-Forces can use them. Use of SpeedTree software or content outside of VR-Forces-based applications is not permitted by the VR-Forces license.

1.6.5. 3D Models, Terrain, and Graphical Content

VR-Forces includes a rich set of 3D models for vehicles, weapons, cultural features and urban clutter (signs, barriers, lampposts, and so on). It also includes several sample terrain databases to help you get started with VR-Forces, and to help demonstrate VR-Forces. Much of this content is derived from 3D data that we have licensed from third parties:

- ♦ Many of the high-quality vehicle models come from SimthetiQ (<http://www.simthetiQ.com>).
- ♦ Some models come from Real DB (<http://www.realdB.qc.ca/company>)
- ♦ Some of the middle-eastern building models and urban clutter objects that are used in our sample terrain databases were licensed from Turbosquid (<http://www.turbosquid.com>)
- ♦ The TerraSim_SampleUrban terrain database comes from TerraSim (<http://www.terrasim.com>)
- ♦ The HarrisAfghanVillage terrain database was created using source data from Harris Inc.
- ♦ The Berkeley Database was developed by Terrex (now Presagis) (<http://www.presagis.com>).

All of this content is distributed in the VR-Forces package solely so that VR-Forces can use them. Use of any of the VR-Forces models, textures, terrains, or other content outside of VR-Forces-based applications is not permitted by the VR-Forces license.

1.6.6. OpenSceneGraph

VR-Forces uses OpenSceneGraph for its underlying scene graph representation, rendering, file loading, and many other capabilities. OpenSceneGraph is an open source, cross-platform graphics toolkit for the development of high-performance graphics applications. The OpenSceneGraph source repository is maintained by Robert Osfield at <http://www.openscenegraph.org/projects/osg>. It is distributed under the OpenSceneGraph Public License (OSGPL), which is based on the GNU Lesser General Public License (LGPL).

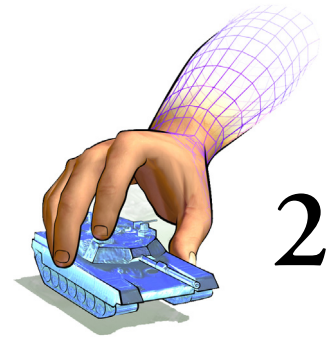
1.6.7. osgEarth

VR-Forces uses osgEarth to import streaming terrain elevation and imagery data. The data can be streamed from external servers and sources or from a directory on the computer running VR-Forces. osgEarth is an open source plug-in to OpenSceneGraph, maintained by Pelican Mapping at <http://osgEarth.org>. osgEarth is distributed under the GNU Lesser General Public License (LGPL).

1.7. Distributed Simulation Standards Supported

VR-Forces supports the 1.3 and 1516 specifications of the High Level Architecture (HLA) and the Distributed Interactive Simulation (DIS) protocol. VR-Forces has built-in support for the HLA RPR-FOM and can support other FOMs through the FOM Mapping feature. For information about FOM mapping, please see [“Specifying a FOM Mapper,”](#) on page 5-36. Please see *VR-Forces Release Notes* for a list of the versions of DIS supported.

VR-Forces supports time management for HLA exercises.



Installing VR-Forces

This chapter explains how to install VR-Forces on your computer. It also explains how to install other software required for use with VR-Forces.

Installing VR-Forces.....	2-2
Installing VR-Forces on Windows	2-2
Installing VR-Forces on a Linux System	2-3
Uninstalling VR-Forces	2-4
The VR-Forces Directory Structure.....	2-5
Installing and Setting Up the License Manager	2-6
Specifying the License Server	2-7
Installing an RTI.....	2-9
Installing the MÄK RTI	2-10
Localizing the Graphical User Interface.....	2-11
Translating Other Interface Files.....	2-14
Applying the Language Files.....	2-14
Merging Translation Files	2-14

2.1. Installing VR-Forces

This section explains how to install VR-Forces on a Windows or a Linux operating system. You must also install the license manager files. (For details, please see [“Installing and Setting Up the License Manager,”](#) on page 2-6.) For HLA, you must install an RTI. (For details, please see [“Installing an RTI,”](#) on page 2-9.)



To install the VR-Forces SDK, you must enter an installation code. Be sure you have received your code from your MÄK salesperson before you start the installation process.

2.1.1. Installing VR-Forces on Windows

Before you install VR-Forces, please read *VR-Forces Release Notes* to see if there are any special instructions for installation.

Installing VR-Forces from Optical Media

MÄK products are supplied on CDs or DVDs with a common installation interface.

To install VR-Forces:

1. Insert the MÄK Products disc into your optical media drive. If the autorun feature on your computer is enabled, the MÄK PC Products Installation window opens. The PC Products Installation window is similar to [Figure 2-1](#).

If the MÄK PC Products Installation window does not open, open the Windows Explorer. In the root directory for your optical media drive, double-click *makInst.exe*. The MÄK PC Products Installation window opens.
2. In the MÄK PC Products Installation window, click VR-Forces or VR-Forces SDK.
3. Follow the instructions of the installation program.

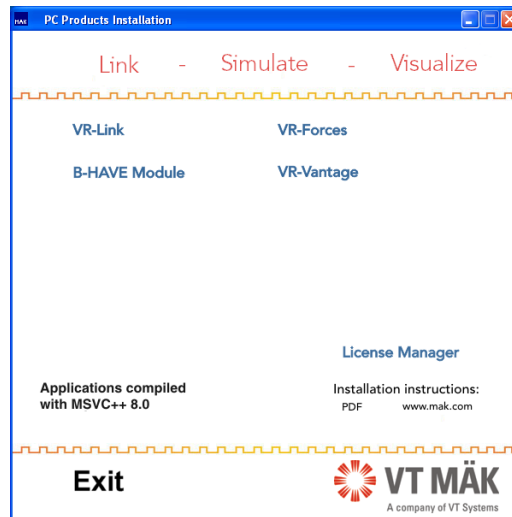


Figure 2-1. MÄK PC Products Installation window

Installing VR-Forces from a Downloaded File

If you downloaded VR-Forces, you received an executable installation file.

- To install VR-Forces from an executable file, run the executable.

2.1.2. Installing VR-Forces on a Linux System

Before you install VR-Forces, please read *VR-Forces Release Notes* to see if there are any special instructions for installation.

Installing VR-Forces from Optical Media

To install VR-Forces:

1. Mount the optical drive.
2. Insert the MÄK Products disc into your optical media drive.
3. Run:


```
./install
```
4. Follow the on-screen instructions. You can install VR-Forces into any directory for which you have write permissions.

Installing VR-Forces from a Downloaded File

If you downloaded VR-Forces, you received a compressed tar file.

To install VR-Forces from a tar file:

1. Create the directory in which you want to install VR-Forces.
2. Copy the tar file to the install directory.
3. Uncompress and untar the tar file:

```
gtar xzf vrforcesxx-os.tar.gz
```

where *xx-os* is the release, operating system, and other product release coding.



You must use `gtar` (from GNU) not the operating system's default `tar`.

2.2. Uninstalling VR-Forces

The VR-Forces application and the VR-Forces SDK have separate installers. Therefore, if you uninstall them, you must uninstall them separately. The uninstaller for the VR-Forces application is in the root directory. The uninstaller for the SDK is in *./SDK*. We recommend that you uninstall the SDK first. If you have installed the B-HAVE Module for VR-Forces, uninstall it before uninstalling VR-Forces.

2.3. The VR-Forces Directory Structure

The VR-Forces installation process creates a directory structure like the one shown in [Figure 2-2](#). On PCs, the default installation directory is *C:\MAK\vrforces4.0.*)

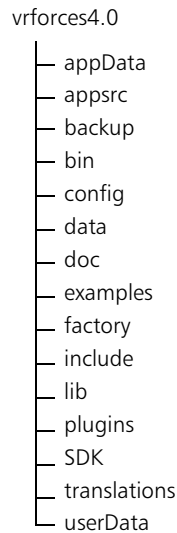


Figure 2-2. Directory structure of VR-Forces files

[Table 2-1](#) describes the contents of the directories installed by VR-Forces.

Table 2-1: Contents of VR-Forces directories (Sheet 1 of 2)

Directory	Directory contents
<i>./appData</i>	Schema, model definitions, default settings and other data required by the front-end.
<i>./appsrc</i>	Source files and project files for API developers.
<i>./backup</i>	Backup configuration files.
<i>./bin</i>	Product executables and DLLs.
<i>./config</i>	Configuration files. For a description of these files, please see <i>VR-Forces Configuration Guide</i> .
<i>./data</i>	The <i>./data</i> directory has several subdirectories that contain: ◆ Bitmaps and icons ◆ Simulation model sets ◆ Terrain database source files ◆ Visual model data.
<i>./doc</i>	VR-Forces documentation.
<i>./examples</i>	Sample code for toolkit users.
<i>./factory</i>	Backups of the application data and settings provided by MÄK.

Table 2-1: Contents of VR-Forces directories (Sheet 2 of 2)

Directory	Directory contents
<code>./include</code>	Header files for use by API developers.
<code>./lib</code>	Library files.
<code>./plugins</code>	Plug-in DLLs.
<code>./SDK</code>	Uninstaller for the VR-Forces SDK.
<code>./translations</code>	Files and an executable for localizing the GUI.
<code>./userData</code>	Scenarios, terrain files and other files that get created by users as they use VR-Forces.

2.4. Installing and Setting Up the License Manager

Before you can use a MÄK product, you must obtain a valid license file, install the 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 MÄK 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 MÄK product, you do not have to install it again. You just need to make sure you have licenses for your newly installed products.

The License Manager installer is included on MÄK installation media. It is separate from the product installers. The installers are also available for download at:

- Windows: <ftp://ftp.mak.com/out/MAKLicenseManager-win-setup.exe>
- Linux: <ftp://ftp.mak.com/out/MAKLicenseManager-linux-setup.tar.gz>

Complete installation and configuration instructions are included with the license manager installer. Instructions are also available at:
http://www.mak.com/products/install_license.php

Some customers use dongle licenses instead of running a license server. Instructions for using dongles are in the license manager documentation.

2.4.1. Specifying the License Server

The first time you run a MÄK application on a particular computer, the License Setup dialog box opens (Figure 2-3). It prompts you to enter the hostname of the license server and optionally, a port number.

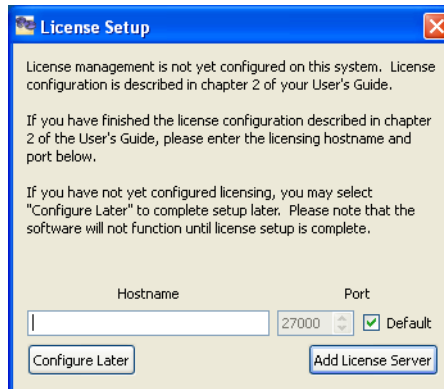


Figure 2-3. License Setup dialog box

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 MÄK applications until you set up license management.

If you know the hostname, type it in the **Hostname** box. Then click **Add License Server**. The application will start.

i

- ♦ If you are running MÄK 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 MÄK application.
 - ♦ You can clear the saved license configuration by deleting the cache file. On Windows it is *C:\Documents and Settings\user_name\Application Data\MAK\licenses<n>.xml*. On UNIX, it is *.mak/licenses<n>.xml*. (There may be more than one cache file, for example, *licenses1.xml* and *licenses2.xml*.)
-

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 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.



To view a video that illustrates this procedure on Windows, please go to <http://www.mak.com/html/setenvvar.htm>

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 products run on.

Windows

To add the MAKLMGRD_LICENSE_FILE in Windows:

1. Open the System Properties dialog box.

Windows XP:

- a. In Windows XP, on the Start menu, choose My Computer.
- b. In the My Computer window, under System Tasks, click View System Information. The System Properties dialog box opens.

Windows Vista or Windows 7:

- a. In Windows Vista, open the Control Panel. On Vista, select System and Maintenance.
 - b. Click System.
 - c. Click Advanced System Settings. The System Properties dialog box opens.
2. Click the Advanced tab.
 3. Click the Environment Variables button. The Environment Variables dialog box opens.
 4. Click the New button. The New System Variable dialog box opens.
 5. In the Variable Name field, enter MAKLMGRD_LICENSE_FILE.
 6. In the Variable Value field, enter `@server_name`, where *server_name* is the name of the license server.
 7. Click OK to back out of each dialog box and set the variable.

Linux

On Linux, you set environment variables in your *.cshrc* (or equivalent startup file). Set the variable similarly to the following example:

```
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 the following example:

```
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 MÄK 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.

i

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 MÄK products, please see the release notes for your MÄK application.

2.5.1. Installing the MÄK RTI

To install the MÄK RTI, follow the instructions in Chapter 2 of *MÄK RTI Users Guide*.

Configuring Your System to Use the MÄK 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 as follows:

- ♦ Linux — `LD_LIBRARY_PATH`
- ♦ Windows — `PATH`.

The MÄK RTI needs to know where to find the following 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 MÄK RTI

In many cases, you do not need an RTI server, such as the `rtiexec`, or a central application to use the MÄK RTI, however you can run the `rtiexec` if you want to. (It is required to use certain features of the MÄK RTI.) Be sure the license server is running, then start the applications, and they should be able to communicate. For more information, please see your RTI documentation.

2.6. Localizing the Graphical User Interface

You can localize your MÄK product's graphical user interface so that it uses languages other than English. To do this, you edit a resource file using the following procedure:

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 2-4).



Figure 2-4. Linguist settings dialog box

3. Select values for the source language, target language, and target region or country.
4. Click OK. The translation file is opened and a list of classes is displayed on the left side of the window (Figure 2-5).

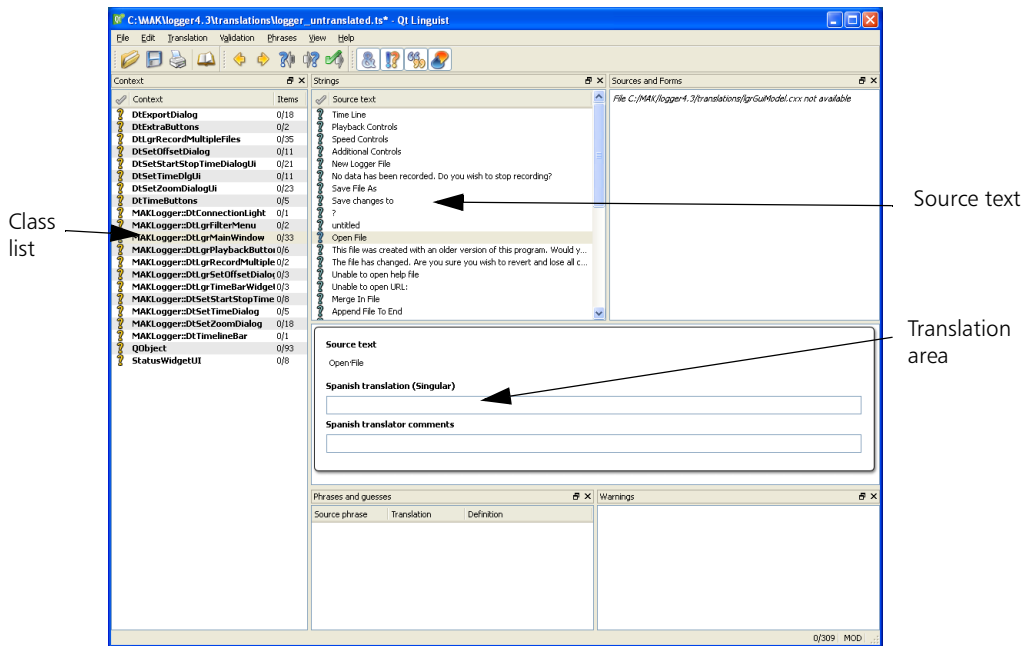


Figure 2-5. Linguist window

5. To translate text, you 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 just have to translate the text. In [Figure 2-5](#), *DtLgrMainWindow* is selected. The Strings window lists text that can be translated. Text that has not been translated has a question mark in the checkmark column.
6. In the Source Text list, select the word you want to translate. The text is displayed under Source Text in the translation area (the window below the source text list.) [Figure 2-6](#) shows the text *Open File* selected.

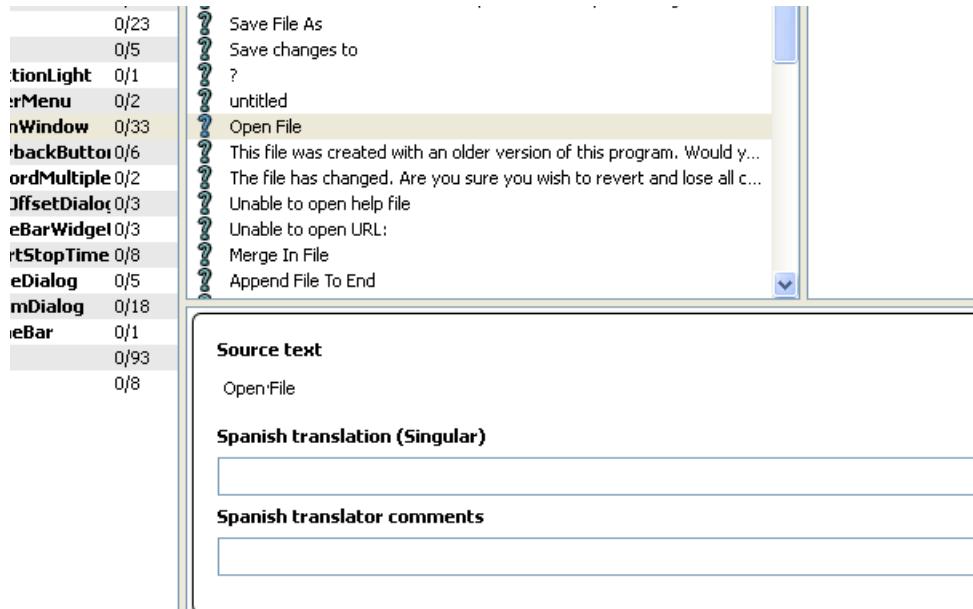


Figure 2-6. Text selected for translation

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 2-7). If you move to a different text string without clicking Done and Next, a yellow question mark is displayed 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.

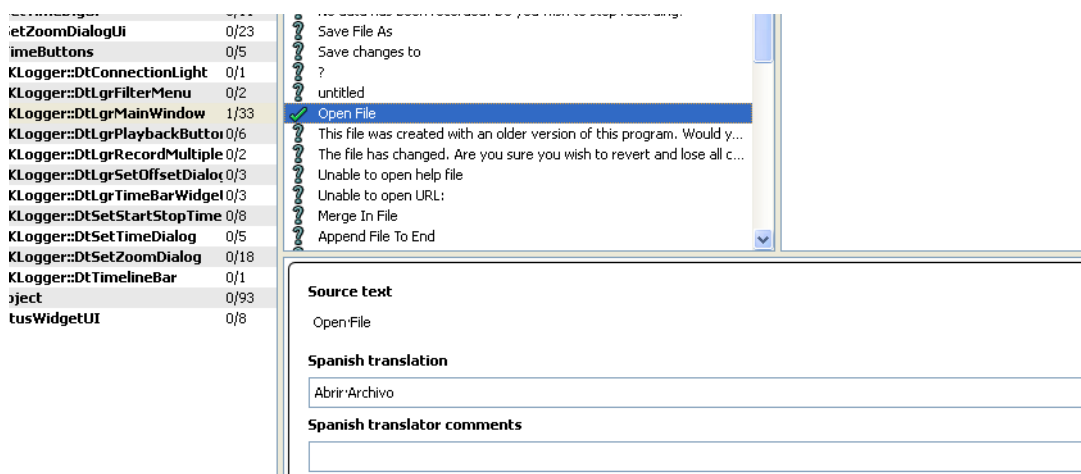


Figure 2-7. Completed translation for Open File

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.

2.6.1. Translating Other Interface Files

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

2.6.2. 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-Forces with the `-G locale` command line option, where *locale* is the two-letter country code for the local language.

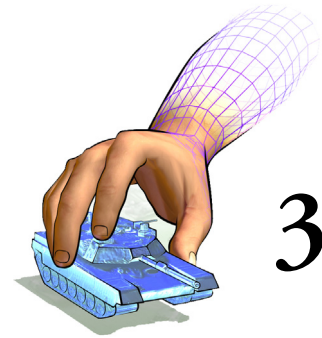
2.6.3. Merging Translation Files

If you localize a version of VR-Forces and upgrade to a newer version, you can merge your localized *.ts* file with the *.ts* file from the new version. Then you only have to translate new or changed text in the new version. You merge translation files with the *translationMerge* utility, which is in the *./bin* directory. The syntax is as follows:

```
translationMerge [-o] [-m] previous.ts current.ts new.ts
```

where:

- ♦ *previous.ts* is the name of the previously localized translation file.
- ♦ *current.ts* is the name of the translation file provided with the current release of VR-Forces.
- ♦ *new.ts* is the name to give to the merged file.
- ♦ `-o` specifies that the merged file removes obsolete messages.
- ♦ `-m` merges modules that have equal text strings.



VR-Forces Simulation Concepts

This chapter describes at a conceptual level the components of a VR-Forces simulation and how they work together. *VR-Forces Scenario Management Guide* explains how to create and run simulations.

The VR-Forces Program	3-3
Front-end and Back-end Concepts.....	3-3
How Front-ends and Back-ends Work Together	3-4
How VR-Forces Back-ends are Identified	3-5
VR-Forces Sessions	3-5
Coordinating Multiple Front-ends.....	3-7
Working with Multiple Back-ends	3-8
Scenarios	3-9
The Scenario File.....	3-9
Terrain Databases	3-9
The Order of Battle File	3-10
The Plan Files.....	3-10
The Object Map File.....	3-10
The Hostility Relationships File	3-10
The Batch Mode File.....	3-11
Objects	3-11
The Object Parameter Database	3-11
Local Objects and Remote Objects.....	3-12
Entities	3-12
How Entities are Identified.....	3-13
How Entities are Organized.....	3-15
How Entities Communicate.....	3-16
VR-Forces Uses a Component Architecture.....	3-17

Entity Behaviors and Tasks	3-19
Aggregate Entities	3-20
How Aggregates Are Organized	3-21
How Aggregates Move	3-22
How Aggregates Carry Out Tasks	3-23
Tactical Graphics	3-24
Points	3-25
Phase Lines	3-25
Routes	3-26
Areas	3-26
Obstacles	3-26
Tasks	3-26
Plans	3-27
Set Data Requests	3-28
Terrain Databases	3-28
Composing Terrains (and Creating MTF Files)	3-29
Coordinate Systems	3-30
MTD Files	3-31
Loading Supported Databases	3-31
Using Large Terrain Databases	3-32
Intervisibility (Line-of-Sight)	3-34
Representing and Managing Time in VR-Forces	3-35
Simulation Time and Exercise Time	3-35
Exercise Clock Modes	3-36
Using HLA Time Management	3-38
Interactions	3-38

3.1. The VR-Forces Program

Unlike most computer applications, which have one executable file, VR-Forces has two executable files: a front-end, or graphical user interface (GUI) and a back-end, or simulation engine. The front-end executable is `vrfGui`. The back-end executable is `vrfSim-protocol`, where *protocol* is DIS, HLA13, or HLA1516.

By separating the simulation engine from the visualization program, VR-Forces allows you to run multiple front-ends, back-ends, or both. If you run multiple front-ends, several VR-Forces users can work together to create and manage scenarios. Using session IDs, you can specify which front-ends control which back-ends. The front-ends also display entities received over the network from other simulations participating in an exercise. (You have a limited ability to interact with these remote objects. For more information, please see “[Local Objects and Remote Objects](#),” on page 3-12, Section 8.9, “[Writing Plans for Remote Entities](#)”, in *VR-Forces Scenario Management Guide*, and Section 8.12, “[Configuring VR-Forces Components on Remote Entities](#)”, in *VR-Forces Configuration Guide*.)

If you run multiple simulation engines, the work of simulating entities and tactical graphics can be shared by multiple computers, which should improve performance. If you save a scenario, VR-Forces records which objects are simulated by each back-end, so that the configuration can be reproduced the next time you run the scenario.

i

In this manual, most procedures apply to a user's interaction with the front-end. For the sake of simplicity, unless we are discussing a subject that applies specifically to either the front-end or back-end, we refer to VR-Forces in the sense of the combined interactions of both executables.

3.2. Front-end and Back-end Concepts

Although the separation of front-ends and back-ends, and the ability to run multiple instances of each adds to the power and flexibility of VR-Forces, you need to understand how the various executables interact to ensure that your scenarios are created and run smoothly. In this section, we describe:

- ♦ How front-ends and back-ends work together
- ♦ How back-ends are uniquely identified
- ♦ Use of sessions
- ♦ How to coordinate multiple front-ends
- ♦ How scenarios are loaded and saved when you run multiple back-ends.

3.2.1. How Front-ends and Back-ends Work Together

The VR-Forces front-end is an interface to the simulation engine. It communicates with the back-end by sending messages over the network. (This relationship exists even if you are not connected to a network.) It displays the state updates and interactions that the back-end generates and sends over the network. [Figure 3-1](#) illustrates loading a scenario and running a simulation.

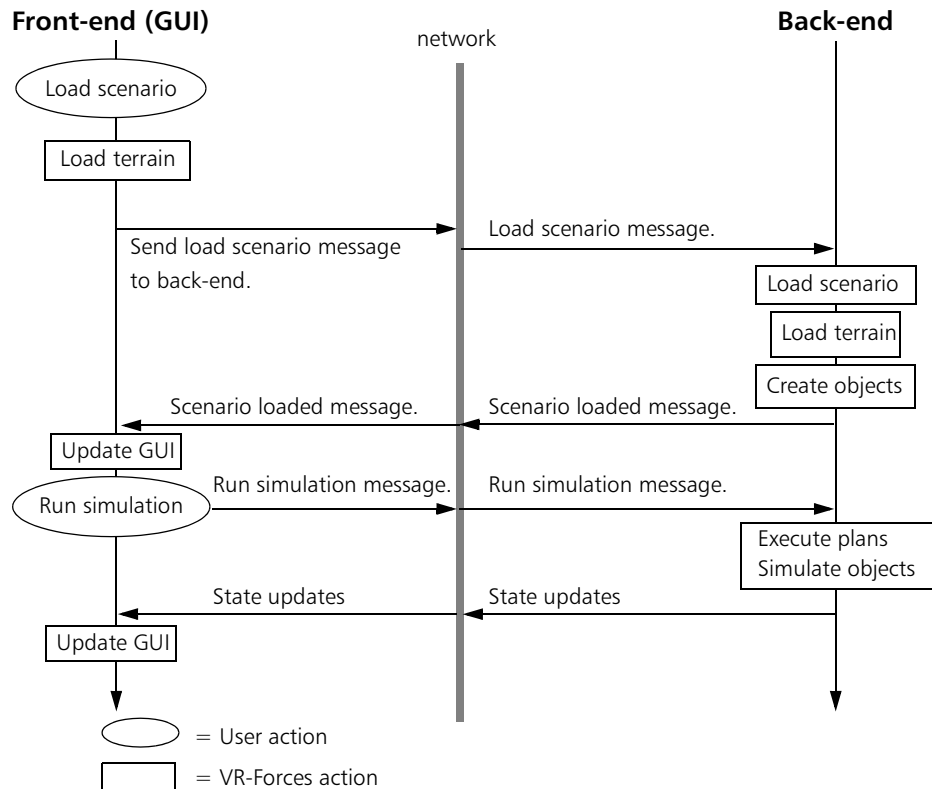


Figure 3-1. Front-end and back-end interaction

3.2.2. How VR-Forces Back-ends are Identified

If you run multiple back-ends, you can specify which back-end you want to use when you create entities and tactical graphics. This feature allows you to spread the simulation load across multiple PCs. However, to do this, each back-end must have a unique ID. VR-Forces back-ends are identified by their site ID and application number. Each back-end must have a unique site:application identifier. You assign the site:application identifier when you start an application. It is up to you to make sure that there are no duplicate identifiers. (For details, please see [“Starting VR-Forces,”](#) on page 5-3.)



The site ID and application number concepts are borrowed from DIS. They apply to VR-Forces when it is running in HLA as well.

3.2.3. VR-Forces Sessions

VR-Forces uses sessions to help manage the relationships between front-ends and back-ends. Back-ends are always part of a session. Front-ends can join a session or operate independently of one. When a front-end is joined to a session, it can control the back-ends in that session. When a front-end is not joined to a session, it cannot control back-ends. However, it can open a terrain and view exercises running on its port. In this case it is just a viewer.

Use of sessions helps ensure consistency among front-ends that are controlling a common set of back-ends. The session mechanism ensures that:

- ♦ All front-ends and back-ends in a session run the same scenario and use the same terrain
- ♦ When a front-end joins a session that is running a scenario, it automatically loads the correct terrain for the scenario
- ♦ If any front-end in a session tries to create a scenario or load a scenario, and a scenario is already running, the user is warned that this action will destroy the currently running scenario
- ♦ If any front-end in a session loads a scenario or creates a new scenario, all of the back-ends and other front-ends in that session take the same action.

Sessions provide the following additional benefits:

- ♦ They allow you to specify which front-ends control which back-ends. This feature could be useful, for example, if you want one set of VR-Forces applications to simulate friendly forces and another set to simulate opposing forces. If all applications were in the same session, participants could send orders to the entities in the opposing force. By using separate sessions, this is not possible.

In [Figure 3-2](#), the front-end with session ID 1 can control the back-ends with that same session ID. The front-end with session ID 2, controls the back-end with session ID 2.

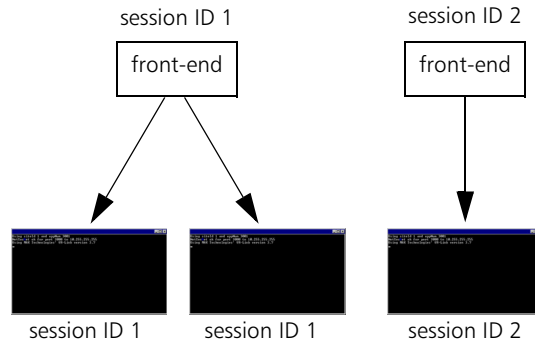


Figure 3-2. Using session ID to control back-ends

- ♦ Sessions allow you to load different terrain databases in different VR-Forces executables that are part of the same exercise. This feature may be helpful if you are participating in an exercise that covers a large geographical area, but only need to observe portions of that total terrain. Loading databases that cover a subset of the total exercise zone could improve the performance of your computer or the resolution of the terrain database.

In [Figure 3-3](#), the two sessions have loaded databases that cover different (but overlapping) portions of California, USA. The entire exercise would cover the terrain covered by two databases.

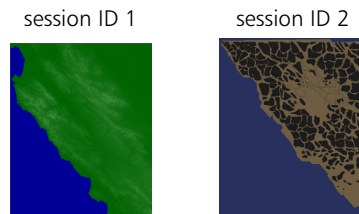


Figure 3-3. Using session IDs to load different terrain databases

If two different sessions use terrain databases that overlap, note the following possible sources of confusion:

- ♦ The front-ends in each session will see entities from the other session that are simulated in the common portions of the terrain. However, the front-end for one session will have no ability to control the entities that are simulated by the other session.
- ♦ If you save the scenario in one of these sessions, it only saves the entities simulated in the session. It does not save the entities from the other session, even though they may be visible in a front-end.
- ♦ If one of the sessions pauses or rewinds, the front-end in the other session will see the entities in the overlapping region pause or return to their origin.

3.2.4. Coordinating Multiple Front-ends

Figure 3-1 shows how a VR-Forces front-end sends messages to a back-end and displays the simulation messages sent out by the back-end. When you run VR-Forces with multiple front-ends, any of them can send control messages to the back-ends and all of them display the state updates. However, you would not want users at different front-ends to send conflicting or overlapping messages. For example, you would not want two different users to try to load the same scenario. Therefore, when you run multiple front-ends, bear the following points in mind:

- ♦ Only one front-end in each session can load a scenario or create a new one (they can all create the objects in the scenario).
- ♦ Any front-end can save a scenario. However, this could lead to multiple, unsynchronized versions of a scenario. It is probably best to reserve saving of scenarios to a specific front-end.
- ♦ Any object that you create in any of the front-ends is visible in the others and can be managed from them.
- ♦ You can close the scenario from any front-end.

VR-Forces provides some help managing multiple front-ends. For example, if one front-end loads a scenario, VR-Forces automatically asks the other front-ends to load the appropriate terrain. And if you close a scenario in one front-end, VR-Forces automatically prompts the other front-ends to close their terrains.

3.2.5. Working with Multiple Back-ends

As noted previously, you can distribute simulation of entities and objects among multiple back-ends. When you run multiple back-ends, bear the following points in mind:

- ♦ When you save a scenario created using multiple back-ends, the object map file ([“The Object Map File,”](#) on page 3-10) records each object and the site and application ID of the back-end on which it was simulated. When you want to replay that scenario, you should run a set of back-ends with the same site and application IDs. (If you do not, you can remap objects to the back-ends that you are running.)
- ♦ Multiple back-ends must all be started and ready when a scenario is loaded. There is no way for a required back-end to join an exercise after it has started and catch up to the current state of the exercise.
- ♦ If, during an exercise, a back-end drops out of the exercise, there is no good way for it to rejoin the exercise. If you load the scenario being run, it will be at the beginning, not at the point that the rest of the participants had reached.
- ♦ If you want a back-end to join an exercise after it has started, you must explicitly load the terrain database using `-T` command-line option. The back-end will not receive any order of battle data, but it can be used to create new entities and tactical graphics going forward in the simulation.
- ♦ When you save a scenario that uses multiple back-ends, the final scenario files are composed of information generated by each back-end and sent to the front-end, where it is combined. Therefore, if any of the back-ends has dropped out of the exercise or is otherwise unavailable, the scenario data will be incomplete.

Remapping Back-ends

The previous section lists the issues you should consider when creating, saving, and loading scenarios that use multiple back-ends. If you choose to run a scenario that requires multiple back-ends and some of the back-ends are not available, VR-Forces lets you remap objects from the missing back-ends to the available back-ends. You can remap back-ends automatically or manually.

If a scenario is missing its object map (*.omp*) file, you can remap the objects and create a new object map file. By default, in combined mode all the entities are remapped to the combined mode back-end. By default, in separate mode the remapping dialog box lists the missing back-end as “UNKNOWN”.

You might also want to manually remap back-ends to load balance a scenario. For more information, please see [“Load Balancing a Scenario,”](#) on page 1-9, in *VR-Forces Scenario Management Guide*.

3.3. Scenarios

Whenever you use VR-Forces, you work in the context of a scenario. A scenario consists of everything that needs to be specified to run a VR-Forces simulation. Each scenario has a scenario file (*.scn*) and the following supporting files:

- ♦ An order of battle file (*.oob*)
- ♦ A plan file (*.pln*)
- ♦ A global plan file (*.gpl*)
- ♦ An object map file (*.omp*)
- ♦ A terrain database for the front-end and back-end
- ♦ A hostility relationships file (*.hst*).

3.3.1. The Scenario File

The scenario file specifies the pathnames of a terrain database file, a simulation model set, and the other files required for a scenario. It specifies the following simulation-wide parameters:

- ♦ Time multiplier (for faster or slower than real-time operation)
- ♦ Random number seed (for repeatability)
- ♦ Frame advance mode
- ♦ Frame time interval
- ♦ Automatic reorganization
- ♦ Simulation time
- ♦ Exercise start time.

3.3.2. Terrain Databases

Each scenario is associated with a terrain database. A terrain database provides three-dimensional information about a geographic area and provides the environment within which simulated entities function. With the information in a terrain database, VR-Forces can calculate height, intervisibility, and other three-dimensional information.

A terrain database does not have to be stored in the same directory as a scenario. It is referenced by the scenario. This lets you use a terrain database, which can be quite large, in more than one scenario, without needing to make multiple copies of it. For more information about terrain databases, please see [“Terrain Databases,”](#) on page 3-28.

3.3.3. The Order of Battle File

The order of battle is a description of the local objects in a simulation. When you create a scenario, you populate it with entities. When you save or checkpoint a scenario, VR-Forces saves a record of each entity, including its state information and geographic location, to the order of battle file. The next time you run the scenario, it starts with all the entities in their saved location and state.

The order of battle file also describes the tactical graphics that have been defined for the scenario. Tactical graphics identify locations and areas in the terrain. You can place a basic set of objects on the terrain. For more information, please see [“Tactical Graphics,”](#) on page 3-24 and Chapter 5, *Overlays and Tactical Graphics*, in *VR-Forces Scenario Management Guide*.

3.3.4. The Plan Files

The plan file (*.pln*) saves the plans for the entities in a scenario. Plans are not saved separately for each entity; they are all in one file. Global plans are saved in a separate file (*.gpl*). For conceptual information about plans, please see [“Plans,”](#) on page 3-27.

3.3.5. The Object Map File

When you start VR-Forces with multiple back-ends, you can specify which back-ends simulate the various objects you create. The object map file records the back-ends on which objects are simulated. The next time you open the scenario, VR-Forces looks for the back-ends specified in the object map file. If any of the specified back-ends is not available, the front-end asks you if you want to remap objects to the back-ends that are present. You can run the simulation without remapping. However, the behavior of a simulation that is missing entities or objects is unpredictable.

If you forget the names of the back-ends you need for a scenario, you can view the object map file in a text editor and determine the site and application numbers for the back-ends you need to run.

3.3.6. The Hostility Relationships File

VR-Forces lets you configure hostility relationships. That is, for any force you can specify which forces it is hostile to. By default Friendly and Opposing forces are hostile to each other. If you change the hostility relationships, they are saved in the hostility relationships file (*.hst*).

3.3.7. The Batch Mode File

A batch mode file contains the information that VR-Forces needs to run scenarios in batch mode. Batch mode offers the following options:

- ♦ Run a particular scenario multiple times.
- ♦ Run a series of different scenarios (batches), each of which can be run multiple times.
- ♦ Run from the command line using one back-end. If a scenario was created using multiple back-ends, you cannot run it in batch mode using this method.
- ♦ Run from the VR-Forces front-end using one or more back-ends.
- ♦ Generate Logger Control messages to have the MÄK Logger record each simulation.

For each batch (one scenario run one or more times) within the batch file, you can:

- ♦ Specify the length of time the simulation is to run (in real time or simulation time)
- ♦ Specify the object parameter database to use
- ♦ Specify that the random number seed used be the value specified in the scenario file, a value specified in the batch file, or be computer generated.

A batch file has the file extension *.bsn*.

For more information, please see Section 1.5, “[Running VR-Forces in Batch Mode](#)”, in *VR-Forces Scenario Management Guide*.

3.4. Objects

All entities and tactical graphics are objects. The procedures for selecting, viewing, and organizing objects are fairly similar. However since their purposes and use are quite different, for the most part we discuss entities and tactical graphics separately.

3.4.1. The Object Parameter Database

VR-Forces creates objects based on parameter specifications in the object parameter database. In the object parameter database, objects are identified by an 8-digit enumeration based on the DIS Enumeration Document (*Enumeration and Bit Encoded Values for Use with Protocols for DIS Applications*). (The DIS enumerations have also been incorporated into the HLA RPR FOM.) In this manual, when we refer to entity or force type enumerations, we are referring to this set of enumerations. For information about the object parameter database and how to edit it, please see *VR-Forces Configuration Guide*.

3.4.2. Local Objects and Remote Objects

From the point of view of a particular VR-Forces back-end, a local object is any object that is simulated in that back-end. In that same back-end, a remote object is an object that is received over the network from an external application, such as another VR-Forces simulation engine, or a non-VR-Forces simulation. A given VR-Forces application participating in a networked exercise may know about a mixture of local and remote objects.

There are two categories of remote objects: objects simulated by VR-Forces applications, and objects simulated by applications other than VR-Forces. When VR-Forces learns about a non-VR-Forces remote object, it knows only what is transmitted in the standard DIS or HLA message. By contrast, VR-Forces remote objects share additional state information over the network, including a string name, an echelon ID (if the entity is organized), and a string label.

i

Since the VR-Forces front-end allows you to manage objects regardless of which back-end is simulating them, in the interest of simplicity, we refer to entities or objects simulated by a VR-Forces back-end, whether local or remote, simply as entities or objects and we refer to entities simulated by applications other than VR-Forces as remote entities or objects.

You cannot control remote objects from VR-Forces. However, you can attach VR-Forces components such as sensors and radios to remote objects and you can write plans that use these components. For more information, please see Section 8.9, “[Writing Plans for Remote Entities](#)”, in *VR-Forces Scenario Management Guide*, and Section 8.12, “[Configuring VR-Forces Components on Remote Entities](#)”, in *VR-Forces Configuration Guide*.

3.5. Entities

The players in a simulation are called entities. There are two broad classes of entities: single entities, such as a tank or a submarine, and aggregate entities, such as a platoon or a company. Since aggregates are entities, they have most of the characteristics of single entities as described in the following sections. Characteristics that are particular to aggregates are described in “[Aggregate Entities](#),” on page 3-20.

Entities can act in and respond to their environment. They execute tasks and they have behaviors. Each entity is part of a force-level organizational unit and many may be members of subordinate organizational units.

The VR-Forces application comes with a set of entity models that you can instantiate in a simulation. Application developers can create additional entities with the VR-Forces toolkit. You can also create new entity types in the Entity Editor. For information about creating new entities, please see *VR-Forces Configuration Guide*.

3.5.1. How Entities are Identified

VR-Forces identifies entities by name, echelon ID, object ID, and label. [Table 3-1](#) summarizes the characteristics of each type of identifier.

Table 3-1: Entity identifiers

Identifier	Characteristics
Name	♦ Must be unique ♦ Persists from exercise to exercise ♦ Set by user ♦ Limits on the length of the character string.
Echelon ID	♦ Must be unique ♦ Persists from exercise to exercise ♦ Managed by VR-Forces. Changes if an entity is organized into an aggregate.
Object ID	♦ Unique ♦ Does not persist from exercise to exercise ♦ No user control.
Label	♦ Does not have to be unique ♦ Persists from exercise to exercise ♦ Controlled by user ♦ No limit on character length.

Entity Names

A name is a text string that, by default, consists of the entity category, as defined in the object parameter database, and an integer that in combination with the category, makes the name unique. For example, if you create a new scenario and add two M1A2 tanks and two Apache helicopters, the default names are M1A2 1, M1A2 2, USA τ RW 1, and USA τ RW 2. You can edit an entity name to make it meaningful to you.

Entity names have the following restrictions on their length:

- ♦ Entities – 11 characters
- ♦ Aggregates – 10 characters in HLA exercises, 32 characters in DIS.



If you are using a character set that requires multiple ASCII characters to represent a visible character, entity and aggregate names will need to be correspondingly shorter.

Echelon IDs

An echelon ID is a text string that identifies the entity's place in the force hierarchy. For example, the hypothetical entities in the previous section would be part of Blue force, or Force 1. The echelon ID for M1A2 1 would be 1 M1A2, 1 Force. The syntax of echelon IDs and the details of entity organization are described in [“How Entities are Organized,”](#) on page 3-15. You cannot edit an entity's echelon ID. The echelon ID changes if you change the force hierarchy by adding, removing, or editing aggregate units.

Object IDs

Each entity has a unique object ID (global object designator). The global object designator is equivalent to the HLA object name in HLA, and the DIS entity identifier in DIS. The object ID is used to identify the entity internally and for network communications.

Entity Labels

An entity label is a text string that you can use to identify entities without the restrictions of uniqueness or character length that affect the other identifiers. For example, if Company A and Company C each have a 1st Platoon, you could not give both of them the name 1st Platoon, because names must be unique. However, you could give each of them the label 1st Platoon.

3.5.2. How Entities are Organized

All entities simulated by VR-Forces exist in the context of a hierarchy. At the top-most level, entities are grouped according to force ID (friendly, opposing, neutral, and so on). At lower levels, entities are grouped according to military organization, in which an entity has a designation in the hierarchy, such as 2nd Platoon, B Company. At each echelon level in the organization, an entity has a single superior entity. Aggregate entities contain one or more subordinate entities. The hierarchy's lowest level contains an individual entity.

Each entity has an echelon ID in the following format:

```
designator [category] [echelon], designator [category] [echelon], ...,  
force_number Force
```

where:

designator is a number or letter that differentiates the entity from other members of the echelon, such as 1 (M1A1), 2 (M1A2), and so on.

category is an optional descriptor for the entity, such as M1A2.

echelon is an optional descriptor for the organizational unit, such as "Platoon".

force_number is the force the entity belongs to, based on the force type enumerations. The force is the highest level in the hierarchy.

When you create a new entity, it exists only within the default force. Therefore its echelon ID would be similar to:

```
1 M1A2, 1 Force
```

If an entity becomes part of an aggregate, the aggregate information becomes part of the echelon ID, and the designator may change, for example:

```
2 M1A2, 2 Plt, 1 Force
```

The M1A2 from the first example is still an M1A2, but it has a new echelon designator: number two in 2 Plt. It is still in Force 1.

How Echelon IDs are Assigned

The category and echelon labels that make up an echelon ID, such as “M1A2”, “Plt”, and so on, are specified in the VR-Forces object parameter database and are associated with object-types. When you create an entity or aggregate of a certain type, the echelon ID category and echelon name associated with that type are assigned to the entity. You cannot change the labels interactively. Although you can change the labels by editing the object parameter database, please note the cautions associated with doing so. For information about changing echelon ID labels, please see Section 7.6, “[Changing the Echelon ID Category and Echelon ID Level](#)” in *VR-Forces Configuration Guide*.

Designators are assigned by VR-Forces. You have no direct control over them. When you create new entities, they are assigned designators sequentially, starting with one (1). If you aggregate a group of entities, the aggregate leader is assigned the designator 1, and the other entities in the aggregate are assigned successive designator numbers.

3.5.3. How Entities Communicate

VR-Forces includes a default communication model called simple-radio-comm-model. Using this simple communication model, radio messages always reach their intended destination (as long as the destination is on a reachable network), and messages are all passed instantly. The simple radio communication model has one parameter, which determines how the radio networks are set up. There are three connection-modes available:

- ♦ **aggregate** mode specifies that each aggregate has its own radio network. Messages sent from an entity can reach any entity that is in the same organization, at the same level (the entity’s peers in the organization), and can also reach the entity’s direct superior. All other entities are outside the network, and are unable to receive messages sent by the entity.
- ♦ **force** mode specifies that there is a radio network for each force in VR-Forces. Messages sent from an entity can reach any other entity of the same force.
- ♦ **all** mode specifies that all entities in VR-Forces are part of one large radio network. In this case, any entity can receive messages sent by any other entity.

For details about configuring radios, please see *VR-Forces Configuration Guide*.

External Communication Model

VR-Forces includes an external communication model that can communicate with an external communication effects server to determine how to deliver radio messages. For details, please see Chapter 9, [Using VR-Forces with an External Comms Effects Server](#), in *VR-Forces Configuration Guide*.

Custom Communication Models

You can create new communication models and install them in VR-Forces. This is useful if you want to define your own connectivity model, or write your own algorithms for determining if radio messages get delivered. For details, please see Section 9.5.1, “[Creating Custom Communication Models](#)” in *VR-Forces Developers Guide*.

3.5.4. VR-Forces Uses a Component Architecture

VR-Forces entities use a component architecture similar to that used in many robotics applications. There are three basic types of components supported: sensors, controllers, and actuators. Components communicate with each other through data ports and port groups, which may provide data as simple as a single number (such as a throttle value) or more complex data such as a list of entities that have been detected. The port groups make it possible to multiplex and prioritize multiple data sources to a single data recipient. For example, the automotive actuator of ground vehicles can receive inputs from a collision-avoidance controller, move-to-point controller, and follow-route controller, but it does not want inputs from all of those controllers at once.

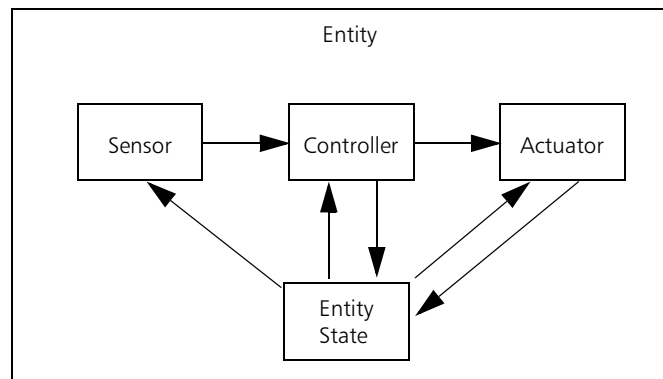


Figure 3-4. Entity component architecture

The component architecture supports any number of sensors, controllers, and actuators in a simulated entity. Components are specified in the object parameter database by using component descriptors. Component descriptors indicate the type of component that should be created. They can also provide parameter information to a component.

VR-Forces Configuration Guide discusses components in detail.

Sensors

Sensor components provide models of the simulated environment, which are used by controller components to make decisions and perform tasks. The simplest sensor might provide (simulated) ground truth, while more sophisticated ones could use complex models for sensing electromagnetic emissions, RADAR, or a cognitive model to simulate a soldier or crew member's perception of the simulated world. Sensor components can get information from:

- ♦ The virtual network
- ♦ From a terrain database
- ♦ By monitoring the state of the entity model
- ♦ Many other potential sources.

The use of sensors is optional.

Controllers

Controller components typically provide control inputs to actuator components (although this is not required). Controllers can use information provided by sensors to perform their functions, but this is not a requirement either. For example, the controller that implements the various forms of the Wait task neither requires sensor information, nor provides control inputs to any actuators.

Actuators

Actuator components:

- ♦ Provide the physical model of the entity being simulated
- ♦ Change the entity's state
- ♦ Interact with other simulated objects (such as by sending messages).

Actuators can use control inputs provided by controller components as parameters to its model each simulation frame. Actuators may:

- ♦ Send messages on the radio system
- ♦ Generate events on the virtual network, such as detonations
- ♦ Modify the state of the entity simulated: position, velocity, damage, and so on.

Actuators can also use data directly from a sensor, when modelled or filtered data is preferred over simulated ground truth for a model that does not require a controller.

3.5.5. Entity Behaviors and Tasks

A behavior is something that an entity knows how to do on its own. Behaviors describe how an entity reacts to a specific condition or situation or how it performs a task. A task is something that you tell an entity to do. An entity does not innately know that it should move to a point. However, if you task an entity to move to a point, it knows how to locate the point and how to move to it.

Behaviors can override task assignments. For example, if an entity is moving along a route and detects another entity in its path, the collision-avoidance behavior overrides the movement task until the entity successfully avoids the obstacle and can return to moving along the route.

VR-Forces creates models of the expected behavior of the entities we supply. These models simulate the behaviors you would expect from live participants and objects functioning in the real world. You can change the parameters for the behavior models. With the VR-Forces toolkit, you can change the models themselves and add additional models. For more information, please see *VR-Forces Developers Guide*.

To learn about changing behavior parameters, please see Chapter 8, [Configuring Object Behaviors](#), in *VR-Forces Configuration Guide*.

i

No semi-automated forces (SAF) system, such as VR-Forces, can completely emulate the behavior of a complex real world entity. The behaviors provided by VR-Forces handle the lower level activities necessary to complete a task, and allow you to focus on the higher level activities involved in assigning tasks and preparing plans.

3.6. Aggregate Entities

The aggregate entities in VR-Forces are multi-resolution models. Their components enable them to be simulated in either of two states:

- ♦ Disaggregated: Subordinates are simulated and published as individual entities ([Figure 3-5](#)). Aggregates in this state can be tasked and given plans, but usually pass responsibility for execution to their subordinates. For example, if you give a disaggregated aggregate a movement task, each member of the aggregate plots a path and follows it.
- ♦ Aggregated: Subordinates are not simulated separately. Tasks and plans are executed by the aggregate. For example, if you give an aggregated aggregate a movement task, the object representing the aggregate moves.

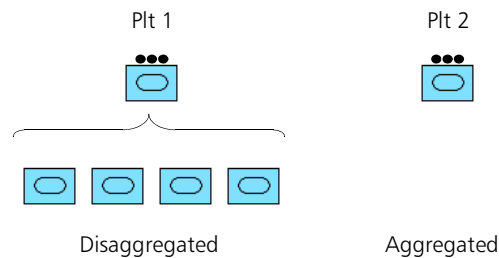


Figure 3-5. Disaggregated and aggregated entities

i

The disaggregated and aggregated states are not the same as expanded and collapsed aggregates (which are explained in [“Expanding and Collapsing Aggregates,”](#) on page 9-2.). Expanded and collapsed aggregates are simply different visual representations of the aggregate. Aggregated and disaggregated states are different simulation states.

An aggregate entity can support either of these states. At any given time, one subset of components and systems is enabled, and the other is disabled. At runtime, you can switch dynamically between these states.

You can configure VR-Forces so that aggregate state is determined automatically according to a set of rules, or to have aggregate state controlled manually.

The choice of whether or not to simulate aggregates in the aggregated or disaggregated state depends on your goals for the simulation. For example, if you need to support large numbers of entities, and do not want or need to simulate individual entity behavior, then you may want to simulate some or all of your aggregates in the aggregated state. Conversely, if you need to simulate the detailed behavior of subordinates, then you will want to simulate aggregates in the disaggregated state.



An aggregate in the aggregated state only simulates movement behavior. It does not model combat behavior. If you want them to model attrition, simulate aggregates in the disaggregated state. However, an aggregate in the aggregated state does keep track of its current resource and attrition state. If an aggregate later transitions to a disaggregated state, this state information is preserved.

For more details about the differences between the two aggregate models, please see Section 4.1, “[Introduction to Aggregates](#)”, in *VR-Forces Scenario Management Guide*.

3.6.1. How Aggregates Are Organized

The organization within a disaggregated aggregate is based on the echelon ID of each member entity. For example, in [Table 3-2](#), four unique entities (identified by name) are assigned designators in 2 Plt.

Table 3-2: Example of names and echelon IDs

Name	Echelon ID
Tank 1	1 M1A2, 2 Plt, 1 Force
Tank 2	2 M1A2, 2 Plt, 1 Force
Tank 3	3 M1A2, 2 Plt, 1 Force
Tank 4	4 M1A2, 2 Plt, 1 Force

Assignment of echelon IDs is handled automatically by VR-Forces when you create an aggregate.

Reorganizing Aggregates

Reorganization is the reassignment of echelon IDs to the members of an aggregate unit when a member entity gets destroyed. If reorganization is enabled, when a member of an aggregate unit gets destroyed, the destroyed entity is assigned the highest numerical ID in the unit, and the surviving members' echelon IDs are moved down numerically by one. For example, assume a platoon as shown in the leftmost table in Figure 3-6. If 2 M1A2 (currently the entity named Tank 2) gets destroyed, reorganization would result in the new echelon ID assignments illustrated in the rightmost table. The former 3 M1A2 is now 2 M1A2, former 4 M1A2 is now 3 M1A2, and the former 2 M1A2, is now 4 M1A2. Their echelon IDs have changed. Their names and object IDs have not.

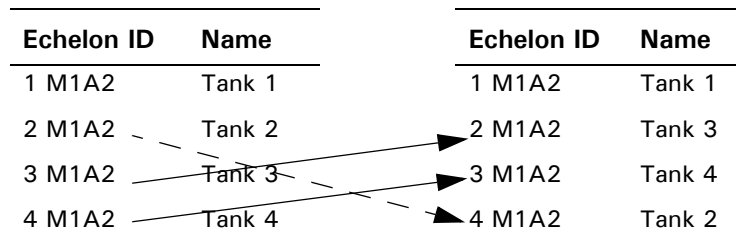


Figure 3-6. Reorganization of Platoon 2

You can configure VR-Forces to reorganize aggregates automatically, or you can reorganize manually. Manual reorganization does not require you to reassign each designator; you simply order the aggregate to reorganize with the Set Reorganize command.



Reorganization only applies to disaggregated aggregates.

3.6.2. How Aggregates Move

When an aggregated aggregate entity receives a movement command, VR-Forces calculates the path it must follow. Then it moves to the location similarly to any individual entity. The entity icon is placed at the center of the bounding box of the area that would be encompassed by the aggregate if it was disaggregated.

When a disaggregated aggregate entity receives a movement command, VR-Forces calculates the path the aggregate leader must follow for movement. Then it calculates parallel paths (taking into account the formation) for each individual member of the aggregate. The individual entities are then responsible for following these paths until the aggregate completes the movement task.

Closing Formation

VR-Forces can close up holes in formations that are created when an aggregate member is removed from a formation. When an aggregate is ordered to form up, it assigns formation positions to each of its member entities. Each position in the formation has an internal ID. If at some point one or more of the aggregate's members are destroyed, removed, or independently tasked, the aggregate reassigns the formation IDs for its list of member entities. Member entities which are gone, destroyed, or independently tasked, are given the least significant formation IDs. The active entities are moved up in the formation to fill the vacancies. Assignment to a different formation position has no effect on echelon designator.

The process of closing formation is internal to an aggregate. Formation position is not reported as part of entity information. The only way to determine an entity's formation position is to observe the formation as the aggregate moves.

Closing formation is an inherent feature of aggregates in the VR-Forces application. You do not need to do anything to put it into effect and you cannot turn it off.

Closing Formation Versus Reorganization

Closing the formation of the entities in an aggregate unit is not the same thing as reorganization. Reorganization changes the echelon designators of its members. Closing formation does not change the echelon designators of aggregate members, or the tasks an entity is assigned. It only affects the relative positioning of members within a formation.

3.6.3. How Aggregates Carry Out Tasks

Like other entities, each aggregate has a plan and you can assign independent tasks to aggregates (plans and tasks are discussed later in this chapter). When a disaggregated aggregate is given a task, it sends radio messages to its subordinates directing them to carry out their role in the task, for example, getting into formation and moving to a waypoint.

If an aggregate is in the aggregated state, it does not send task messages to subordinates. However, if while carrying out its task the entity disaggregates, it sends appropriate task messages to its subordinates to continue with the task execution. For example, suppose an aggregate is moving to a point and its path to the point passes through a disaggregation area. When it enters the disaggregation area, it would disaggregate, calculate routes for its subordinates, and send them Move Along Route tasks. When the aggregate exits the disaggregation area, it aggregates again and the tasks for its subordinates are abandoned.



Disaggregation areas are special control objects. If automatic aggregation and disaggregation is enabled, when an aggregate enters a disaggregation area, it automatically disaggregates. When it leaves the area it aggregates. For more information, please see Section 4.6.3, “[Using Disaggregation Areas](#)”, in *VR-Forces Scenario Management Guide*.

The precise way in which an aggregate sends out messages to its subordinates is part of the modeling of the aggregate that is built into the VR-Forces application.

3.7. Tactical Graphics

VR-Forces lets you add tactical graphics (also called control objects) to scenarios. It supports the following types:

- Waypoint – a specific X, Y, Z coordinate.
- Phase line – infinitely long line defined by two points.
- Route – a sequence of points in three-dimensional space.
- Area – a polygonal area. Areas are considered to have volume, so that air and subsurface entities can interact with them.
- Obstacle – a polygonal area on the terrain that entities try to avoid if obstacle avoidance is enabled.
- Disaggregation area – An area in which an aggregated aggregate automatically disaggregates if automatic disaggregation is enabled.
- Terrain Page-in Area – An area in which streaming or paged terrain is given priority for paging in.

Tactical graphics have names, which, like entity names, must be unique. New objects are assigned default names in the format `Point 1`, `Point 2`, `Route 1`, `Route 2`, and so on. You can specify an alternative name when you create the object. A graphical object's name can be up to 255 characters long.

Entities are aware of points, lines, and areas and can respond to them. For example, an entity can follow a route, or move to a point.

Tactical graphics are assigned to a force. Waypoints are displayed using their force color. To determine the force of other tactical graphics, examine the Force attribute in the Environmental State page of the graphic's Information dialog box. For more information, please see “[Displaying Information About an Object](#),” on page 6-10.

The following sections provide more information about the basic tactical graphic types.

3.7.1. Points

A point identifies an X, Y, Z coordinate in the terrain. You can command an entity to travel to a point and to travel back and forth between points. [Figure 3-7](#)) illustrates waypoints in the 3D and 2D views. In 3D, the bottom of the symbol points to the location of the waypoint. In 2D, it is centered. You can place several different kinds of points on overlays.

A point can have an altitude value. This allows you to send aircraft to waypoints or have them patrol between points.

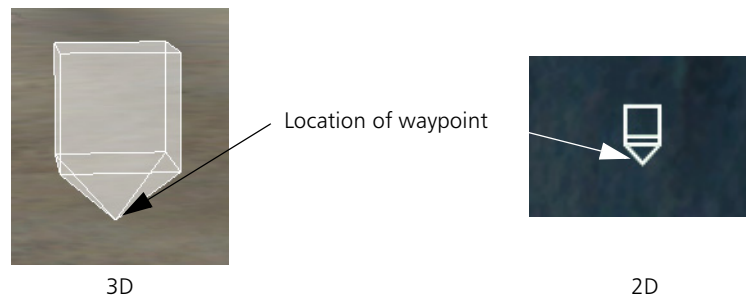


Figure 3-7. Waypoint symbol

3.7.2. Phase Lines

A phase line is defined by two points, but is considered to extend indefinitely in either direction and have infinite height. A phase line is used to test the location of an entity, for example: Is entity A to the left of phase line B? The left side of the line is determined relative to the direction in which the line is drawn. A phase line's label is placed at the beginning of the line. Therefore, you can always determine which side of a line is the left side. [Figure 3-8](#) illustrates two phase lines. The arrows are placed to the left of the line and indicate the direction of the line.



Entities cannot move along phase lines, so do not create a phase line if you want a two point route.



Figure 3-8. Phase line direction

3.7.3. Routes

A line is a sequence of points in three-dimensional space. Like phase lines, lines have a direction. A line's name is displayed at its beginning.

Entities can move along lines from one end to the other, and can patrol back and forth along them.



Do not create a two-point route if you want a phase line. VR-Forces cannot determine whether an entity is to the left or right of a route.

3.7.4. Areas

An area defines a polygonal area on the terrain. (This is not a terrain polygon, it is just a graphic.) An area is used to test the location of an entity, for example: Is entity A in area B?

3.7.5. Obstacles

Obstacles represent areas of the terrain that would cause an entity problems, such as a ditch or minefield. If obstacle avoidance is enabled, if a ground entity encounters an obstacle in its path, it tries to drive around the obstacle.

3.8. Tasks

VR-Forces entities can execute a variety of movement and tactical tasks. These tasks can be assigned in plans or through the Task menu. Tasks assigned outside of a plan are called independent tasks.

Some tasks are mutually exclusive with other tasks, for example, an entity cannot execute two different movement tasks at the same time. Other tasks, such as Send Text Message, can run concurrently with movement tasks. When you task an entity, if the new task is mutually exclusive with the current task, the entity immediately stops the current task and begins the new one. If the new task is not mutually exclusive with the current task, the entity executes the new task and continues with the previous task. For information about how to configure mutually exclusive tasks, please see Section 8.8, “[Configuring Task Execution Rules](#)”, in *VR-Forces Configuration Guide*.

If an entity has a plan, when you give it an independent task, it abandons the rest of the plan.

Independent tasks are useful for assigning tasks to entities without the need to edit a plan. You can assign them interactively, so simulation participants can respond immediately to events in a simulation by assigning new tasks to an entity as they might have to do in the field. Tasks are described in detail in Chapter 6, [Assigning Tasks](#), in *VR-Forces Scenario Management Guide*.

3.9. Plans

VR-Forces has two kinds of plans, global plans and entity plans. A plan can assign tasks and set data requests to an entity. Plans can also contain commands that create or delete entities and objects. The commands in a plan can be conditional, that is, they will be executed only if a condition is satisfied.

Global plans are created and executed independent of any particular entity. Commands sent to entities from a global plan affect an entity as if it received an independent task through the Task menu — they interrupt the current task, if any, and cause the entity to abandon its plan, if it has one.

An entity plan is a set of statements that order an entity to complete a sequence of tasks, change its state, or both. The entity carries out the tasks in order unless it is interrupted by an independent task or global command. When you open a scenario, it is in pause mode (unless you start it with the `-1` command line option). When you start the simulation, the plans start executing.

An entity owns its plan. Changing its name, echelon relationship, or force does not change its plan.

If an entity does not have a plan, it does not mean that the entity will have nothing to do. An entity may be assigned tasks by its superior in the Force hierarchy. It can also be assigned tasks interactively from the front-end. If you want an entity to take planned actions independently of those assigned by its superior, you need to add statements to its plan.

You can view an entity's plan at any time in the Plan window. The task that an entity is currently executing is highlighted. A simple plan might look like the following:

```
Move-To Waypoint:"red point"  
Wait-Duration Seconds-ToWait:900  
Move-To Waypoint:"white point"  
Set Engagement-Rules="Fire-at-will"
```



You can edit plans while a simulation is running. However, when you change an entity plan during a simulation, the entity starts executing the plan from the beginning, which may cause its activities to be out of synchronization with the other entities in the simulation.

Plans are discussed in detail in Chapter 8, [Writing Plans](#), in *VR-Forces Scenario Management Guide*.

3.10. Set Data Requests

Set data requests affect the state or behavior of an entity. They take effect immediately. Assignment of a set data request does not affect the task that an entity is executing. By contrast, if you assign an entity a task, it stops its current task and begins the new task. You can send set data requests from a plan or from the Set menu. The various set data requests are described in detail in Chapter 7, *Issuing Set Data Requests*, in *VR-Forces Scenario Management Guide*.

3.11. Terrain Databases

Each simulation takes place within the extents of a terrain database. A terrain database provides three-dimensional information about a geographic area and provides the environment within which simulated entities function. With the information in a terrain database, VR-Forces calculates height, intervisibility, and other three-dimensional information.

VR-Forces allows you to load terrain data in the following basic ways:

- Compose a terrain from elevation, imagery, and feature data
- Open an MTF or MTD file
- Open a terrain database in a supported format, such as FLT, DTED, or GDB.

The recommended work flow is:

1. Compose a terrain in the VR-Forces front-end and save it as an MTF file.
2. If your database requires processing that is not available in the front-end, but which is available in the MÄK Terrain Database Tool (TDB Tool), such as the ability to create point or areal features, export the terrain to GDB, process it in the TDB Tool, and add the GDB to the composed terrain.
3. Create a scenario and choose the MTF file as the scenario database.

The rest of this section explains the terrain loading options. The decision making process for choosing a terrain when you create a scenario is described in Chapter 1, *Creating and Running Scenarios*, in *VR-Forces Scenario Management Guide*.

3.11.1. Composing Terrains (and Creating MTF Files)

VR-Forces lets you compose terrains from a variety of supported data formats. You can combine elevation data (terrain patches), imagery, and feature data to create terrains.

VR-Forces offers the following basic methods for creating terrains:

- ♦ Compose a terrain by loading individual elevation, image, and feature data files. This method is best for relatively small terrains that do not require paging. It supports all of the coordinate systems that VR-Forces supports. For more information, please see Section 11.1, “[Creating a Composed Terrain](#)”, in *VR-Forces Configuration Guide*.
- ♦ Specify a set of files to load using a *.earth* file. Use this method to stream terrain data from local or remote sources. This method is best for terrains with large amounts of elevation or image data, which would benefit from paging. SpeedTrees do not work with this terrain format. For more information, please see Section 11.5, “[Connecting to Terrain Servers](#)”, in *VR-Forces Configuration Guide*.

When you compose a terrain, you can save it as a MÄK terrain file (MTF), which is essentially a description of all of the data used to visualize the simulated 2D and 3D environment. Saving a terrain lets you automatically load all of the data that you loaded manually when you first created it.

You can save terrains in a cached format that loads terrain data more quickly than if the terrain is created from the source data. For details, please see Section 11.9, “[Configuring File Caching](#)”, in *VR-Forces Configuration Guide*.

Elevation Data

VR-Forces imports elevation data as terrain patches. It supports the following formats:

- ♦ MÄK terrain database (*.gdb*)
- ♦ MÄK encrypted data format (*.medf*)
- ♦ MÄK Stealth format (*.mtl*)
- ♦ MÄK terrain project or terrain descriptor file (*.mtp*, *.mtd*)
- ♦ OpenFlight (*.flt*)
- ♦ MetaFlight (*.mft*)
- ♦ DTED (*.dt0*, *.dt1*, *.dt2*, *.dted*)
- ♦ CTDB 4, 7, 7L (*.c4b*, *.c7b*, *.c7l*)
- ♦ Shape terrain data (*.shp*)
- ♦ Terrapage (*.txp*)
- ♦ 3DS
- ♦ OBJ
- ♦ OSG
- ♦ LWO.

Images

You can import images in the following formats:

- ♦ GEO Tiff (.tif, .tiff)
- ♦ JPEG (.jpg, .jpeg)
- ♦ PNG
- ♦ RGB
- ♦ Windows bitmap (.bmp)
- ♦ GIF.

Feature Layers and Props

A terrain can have objects such as buildings and trees embedded in it. These objects cannot be manipulated in this format. However, VR-Forces lets you extract these types of objects from the terrain and render them independently as props.

Props are terrain elements that you can manipulate as separate objects. You can add props to a terrain by extracting them from a terrain patch or from a feature layer.

A feature layer is a file that contains source data from which you can add props to a terrain. A feature layer can be a MÄK Terrain Format (GDB) file, a shapefile (SHP), or DFAD data (DFD).

For more information, please see Section 11.8, “[Adding Props to a Terrain](#)”, in *VR-Forces Configuration Guide*.

3.11.2. Coordinate Systems

VR-Forces supports UTM, geocentric, flat earth, and Lambert Conformal Conic coordinate systems. There is no presumption that a particular database format, such as OpenFlight or DTED, will use a particular coordinate system.

The UTM Coordinate System

VR-Forces includes several databases that use Universal Transverse Mercator (UTM) coordinates. UTM databases are well-suited to two-dimensional views and simulations on a relatively small geographic area. However, their accuracy decreases as you move away from the origin, so they are not well-suited for exercises that need to simulate over areas greater than one UTM zone.

There are several types of UTM systems. In general, the earth is divided into 60 zones generally 6 degrees wide in longitude. These zones may be further subdivided into quadrangles by using letter designations from south to north. This complicates references to UTM. For example, standard offset could be used with or without the quadrangles (which affects the northing portion of the coordinate), non-offset does not use the quadrangles, and Milgrid requires the quadrangles.

More specifically, UTM coordinates define two dimensional, horizontal, positions. UTM zone numbers designate 6 degree longitudinal strips extending from 80 degrees South latitude to 84 degrees North latitude. UTM zone characters designate eight degree zones extending north and south from the equator. There are special UTM zones between 0 degrees and 36 degrees longitude above 72 degrees latitude and a special zone 32 between 56 degrees and 64 degrees north latitude.

VR-Forces uses the following definition:

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. When in UTM mode (the default), VR-Forces uses an offset UTM coordinate system, one in which coordinates are expressed with respect to an origin located at the latitude and longitude specified in a configuration file. Typically, this will be the location of the origin of your terrain database.

Geocentric Terrain Databases

To support large play areas, simulations can use geocentric coordinates. In the geocentric coordinate system, the origin is at 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 through the North Pole.

3.11.3. MTD Files

An MTD file is a descriptive file created by the MÄK Terrain Database Tool (TDB Tool). The TDB Tool is a utility included with VR-Forces. It allows you to customize the parameters used to convert terrain databases to MÄK Terrain Format (GDB) format. It also allows you to add vector data to databases and to edit the resulting GDB terrains. The TDB Tool creates MTD files for each database it creates.

3.11.4. Loading Supported Databases

MTF and MTD files reference elevation, image, and feature files that VR-Forces processes and loads when you load a scenario. However, you can also load terrain databases directly into VR-Forces. They will be converted to the VR-Forces internal format using default values. This process is less efficient than loading MTF or MTD terrains, may take more time, and must be repeated each time you load the terrain as part of a scenario or in the front-end.

If you load a terrain directly, you can still save it to MTF or GDB.

3.11.5. Using Large Terrain Databases

VR-Forces can support large databases, where large might mean a high polygon count, a large geographic area, or both. VR-Forces supports two methods for dealing with large databases – terrain paging and streaming terrain. Both approaches have a common goal – providing access to large amounts of data without loading all of that data into memory at any one time.

VR-Forces cannot arbitrarily page or stream terrains. It can page terrains that have been designed for paging and can stream terrains that are accessed through a terrain server, such as VR-TheWorld Server. For details about support for paged terrains, please see Section 11.6, “[Loading MetaFlight Terrains](#)”, in *VR-Forces Configuration Guide*. For details about connecting to terrain servers, please see Section 11.5, “[Connecting to Terrain Servers](#)” and Section C.2, “[Adding a Terrain Server](#)”, in *VR-Forces Configuration Guide*.

Terrain Paging

Terrain paging breaks a database up into multiple cells which can be loaded independently. By comparison, a non-pageable terrain must be loaded in its entirety. When VR-Forces loads a paged terrain, the front-end loads the terrain at whatever level of detail is required by the zoom level. The simulation engine always loads terrain at the highest level of detail. Therefore, if you rapidly zoom into a section of the terrain in the front-end, there may be a delay in correlation between entities and the terrain until the front-end catches up to the proper level of detail.

The simulation engine does not load any terrain until it actually needs the terrain to support the objects in the scenario. For example, if you create a new scenario and specify a paged terrain, the simulation engine does not load any terrain until you create an entity or object that needs to be placed on the terrain. Then it only loads the terrain page required for the object.

When you run a scenario, as entities move, the simulation engine tries to predict which terrain pages it will need to support the entities and tries to page them in before the entities need them. If the total polygon count required to page in required terrain exceeds the maximum polygon count configured, VR-Forces pages out (removes) unused pages. The simulation engine pages out the least recently used data.



Predictive terrain paging is done in real time, not just when a scenario is running. For example, when you load a scenario, all required terrain is paged in and when you add objects during scenario creation, terrain is paged in.

You can also manually page in terrain using terrain page-in areas. For details, please see Section 11.7, “[Manually Paging-In Terrain](#)”, in *VR-Forces Configuration Guide*.

Paging is done based on the following priorities:

- ♦ If an entity is moving, paging in terrain that it needs is a higher priority than predictive paging.
- ♦ Blocking terrain calls have a higher priority than non-blocking calls. (A blocking call is one that prevents the simulation from progressing until the call is completed. For more information, please see the `assertOnBlockingTerrainCalls` parameter in [Table 2-1](#), in *VR-Forces Configuration Guide*.)
- ♦ Paging out terrain to recover memory is a higher priority than paging in terrain. The need to page out is calculated based on the polygon count of the terrain, not on its memory usage.

VR-Forces can display information about paged terrains that can help you understand how your terrain is performing and why objects may be interacting with it as they do. For details, please see Section 11.11, “[Displaying Information About Paged and Streaming Terrains](#)”, in *VR-Forces Configuration Guide*.

Terrain Streaming

The term terrain paging is used in reference to static, local terrains. Another way to load terrain data for a large area is to stream it from a terrain server. The idea is similar to terrain paging. Only the data you need at a particular location is loaded. The difference is that a terrain server can serve many computers at a time and normally holds much more data than a static local terrain. It can stream elevation data, imagery, map data, and feature data. It may have data at multiple levels of resolution for a particular portion of the terrain. VR-Forces supports connection configurations to terrain servers, including MÄK's VR-TheWorld Server.

3.12. Intervisibility (Line-of-Sight)

You can display intervisibility (line-of-sight or LOS) between points, within areas, and between a point and all entities in a defined area. Intervisibility displays can be transient or permanent. Permanent intervisibility displays are objects. You can resize them and move them around the terrain. Intervisibility objects can be tied to a particular entity.

You can display range and heading data for point-to-point intervisibility lines.

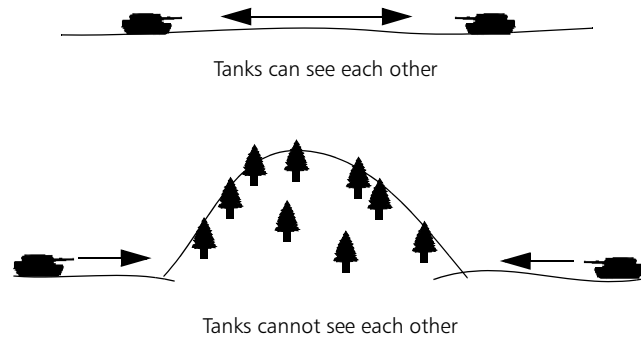


Figure 3-9. Intervisibility

You can configure parameters for intervisibility objects, such as the height above the terrain from which intervisibility is calculated. You can also change the colors used to display intervisibility. For procedures, please see Chapter 13, [Intervisibility](#).

i

- ♦ By default, intervisibility between entities is calculated from the center of their bounding volumes. The VR-Forces target detection sensor uses an offset, which is often above the center of the entity. Unless you change the Origin LOS Elevation value in the Intervisibility Options dialog box to use the same offset as the target detection sensor, entities may shoot at targets for which they do not appear to have line-of-sight.
 - ♦ Linear and areal features do not block line-of-sight. Only point features block LOS.
-

3.13. Representing and Managing Time in VR-Forces

VR-Forces employs the concepts of simulation time and exercise time. In HLA federations, it may be affected by federation time. These concepts also have a relationship to wall-clock, or real time. The following sections discuss how these concepts interrelate.

3.13.1. Simulation Time and Exercise Time

VR-Forces maintains an exercise clock. The clock keeps track of current elapsed simulation time and simulated exercise time. Elapsed simulation time is the amount of time that has passed since the simulation started, and is expressed in days, hours, minutes, and seconds in the format 00:00:00:00. Simulated exercise time is the simulated time of day, and is expressed in days, hours, minutes, and seconds in the format 00:00:00:00.

You can set the starting simulation time and exercise time for new scenarios. Application developers can manipulate simulation time. Unless a local developer tells you otherwise, simulation time starts at 00:00:00:00. Exercise time starts at 01:00:00:00.

As illustrated in [Figure 3-10](#), elapsed simulation time starts at 00:00:00:00 and represents the total amount of time since the simulation began. Simulated exercise time starts on the first day at some specified time of the 24 hour clock and continues to represent the day of the exercise and 24-hour time.



When you use the days, hours, minutes, seconds format for exercise time, days start at one, not zero. Zero seconds exercise time is 01:00:00:00.

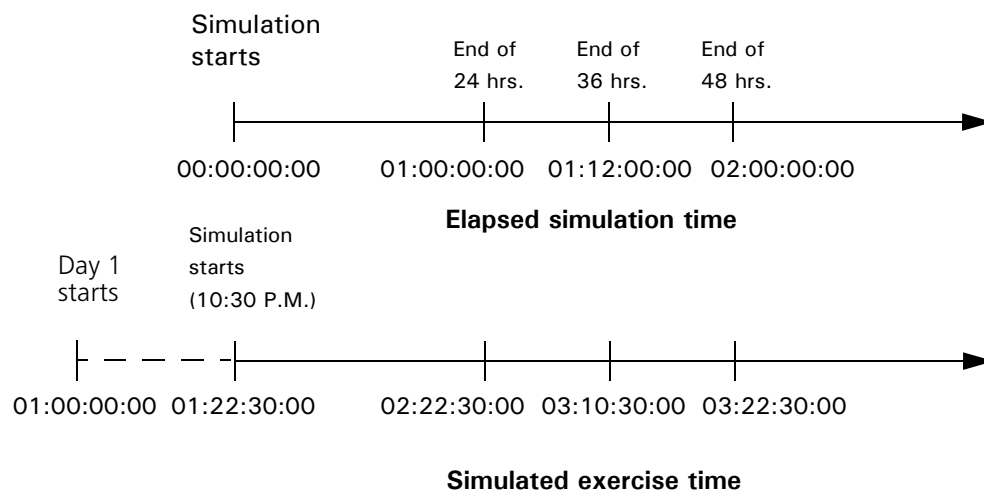


Figure 3-10. Elapsed simulation time compared to exercise time

3.13.2. Exercise Clock Modes

The exercise clock supports the following modes:

- ♦ Variable-Frame Run-To-Complete mode advances simulation time by the amount of time passed since the last time the exercise clock was ticked. This mode is typical for distributed, interactive simulations. It does not provide repeatable results. Do not use this mode in time-managed HLA federations.
- ♦ Fixed-Frame Best-Effort mode advances simulation time by a fixed amount each frame, even if a frame takes longer than the fixed amount to compute. If the simulation takes less than the fixed frame time to compute, it waits until the frame time has elapsed before starting another.

This mode is suitable for viewing or interactive use if your hardware can handle the frame rate. It is suitable for distributed use only in time-managed HLA federations.

- ♦ Fixed-Frame Run-To-Complete mode advances simulation time by a fixed amount each frame, even if a frame takes longer than the fixed amount to compute, just as in Fixed-Frame Best-Effort mode. However, if the simulation takes less than the fixed frame time to compute, it does not wait for the remainder of the frame time to elapse before starting the next frame.

This mode is most useful for situations where you want a simulation to run with internal consistency and high fidelity, and want it to run to completion, but do not need to observe the simulation. So, for example, you might run a simulation overnight and view the results the following day. Fixed-Frame Run-To-Complete mode is not suited for interactive use. It is suitable for distributed use only in time-managed HLA federations.



If you want to change the default clock mode, you must use the correct syntax. For information about how to set the exercise clock mode, please see Chapter 3, *Scenario Files* in *VR-Forces Configuration Guide*.

Repeatability

Repeatability is the ability to get the same results from a simulation each time you run it. Simulations that are run interactively or across a network are not repeatable, because you cannot guarantee consistent behavior by participants or the network.

For repeatable results:

- ♦ Use either Fixed-Frame Best-Effort mode or Fixed-Frame Run-To-Complete mode.
- ♦ Run with just one back-end.
- ♦ Do not interact with the scenario while it is running.
- ♦ Use the same random number generator seed for each iteration of the simulation.
- ♦ Do not use real time to measure the length of batch runs. Batches must be specified using simulation-run-duration.
- ♦ Do not use asynchronous I/O in DIS. (It uses another thread.)
- ♦ Do not use the path planning method for ground entity movement. (It runs in a separate thread.)
- ♦ Configure components with the `tick-period-uses-real-time` parameter set to False. For more information about this parameter, please see Section 5.4.5, “[Ticking Components](#)”, in *VR-Forces Developers Guide*.

Running a Simulation Faster or Slower than Real Time

VR-Forces supports simulation at rates faster or slower than real-time by applying a multiplier to the tick time.

i

- ♦ Do not run your simulation at faster or slower than real-time if you are interacting with other real-time simulations.
 - ♦ Running a simulation faster than real-time can cause performance of entity models to degrade.
-

3.13.3. Using HLA Time Management

In HLA, use of time management allows the rate at which a federate proceeds to be controlled by another federate. Federates being controlled by other federates are said to be constrained. Federates that control other federates are time regulating. A federate can be both time regulating and constrained. It can also be unconstrained.

The main reasons why you might want to enable time management for VR-Forces are to run VR-Forces as part of a wider federation that is using time management or to synchronize multiple VR-Forces back-ends. Without time management, multiple back-ends can stay synchronized only if they all are able to maintain the same frame rate. However, if they cannot do so, they will no longer have a shared concept of simulation time and will become unsynchronized. By using time management services, multiple back-ends can be configured to run in lock-step with respect to simulation time, ensuring that events will always happen in the same order.



VR-Forces supports time management in the back-end only.

For details about enabling time management, please see [“Configuring Time Management for HLA Exercises,”](#) on page 5-15.

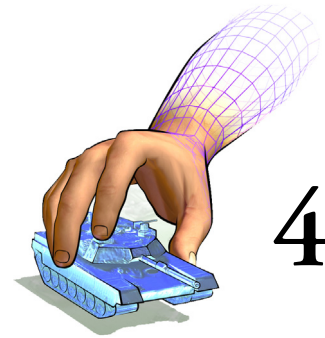
VR-Forces Simulation Time Versus Federation Time

Time management synchronizes federates to federation time. Federation time is the common model of time for the federation as whole. It is the time used in time stamp ordered messages (TSO), exchanged by time-managed federates. During the lifetime of a federation, federation time always increases. (There is no notion of rewinding back to an earlier time as with VR-Forces simulation time.)

VR-Forces simulation time is not the same as HLA federation time. It is the elapsed time within a scenario (the time displayed in the Simulation Time toolbar). When you first start a simulation, federation time and simulation time may be similar. However, if you rewind a time-managed simulation, the discrepancy between the two time streams will increase. This should not have any effect on running and rewinding scenarios. As long as all the back-ends start from the same simulation state when you begin a simulation, they will be correctly synchronized (against federation time) as the simulation progresses.

3.14. Interactions

VR-Forces displays interactions, such as detonations, that it receives in interaction messages or PDUs. You cannot select or otherwise manipulate them. You can only view them.



VR-Forces Visualization Concepts

This chapter briefly describes the concepts underlying visualization of simulations in the VR-Forces window.

The Display Engine	4-2
Window Types	4-2
The Observer	4-2
Observer Modes	4-3
Introduction to VR-Forces Object Modeling	4-3
Entity, Object, and Effects Model Formats	4-6
Model Sets.....	4-6
Mapping Models to Entities and Objects.....	4-7

4.1. The Display Engine

The VR-Forces front-end has a component called a display engine. The display engine renders 3D data and can display data in one or more windows, which can each have zero or more channels. The windows and channels being rendered at any particular time make up a display configuration. You can save display configurations and can load them to quickly recreate a particular configuration.

4.1.1. Window Types

VR-Forces supports the following types of windows:

- ♦ Full screen. A full screen window uses the entire area of a monitor. It does not have any menus, panels, or other GUI controls.
- ♦ Resizable. A resizable window can be moved, resized, maximized, and minimized. It does not have any GUI controls. An inset view is an example of a resizable window. (For information about inset views, please see [“Inset Views,”](#) on page 8-6.)
- ♦ Embedded. An embedded window is part of an application. The main display window in the VR-Forces front-end is an embedded window. The application provides a variety of controls that affect the view in the window. You could, if you wanted to, change VR-Forces’s window to a different type, in which case it would no longer be embedded in the VR-Forces window.

A window can have zero or more channels. Each channel defines a viewable area on the screen. You must have at least one channel to see a rendered image in the window. Each channel can have its own observer.

4.2. The Observer

The observer, or eyepoint, is the location in the 3D or 2D environment from which you observe the scene. You can move the observer (navigate) through the scene. You can attach the observer to an entity or prop. When the observer is attached to an entity, it moves automatically with that entity. You can also control the observer from a remote application using view control messages.

You can have multiple observers in a VR-Forces session. They might be associated with different windows. For example you could have an observer for the main window and one for an inset window.

The observer is such a pervasive component of the VR-Forces front-end that it is not covered in just one section of the manual. For more information, please see:

- ♦ Chapter 10, [The Observer and Observer Modes](#)
- ♦ Chapter 11, [Moving the Observer](#)
- ♦ Chapter 12, [Attaching the Observer](#)
- ♦ [“Displaying Models for Remote Observers,”](#) on page 10-7.

4.2.1. Observer Modes

Observer modes let you configure VR-Forces settings that affect how the observer moves and what it sees and package the settings together. You can then quickly change from one group of settings to another by switching between observer modes. An observer mode specifies:

- ♦ A model set. A collection of models of a particular type, such as 3D models.
- ♦ A key map. A coordinated set of key mappings to movement functions.
- ♦ A projection (2D or 3D).
- ♦ Settings for entity graphics and observer movement.

[Table 4-1](#) lists the observer modes and default settings provided with VR-Forces.

Table 4-1: Observer modes

Observer Mode	Projection	Key Map
Out-the-Window	3D	Observer Frame
Stealth	3D	Observer Frame
XR	3D	Observer Frame
Plan View	2D	2D Level Observer Frame
The observer modes all support the three model sets: 3D models, 3D colorized models, and 2D icons.		

You can edit observer modes and add new ones. For details, please see [“Creating and Editing Observer Modes,”](#) on page 10-2.

4.3. Introduction to VR-Forces Object Modeling

VR-Forces can display the simulated world using different views (2D and 3D). In each view, it can display appropriate types of models (2D icons, 3D models, and so on). Models are organized into model sets, and at any one time only the models in a particular model set are used.

VR-Forces can apply various types of visual effects to a model, such as trailing effects, height-above-terrain lines, track histories, and so on. Some of these effects are appropriate to a particular view, some are available for both views. Visual effects are implemented using visualizers.

Visual effects and model sets are managed by the integration of schemas, model definitions, element definitions, and model mapping.

- A schema is a set of parameters for a scene element (such as an entity), or a visualizer (such as a trailing effect). VR-Forces uses schemas to validate model definitions. It uses visual definition schemas to assign visualizers to scene elements.
- A model definition specifies values for the required and optional parameters for an instance of a schema. For example the model definition for an M1A2 tank is an instance of the Entity schema. The model definition for a lifeform is an instance of a DiGuyCharacter schema. Of particular importance, a model definition specifies the 3D model to be used to represent an element in the scene. (2D icons are specified by the Military Symbol Icon Visualizer in the entity definition.)
- Element definitions (where elements can be entities, tactical graphics, or interactions) configure visual definition schemas on scene elements. For any given element, you can create a definition for each model set in which you want the element to be visualized. For example, you might want to visualize a tank in the 3D Models, 2D icons, and 3D Colorized Models model sets. You might want a tree to be visualized only in the 3D Models model set.
- Model definitions are associated with particular entity types by mapping an element definition to an entity type enumeration.

Figure 4-1 illustrates these relationships using an M1A2 entity. The model definition called Tracked_M1A2_DESERT uses an Entity schema. One of the parameters of the schema is the filename, which for this model definition is defined as *M1A2.medf*. The entity definition called M1A2 Abrams has a Model Visualizer schema. The modelDefinition attribute in the visualizer is specified as Tracked_M1A2_DESERT, which means that it will use the *M1A2.medf* model. The entity type 1:-1:-1:-1:-1:-1:-1 is mapped to the M1A2 Abrams entity definition. So, when VR-Forces discovers an entity with the enumeration 1:-1:-1:-1:-1:-1:-1, it will look up the M1A2 Abrams entity definition, which will tell it to use a Model Visualizer for this entity that uses the Tracked_M1A2_DESERT model definition, which specifies that VR-Forces use the *M1A2.medf* model to render the entity in the scene.

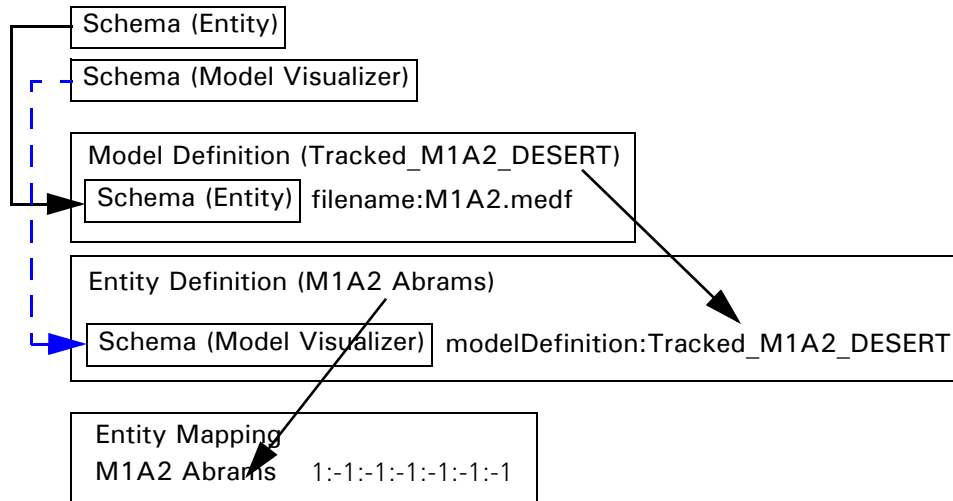


Figure 4-1. Element definitions and model mapping

To apply these abstract concepts to the concrete goals of a typical VR-Forces user, we could say that:

- If you want to visualize a completely new class of scene elements in VR-Forces, such as a celestial body, you would need to create a schema that specifies its parameters. If you want to change the parameters of an existing object, such as an entity, you would edit its schema. For details, please see Section 13.1, “[Creating and Editing Schemas](#)”, in *VR-Forces Configuration Guide*.
- If you want to add a new instance of an element, such as a star, you would create a model definition using the celestial body schema and specify its parameters. If you want to change an element definition by, for example, changing the model displayed for it, you would edit its model definition. For details, please see Section 13.2, “[Creating and Editing Model Definitions](#)”, in *VR-Forces Configuration Guide*.
- If you want your scene element to have visual effects, you would add appropriate visualizers in the entity definitions page or a specialized page. (If your new element requires visualizers that are not included with VR-Forces, you would have to use the VR-Forces toolkit to add them.) If you want to change the visualizers on an existing scene element, such as an entity, you would edit its entity definition. For details, please see Section 13.3, “[Creating and Editing Element Definitions](#)”, in *VR-Forces Configuration Guide*.
- Finally, if the new object will be displayed based on simulation data received over the network, you must map it to an entity enumeration. For details, please see Section 12.1, “[Mapping Models to Entities and other Scene Elements](#)”, in *VR-Forces Configuration Guide*.

4.3.1. Entity, Object, and Effects Model Formats

VR-Forces includes 3D models for vehicles, lifeforms, environmental objects, and effects. The models are in a compressed, encrypted format (extension *.medf*).



VR-Forces includes a utility, the MEDF Tool, that lets you encrypt files in MEDF format. You cannot use it to decrypt encrypted files. For more information, please see [“Compressing Model Files,”](#) on page 12-8, in *VR-Forces Configuration Guide*.

You can provide your own models if they are in one of the following supported formats:

- ♦ OpenFlight
- ♦ 3ds
- ♦ OBJ.

4.3.2. Model Sets

VR-Forces includes the following model sets:

- ♦ 3D Models. Realistic 3D models. This is the default model set for the Stealth observer mode.
- ♦ 3D Colorized Models. 3D colorized models are cartoon-like models that create an exaggerated notional view of the world. This is the default model set for XR observer mode.
- ♦ 2D Icons. 2D MIL-STD 2525B icons. This is the default model set for the Plan View observer mode.



The model sets are fixed. You cannot add model sets or edit the model sets provided.

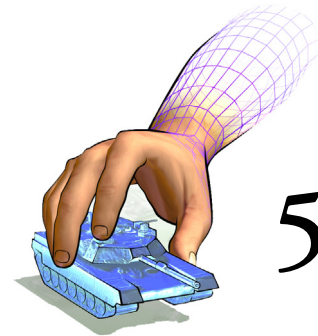
For more information about observer modes, please see [“Observer Modes,”](#) on page 4-3. For information about changing model sets, please see [“Changing the Model Set,”](#) on page 7-10.

4.3.3. Mapping Models to Entities and Objects

Model mapping is the process of associating an entity or other scene element with an element definition.

DIS and HLA objects are specified by an entity type enumeration. An entity type enumeration is a list of seven components as specified by the DIS protocol and HLA RPR FOM. If none of the seven components contains a wildcard (-1) value, then the entity type refers to a specific 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. For complete details about entity type enumerations, please see *Enumeration and Bit Encoded Values for Use with Protocols for DIS Applications* (DIS enumerations document). It is available at the IEEE World Wide Web site: <http://shop.ieee.org/store/> and at the SISO web site: <http://www.sisostds.org/>.

For more information about mapping models, please see Chapter 12, *Mapping Models and Effects* and Chapter 13, *Model and Element Definitions*, in VR-Forces Configuration Guide.



Starting VR-Forces

This chapter explains how to start and stop VR-Forces and load terrain databases.

Starting VR-Forces.....	5-3
Starting VR-Forces from the VR-Forces Launcher.....	5-3
Starting Independent VR-Forces Executables.....	5-6
Specifying a Session ID.....	5-7
VR-Forces Startup Tutorial.....	5-7
The VR-Forces Window	5-9
Switching Between the 2D and 3D Views	5-11
Opening New Windows.....	5-11
Managing a Front-End's Session Connection	5-12
Joining a Session.....	5-13
Resigning from a Session	5-13
Configuring Session Messages and Join at Startup	5-14
Configuring Time Management for HLA Exercises	5-15
RTI Requirements for Time Management.....	5-15
Enabling Time Management for the Back-end	5-16
Opening a Terrain Database.....	5-17
Loading a Terrain Database at Startup.....	5-18
Saving a Terrain Database in GDB Format.....	5-18
Closing a Terrain	5-18
Managing VR-Forces Settings	5-19
Synchronizing Settings Among VR-Forces Installations.....	5-21
Global Settings and Observer-Specific Settings	5-21
Exiting VR-Forces.....	5-21
Configuring Simulation Connections	5-22

Opening the Simulation Connections Configuration Dialog Box.....	5-22
Adding a Simulation Connection	5-23
Editing a Simulation Connection	5-24
Deleting a Simulation Connection	5-24
Configuring Auto Connect	5-24
Command Line Options.....	5-25
Command-Line Options for vrfGui	5-25
Command Line Options for vrfSim	5-28
Command Line Options for vrfLauncher	5-32
Protocol-Independent Command Line Options	5-33
Command Line Options for HLA Federations	5-35
Command Line Options for DIS Exercises.....	5-37
Managing Plug-ins.....	5-39
Loading a Plug-in	5-39
Adding a Plug-in	5-40
Specifying the DLLs for a Plugin.....	5-41
Adding a Plug-in Configuration	5-42
Deleting a Plug-in Configuration	5-42
Deleting a Plug-in	5-43
Viewing a List of Loaded Plug-ins	5-43
The VR-Forces Log Files.....	5-44

5.1. Starting VR-Forces

In “[The VR-Forces Program](#),” on page 3-3, we describe the VR-Forces front-end and back-end applications. You can start front-ends and back-ends separately, which is called separate mode, or you can use the VR-Forces Launcher to start them together, which is called combined mode. The only real difference between the two startup modes is that in combined mode, when you shut down the front-end, the back-end automatically shuts down also.

By default, all front-ends and back-ends start with session ID 1. Front-ends automatically try to join a session. If no back-ends are running when a front-end starts, it does not join a session. However, if at a later time a back-end with the same session ID starts, the front-end will automatically join the session.



If you run VR-Forces for HLA, be sure that you have installed an RTI and configured your system to use it. For details, please see “[Installing an RTI](#),” on page 2-9.

For convenience you can create batch files or scripts to start VR-Forces. Please see “[Command Line Options](#),” on page 5-25 for a list of command line options.

5.1.1. Starting VR-Forces from the VR-Forces Launcher

You can run VR-Forces using DIS, HLA 1.3, or HLA 1516. For either HLA specification, you can run using one of several different versions of the RPR FOM. With all these different possible startup configurations, particularly for HLA, it can be confusing to keep track of the options for starting VR-Forces. The VR-Forces Launcher simplifies the process of specifying the simulation standard to use when starting VR-Forces.

The VR-Forces Launcher lets you start in combined mode or separate mode.



The VR-Forces Launcher uses default values for the site ID and application number. If you want to run multiple front-ends, back-ends, or both, you must start the additional executables from the command-line using unique siteID:application number combinations. For details, please see “[Starting Independent VR-Forces Executables](#),” on page 5-6.

To start VR-Forces using the VR-Forces Launcher:

1. On Windows, on the Start menu, choose **Programs** → **MAK Technologies** → **VR-Forces 4.0** → **VR-Forces configuration**, where *configuration* is:
 - VRF GUI
 - Simulation Engine (back-end) only
 - VRF GUI + Simulation Engine.

On Linux (or from the Windows command line), to start the Launcher, in a console window, enter the following command:

```
vrfLauncher options
```

For descriptions of the vrfLauncher command-line options, please see “[Command Line Options for vrfLauncher](#),” on page 5-32. The basic startup configurations for vrfLauncher are:

- Start vrfGui and vrfSim (combined mode):

```
vrfLauncher -Z vrfSim
```

- Start vrfGui only:

```
vrfLauncher
```

- Start vrfSim only:

```
vrfLauncher --app vrfSim
```

If this is the first time that you are running VR-Forces, or if you have not specified a default startup connection, the Simulation Connections Configuration dialog box opens ([Figure 5-1](#)).

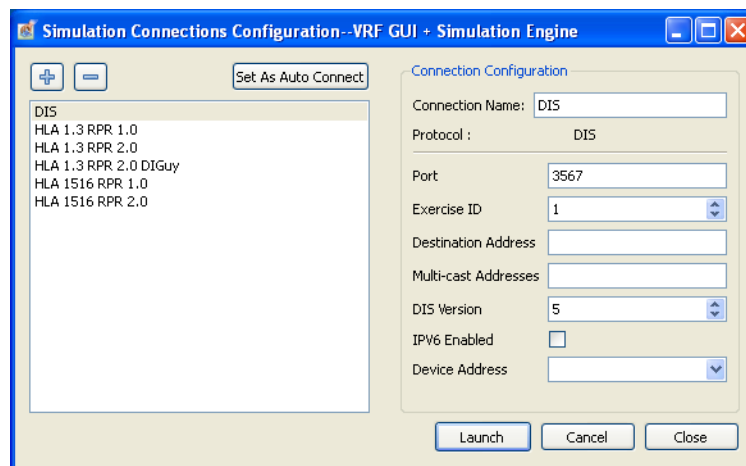


Figure 5-1. Simulation Connections Configuration dialog box

2. In the Simulation Connections Configuration dialog box, select a connection configuration. (Optionally, create a new configuration. For details, please see “[Configuring Simulation Connections](#),” on page 5-22.)

3. Click Launch. The VR-Forces executables start. When the VR-Forces applications finish their startup process, the Session Startup dialog box opens (Figure 5-2). It lets you quickly create or load a scenario using common options.



Figure 5-2. Session Startup dialog box

4. To create a scenario or load a scenario, select one of the options and click OK. The options are as follows:
 - Create a new scenario using streaming terrain from VR-TheWorld Online. This option connects to the VR-TheWorld terrain server. It lets you choose map configurations that have different levels of resolution to suit the performance needs of your scenario. The Status field indicates whether or not VR-Forces is able to connect to the terrain server using the specified connection.
 - Create a new scenario on a local terrain. This option is the same as choosing **File** → **New Scenario**.
 - Load a scenario. This option is the same as choosing **File** → **Load Scenario**.
5. If you do not want any of the options, click Close.

Enabling and Disabling the Session Startup Dialog Box

If you do not want VR-Forces to display the Session Startup dialog box every time it starts up, you can disable it. You can enable the dialog box at a later time.

To enable or disable the Session Startup dialog box:

1. Choose **Settings** → **Application**. The Application Settings dialog box opens.
2. Select the Session Options page (Figure 5-3).

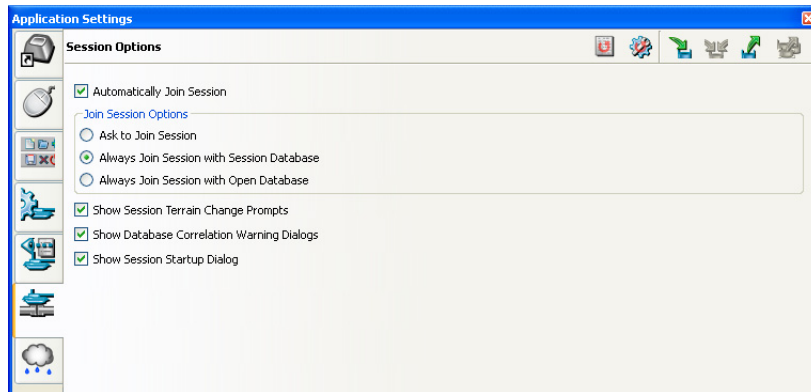


Figure 5-3. Session Options page

3. Select or clear the Show Session Startup Dialog check box.



You can also disable display of the Session Startup dialog box by selecting Do Not Show This Dialog At Startup on the dialog box itself.

5.1.2. Starting Independent VR-Forces Executables

To start independent VR-Forces executables:

1. Open a console window.
2. Change to the `./bin` directory.
3. Enter a command for each VR-Forces executable you want to run. Specify a unique site:application identifier for each executable, for example:

```
vrfSimprotocol -s 1 -a 3001
vrfSimprotocol -s 2 -a 3001
vrfGui -s 1 -a 3000 [--{dis | hla13 | hla1516} options]
```

A VR-Forces window opens for each front-end you start and a console window opens for each back-end. For an example of running multiple front and back-ends, please see “VR-Forces Startup Tutorial,” on page 5-7.

i

In the remainder of this manual, generic examples of starting the back-end on the command line will simply use *vrfsim*. It is assumed that users understand they must use the version of *vrfsim* for the protocol that they want to use.

5.1.3. Specifying a Session ID

- To assign a session ID, use the `-i` command line option.

```
vrfgui -s 1 -a 3000 -i 25
vrfsim -s 2 -a 3002 -i 25
```

The default session ID is 1. You cannot change the session ID during runtime.

5.1.4. VR-Forces Startup Tutorial

This section is a brief example of how to run multiple front and back-ends on multiple computers. Examples are provided for DIS and HLA. For both protocols, assume the configuration in (Figure 5-4). This configuration has two front-ends and three back-ends. Each is on a different computer. Assume that they are all running in the same session.

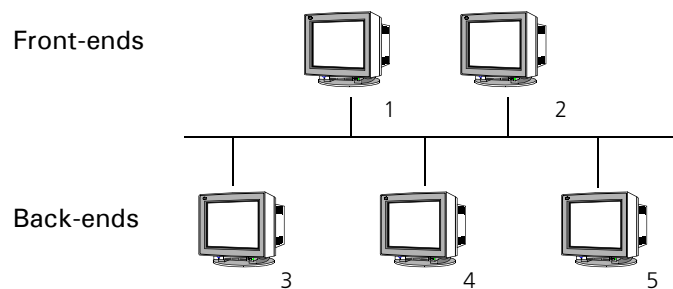


Figure 5-4. VR-Forces multicomputer configuration

i

As an alternative to starting individual front-ends and back-ends from the command line and specifying their protocol information in the command line, if you do not need to specify other command-line options, you can use *vrflauncher* to start the front-ends and back-ends. Specify the site and application numbers on the *vrflauncher* command line and choose the connection configuration in the Launcher. For details about *vrflauncher* command syntax, please see “[Command Line Options for vrflauncher](#),” on page 5-32.

DIS Configuration

In a DIS exercise, you typically specify a port so that the exercise traffic is isolated from other network traffic. Each front-end and back-end must have a unique site:application ID.

Run VR-Forces on each PC using the following commands:

- ♦ PC 1:
`vrfGui -s 1 -a 3000 --dis -P 1234`
- ♦ PC 2:
`vrfGui -s 1 -a 3001 --dis -P 1234`
- ♦ PC 3:
`vrfSimDIS -s 1 -a 3002 -P 1234`
- ♦ PC 4:
`vrfSimDIS -s 1 -a 3003 -P 1234`
- ♦ PC 5:
`vrfSimDIS -s 1 -a 3004 -P 1234`

HLA Configuration

In an HLA exercise, you could simply specify the site:application ID and use default values to specify the HLA configuration parameters. Specification of the site:application ID is identical to that in DIS. For this example, we use HLA 1.3 and specify a FOM Mapper other than the default (RPR FOM 1).

Run VR-Forces on each PC using the following commands:

- ♦ PC 1:
`vrfGui -s 1 -a 3000 --hla -x VR-Link20017-1
--rprFomVersion 2.0017 -F VR-Link20017-1.fed`
- ♦ PC 2:
`vrfGui -s 1 -a 3001 --hla -x VR-Link20017-1
--rprFomVersion 2.0017 -F VR-Link20017-1.fed`
- ♦ PC 3:
`vrfSimHLA13 -s 1 -a 3002 -x VR-Link20017-1
--rprFomVersion 2.0017 -F VR-Link20017-1.fed`
- ♦ PC 4:
`vrfSimHLA13 -s 1 -a 3003 -x VR-Link20017-1
--rprFomVersion 2.0017 -F VR-Link20017-1.fed`
- ♦ PC 5:
`vrfSimHLA13 -s 1 -a 3004 -x VR-Link20017-1
--rprFomVersion 2.0017 -F VR-Link20017-1.fed`

5.2. The VR-Forces Window

VR-Forces shows all the simulated entities and interactions on a terrain database. The VR-Forces window has the following main components (Figure 5-5):

- ♦ Menus. Menus for accessing VR-Forces commands.
- ♦ Simulation control toolbars:
 - Simulation Control toolbar. Allows you to start, pause, and rewind a scenario
 - Simulation Selection toolbar. Displays site and application numbers of multiple back-ends, if present
 - Simulation Time toolbar. Displays the elapsed simulation time
 - Simulation Time Scale toolbar. Allows you to control the speed at which the simulation runs.
- ♦ Application panels. Dockable panels that provide information and lets you manage the observer.
- ♦ Convenience toolbars. Toolbars that let you quickly select frequently used menu commands and configuration settings.
- ♦ Object creation palettes. Palettes for creating entities, control objects, and props.

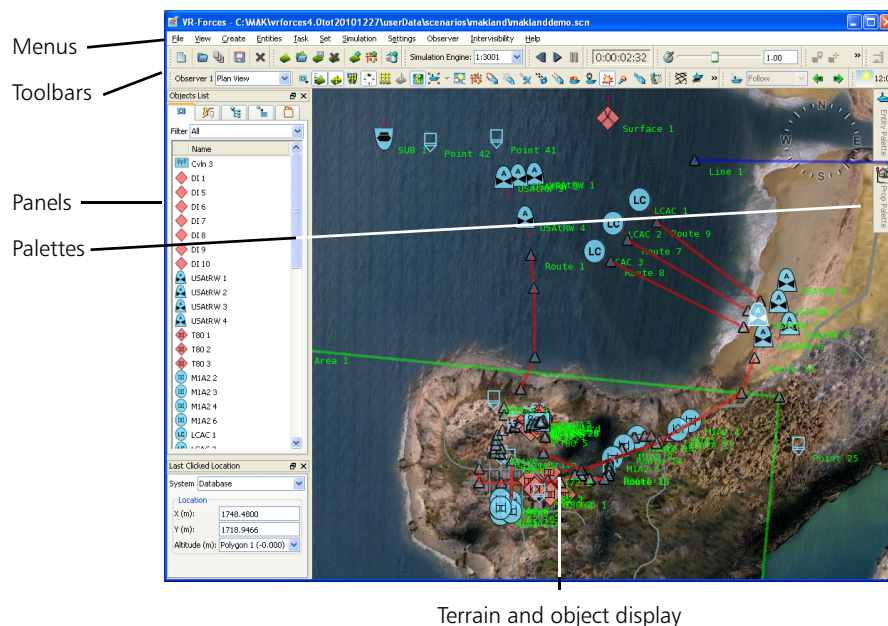


Figure 5-5. VR-Forces window and toolbars (2D)

You can view the terrain in 2D, 3D (Figure 5-7), and exaggerated reality (XR) (Figure 5-7) modes.

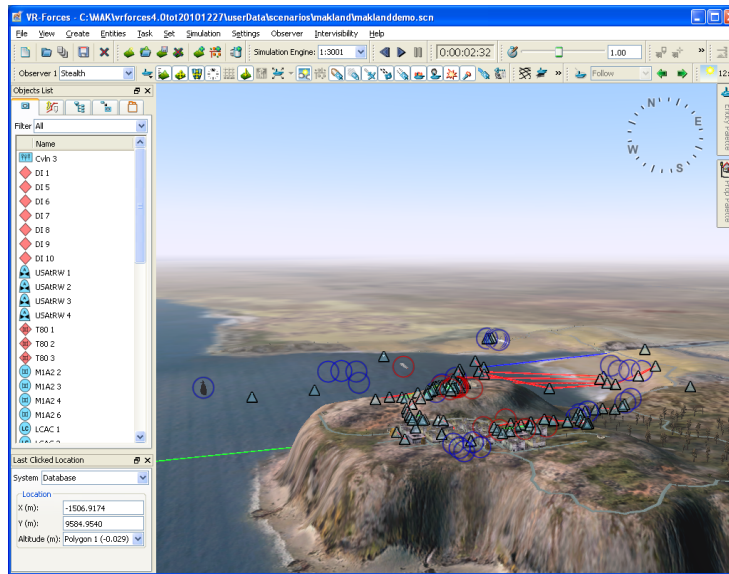


Figure 5-6. VR-Forces window (3D)

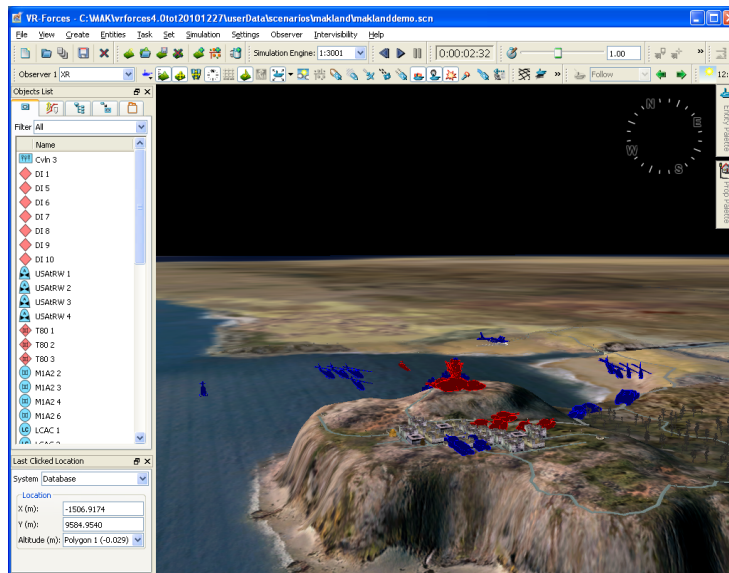


Figure 5-7. VR-Forces window (XR)

You can undock toolbars and panels and display them anyplace on your desktop. You can hide any toolbar that you do not want to be visible. (For details, please see Appendix A, *Common GUI Controls and Behaviors*.)

5.2.1. Switching Between the 2D and 3D Views

VR-Forces supports both 2D and 3D views. You can switch between them simply by changing the observer mode from a 3D observer mode, such as Stealth or XR, to the Plan View observer mode. (Changing observer modes is described in [“Changing the Observer Mode,”](#) on page 10-2.) When you go from a 3D observer mode to 2D, VR-Forces calculates a zoom level so that the observer should appear as if it is roughly the same height above the terrain. When you go from a 2D observer mode to 3D, VR-Forces calculates an altitude to do the same thing, and keeps the observer pointing down. So when you switch between 2D and 3D, the view of the terrain does not change much; however, the view of entities changes significantly because they transition from 2D icons that are quite large relative to the terrain, to 3D models, which are normally sized to real-world proportions.

5.2.2. Opening New Windows

In addition to the main VR-Forces window, you can open secondary windows. These windows do not have any of the menus or toolbars of the main window. However, you can navigate the windows using the same keyboard and mouse options as the main window. When a secondary window has focus, the settings on the Observer Settings toolbar apply to that secondary window and changes you make affect the window.

You can open windows with a 2D view or 3D view. Once a window is open, you can change the view by changing the observer mode.

Opening a secondary window is not the same thing as running a second instance of the front-end. When you close a front-end, all secondary windows associated with it are also closed.

- To open a new window, choose **File → New Stealth Window** or **File → New Plan View Window**.

5.3. Managing a Front-End's Session Connection

A front-end can be joined to a session or not joined. When a front-end is not joined to a session, it cannot create, load, or manage a scenario. It can only load a terrain and function as a viewer. You can tell a front-end's session status by the availability of Join Session and Resign Session commands on the File menu.

When you start a front-end it automatically tries to join a session. If there is no session running, it will poll the network until one becomes available and then automatically join it. Once a front-end is joined to a session, you can manually resign from and rejoin it.

If you have one front-end running, when you open and close a scenario, the front-end opens and closes the terrain database for the scenario, as you would expect. If you have two or more front-ends loaded, the front-end from which you load a scenario automatically loads the terrain. However, all other front-ends prompt you to load the terrain. (The front-end knows which terrain to load, it is just asking permission to do so.) Similarly, when you close a scenario, the front-end from which you close the scenario automatically closes the terrain. All other front-ends ask permission to close the terrain. In either of these cases, you do not have to accept the default behavior. You can load an alternative terrain when a scenario opens and you can keep the terrain loaded when a scenario closes.



If you do not load the terrain specified by the scenario, it is possible that visual and simulation terrains will not be synchronized.

5.3.1. Joining a Session

When you join a session, all menu commands related to scenario management are enabled. If a scenario is loaded in the session, the terrain is closed.

To join a session:

1. Choose **File** → **Join Session**. If VR-Forces is configured to always join a session with the session database, the front-end loads the session database and joins the session. If VR-Forces is not configured to automatically use the session database, the Join Session dialog box opens (Figure 5-8). If a scenario is running, its details are listed.

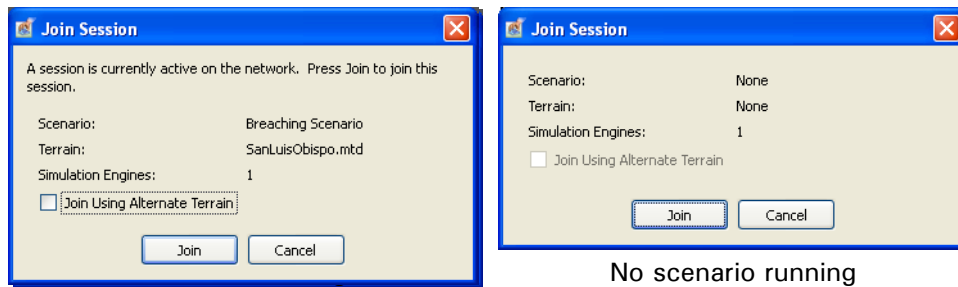


Figure 5-8. Join Session dialog box

2. If a scenario is running, optionally, specify that you want to load an alternate terrain:
 - a. Select the Join Using Alternate Terrain check box.
 - b. Click Join. An Open Database dialog box opens.
 - c. Select the terrain that you want to load.
 - d. Click Open. The terrain is loaded and the front-end joins the session.



If you join a session using an alternate terrain, the front-end (visual) and back-end (simulation) terrains might not be synchronized.

3. Click Join. The front-end joins the session.

5.3.2. Resigning from a Session

When you resign from a session, all menu commands related to scenario management are disabled. If you have a scenario loaded, the terrain is closed.

- To resign from a session, choose **File** → **Resign From Session**.

5.3.3. Configuring Session Messages and Join at Startup

You can configure the display of session-related messages and whether or not the front-end tries to join a session automatically at startup.

To configure sessions:

1. Choose **Settings** → **Application**. The Application Settings dialog box opens.
2. Select the Session Options page (Figure 5-3).
3. Select the options that you want to enable, as follows:
 - Automatically Join Session. If checked, when the front-end starts, automatically join the session if it is available. If cleared, the front-end does not join the session and does not prompt you to do so.
 - Ask to Join Session. Provides session data (Figure 5-8) and prompts you to join whenever you join a session.
 - Always Join Session with Session Database. Always join a session using the database in the loaded scenario. (Does not display a Join Session dialog box.)
 - Always Join Session with Open Database. Always joins a session with whichever terrain is currently open in the front-end. This might generate a terrain correlation warning.
 - Show Session Terrain Change Prompts. Enable if you want to get prompted when the terrain in a session changes
 - Show Database Correlation Warning Dialogs. If enabled, VR-Forces displays a warning if the terrain in the front-end might not correlate with the terrain loaded in the back-end.
 - Show Session Startup Dialog. Enable if you want be prompted when a new session starts.
4. Click OK.

5.4. Configuring Time Management for HLA Exercises

To use time management services in an HLA exercise, you must properly configure an RTI and enable time management in the VR-Forces back-end.

Time management is enabled at the application level, not the scenario level. Scenarios created without use of time management work in time-managed mode. No time management-specific information is saved as part of a scenario.

If you enable time management in the back-end, then whenever you create or load a scenario that specifies a fixed-frame mode (that is, the **frame-mode** parameter in the scenario file is set to either fixed-frame best effort or fixed-frame-run-to-complete), the back-end will become a regulating and constrained federate. It will remain in this state until the scenario is closed.

A time-managed back-end advances simulation time at a rate that is a function of the frame-mode specified in the scenario it is executing. If the **frame-mode** is fixed-frame, then it tries to advance simulation time at a rate approximating real time. However, if the back-end has a heavy simulation load, or another time-constrained federate is advancing at a relatively slower rate, the back-end may not achieve this rate. If the **frame-mode** is fixed-frame-run-to-complete, the back-end tries to advance simulation time as fast as possible, independent of wall-clock time.

For conceptual information about time management, please see [“Using HLA Time Management,”](#) on page 3-38.

5.4.1. RTI Requirements for Time Management

To use time management in VR-Forces, your RTI must be properly configured to support this service. If you are using the MÅK RTI, set the **RTI_useRtiExec** and **RTI_timeMgmt** parameters to 1. When you start the federation, you must run the **rtiexec**.

If you are using another RTI, please consult its documentation for configuration instructions.

5.4.2. Enabling Time Management for the Back-end

To enable time management in the back-end, use one of the following methods:

- ♦ Set `runInTimeManagementMode` to 1 in *vrfSim.mtl*.
- ♦ Start the back-end with the `--timeManagement` command-line option. For example, to start the back-end in time management mode using the sample FED file configured for use with time management and the federation name VR-LinkTM:

```
vrfSim -s 1 -a 3001 --timeManagement -F VR-LinkTM.fed  
-x VR-LinkTM
```



- ♦ The back-end will only run in time management mode (time regulating and constrained) while it has a scenario loaded.
 - ♦ If a scenario is configured with `frame-mode` set to variable-frame, the back-end will not become time regulating or constrained. It will print a warning to the console and run without time management enabled.
-

5.5. Opening a Terrain Database

If you are using multiple front-ends, when you load or create a scenario, the front-end on which you issue the command loads the correct terrain database. The other front-ends prompt you to load the terrain. However, you can choose to load a different terrain. Also, if you want to use the front-end as a viewer, you can manually load a terrain database. (For a brief description of the databases supplied with VR-Forces, please see [“Terrain Databases Provided with VR-Forces,”](#) on page 14-13.)



It is possible to open a terrain without having a scenario loaded. In such cases you cannot create any type of VR-Forces object. To create and manipulate entities and objects, you must load or create a scenario.

To open a terrain database:

1. In the VR-Forces window, choose **File** → **Open Terrain**. The Open Terrain dialog box opens.
2. Select the terrain that you want to load. By default, MTF files are listed. However, you can also load MTD files, or any supported terrain database format.
3. Click Open. The database is displayed in the VR-Forces window ([Figure 5-9](#)).

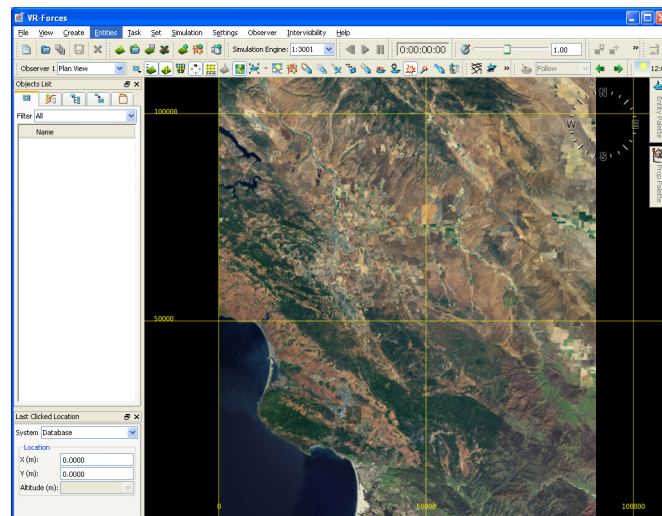


Figure 5-9. VR-Forces window with a database loaded



If you load a terrain database while one is currently loaded, VR-Forces closes the current database and then opens the new one.

5.5.1. Loading a Terrain Database at Startup

- To load a terrain database when you start VR-Forces, use the `-T` command line option.

```
vrfSim -s 1 -a 3001 -T "../data/terrain/myDatabase.mtd"
```

5.5.2. Saving a Terrain Database in GDB Format

VR-Forces lets you save terrains in the GDB format. You might want to load a GDB file in the TDB Tool to take advantage of TDB Tool features that are not supported in the front-end, such as adding feature data. If you are using the B-HAVE module, you might need to generate PathData, which requires opening a GDB file in the TDB Tool.

To save a database in the GDB format from the VR-Forces GUI:

1. Choose **File** → **Export GDB**. The Export GDB dialog box opens.
2. Type a name for the database.
3. Click Save.

5.5.3. Closing a Terrain

To close a terrain, use one of the following methods:

- ♦ Choose **File** → **New Terrain**.
- ♦ Open a different terrain.

5.6. Managing VR-Forces Settings

VR-Forces has the following types of settings:

- ♦ Independent, or display, settings are the individual settings managed on the Settings menu and the pages of the Display Settings dialog box. VR-Forces also saves the layout of toolbars and panels. Changing an independent setting, such as enabling track histories or wireframe mode, does not affect other settings.

Independent settings are categorized as being global settings or observer-specific settings. Global settings apply regardless of the observer mode. Observer-specific settings can have different settings depending on the observer mode. For more information, please see [“Global Settings and Observer-Specific Settings,”](#) on page 5-21.

- ♦ Dependent settings are the schemas, model definitions, element definitions, and entity type mappings that collectively determine how entities and objects are modeled in the scene. Changing a dependent setting, such as a model definition may affect settings that depend on it, such as an entity definition.

VR-Forces manages settings as follows:

- ♦ As you make changes to settings during a session, it saves the changes
- ♦ It remembers the settings in effect when you exit and uses them as the default settings in your next session
- ♦ During any particular session, you can revert to the default settings
- ♦ You can save the current settings to a file
- ♦ You can import saved settings
- ♦ You can restore settings to the settings that were in effect when VR-Forces was first installed (factory settings).

Dependent settings have two additional options that independent settings do not have:

- ♦ You can save or restore a specific setting
- ♦ You can merge settings from a file into the current settings.

The procedures for saving and restoring settings are the same regardless of the types of settings. You start them by clicking an icon on a settings dialog box page. However, the way that VR-Forces saves or restores settings depends on the type of setting, as follows:

- ♦ When you initiate a save or restore operation on independent settings it only affects the settings on the dialog box page from which you started the save or restore. If you have changed a setting on another page, it is not affected.
- ♦ When you initiate a save or restore operation on dependent settings, all of the changes to all of the related settings get saved or restored. This ensures that the relationships do not get broken.

([Figure 5-10](#)) illustrates the settings management icons. They are described in the list.



Figure 5-10. Settings icons

- ♦ Revert to the default settings (1). Override any changes made to settings during this session.
- ♦ Restore the factory settings (2). Change settings to the settings used when VR-Forces was installed. (The factory settings and default settings may be different.)
- ♦ Import settings from a file (3). Replace the current settings with those stored in the selected file.
- ♦ Merge in settings from a file (4). (Only supported for dependent settings.) Merging works as follows:
 - Saved settings that do not have a match in the current settings are added.
 - Saved settings replace matching settings.
 - Current settings that are not replaced (that is, for which there is no match in the saved settings) remain valid.

Figure 5-11 illustrates how merged settings works. Setting A in the saved settings has no match in current settings, so it is added to the merged settings. Current settings B and C are replaced with saved settings. Current setting D has no match in saved settings, so it is retained in the merged settings.

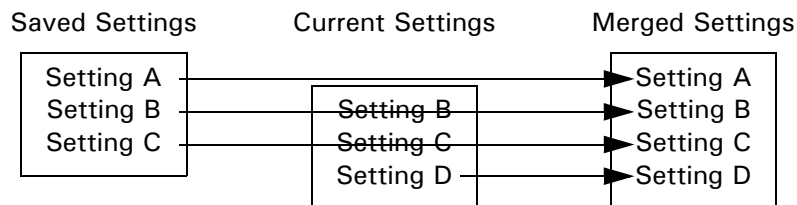


Figure 5-11. Merged settings

- ♦ Export settings to a file (5). Save the current settings to the selected file.
- ♦ Save the selected setting to a file (6). (Only supported for dependent settings.)

i

- ♦ The icons described in this section are disabled on dialog box pages on which their function is not supported. This includes most of the pages on the Terrain Settings dialog box.
 - ♦ VR-Forces saves settings to *./appData/settings/vrfGui*.
 - ♦ If you load an alternative settings file at startup, it is treated like an imported settings file. If you save current settings as default, they get saved to the default files in *./appData/settings/*. They do not get saved back to the alternative file unless you explicitly export them to that file.
-

5.6.1. Synchronizing Settings Among VR-Forces Installations

If your simulation is using several front-ends and back-ends on multiple computers, you may want to use the same settings across all of them, particularly if you have added model definitions and mappings to the factory settings. To synchronize settings, copy the data directories (*./appData*, *./data*, and *./userData*) from the computer on which you have updated them to the computers on which the other front-ends and back-ends are installed. Then import the settings into any Settings page that you customized on the source VR-Forces machine.

5.6.2. Global Settings and Observer-Specific Settings

VR-Forces has two types of independent settings – global settings and observer-specific settings.

Global settings affect the display regardless of which observer mode you are using. Global settings include entity labels, tactical graphics, and cockpit displays. The settings on the Settings menu are all global settings.

Observer-specific settings can be different for each observer mode. In fact, given a similar projection and model set, it is the different settings that distinguish one observer mode from another. For example, one difference between Stealth observer mode and Out-the-Window observer mode is that Stealth mode has view controls enabled and Out-the-Window mode has them disabled.

5.7. Exiting VR-Forces

When you exit VR-Forces, all GUI windows close. In combined mode, the application terminates. If you are not running in combined mode, you must also close all simulation engines.

- To exit VR-Forces, choose **File** → **Exit**.

5.8. Configuring Simulation Connections

A simulation connection specifies the connection parameters for a DIS or HLA simulation connection. VR-Forces comes with the following connection configurations:

- ♦ DIS (port 3000)
- ♦ HLA 1.3 RPR FOM 1.0
- ♦ HLA 1.3 RPR FOM 2.0, draft 17
- ♦ HLA 1.3 RPR FOM 2.0, draft 17, with support for DI-Guy
- ♦ HLA 1516 RPR FOM 1.0
- ♦ HLA 1516 RPR FOM 2.0, draft 17

You can add your own connection configurations that use, for example, a different DIS port or your own FOM Mapper.

5.8.1. Opening the Simulation Connections Configuration Dialog Box

Simulation connection configurations are maintained in the Simulation Connections Configuration dialog box. This dialog box opens automatically when you start VR-Forces, unless you choose a configuration as the default connection for auto connection. If Auto Connect is enabled and you want to change the connection, you can open the dialog box from the Tools menu.

- To open the Simulation Connections Configuration dialog box, on the Windows Start menu, choose **Programs → MAK Technologies → VR-Forces 4.0 → Tools → Configure Connections**.

5.8.2. Adding a Simulation Connection

To add a simulation connection:

1. If the Simulation Connections Configuration dialog box (Figure 5-12) does not open automatically when you start VR-Forces, choose **Programs** → **MAK Technologies** → **VR-Forces 4.0** → **Tools** → **Configure Connections**.

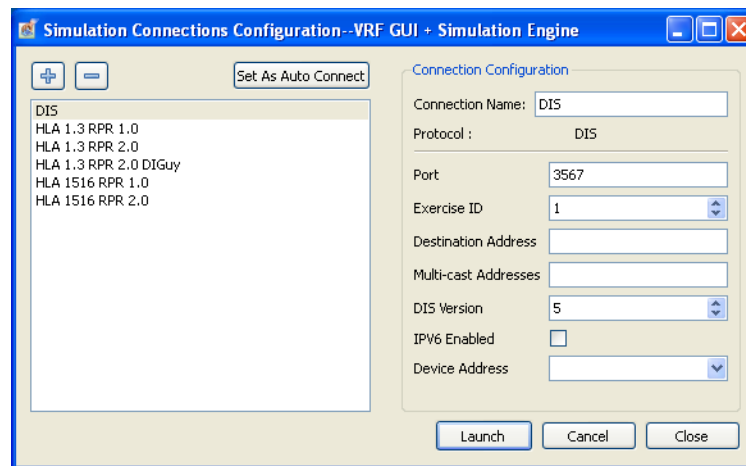


Figure 5-12. Simulation Connections Configuration dialog box

2. Click the Plus (+) button. The Create Connection dialog box opens (Figure 5-13).



Figure 5-13. Create Connection dialog box

3. In the Connection Name box, type a name for the connection.
4. Select a protocol from the Protocol list.
5. Click OK.
6. The connection is added to the list in the Simulation Connections Configuration dialog box.
7. Select the new connection in the list.
8. Edit the parameters in the Connection Configuration group box as desired.

5.8.3. Editing a Simulation Connection

To edit a simulation connection:

1. If the Simulation Connections Configuration dialog box (Figure 5-12) does not open automatically when you start VR-Forces, choose **Programs** → **MAK Technologies** → **VR-Forces 4.0** → **Tools** → **Configure Connections**.
2. In the list of connections, select the connection that you want to edit.
3. Edit the parameters in the Connection Configuration group box as desired.

5.8.4. Deleting a Simulation Connection

To delete a simulation connection:

1. If the Simulation Connections Configuration dialog box (Figure 5-12) does not open automatically when you start VR-Forces, choose **Programs** → **MAK Technologies** → **VR-Forces 4.0** → **Tools** → **Configure Connections**.
2. In the list of connections, select the connection that you want to delete.
3. Click the Minus (–) button. A confirmation prompt opens.
4. Click Yes.

5.8.5. Configuring Auto Connect

If you know that you will usually or always want to use the same simulation connection, you can set up Auto Connect and you will not have to select a connection every time you start VR-Forces.

To configure Auto Connect:

1. If the Simulation Connections Configuration dialog box (Figure 5-12) does not open automatically when you start VR-Forces, choose **Programs** → **MAK Technologies** → **VR-Forces 4.0** → **Tools** → **Configure Connections**.
2. In the list of connections, select the connection that you want to use every time you start VR-Forces.
3. Click Set as Auto Connect. A star is displayed next to the selected connection.

Disabling Auto Connect

To disable Auto Connect:

1. Choose **Programs** → **MAK Technologies** → **VR-Forces *x.x*** → **Tools** → **Configure Connections**. The Simulation Connections Configuration dialog box opens (Figure 5-12).
2. In the list of connections, select the connection that is configured for auto connect.
3. Click Set as Auto Connect. Auto connect is disabled and the star is removed from next to the connection.

5.9. Command Line Options

VR-Forces has command line options for vrfGui and vrfSim.

5.9.1. Command-Line Options for vrfGui

Table 5-1 lists the command line options for vrfGui.

Table 5-1: vrfGui command-line options (Sheet 1 of 3)

Option	Description
<code>(-- --ignore_rest)</code>	Ignore all command-line options after this one.
<code>--appDataDir directory</code>	Specifies the location of application data.
<code>-B --bindObservers "observer1", ..., "observern", saved_views_file</code>	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 configuration gets a view assigned to it. If observer names have spaces, such as Observer 1, they must be enclosed in quotes.
<code>(-C --autoconnect) {filename connector_name}</code>	Automatically connect to the network using the specified connector.
<code>--dataDir directory</code>	Specifies the location of the data directory (terrains and so on).
<code>--dis</code>	Start in DIS mode, using DIS command-line arguments to specify connection parameters.
<code>--dispSetting filename</code>	Specifies the display configuration to load.
<code>--doNotCheckPluginVersions</code>	Do not check the version in a plug-in to make sure it can load in VR-Forces. A plug-in gets built against a particular version of VR-Forces. Ordinarily, when VR-Forces loads a plug-in, it checks to be sure it was created with that version of VR-Forces. This option prevents that check.

Table 5-1: vrfGui command-line options (Sheet 2 of 3)

Option	Description
<code>--doNotLoadVrfPlugins</code>	Prevents loading of plug-ins in the <code>./plugins</code> directory.
<code>--entDispSetting filename</code>	Specifies the entity display setting to load.
<code>--envSetting filename</code>	Specifies the environment configuration file to load.
<code>(-E --entTypeMap) filename</code>	Load the specified entity type mapping file. The file extension determines the type of mapping.
<code>--fullScreen</code>	Start with window in full screen mode.
<code>(-G --locale) language</code>	Specifies the locale (for localization).
<code>(-h --help)</code>	Display command-line usage information.
<code>--hla13</code>	Start in HLA 1.3 mode, using HLA command-line arguments to specify connection parameters.
<code>--hla1516</code>	Start in HLA 1516 mode, using HLA command-line arguments to specify connection parameters.
<code>(-l --logFileName) filename</code>	Specifies the log file name.
<code>(-L --loadObservers filename1 filename2)</code>	Loads the saved observers in the specified files.
<code>--loadPlugin filename</code>	Specifies plug-ins to load. Accepted multiple times. The filename can be a DLL or SO or an XML file that specifies the plug-ins to load. For details, please see “Managing Plug-ins,” on page 5-39.
<code>(-n --notifyLevel) notify_level</code>	Specifies the notification level for application messages, where <code>notification_level</code> is a number from 0 through 4. The least verbose message level is 0, the highest is 4. The default is 2. For details, please see “Setting the Notification Level,” on page 5-34.
<code>--noGui</code>	Start without the graphical user interface.
<code>--observerMode mode</code>	Specifies the observer mode for each observer in the startup configuration, for example Observer 1, Plan View. If you specify an observer that does not exist, the mode is not set. Accepted multiple times.
<code>--plugin filename</code>	Load the specified plug-in. Can be used multiple times.
<code>--scenarioFile filename</code>	Specifies a scenario to load.
<code>--showConsole</code>	Display a console window.
<code>terrain_MTF_file</code>	Specifies a terrain (MTF) file to load.
<code>--unplug filename</code>	Do not load the specified plug-in. Can be used multiple times.
<code>--userDataDir directory</code>	Specifies the user data directory.

Table 5-1: vrfGui command-line options (Sheet 3 of 3)

Option	Description
(-v --version)	Display version information and exit.
--vsync	Start with vsynched windows.
-W --withDisplayEngine port@netAddress;display ConfigFilename1 n	Connect to a display engine running on another computer.
HLA Only	These options are available if you specify --hla13 or --hla1516.
(-f --fomMapperLib) libname	FOM Mapper library name.
(-F --fedFileName) fedfile_name	FED file name.
--fomMapperInitData data	FOM Mapper initialization data.
(-N --federateName) federate_name	Federate name. Default: MAK VR-Forces GUI HLA 1.3 Driver or MAK VR-Forces GUI HLA 1516 Driver.
--rprFomRevision	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, or 2.0017)
--useAdvisories	Enables or disables HLA advisory messages. Default: true.
--useGeographicDdm	Specifies use of geographic DDM if you are using the MAK RTI. Default: false.
(-x --execName) exec_name	Execution name. Default: VR-Link.
DIS Only	These options are available if you specify --dis.
(-a --appNumber) number	Application number.
(-A --destAddrString) address	Destination address.
--deviceAddress address	Specifies the address of the network card to use for UDP traffic.
--disVersion	The DIS version. Default: 5.
--mcastTtl	Specifies the multicast time-to-live. Default: -1.
--multicastAddresses address1 ... addressn	Specifies one or more multicast addresses.
(-P --disPort) port	DIS port.
--recvBufferSize size	Receive buffer size
(-s --siteId ID	Site ID.
--sendBufferSize size	Send buffer size. Default: -1.
--useAsyncIO	Use asynchronous I/O. Default: False.
(-x --exerciseId) ID	Exercise ID.

5.9.2. Command Line Options for vrfSim

Table 5-2 describes the command line options for vrfSim.

Table 5-2: vrfSim command line options

Option	Description
(-a --appnumber) <i>application_number</i>	Specifies the application number. For details, please see “Specifying the Site ID and Application Number,” on page 5-33.
--appIdRange <i>lower-upper</i>	Specifies a range of application IDs that represent objects generated by VR-Forces.
(-B --batchScenarioFileName) <i>batch_file</i>	Runs the specified batch file. (For use with one back-end only.) For more information, please see Running in Batch Mode from the Command Line , under Section 1.5.3, “Running VR-Forces in Batch Mode” , in <i>VR-Forces Scenario Management Guide</i> .
(-c --frontEndPID) <i>PID</i>	The process ID for the front-end when running in combined mode.
(-d --settingsFile) <i>file_name</i>	Specifies a configuration file to load instead of <i>vrfSimSettings.xml</i> .
--daemon	Starts that back-end as a daemon process (Linux only). If you start <i>vrfSim</i> as a daemon, it does not try to read characters from the console. It forks a background process at startup. It will exit cleanly if it receives a SIGTERM, SIGQUIT, or SIGINT signal.
--diGuyAnimationsFile <i>file_name</i>	Specifies an additional configuration file for DI-Guy animations.
--diGuyCharacerDataFile <i>file_name</i>	Specifies an additional configuration file for DI-Guy character data.
--doNotLoadVrfPlugins	Do not load any VR-Forces plug-ins from the <i>./plugins</i> directory.
--fullyLoadTerrain	For paged terrains, load all pages.
(-h --help)	Displays help information in a console window.
(-i --sessionId) <i>session_ID</i>	Specifies a session ID. For details, please see “Specifying a Session ID,” on page 5-7.

Table 5-2: vrfSim command line options

Option	Description
<code>(-L --scenarioFileName)</code> <code>"scenario_pathname"</code>	Loads the specified scenario. The pathname can be absolute or relative from the <code>./bin</code> directory. Mutually exclusive with the <code>-T</code> option. For details, please see Section 1.2.2, “Loading a Scenario from the Command Line” , in <i>VR-Forces Scenario Management Guide</i> .
<code>--loadPlugin plugin_file</code>	Specifies one or more plug-ins to load. (Accepted multiple times.)
<code>--logFrameRateStatistics</code>	Specifies that the back-end should log frame rate statistics.
<code>(-n --notifyLevel)</code> <code>notification_level</code>	Sets the level for messages written to the console or the VR-Forces log file (<i>vrf.log</i>), where <code>notification_level</code> is a number from 0 through 4. The least verbose message level is 0, the highest is 4. The default is 2. For details, please see “Setting the Notification Level,” on page 5-34.
<code>--noPluginDirectory</code>	Specifies that VR-Forces not load the configuration files in the <code>./plugins</code> directory.
<code>--outputLogFile file_name</code>	The file to use to write out application output.
<code>--physicalWorldFile file</code>	Specifies a physical world configuration file to load.
<code>(-q --doNotUseConsole)</code>	Specifies that all <i>vrfSim</i> 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 back-ends running on Windows. To create a log file on Linux, redirect the output of <i>vrfSim</i> , for example: <code>vrfSim > mylog.txt</code> For more details about log files, please see “The VR-Forces Log Files,” on page 5-44.

Table 5-2: vrfSim command line options

Option	Description
(-r --startInRunMode)	Starts the back-end in run mode. If you start the back-end without specifying a scenario (with -L) or a terrain database (with -T), the back-end starts in pause mode.
(-s --siteId) <i>site_id</i>	Specifies the site ID. For details, please see “Specifying the Site ID and Application Number,” on page 5-33.
--simulationModelSet <i>sms</i>	Specifies the simulation model set. It must be declared separately for front-end and back-end even in combined mode.
(-T --terrainDatabase) <i>terrain_database</i>	Specifies a terrain database to load in the back-end. Mutually exclusive with the -L option. For details, please see “Working with Multiple Back-ends,” on page 3-8 and “Loading a Terrain Database at Startup,” on page 5-18.
--targetFrameRate <i>rate</i>	Specifies the frame rate above which vrfSim will yield CPU time to other processes.
--timeManagement	Enables time management for the back-end.
(-v --version) <i>version</i>	Displays version information.
(-- --ignore_rest)]	Ignore all of the command-line options that are listed after this argument. This lets you quickly limit the arguments applied if you want to edit a script or batch file without losing the original command line.
HLA Only	
(-f --fomMapperLib) <i>libname</i>	FOM Mapper library name.
(-F --fedFileName) <i>fedfile_name</i>	FED file name.
--fomMapperInitData <i>data</i>	FOM Mapper initialization data.
(-N --federateName) <i>federate_name</i>	Federate name. Default: VR-Forces.
--rprFomRevision	RPR FOM revision for RPR FOM 2.0, draft 17.
--rprFomVersion <i>version_number</i>	RPR FOM version. (0.5, 0.7, 0.8, 1.0, 2.0006, 2.0014, or 2.0017)
--useAdvisories	Enables or disables HLA advisory messages. Default: true.

Table 5-2: vrfSim command line options

Option	Description
<code>--useGeographicDdm</code>	Specifies use of geographic DDM if you are using the MÄK RTI. Default: false.
<code>(-x --execName) exec_name</code>	Execution name. Default: VR-Link.
DIS Only	
<code>(-a --appNumber) number</code>	Application number.
<code>(-A --destAddrString) address</code>	Destination address.
<code>--deviceAddress address</code>	Specifies the address of the network card to use for UDP traffic.
<code>--disVersion</code>	The DIS version. Default: 5.
<code>--mcastTtl</code>	Specifies the multicast time-to-live. Default: -1.
<code>--multicastAddresses address1 ... addressn</code>	Specifies one or more multicast addresses.
<code>(-P --disPort) port</code>	DIS port.
<code>--recvBufferSize size</code>	Receive buffer size
<code>--sendBufferSize size</code>	Send buffer size. Default: -1.
<code>--suppressSelfReflect</code>	Represses self reflection of the DIS exercise connection.
<code>--useAsyncIO</code>	Use asynchronous I/O. Default: False.
<code>--useIpv6</code>	Specifies that the DIS exercise connection use IPV6.
<code>(-x --exerciseId) ID</code>	Exercise ID.

5.9.3. Command Line Options for vrfLauncher

[Table 5-3](#) describes the command line options for vrfLauncher.

Table 5-3: vrfLauncher command line options

Option	Description
<code>-a application_number</code>	Specifies the application number. For details, please see “Specifying the Site ID and Application Number,” on page 5-33.
<code>--app vrfSim</code>	Start Launcher for back-end only.
<code>--configureConnections</code>	Open Simulation Connections Configuration dialog box for creating and editing connection configurations. For details, please see “Configuring Simulation Connections,” on page 5-22.
<code>--configFilename filename</code>	Specifies the connection configurations file to use. Default: <code>./config/connectionsSettings.xml</code> .
<code>-h</code>	Displays help information in a console window.
<code>-s site_id</code>	Specifies the site ID. This option applies only to the front-end in combined mode. For details, please see “Specifying the Site ID and Application Number,” on page 5-33.
<code>-v</code>	Displays version information.
<code>-Z vrfSim</code>	Start Launcher in combined mode. (Case sensitive; must be capital Z.)
<code>--</code>	Ignore all of the command-line options that are listed after this argument. This lets you quickly limit the arguments applied if you want to edit a script or batch file without losing the original command line.

5.9.4. Protocol-Independent Command Line Options

This section describes in greater detail some of most commonly used command line options that you can use with both DIS and HLA.

Specifying the Site ID and Application Number

Each front-end and back-end must have a unique site ID:application number pair. To specify the site ID, use the `-s` option. To specify the application number, use the `-a` option, for example:

```
vrfSim -s 2 -a 3005
```

The default application number for `vrfGui` is 3000; for `vrfSim` it is 3001.



If you record a VR-Forces simulation with the MÄK Logger and you want to view the Logger file in VR-Forces, when you view the Logger file, you must run VR-Forces with an application number that is different from the one you used when you recorded the simulation.

Specifying the Language to Use in the GUI

VR-Forces decides which translation files to use based on the language setting for your computer. However, you can override the language setting with the `-G` command line option, for example, the following command sets the language to German:

```
vrfGui -s 1 -a 3000 -G de
```

If you use this command line option, you must have a file named *vrf_language_code* (for example, *vrf_de*) in the *.translations* directory. For more information, please see [“Localizing the Graphical User Interface,”](#) on page 2-11.



You cannot specify a language file for messages sent from the back-end.

Setting the Notification Level

You can control the types of messages sent to the console (and log file on a PC) by specifying a notification level.

- To specify a notification level, use the `-n` option, for example:

```
vrfSim -s 1 -a 3001 -n 1
```

The notification levels you can specify are:

- ♦ 0 – Only fatal messages are printed.
- ♦ 1 – Warning messages and fatal messages are printed.
- ♦ 2 – Some diagnostic information is printed with warnings and fatal messages.
- ♦ 3 – Verbose information is printed.
- ♦ 4 – Debug information is printed.

The default notification level is 2.



The Linux version of VR-Forces does not have a log file. Informational messages are sent to the window from which you start VR-Forces. To capture informational messages, redirect output to a file.

Loading Plug-ins

You can load plug-ins from the command line or by specifying them in a configuration file. (For details about creating a configuration file, please see [“Managing Plug-ins,”](#) on page 5-39.)

- To load a plug-in from the command line, use the `--loadPlugin` command line option, for example:

```
vrfGui -s 1 -a 3000 --loadPlugin pluginA
--loadPlugin pluginB
vrfSim -s 1 -a 3001 --loadPlugin pluginA
--loadPlugin pluginB
```



Do not specify the extension of the plug-in (for example, `.dll`). If the extension is included, the plug-in will not load.

VR-Forces can display a list of the plug-ins that are loaded. For details, please see [“Viewing a List of Loaded Plug-ins,”](#) on page 5-43.

5.9.5. Command Line Options for HLA Federations

VR-Forces listens to whatever port is specified in the RID file. For information about the RID file, consult your RTI documentation.

Specifying the Federation Execution

By default, VR-Forces uses the *VR-Link.fed* file and the VR-Link federation execution.

- To specify the HLA federation execution name, use the `-x` option, for example:

```
vrfsimHLA13 -s 1 -a 3001 -x vrlink
```



If you are using the MÄK RTI and are running multiple, concurrent federation executions, you must run the `rtiexec`. In other words, you cannot use the MÄK RTI in lightweight mode if you are running multiple federations. Please see the MÄK RTI documentation for details about how to configure it to use the `rtiexec`.

Specifying the Federate Name

You can specify the federate name to use for VR-Forces.

- To specify the HLA federate name, use the `-N` option, for example:

```
vrfsimHLA13 -s 1 -a 3001 -N vrforces
```

Specifying the FED File

In combined mode, specifying `-F` in the command line also applies to the back-end. If you are running a back-end independently and want to specify a FED file name, you must use `-F`.

- To specify a FED file at startup, use the `-F` option.

Specifying the RPR FOM Version

VR-Forces has built-in support for the RPR FOM. The default version is 1.0.

You can specify a different RPR FOM version (0.5, 0.7, 0.8, 2.0006, 2.0014, or 2.0017). If you specify a different FOM version, you must also specify the correct FED file (with `-F`).

```
vrfsimHLA13 -s 1 -a 3001 --rprFomVersion 2.0017  
-F VR-Link20017-1.fed
```



Please see *VR-Forces Release Notes* for a list of the RPR FOM versions supported by your version of VR-Forces.

Specifying a FOM Mapper

VR-Forces supports alternative FOMs through the VR-Link FOM Mapper. You can specify which FOM Mapper to use with the `-f` command line option:

```
vrfsimHLA13 -s 1 -a 3001 -f myFomMapper
```

Specifying the RPR FOM Revision

RPR FOM revision 2 for version 2.0017 fixes an encoding and decoding size issue with object ID lists to match RPR FOM 2 draft 17.

- To specify the RPR FOM revision, use the `--rprFomRevision revision` command-line option, for example:

```
vrfsimHLA13 -s 1 -a 3001 --rprFomVersion 2.0017  
-F VR-Link20017-1.fed --rprFomRevision 2
```

Specifying FOM Mapper Initialization Data

Some FOM Mappers require initialization data. If you specify a FOM Mapper that requires such data, specify the data with the `--fomMapperInitData` option:

```
vrfsimHLA13 -s 1 -a 3001 -f myFomMapper --fomMapperInitData  
ABC
```

The string “ABC” is passed to the FOM Mapper creation function in the shared library as an initialization parameter.

Specifying a FED File That is Appropriate for the FOM Mapper



The `-f` option only tells the application which FOM Mapper to use. It does not ensure that you are using the right FED file for the FOM Mapper. Make sure that you are using a FED file that is consistent with the FOM Mapper you have chosen.

By default VR-Forces uses a FED file called *VR-Link.fed*, but you can specify an alternate FED file using the `-F` option.

The following example instructs the application to use the RPR FOM 2.0, version 17 FOM mapping code, in the federation execution VR-Link20017, with the *VR-Link20017-1.fed* file:

```
vrfsimHLA13 -s 1 -a 3001 --rprFomVersion 2.0017  
-x VR-Link20017 -F VR-Link20017-1.fed
```

Enabling Time Management

You can enable time management for the back-end from the command line, as follows:

```
vrfSimHLA13 -s 1 -a 3001 --timeManagement -F VR-LinkTM.fed  
-x VR-LinkTM
```

5.9.6. Command Line Options for DIS Exercises

This section describes command line options that apply only to using VR-Forces with DIS exercises.

Specifying the Port Number

By default, VR-Forces listens to port 3000.

- To change the port that VR-Forces listens to, use the `-P` option, for example:

```
vrfSimDIS -s 1 -a 3001 -P 3500  
vrfGui -s 1 -a 3000 --dis -P 3500
```

Specifying Point-to-Point or Multicast Operation

You can specify that VR-Forces run in point-to-point mode or multicast.

- To specify point-to-point, use the `-A` option, for example:

```
vrfSimDIS -s 1 -a 3001 -A 123.123.12.1
```

- To specify multicast, use the `-S` option, for example:

```
vrfSimDIS -s 1 -a 3001 -S 233.233.22.3
```

You can specify more than one multicast address, as follows:

```
vrfSimDIS -s 1 -a 3001 -S address -S address ...
```



The default is broadcast mode.

Specifying the Multicast Time-to-Live

The multicast time-to-live specifies the number of routers that a message can pass through when PDUs are sent to multicast addresses.

- To specify the multicast time-to-live value, use the `--mcastTtl` option, for example:

```
vrfSimDIS -s 1 -a 3001 -S address -S address --mcastTtl 3
```

Using Asynchronous I/O

If you want to use asynchronous I/O for DIS exercises, start VR-Forces with the `--useAsyncIO` option. Using asynchronous I/O can help prevent dropped packets and the resulting loss of information. There is no comparable command line option for HLA. To use asynchronous I/O in HLA, you set a RID file parameter.

Specifying the Exercise ID

- To set the VR-Forces exercise ID, use the `-x` option, for example:

```
vrfSimDIS -s 1 -a 3001 -x 6
```

Specifying the DIS Version

- To specify the DIS version, use the `--disVersion` command line options, for example:

```
vrfSimDIS -s 1 -a 3001 --disVersion 6
```

Please see *VR-Forces Release Notes* for a list of supported versions.

5.10. Managing Plug-ins

One way to extend the functionality of VR-Forces is by loading plug-ins that have been written for it by MÄK or other software vendors. The B-HAVE Module for VR-Forces is an example of a plug-in. You can manage which plug-ins get loaded when VR-Forces starts up. You can also view information about the plug-ins that have been loaded.

Plug-ins are implemented as one or more dynamic linked libraries (DLL or SO files). The DLLs can be release or debug, for vrfSim or vrfGui, for Windows or Linux, and for a specific protocol or protocol-independent. The Plug-ins Editor lets you specify the DLLs to use for each of the desired configuration types.



Changes to plug-in configurations do not take effect until the next time you start the affected VR-Forces application.

5.10.1. Loading a Plug-in

To specify that a plug-in be loaded or not loaded:

1. Choose **Settings** → **Plugins**. The Plugins dialog box opens (Figure 5-14).

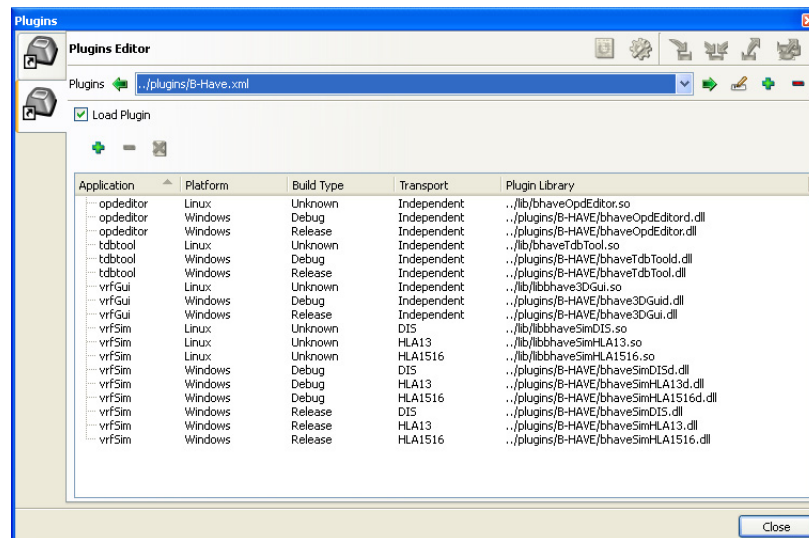


Figure 5-14. Plugins dialog box, Plugins Editor page

2. Select the Plugins Editor page.
3. In the list of plug-ins at the top of the page, select the one that you want to edit.
4. Select or clear the Load Plugin check box.

5.10.2. Adding a Plug-in

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

To add a plug-in:

1. Choose **Settings** → **Plugins**. The Plugins dialog box opens (Figure 5-14).
2. Select the Plugins Editor page.
3. Click the Add icon (+) at the upper right corner of the page. The New Plugin dialog box opens.
4. Type a name for the plug-in.
5. Click OK. The plug-in is added to the plug-in list. It has configurations for each of the combinations of application, platform, build type, and transport (Figure 5-15).

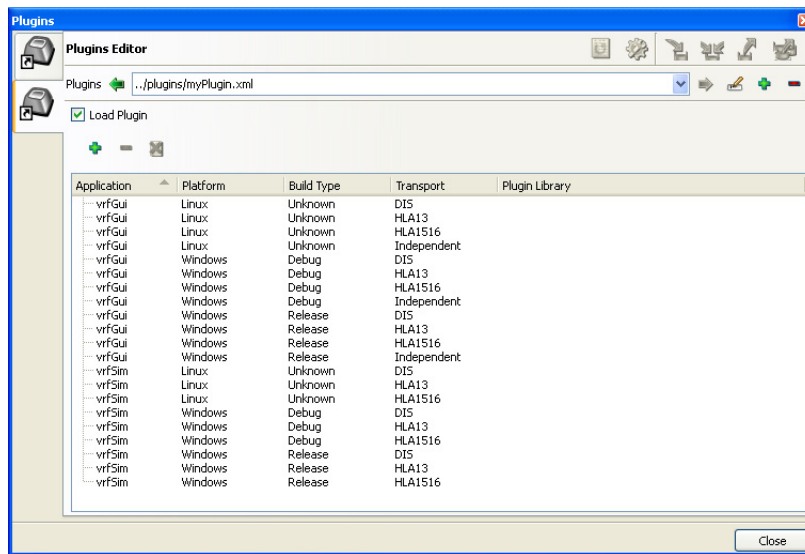


Figure 5-15. New plug-in

6. Specify the DLL for each configuration that you want to support.

5.10.3. Specifying the DLLs for a Plugin

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** → **Plugins**. The Plugins dialog box opens ([Figure 5-14](#)).
2. Select the Plugins Editor page.
3. In the list of plug-ins at the top of the page, select the one that you want to edit.
4. Click the Plugin column for the configuration that you want to edit. The Select Plugin dialog box opens.
5. Select the DLL that you want to load for this configuration.
6. Click Open. The value is changed.

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** → **Plugins**. The Plugins dialog box opens ([Figure 5-14](#)).
2. Select the Plugins Editor page.
3. Select the configuration whose DLL assignment you want to remove.
4. Click the Erase icon (.

5.10.4. Adding a Plug-in Configuration

To add a plug-in configuration:

1. Choose **Settings** → **Plugins**. The Plugins dialog box opens ([Figure 5-14](#)).
2. Select the Plugins Editor page.
3. In the list of plug-ins at the top of the page, select the one that you want to edit.
4. Click the Add icon (+) above the list of configurations. The Add Plugin dialog box opens ([Figure 5-16](#)).

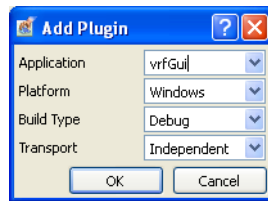


Figure 5-16. Add Plugin dialog box

5. Select a values from the drop-down lists for Application, Platform, Configuration, and Transport.
6. Click OK. The configuration is added to the list.
7. Specify a DLL for the configuration, as described in “[Specifying the DLLs for a Plugin,](#)” on page 5-41.

5.10.5. Deleting a Plug-in Configuration

To delete a plug-in configuration:

1. Choose **Settings** → **Plugins**. The Plugins dialog box opens ([Figure 5-14](#)).
2. Select the Plugins Editor page.
3. In the list of plug-ins at the top of the page, select the one that you want to edit.
4. Select the configuration that you want to delete.
5. Click the Delete icon (-) that is above the list of configurations.

5.10.6. Deleting a Plug-in

When you delete a plug-in, you delete the XML configuration file in `./plugins` that specifies the plug-in configurations and whether or not to load it.

To add a plug-in:

1. Choose **Settings** → **Plugins**. The Plugins dialog box opens (Figure 5-14).
2. Select the Plugins Editor page.
3. In the list of plug-ins at the top of the page, select the one that you want to delete.
4. Click the Delete icon () to the right of the plug-ins list.

5.10.7. Viewing a List of Loaded Plug-ins

You can view a list of the plug-ins that are loaded when you start up VR-Forces. Plug-ins may include a description that you can also view.

To view the details of loaded plug-ins:

1. Choose **Settings** → **Plugins**. The Plugins dialog box opens (Figure 5-17).
2. Select the Plugins Loaded page (Figure 5-17).

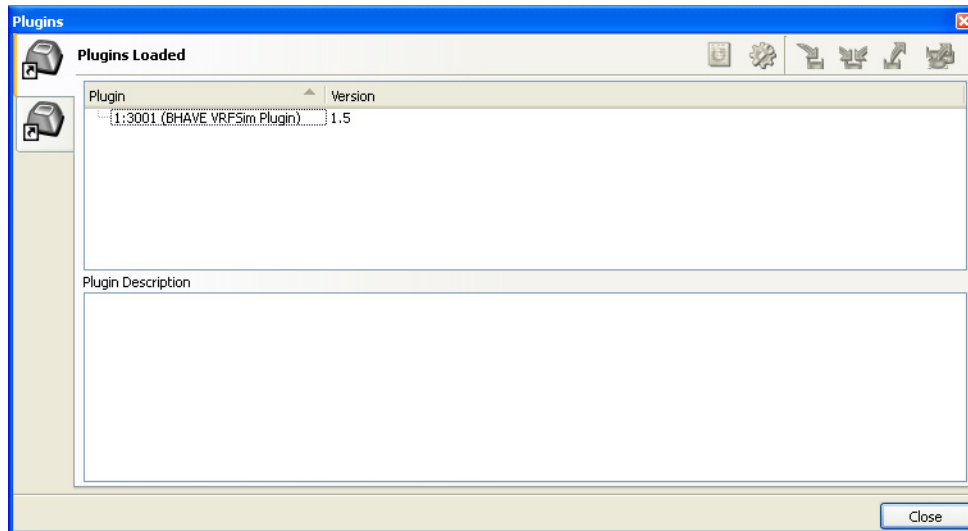


Figure 5-17. Plugins dialog box, Plugins Loaded page

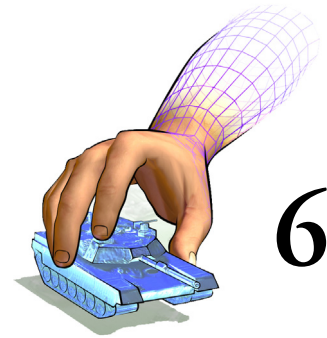
3. To view details about a plug-in, select it in the list.

5.11. The VR-Forces Log Files

On Windows, VR-Forces creates two log files, *vrfSim.log* and *vrfGui.log*. Both log files are written to the directory from which you run VR-Forces.

On Linux, in combined mode, VR-Forces automatically creates a log file, *vrfSim.log* for back-end output. Front-end output is sent to the console. If you want to view back-end output, open a second console window, and do `tail -f vrfSim.log`. If you want to send front-end output to log files, or if you are running in separate mode and want to create a log file for either the front-end or back-end, you need to redirect your output to a file, for example:

```
vrfGui > vrfGui.log
```



Managing Objects

This chapter describes procedures and features that are common to entities and tactical graphics.

The Objects List Panel	6-2
Selecting Entities, Tactical Graphics, and Props.....	6-4
Selecting Objects in the Window.....	6-5
Selecting Objects in the Objects List Panel	6-6
Unselecting Objects.....	6-6
Showing and Hiding Tactical Graphics	6-7
Showing and Hiding Tactical Graphics.....	6-8
Showing and Hiding Vertex Labels.....	6-9
Displaying Height-Above-Terrain Lines for Vertices	6-9
Displaying Information About an Object.....	6-10

6.1. The Objects List Panel

There is a great deal of similarity in how you select, move, and view the different types of objects. You can manage them on the terrain, to some extent from the keyboard, and in the Objects List Panel, which is the central repository of object information.

The Objects List Panel has the following views:

- Entity list: displays entities in the simulation (Figure 6-1)
- Environmentals list: lists tactics graphics (control objects) (Figure 6-1)
- Echelon view: displays the entity hierarchy (Figure 6-2)
- Embarkation view: displays the embarkation relationships of entities (Figure 6-2)
- Overlays view: displays overlays and the objects that are on them (Figure 6-3).

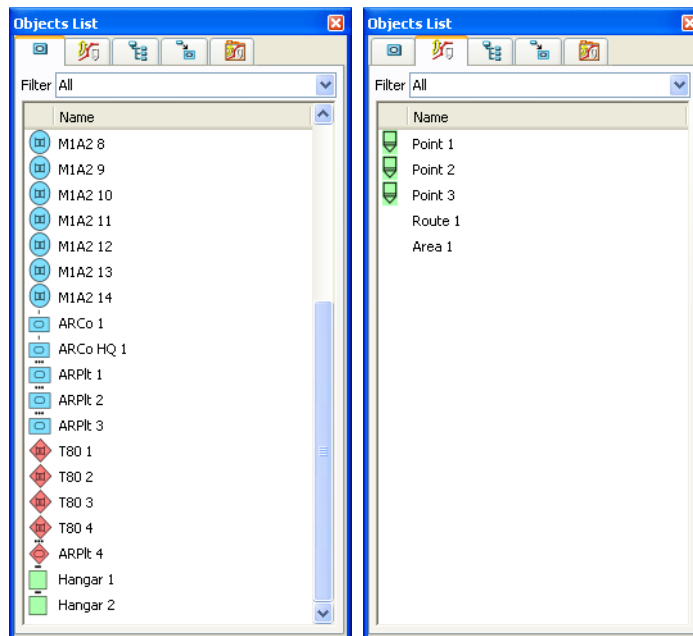


Figure 6-1. Objects List Panel – Entities view and Environmentals view

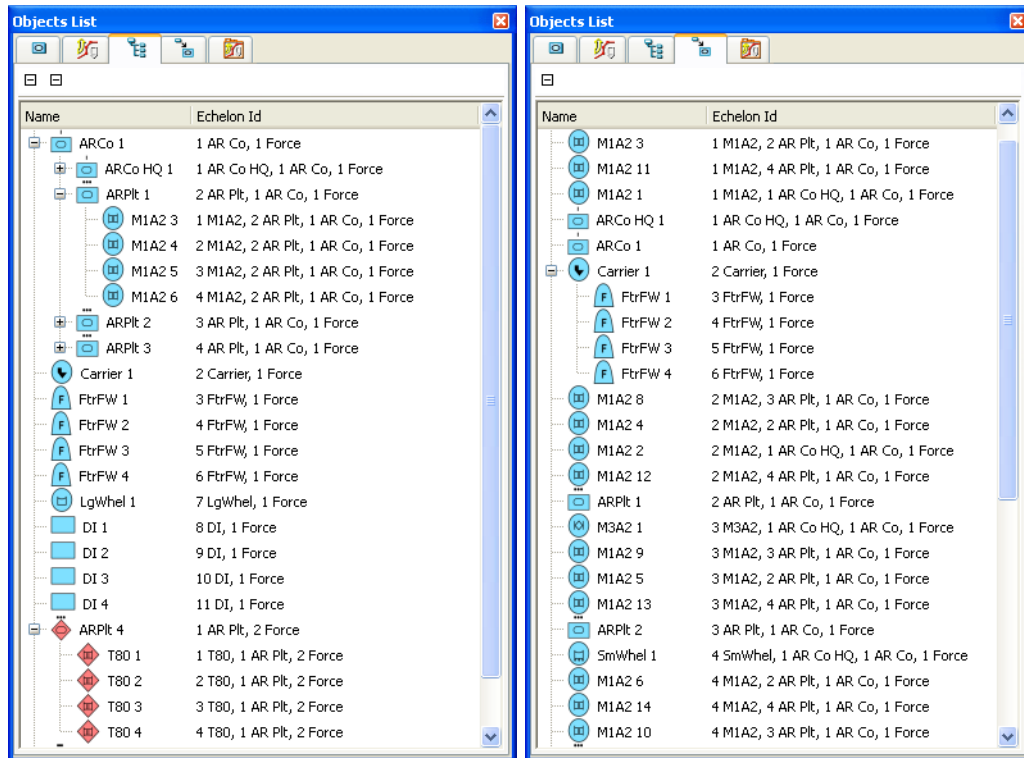


Figure 6-2. Objects List Panel – Echelon view and Embarkation view

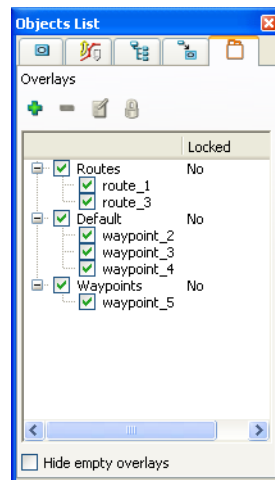


Figure 6-3. Objects List Panel – Overlays view

6.2. Selecting Entities, Tactical Graphics, and Props

VR-Forces displays the following kinds of objects that you might want to select:

- ♦ Entities. Any of the entity types that you can create from the Entity Palette or Create menu.
- ♦ Cultural features. Objects in the Building and Cultural Feature categories on the Entity Palette and Create menu.
- ♦ Tactical graphics. Waypoints, phase lines, routes, and areas on the Entity Palette and Create menu.
- ♦ Props. Any of the props that you can create from the Prop Palette or which are already part of the terrain.

You can select all types of objects in the window. You can select entities and tactical graphics in the Objects List Panel (Figure 6-1), and you can select props on the Props page of the Terrain Settings dialog box (Figure 6-4).

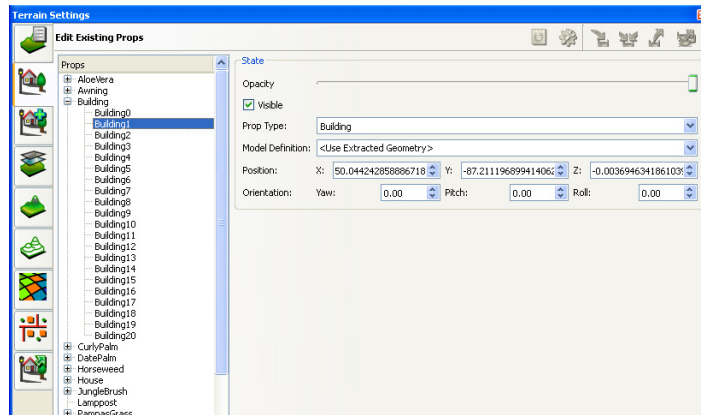


Figure 6-4. Terrain Settings dialog box, Props page

When you select an entity in the 2D view, its border color changes to white. All other objects display a bounding box. In the 3D views, all selected objects display a bounding box.

6.2.1. Selecting Objects in the Window

To select objects in the window, use any of the following mouse actions:

Mouse Action	Result
Click the object.	Selects an object (and unselects any other selected objects).
Ctrl -click the object.	Selects an object while keeping existing selections.
Double-click the object.	Selects the object and makes it editable. This is the equivalent of selecting an object and choosing Entities → Edit .
Alt -drag a bounding box.	Selects multiple objects by dragging a selection box around them.
Right-click the object.	If no objects are selected, right-clicking selects an object and displays a menu. If one or more objects are selected, right-clicking a selected object maintains the selection and displays a menu. Right-clicking an unselected object changes the selection to the object you right-clicked.

When you select an object in the window, the views on the Objects List Panel add a gray highlight to the entry for the object to indicate that it is selected.

6.2.2. Selecting Objects in the Objects List Panel

To select an object in the Objects List Panel:

1. Select a view in the Objects List Panel ([Figure 6-1](#)).
2. Locate the object you want to select. You can filter the list to make it easier to find particular objects. (For details, please see [“Filtering the Object Lists,”](#) on page 6-6.) Disaggregate and expand aggregates if necessary. (For details, please see [“Expanding and Collapsing Aggregates,”](#) on page 9-2 in this manual, and [“Changing the Aggregation State of an Aggregate Entity,”](#) on page 4-13, in *VR-Forces Configuration Guide*.)
3. Click the entry for the object. The selected object is highlighted in the Objects List Panel and on the terrain.
4. Optionally, select multiple objects, as follows:
 - **Shift**-click to select a contiguous range of entities
 - **Ctrl**-click to select multiple non-contiguous entities.

Filtering the Object Lists

You can filter the display in the Entities List and the Environmental List.

- To filter the display of objects, select a category to view from the list at the top of the tab ([Figure 6-1](#)).

6.2.3. Unselecting Objects

To unselect objects in the window, choose one of the options in [Table 6-1](#).

Table 6-1: Unselecting objects on the terrain

Action	Result
Click in the window away from the selected object or objects.	Unselects all objects.
Right-click.	Exits edit mode, but object remains selected.
Click a selected object.	If multiple objects were selected, unselects all except the object you clicked.
Ctrl -click a selected object.	Unselects the object. If multiple objects were selected, does not unselect other objects.

To unselect objects on the Objects List Panel, choose one of the options in [Table 6-2](#).

Table 6-2: Unselecting objects in the Objects List Panel

Action	Result
If one object is selected.	Ctrl -click the object.
If a group of objects is selected:	
Ctrl -click an object.	Unselects the object without affecting other selections.
Click an object.	Selects the clicked object and unselects all other objects.

6.3. Showing and Hiding Tactical Graphics

Tactical graphics are graphical objects that represent points, lines, and areal objects used for various purposes in simulations ([Figure 6-5](#)). They could represent routes that entities are following, a destination point, or a way to show some sort of relationship, such as height-above-terrain lines. You can configure the display of tactical graphics as follows:

- Enable and disable the display of tactical graphics
- Show or hide vertex indicators for point, linear, and areal objects
- Show or hide height-above-terrain lines
- Display height-above-terrain lines only for selected graphics.

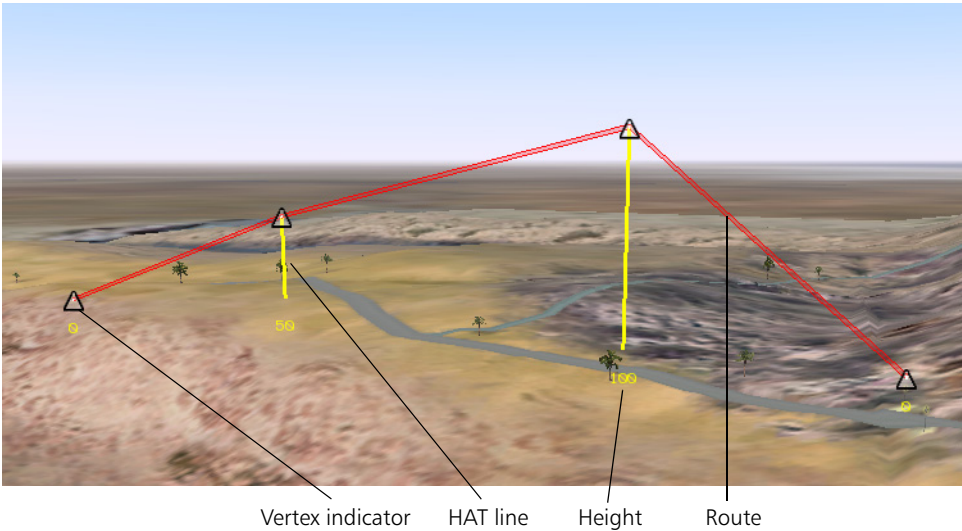


Figure 6-5. Tactical graphics height-above-terrain lines

6.3.1. Showing and Hiding Tactical Graphics

To show or hide tactical graphics:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Tactical Graphics Display Settings page (Figure 6-6).

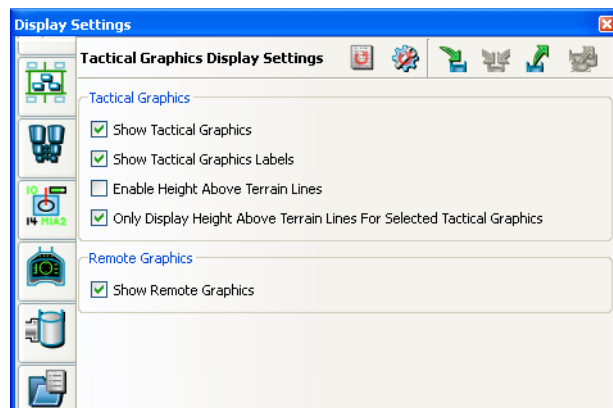


Figure 6-6. Tactical Graphics Display Settings page

3. Select or clear the Show Tactical Graphics check box.



You can also enable and disable tactical graphics by choosing **Settings** → **Tactical Graphics Labels** or clicking the the Tactical Graphics icon () on the Display Settings toolbar.

6.3.2. Showing and Hiding Vertex Labels

In addition to displaying models for tactical graphics themselves, you can display graphics (labels) for the vertices of these objects. Vertex labels are enabled in [Figure 6-5](#).



Displaying vertex labels can affect performance.

To show or hide vertex labels:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Tactical Graphics Display Settings page ([Figure 6-6](#)).
3. Select or clear the Show Tactical Graphics Labels check box.



You can also enable and disable tactical graphics labels by choosing **Settings** → **Tactical Graphics Labels** or clicking the the Tactical Graphics Labels icon () on the Display Settings toolbar.

6.3.3. Displaying Height-Above-Terrain Lines for Vertices

You can display height-above-terrain (HAT) lines for the vertices of tactical graphics. These vertical lines between an object's vertices and the ground help you understand the relationship of the object to the terrain. The height, in meters, for each vertex is displayed ([Figure 6-5](#)).

HAT lines are shown only for tactical graphics that could be above or below the ground, such as routes, and waypoints. You can show and hide these lines and you can specify that they be shown for all objects or only for the selected objects.



Use of height above terrain lines can reduce the frame rate.

To show or hide height-above-terrain lines:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Tactical Graphics Display Settings page ([Figure 6-6](#)).
3. To display HAT lines, select the Enable Height Above Terrain Lines check box.
4. To only display lines for selected objects, select the Only Display Height Above Terrain Lines For Selected Tactical Graphics check box.

6.4. Displaying Information About an Object

You can display information about individual objects, including entities, aggregates, environmental objects, and Stealths (Figure 6-7 and Figure 6-8). You cannot edit this information. You can also display entity labels. For details about entity labels, please see “Displaying Entity Labels,” on page 7-2.

To display an object information dialog box:

1. Select an object.
2. Choose **Entities** → **Information**, or press **i**.

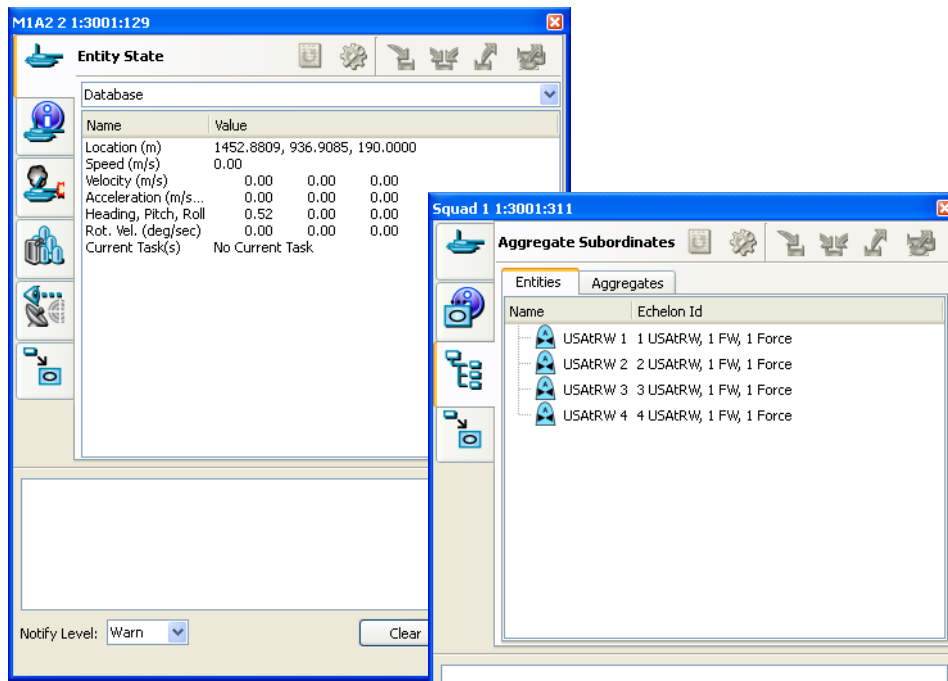


Figure 6-7. Entity and aggregate information dialog boxes

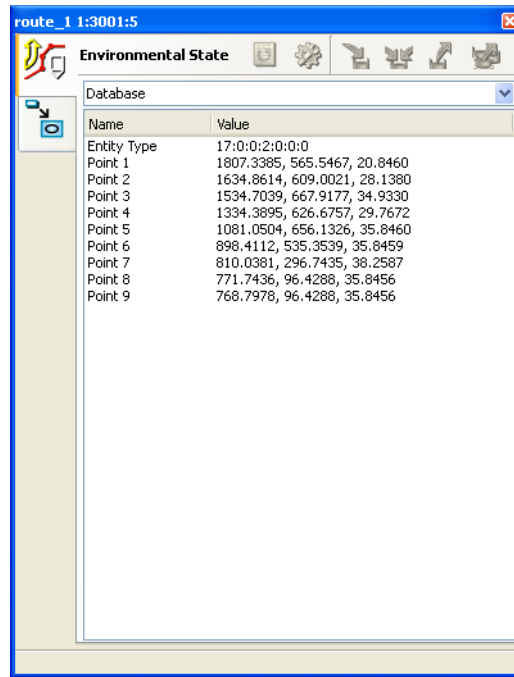
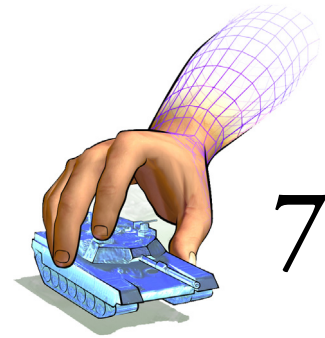


Figure 6-8. Environmental object information dialog box



Viewing Entities and Entity Information

This chapter describes how to get information about entities and how to attach the observer to entities.

Displaying Entity Labels	7-2
Displaying Text Labels for 3D Models	7-4
Turning Entity Labels On and Off	7-5
Pinning 3D Text Labels to the Window	7-5
Displaying Entity Labels for 2D Icons	7-6
Entity Icons	7-8
Changing the Size of 2D Icons	7-9
Model Sets	7-10
Changing the Model Set	7-10
Exaggerating the Scale of 3D Entity Models	7-10
Changing Force Colors	7-13
Displaying Height-Above-Terrain Lines for Entities	7-14
Displaying Entity Resources	7-15
Viewing Numerical Resource Data	7-15
Configuring Entity Console Messages	7-16
Setting the Notification Level for Console Messages	7-16
Logging Console Messages to a File	7-17
Clearing the Object Console	7-17

7.1. Displaying Entity Labels

VR-Forces provides extensive information about entities in the Entity Information dialog box (“[Displaying Information About an Object](#),” on page 6-10). It also provides an abbreviated set of information in the form of entity labels.

For 3D models, VR-Forces displays entity labels that help you locate and identify entities (Figure 7-1). The entity label consists of an entity locator circle, which is always visible when entity labels are enabled, and a text label that is displayed when you mouse over an entity. All entities within the view have locator circles, whether the model is visible or not. The color of an entity’s label represents its force – blue for friendly, red for opposing, and green for neutral.

A 3D text label displays the entity’s:

- ♦ Marking text
- ♦ Entity type enumeration
- ♦ Force
- ♦ Name
- ♦ Speed
- ♦ Heading
- ♦ Latitude and longitude
- ♦ Altitude.

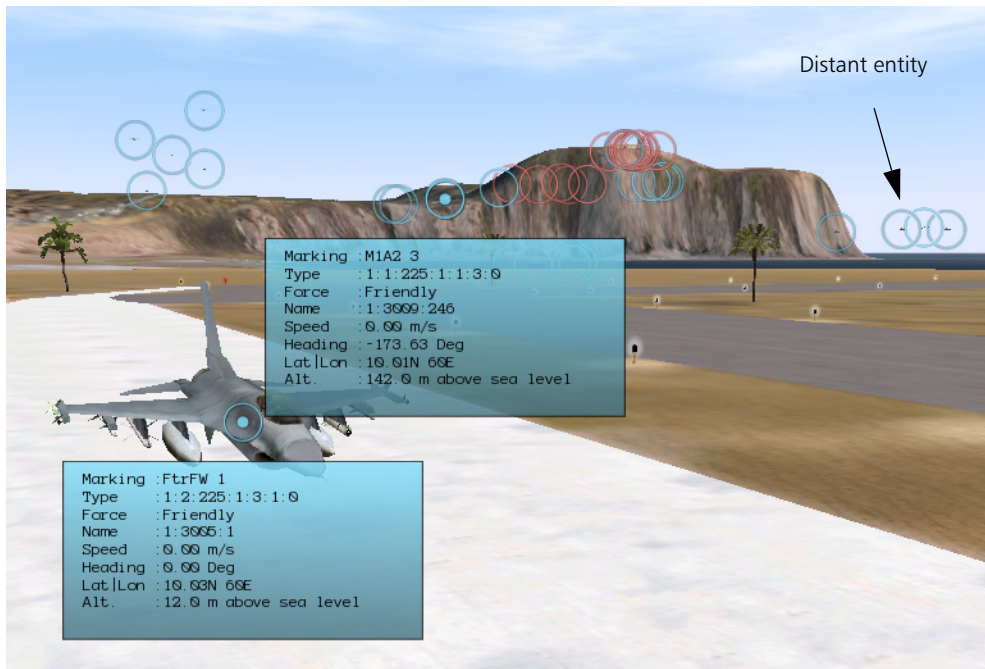


Figure 7-1. Entity labels for 3D models

For 2D icons, entity labels are a set of individually enabled text labels (Figure 7-2). Entity labels for 2D icons are supported both in the 2D view and the 3D view. If you resize 2D icons, the labels also get resized. 2D icons can display the entity's:

- ♦ Name
- ♦ Orientation
- ♦ Velocity
- ♦ Acceleration
- ♦ Heading Indicator.

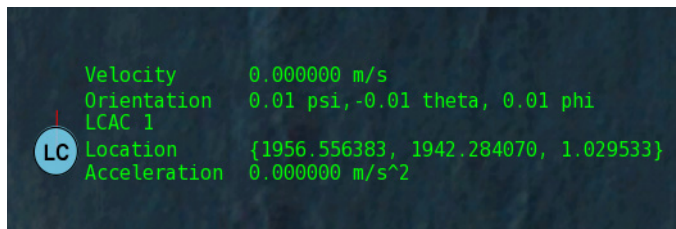


Figure 7-2. Entity labels for 2D icons

7.1.1. Displaying Text Labels for 3D Models

To view a 3D text label, use one of the following methods:

- ♦ Move the cursor over the circle that represents the entity.
- ♦ On the Display Settings dialog box, Entity Display Settings page, select Only Display Labels for Selected Entities (Figure 7-3). Then select the entity whose label you want to see.

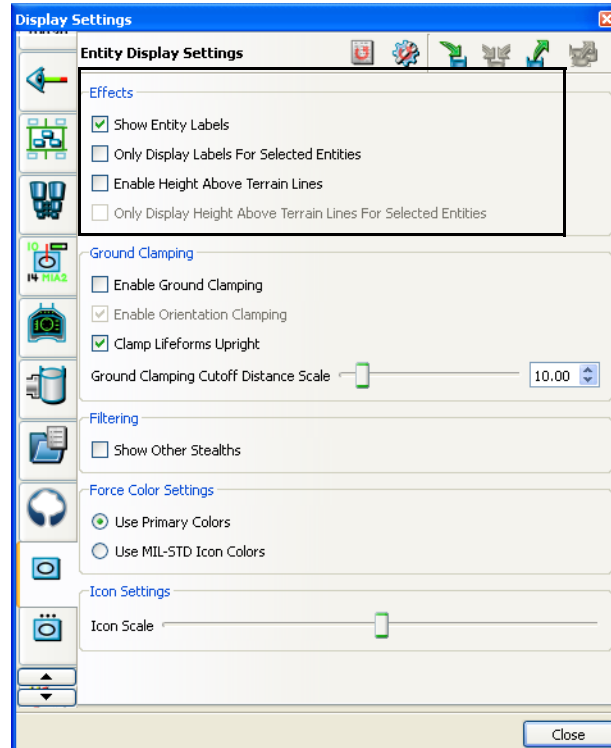


Figure 7-3. Display Settings, Entity Display Settings page

7.1.2. Turning Entity Labels On and Off

To turn the display of entity labels on or off:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Entity Display Settings page (Figure 7-3).
3. Select or clear the Show Entity Labels check box.



You can also enable and disable entity labels by choosing **Settings** → **Entity Labels** or by clicking the Entity Labels icon () on the Display Settings toolbar.

These settings do not affect the labels for 2D icons.

7.1.3. Pinning 3D Text Labels to the Window

You can pin 3D text labels to the window, making them stay visible after you move the cursor away from the entity.

To pin a 3D text label to the window or hide a pinned label:

1. Right-click the entity's locator circle. A menu is displayed.
2. Choose **Toggle Pinned Label**.

Pinning Multiple 3D Text Labels

You can display multiple text labels at one time.

To pin multiple text labels use either of the following methods:

- ♦ Right-click and choose **Toggle Pinned Label** for each entity whose label you want to pin.
- ♦ On the Display Settings dialog box, Entity Display Settings page, select Only Display Labels for Selected Entities (Figure 7-3). Then, in the Objects List Panel, multi-select the entities whose labels you want to see.

7.1.4. Displaying Entity Labels for 2D Icons

You can enable and disable the display of labels for 2D icons by label type. This allows you to control how much screen space is taken up by the labels. You can control the display of entity labels in the Display Settings dialog box or on the Display Settings toolbar.

To enable and disable display of 2D icon labels:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Symbol Decoration Settings page (Figure 7-4).

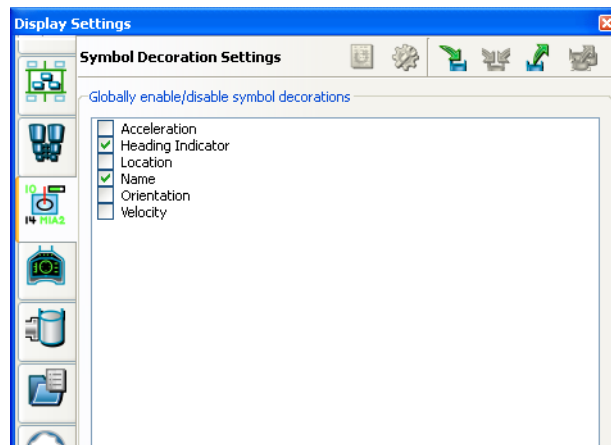


Figure 7-4. Symbol Decoration Settings page

3. Select the check box for each label that you want to display. Clear the check box for each label that you want to hide.

Enabling 2D Labels on the Display Settings Toolbar

The Entity Labels icon on the Display Settings toolbar lets you open a panel that lists the symbol decorations that you can show or hide.

To enable or disable 2D labels on the Display Settings toolbar:

1. Click the right-most arrow on the Entity Icon Scale icon (↔↕↔). A small panel is displayed. It is a persistent window.

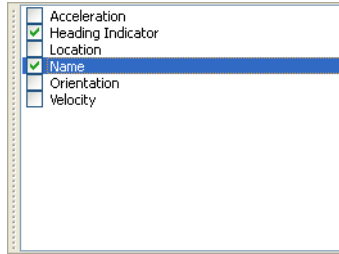


Figure 7-5. Symbol Decoration panel

2. Select the symbol decorations that you want to display. Clear the ones you want to hide.
3. To hide the symbol decorations panel, click the right-most arrow on the Entity Icon Scale icon.

7.2. Entity Icons

The 2D Icons model set uses MIL-STD 2525B icons. The figure [Figure 7-6](#) illustrates basic icon shapes for friendly forces.

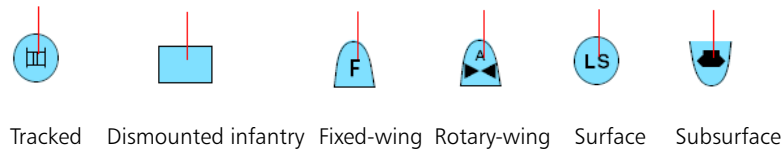


Figure 7-6. MILSTD-2525B icons

The color and shape code for 2D entity icons is:

- ♦ Light blue – friendly
- ♦ Salmon – hostile
- ♦ Light green – neutral.

Symbols have a black border except as follows:

- ♦ If there is no border on the bottom, the entity domain is above ground, for example, helicopters.
- ♦ If there is no border on the top, the entity domain is subsurface, for example, submarines.
- ♦ Selected entities have a white border.

Symbols that represent aggregate entities have aggregate identifiers above the symbol, as illustrated in [Figure 7-7](#).

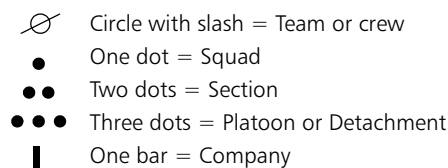


Figure 7-7. Unit echelon indicators

7.2.1. Changing the Size of 2D Icons

The Plan View observer mode uses the 2D Icons model set for entities. (You can also use these icons in the 3D observer modes.) You can change the size of 2D entity icons. When many entities are close together, this can reduce overlap among icons. (When you change the size of 2D icons, the labels, if enabled, also change size.)

To change the size of entity icons:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Entity Display Settings page (Figure 7-8).

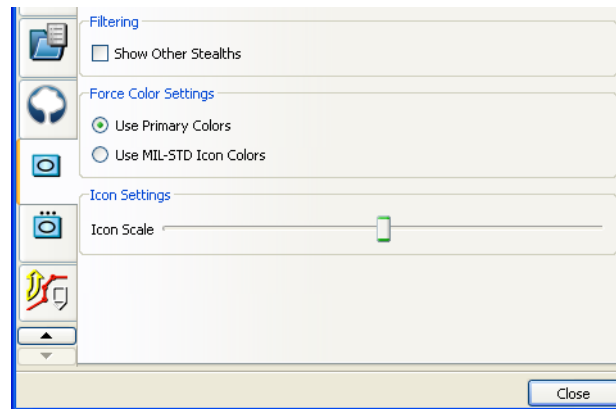


Figure 7-8. Entity Display Settings page

3. In the Icon Settings group box, adjust the Icon Scale slider.

You can also change the size of 2D icons from the Display Settings toolbar.

To change the size of entity icons from the toolbar:

1. On the Display Settings toolbar, click the left-most arrow on the Entity Icon Scale icon (🔍 ↕). The Entity Icon Scale Factor slider opens.
2. Adjust the slider to change the size of the icons.



The Entity Icon Scale Factor slider is a dockable widget that you can undock and leave visible as long as icon scaling is enabled.

7.3. Model Sets

VR-Forces includes the following model sets:

- ♦ 3D Models. Realistic, 3D, textured models with articulated parts.
- ♦ 3D Colorized Models. 3D model caricatures with minimal articulation. These models do not have textures, they are solid colors, by force. You can change the coloring between primary colors and MIL-STD icon colors.
- ♦ 2D icons. MIL-STD 2525B icons.

You can use any of the model sets in both 3D and 2D views. In the 2D view, the 3D models can only be viewed from the top.

7.3.1. Changing the Model Set

Each observer mode specifies a model set. You can change the model set associated with an observer mode by editing it (for details, please see “[Editing an Observer Mode](#),” on page 10-4), but you can also dynamically change the model set being used.

- To change the model set, choose **Observer** → **Set Model Set** → *model_set*, where *model_set* is one of the option on the menu, or select a model set from the Set Model Set icon () on the Observer Settings toolbar. (The icon changes to show the currently selected model set.)

7.4. Exaggerating the Scale of 3D Entity Models

If you view the terrain in Stealth observer mode, beyond a certain distance most 3D entity models are too small to see. Model scaling lets you exaggerate the scale of entity models to see them at a great distance. [Figure 7-9](#) shows the same scene with model scaling off and on. When model scaling is off, only the largest entities are visible. When models are scaled, they are all visible.



The visual effect of model scaling depends on the observer’s distance from an entity. When the observer is close to an entity, scaling models may not make any noticeable difference in its size.

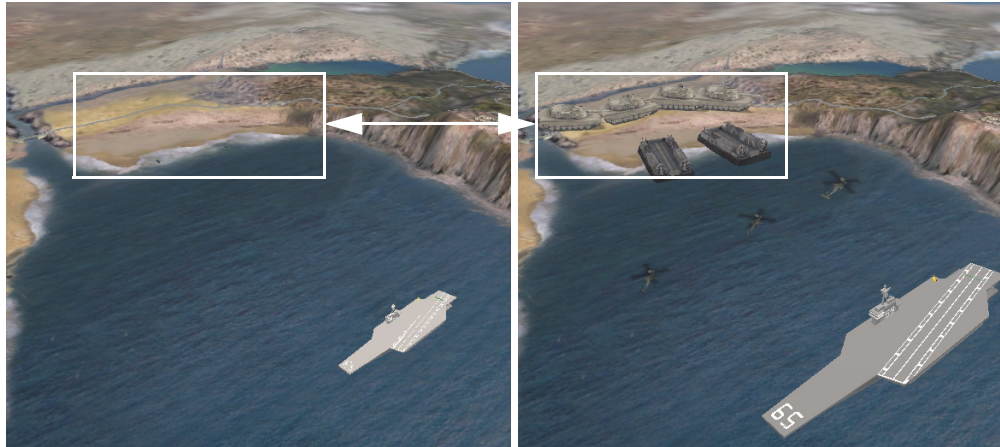


Figure 7-9. Model scaling off and on

Another way to quickly change the visibility of entities is to switch to XR observer mode. In XR mode, entities are immediately scaled and are represented by colorized 3D models to enhance their visibility.



Figure 7-10. XR mode

To scale entity models:

1. Choose **Observer** → **Advanced Settings**. The Display Settings dialog box opens to the Observer Mode Settings page (Figure 7-11).

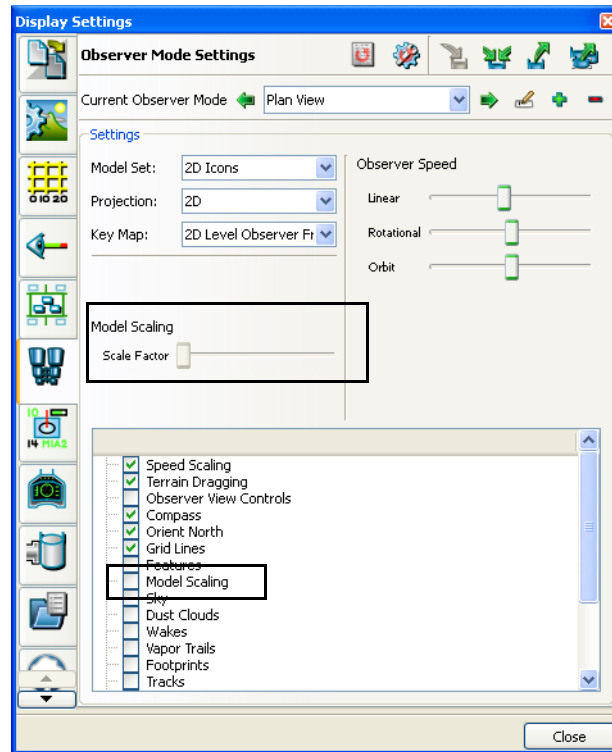


Figure 7-11. Observer Mode Settings page

2. In the list of observer attributes, select Model Scaling.
3. Adjust the Scale Factor slider to the amount of scaling that you want.

You can also scale models from the Observer Settings toolbar and the Observer Control Panel.

To scale models from the Observer Settings toolbar:

1. If model scaling is not enabled, click the Model Scaling icon () on the Observer Settings toolbar.
2. Click the down arrow on the icon. The Scale Factor slider is displayed.



Figure 7-12. Scale Factor slider

3. Adjust the slider to scale the models.



The Scale Factor slider is a dockable widget that you can undock and leave visible as long as model scaling is enabled.

To scale models from the Observer Control panel:

1. If model scaling is not enabled, enable model scaling in the Observer Mode Settings page or click the Model Scaling icon () on the Observer Settings toolbar. The scale on the Observer Control Panel becomes enabled ([Figure 7-13](#)).

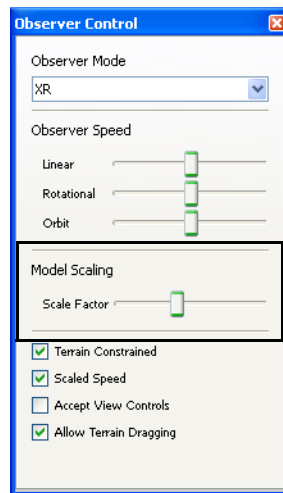


Figure 7-13. Observer Control Panel

2. Adjust the slider to scale the models.

7.5. Changing Force Colors

By default, entity labels, detonation lines, and other force-related entity graphics (including the 3D Colorized Model model set) are displayed using primary colors (red, blue, and green). You can change the force coloring to use MIL-STD 2525B colors (light blue, light green, and salmon).

To change force coloring:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Entity Display Settings page ([Figure 7-8](#)).
3. In the Force Color Settings group box, select the coloring option that you want.

7.6. Displaying Height-Above-Terrain Lines for Entities

Height-above-terrain (HAT) lines are vertical lines that run from an entity to the ground. They help you understand the relationship of the entity to the terrain. The height, in meters, is displayed ([Figure 7-14](#)).

HAT lines are shown only for entities that could be above or below the ground, such as fixed and rotary-wing vehicles. You can show and hide these lines globally and you can specify that they be shown only for the selected entities.

When you create or edit an entity, VR-Forces displays HAT lines even if they are otherwise disabled.

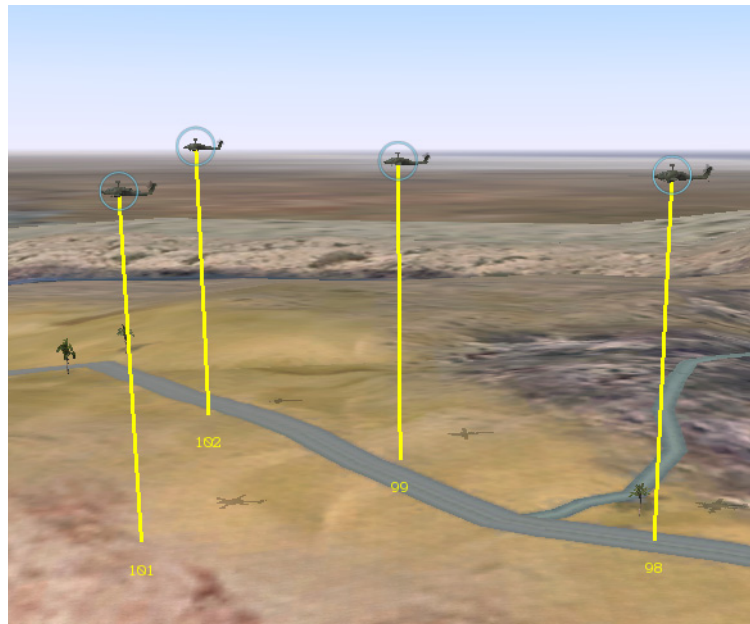


Figure 7-14. Entity height-above-terrain lines



Displaying height above terrain lines can reduce the frame rate.

To show or hide height-above-terrain lines:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Entity Display Settings page ([Figure 7-3](#)).
3. To display lines, select the Enable Height Above Terrain Lines check box.
4. To display lines only for selected objects, select the Only Display Height Above Terrain Lines For Selected Entities check box.



You can also enable and disable entity HAT lines by choosing **Settings** → **Entity Height Above Terrain Lines** or by clicking the Entity Height Above Terrain Lines icon () on the Display Settings toolbar.

7.7. Displaying Entity Resources

VR-Forces tracks the amount of resources that an entity has. You can view the status of resources in the Entity Information dialog box.

7.7.1. Viewing Numerical Resource Data

The Entity Information dialog box lists the entity's current amount of resources, its starting amount, and the percentage remaining ([Figure 7-15](#)).

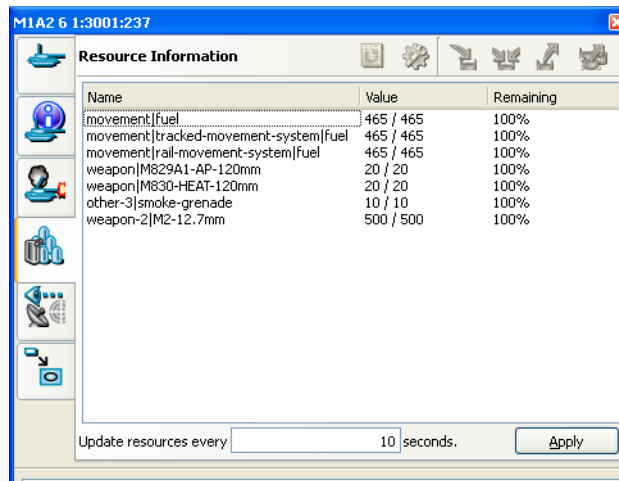


Figure 7-15. Entity resources

- To view entity resource status, on the Entity Information dialog box, select the Resource Information page.

7.8. Configuring Entity Console Messages

The Object Console of the Entity Information dialog box displays messages sent from the simulation engine, from an entity's plan, and from other entities. You can set the notification level that controls the display of these messages. You can save them to a log file, and you can clear the display.

7.8.1. Setting the Notification Level for Console Messages

Console messages for an entity are displayed only if they match or exceed the notification level set for that entity. The entity notification level is set individually for each entity. You can set the notification level in the Entity Information dialog box, with the Set Notify Level set data request, or with the `objectConsoleNotifyLevel` parameter in *vrfSim.mtl*. (For the set data request procedure, please see “Set Notify Level,” on page 7-21.)

To set the notification level for an entity's console in the Entity Information dialog box:

1. Select the entity.
2. Choose **Entities** → **Information**. The Entity Information dialog box opens. The console is the panel at the bottom of the dialog box (Figure 7-16).

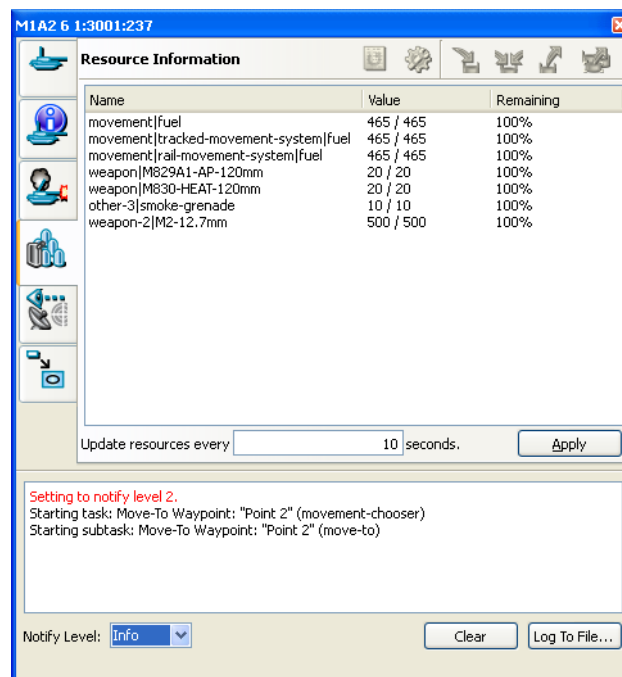


Figure 7-16. Entity Information dialog box, Object Console

3. Select a notification level from the Notify Level list.

7.8.2. Logging Console Messages to a File

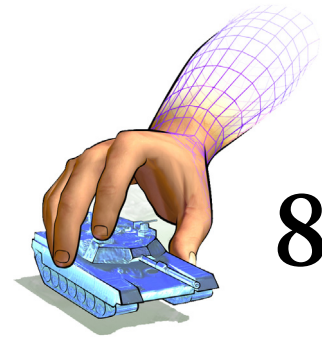
You can save the messages that are sent to the entity console to a log file. Messages are saved only for the current run of the simulation.

To log entity console messages to a file:

1. Select the entity.
2. Choose **Entities** → **Information**. The Entity Information dialog box opens ([Figure 7-16](#)).
3. Click Log to File. The Log to File dialog box opens.
4. Type a name for the log file.
5. Click Save.

7.8.3. Clearing the Object Console

- To clear the messages from the object console, in the Console group box, click Clear.



Managing Entity Effects

This chapter describes how to manage special visual effects that can be applied to entities and the VR-Forces scene.

Using Ground Clamping	8-2
Configuring Ground Clamping.....	8-3
Trajectory Smoothing.....	8-4
Configuring Trajectory Smoothing.....	8-5
Inset Views.....	8-6
Displaying Trailing and Decal Effects.....	8-7
Specifying Models for Trailing Effects and Decal Effects.....	8-8
Displaying Track Histories	8-9
Viewing Fire and Detonation Graphics	8-10
Displaying Sensor Volumes	8-11
Displaying Cockpit Displays	8-12
Displaying Shadows	8-13
Configuring Shadows	8-14
Disabling Lighting and Clouds	8-14

8.1. Using Ground Clamping

Because of differences in terrain databases among exercise participants, entities can sometimes appear to be underground or hovering above the terrain surface. To compensate, VR-Forces can clamp land and sea entities to the terrain surface. When ground clamping is enabled, VR-Forces keeps an entity anchored to the terrain surface regardless of the altitude data contained in its state update.



Use of ground clamping can reduce your frame rate. If you correlate the simulation and visualization terrain databases, you can eliminate the need to use ground clamping.

To enable or disable ground clamping:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Entity Display Settings page (Figure 8-1).

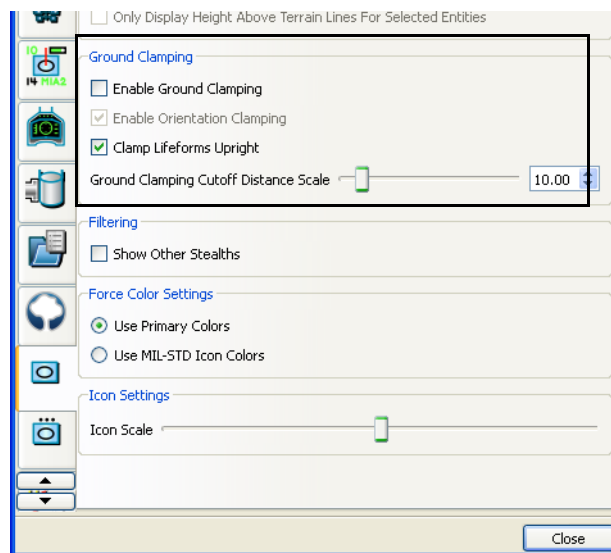


Figure 8-1. Ground clamping settings

3. Select or clear the Enable Ground Clamping check box.



You can also enable and disable ground clamping by choosing **Settings** → **Ground Clamping**, or clicking the Ground Clamping icon () on the Display Settings toolbar.

8.1.1. Configuring Ground Clamping

You can configure the following aspects of ground clamping:

- ♦ Enable or disable ground clamping
- ♦ Enable or disable orientation clamping
- ♦ Force lifeforms to remain upright relative to the plane of their location when moving up slopes or steps, rather than being perpendicular to the terrain
- ♦ The distance from the terrain within which entities are clamped.

To configure ground clamping:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Entity Display Settings page (Figure 8-1).
3. To enable or disable a ground clamping option, select or clear its check box.
4. To specify the distance from the terrain in which entities are ground clamped, adjust the Ground Clamping Cutoff Distance Scale slider or type a value, in meters, in the spinner box. You can also access the Ground Clamping Cutoff Distance Scale slider by clicking the down arrow next to the Ground Clamping icon () on the Display Settings toolbar.



The setting for the Ground Clamping Cutoff Distance Scale maximizes at 100 meters. If you set it to the maximum, the distance is infinite and all entities are ground-clamped.

8.2. Trajectory Smoothing

From the time VR-Forces receives a state update message for an entity, until it receives the next update, it calculates the location of the entity based on the heading and acceleration in the most recent state update. This process is called dead-reckoning. When the next state update arrives for the entity, VR-Forces moves it to the new position. If the actual position is significantly different from the dead-reckoned position, the entity could appear to jump from the dead-reckoned position to the actual position. VR-Forces can eliminate these discontinuities by smoothing entity trajectories and rotation paths.

Figure 8-2 illustrates dead-reckoning and smoothing.

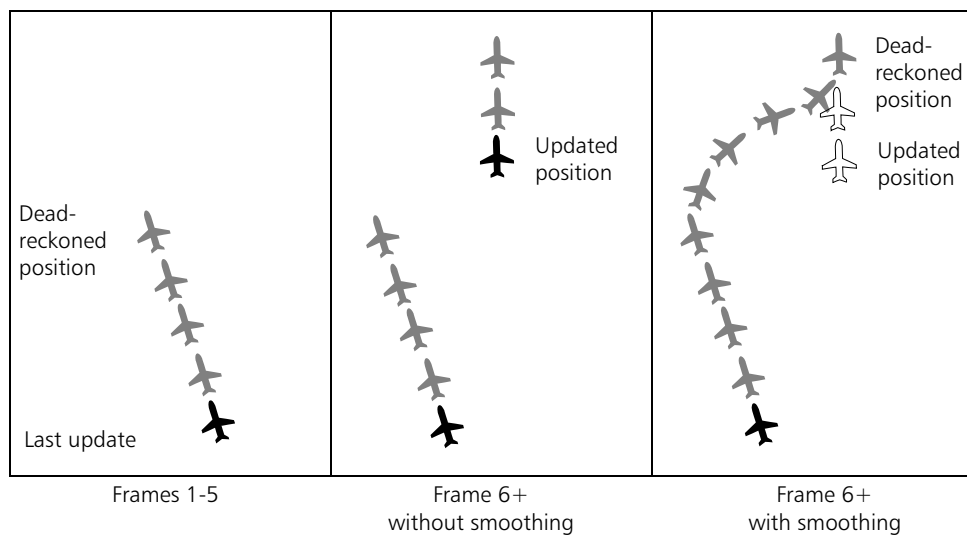


Figure 8-2. Dead-reckoning and smoothing

The smooth period is the period of time over which smoothing takes place. The default period is 1/10 second. If course corrections appear abrupt with this setting, you may want to increase the smooth period.



Smoothing makes fast-moving entities, such as fixed wing aircraft, look better. Slow moving entities, such as dismounted infantry, may look worse. If your simulation is mostly one or the other of these types of entities, enable or disable smoothing as appropriate for the best visualization experience.

8.2.1. Configuring Trajectory Smoothing

To configure trajectory smoothing:

1. Choose **Settings** → **Application**. The Application Settings dialog box opens.
2. Select the Performance Options page (Figure 8-3).

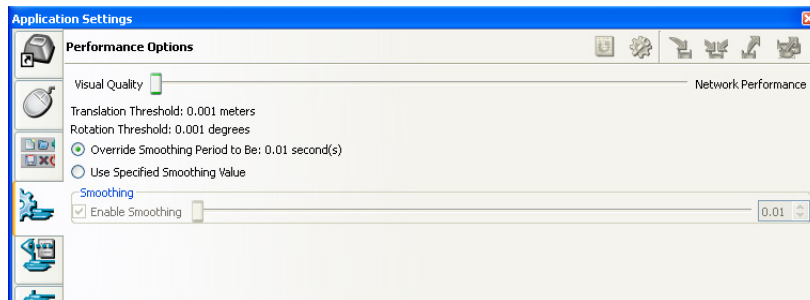


Figure 8-3. Performance Options page

3. Select the User Specified Smoothing Value option. The Enable Smoothing check box becomes enabled.
4. To enable or disable smoothing, select or clear the Enable Smoothing check box.
5. To change the smooth period, adjust the Smoothing Period slider or type a value, in seconds, in the spinner box.

8.3. Inset Views

You can display inset views of individual entities. Figure 8-4 shows a helicopter hovering over the runway in the Makland terrain with an inset view of an M1A2 tank driving through the town.



Figure 8-4. Inset view



By default, when you create an inset view, the observer in the inset view is attached to the entity in Mimic mode. This has the side effect of causing the model in the 3D view to disappear. (The reverse occurs if you attach the observer in the 3D view to the entity in Mimic mode.) To make the model reappear, select the inset view and change the attach mode or detach the observer.

You can change the observer position and orientation in an inset view using keyboard controls. You can also edit window and channel properties for the window in the Display Configuration Panel.

To open an inset view of an entity:

1. Right-click the entity (or its entry in the Objects List Panel). A menu opens.
2. On the menu, choose **Create Inset View**.

8.4. Displaying Trailing and Decal Effects

VR-Forces supports trailing effects such as vapor trails for missiles and aircraft and dust clouds for vehicles. It also supports decal effects, such as tire tracks, footprints, and wakes. In essence, trailing effects are effects that are dispersed through the air, while decal effects are effects that are printed on the ground.



Trailing and decal effects are not supported in the 2D view.

Trailing effects and decal effects are properties of an observer mode. You can enable and disable them on a per-observer basis.

Figure 8-5 illustrates some trailing effects.



Figure 8-5. Tank tracks, dust cloud

To enable or disable trailing effects or decal effects:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Observer Mode Settings page (Figure 8-6).

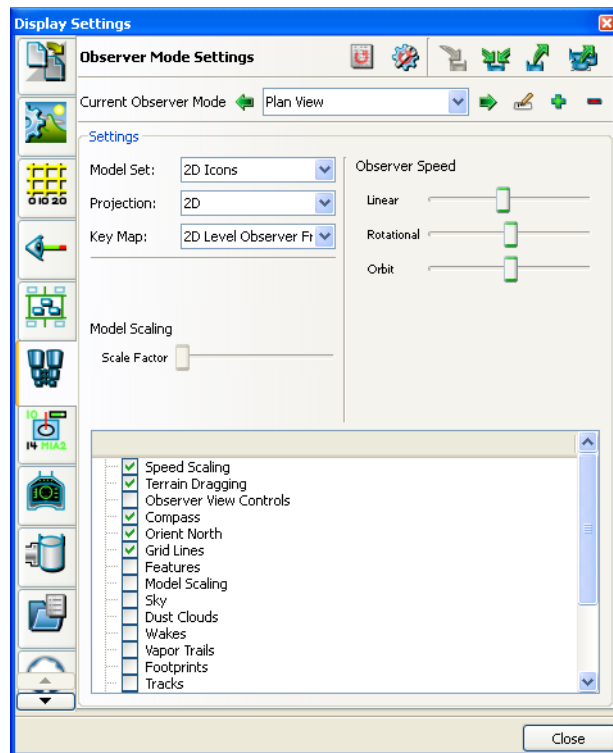


Figure 8-6. Display Settings dialog box, Observer Mode Settings

3. Select or clear the check box for each effect that you want to change.



You can also enable and disable trailing and decal effects by choosing **Observer** → *effect* or clicking the appropriate icon on the Observer Settings toolbar.

8.4.1. Specifying Models for Trailing Effects and Decal Effects

Trailing effects and decal effects are configured through model definitions. (For the purposes of configuration, trailing effects are categorized as decals.) Each effect must have a model definition that uses a decal schema. Then, in an entity's entity definition, you configure a visualizer for the effect. For details about creating model definitions, please see Section 13.2, "[Creating and Editing Model Definitions](#)", in *VR-Forces Configuration Guide*.

8.5. Displaying Track Histories

Track histories allow you to track the paths of entities during an exercise, including the path of munitions as they travel from the shooter to the target. When track histories are enabled, each entity leaves a track ribbon. The ribbon is colored based on the entity's force. [Figure 8-7](#) illustrates track histories.

Track histories are a property of an observer mode. You can enable and disable them on a per-observer basis.



Figure 8-7. Track history

To enable or disable display of all track histories:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Observer Mode Settings page ([Figure 8-6](#)).
3. Select the observer mode for which you want to change the Track Histories setting.
4. Select or clear the Track Histories check box.



You can also enable or disable track histories by choosing **Observer** → **Track Histories** or clicking the Track Histories icon () on the Observer Settings toolbar.

8.6. Viewing Fire and Detonation Graphics

If fire and detonation graphics are enabled, when an entity fires a munition, a line is drawn between the entity and the location of the detonation. The line is colored blue, red, or green for the force of the shooter (Figure 8-8).

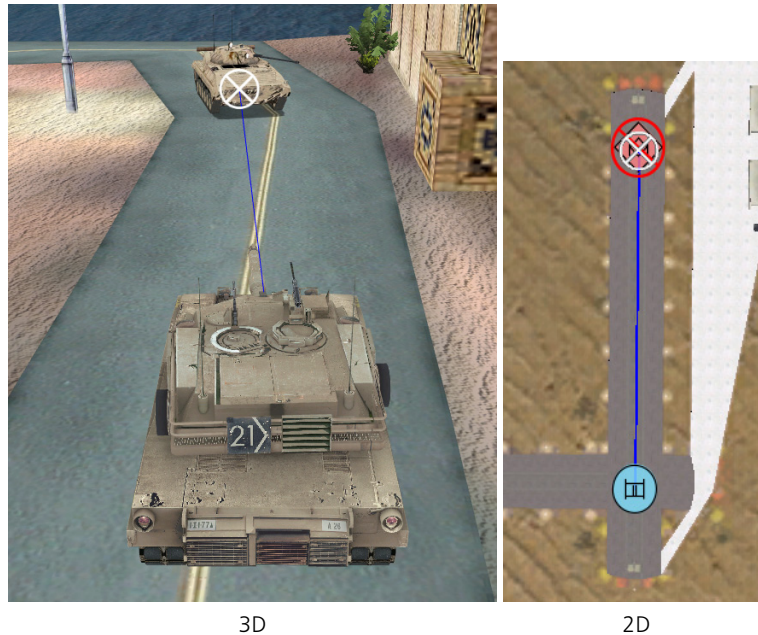


Figure 8-8. Fire line and detonation

A detonation is identified by a circle with an X through it. The detonation color indicates the result of the detonation, as follows:

- ♦ Entity impact – white
 - ♦ Ground impact – gray
 - ♦ Airburst – black
 - ♦ Other – white.
- To enable or disable display of fire and detonation graphics, choose **Settings** → **Fire / Detonate Lines** or click the Fire/Detonate Lines icon (🔥) on the Display Settings toolbar.

8.7. Displaying Sensor Volumes

VR-Forces can display sensor volumes for entities that have emitter systems (Figure 8-9). You can configure the colors used to represent emitter frequencies and you can configure the size of the sensor volumes. (For details about configuring sensor volumes, please see Section 8.7, “Configuring Sensor Volumes”, in *VR-Forces Configuration Guide*.)

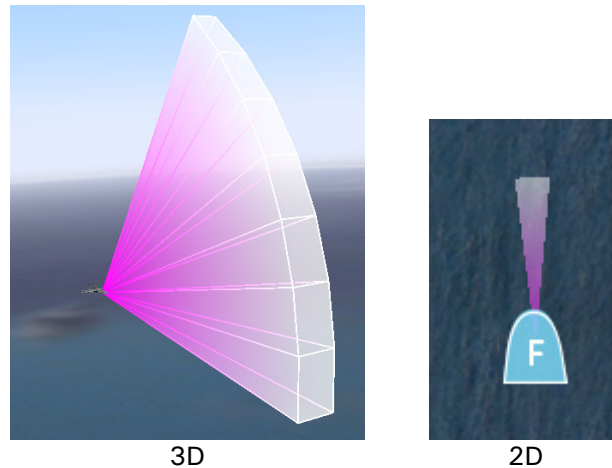


Figure 8-9. Sensor volumes

Display of sensor volumes is an observer-specific setting. It only affects entities that have emitter systems.

To enable or disable display of sensor volumes:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Observer Mode Settings page (Figure 8-6).
3. Select the observer mode for which you want to change the Sensor Volumes setting.
4. Select or clear the Sensor Volumes check box.



You can also enable or disable sensor volumes by choosing **Observer** → **Sensor Volumes** or clicking the Sensor Volumes icon (🔍) on the Observer Settings toolbar.

8.8. Displaying Cockpit Displays

VR-Forces can display cockpit displays created with GL Studio. It includes cockpits for fixed-wing, rotary-wing, and ground vehicle entities. As with other entity effects, cockpits have schemas and model definitions that you map to entity types. For details about cockpit display models, please see Section 13.5, “[Adding New Cockpit Display Models](#)”, in *VR-Forces Configuration Guide*.



Cockpits are displayed only when the observer is attached to an entity in mimic mode.

To display cockpits:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Cockpit Display Settings page ([Figure 8-10](#)).

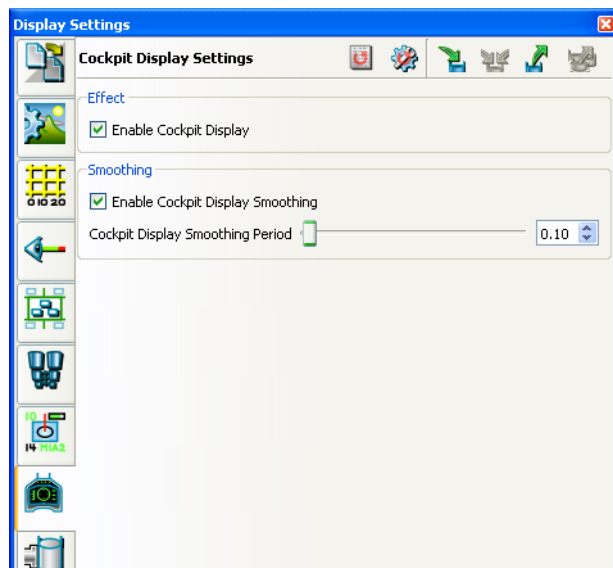


Figure 8-10. Cockpit display settings

3. To enable or disable cockpit displays, select or clear the Enable Cockpit Display check box.
4. Attach the observer to an entity in mimic mode.



You can also enable and disable cockpit displays by choosing **Settings** → **Cockpit Displays** or by clicking the Cockpit Displays icon () on the Display Settings toolbar.

8.9. Displaying Shadows

VR-Forces can display shadows for entities. Shadows are cast only on the terrain. They are not cast on buildings (props) or on other entities. Therefore, for example, entities on an aircraft carrier do not cast shadows on the deck.

You can enable and disable shadows and you can configure shadow quality. By default, fixed-wing, rotary-wing, and DI-Guy entities cast shadows. You can configure other entities to cast shadows by setting the shadowCaster attribute in their model definitions to 1. (For information about model definitions, please see Chapter 13, *Model and Element Definitions*, in *VR-Forces Configuration Guide*.)



Updated model definitions do not take effect immediately. To view shadows, you must restart VR-Forces.

To enable and disable shadows globally:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Render Settings page (Figure 8-11).

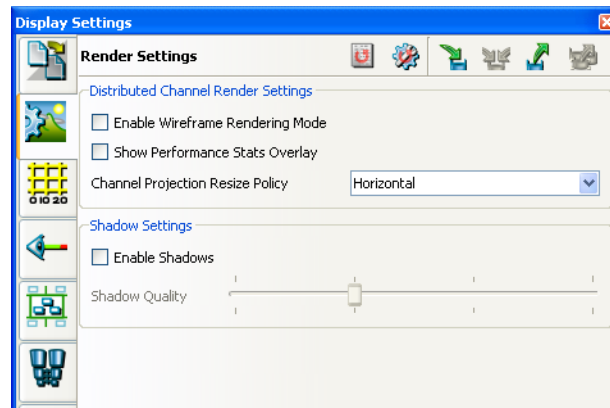


Figure 8-11. Display Settings, Render Settings page

3. In the Shadow Settings group box, select or clear the Enable Shadows check box.



You can also enable and disable shadows by choosing **Settings** → **Shadows** or clicking the Shadows icon () on the Display Settings toolbar.

8.9.1. Configuring Shadows

You can configure the quality of shadows. To use this feature you must have high quality hardware.

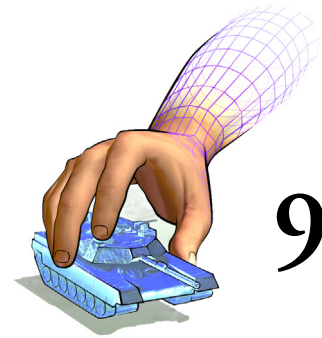
To configure how shadows are rendered:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Render Settings page ([Figure 8-11](#)).
3. Select Enable Shadows. The Shadow Quality slider becomes enabled.
4. Adjust the Shadow Quality slider.

8.10. Disabling Lighting and Clouds

When you are using XR observer mode and you zoom out from the terrain, the lighting and cloud effects can interfere with your ability to view entities. Therefore, VR-Forces lets you quickly disable lighting effects and clouds. By default, they are enabled in Stealth observer mode and disabled in XR observer mode.

- To enable or disable lighting and clouds, choose **Observer** → **Sky**, or click the Show/Hide Sky icon () on the Observer Settings toolbar.



Aggregate Entities

Many aspects of managing aggregate entities are the same as for individual entities. This chapter focuses on features that are unique to aggregates.

Selecting Aggregates	9-2
Expanding and Collapsing Aggregates	9-2
Expanding and Collapsing Aggregates in the Echelon View	9-4
Expanding and Collapsing the Echelon View by Level of Aggregation	9-4
Displaying Ghosted Icons	9-5
Displaying Aggregate Icons and Bounding Volumes	9-8
Specifying the Color Scheme for Aggregate Bounding Volumes	9-8

9.1. Selecting Aggregates

You can select aggregates in the same ways that you can select entities. (For details about selecting entities, please see [“Selecting Entities, Tactical Graphics, and Props,”](#) on page 6-4.) However, to select an aggregate, you must select the aggregate icon, not individual members of the aggregate. If a disaggregated aggregate is expanded, the aggregate icon might not be visible. To select the aggregate, you can collapse it, or you can select the aggregate level entity in the Echelon View.

9.2. Expanding and Collapsing Aggregates

You can display disaggregated aggregates in expanded or collapsed mode. When an aggregate is collapsed, it is displayed as one icon. When an aggregate is expanded, all the members of the aggregate are visible as individual entities and the aggregate icon is hidden. Multi-level aggregates can expand and collapse at each level of aggregation. For example a company could be collapsed to the company level (one icon), the platoon level (an icon for each platoon), or fully expanded.

When you Expand or Collapse an aggregate, you can expand or collapse one level of aggregation or all levels. When you Expand or Expand All, expansion is downward from the current aggregate and does not affect any sibling aggregates. When you Collapse, contraction is upward in the hierarchy. Collapse All collapses all members of the aggregate to the topmost level.

[Figure 9-1](#) illustrates these relationships. Assume a company with three platoons, each of which has four vehicles.

- ♦ If you Expand the top level aggregate, it expands to the platoon level. (1)
- ♦ If you select Plt C and choose Expand or Expand All, it expands just Plt C. (2)
- ♦ If you select the top level aggregate and expand all, it expands to the leaf level – 12 individual vehicles. (3)
- ♦ If the aggregate is fully expanded and you select one member of Plt C and choose Collapse, just Plt C collapses. (4)
- ♦ If you select any member of an expanded aggregate and select Collapse All, the entire aggregate collapses. (5)

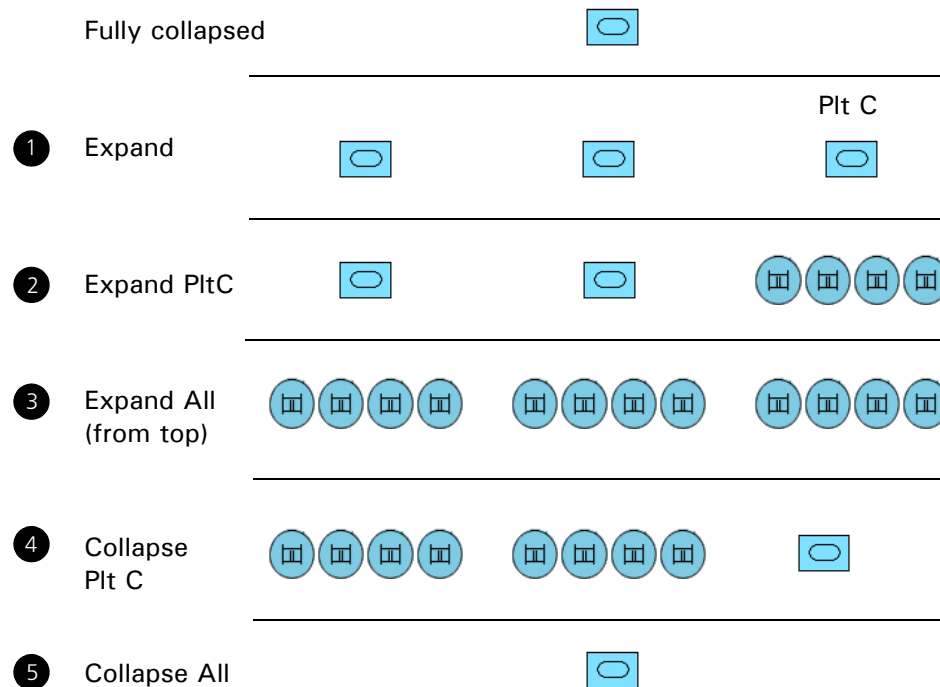


Figure 9-1. Aggregate expand and collapse

i

An aggregate's icon does not indicate whether it is aggregated or disaggregated. However you can easily infer its state from the Entities menu or the Echelon View. The aggregate commands on the Entities menu are context sensitive. If you select an aggregate and the Disaggregate command is available, then the aggregate is in the aggregated state. Similarly, if you view an aggregate in the Echelon View, if it is aggregated, you cannot expand it in the tree. If it is disaggregated, you can expand it to see its subordinates.

- To expand an aggregate, right-click the aggregate and choose **Expand** on the popup menu.
- To expand an aggregate and all its subordinate aggregates, right-click the aggregate and choose **Expand All** on the popup menu.
- To collapse an aggregate one level, right-click any member of the aggregate and choose **Collapse** on the popup menu.
- To collapse an aggregate to its root level, right-click any member of the aggregate and choose **Collapse All** on the popup menu.

9.2.1. Expanding and Collapsing Aggregates in the Echelon View

You can expand and collapse aggregates in the Echelon View on the Objects List Panel.

- To expand an aggregate in the Echelon View, click the plus sign next to its icon.
- To collapse an aggregate in the Echelon View, click the minus sign next to its icon.

9.2.2. Expanding and Collapsing the Echelon View by Level of Aggregation

In addition to expanding and collapsing the order of battle on a node-by-node basis, as in typical hierarchical view windows, the Echelon View enables you to view entities by the level of aggregation.

Individual entities are directly subordinate to the force they belong to. If an entity becomes part of an aggregate, the aggregate is subordinate to the force and the individual entity is one level further away. Seen another way, you can say that the force is the parent and all other entities in that force are children, grandchildren, and so on.

The Echelon View displays the entities in columns that are equivalent to their relationship to the force – the first column is children, the second column is grandchildren, and so on.

At the top of each column in the Echelon View is a plus (+) or minus (-) icon that works like the icons within the hierarchical display. A plus indicates that at least one entity in that column is an aggregate entity and that none of the aggregates in the column are expanded. A minus indicates that at least one aggregate at that level is expanded. Clicking a minus sign collapses all aggregate entities in the column. Clicking a plus sign expands all aggregates in the column.

- To change the hierarchical level view in the Echelon View, click the level controls in the horizontal band across the top of the window.

[Figure 9-2](#) illustrates an Echelon View with several levels of aggregates, some of which are collapsed and some of which are expanded.

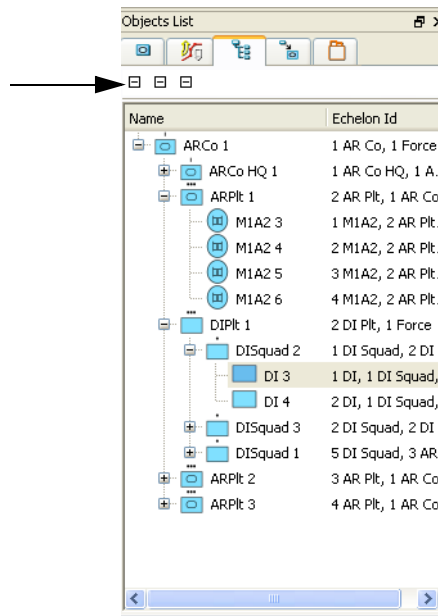


Figure 9-2. Echelon View

9.3. Displaying Ghosted Icons

If you want to select aggregates, you need to collapse them so that you can select the aggregate icon. However, when you do this, you can no longer see where individual subordinate entities are located and if subordinates are destroyed, you cannot see which ones are destroyed and which are functional. VR-Forces provides the following options for how aggregate icons are displayed so that you can work with them in a way that best meets your visualization and entity management needs:

- Display ghosted icons for hidden entities
- Allow ghosted icons to be selected
- Display all aggregate members
- Manually expand and collapse the display
- Display leaf-level aggregate members.

In the Plan View observer mode, when you display ghosted aggregates, the aggregate icon is displayed normally, and the subordinate members are displayed using semi-transparent icons. In 3D observer modes, aggregates that are subordinate to the top level aggregate are displayed using semi-transparent icons. Leaf-level entities are displayed using the appropriate model for the observer mode ([Figure 9-3](#)).

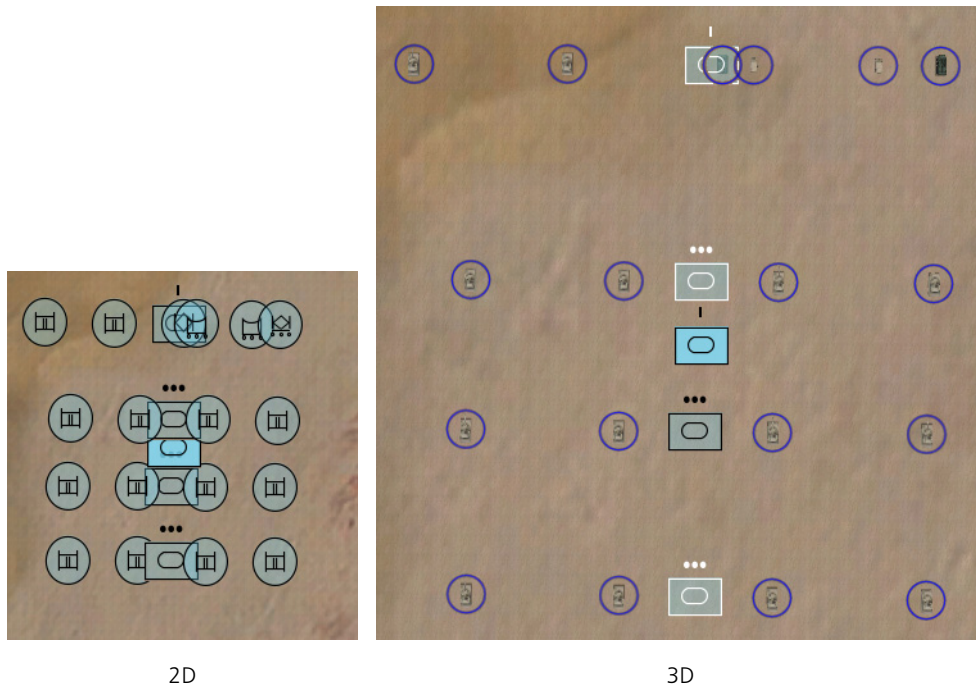


Figure 9-3. Ghosted icons

To configure the display of aggregate icons:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Aggregate Display Settings page ([Figure 9-4](#)).

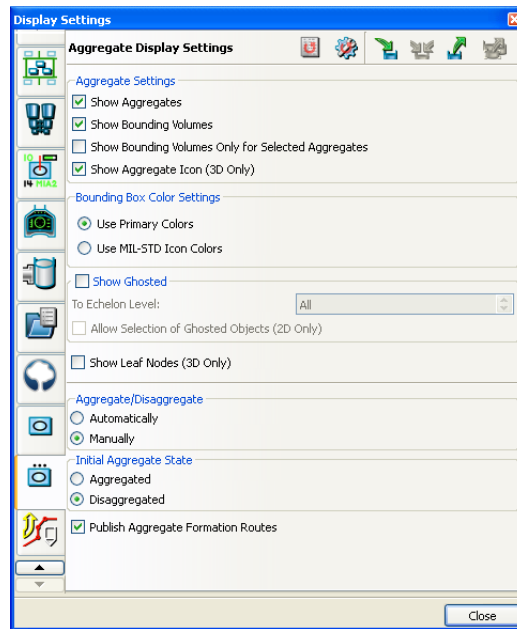


Figure 9-4. Aggregate Display Settings page

3. To display aggregate members based on the current level of aggregation only, clear Show Ghosted.
4. To display ghosted icons:
 - a. Select Show Ghosted.
 - b. In the To Echelon Level box, select the level of aggregation to show ghosted icons for. To show all levels without needing to consider how many levels there are, select All.
 - c. To allow selection of ghosted icons (2D view only), select Allow Selection of Ghosted Icons. (In 3D views, you can always select entities at the leaf level.)

9.4. Displaying Aggregate Icons and Bounding Volumes

VR-Forces lets you configure when to display aggregate icons and bounding volumes. Bounding volumes are translucent polygons that show the extents of the aggregate. This is useful when the individual entity icons or models are not displayed. You can configure icons and bounding volumes as follows:

- Enable and disable the display of aggregates. When disabled, only the 2D icons or 3D models for leaf level aggregate members are displayed.
- Enable and disable the display of bounding volumes. Bounding volumes are displayed only for collapsed aggregates. You can restrict the display of bounding volumes to the selected aggregates.
- In 3D observer modes, enable or disable the display of aggregate icons.

You can display aggregate icons without showing bounding volumes and vice versa.

To configure the display of aggregate icons and bounding volumes:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Aggregate Display Settings page (Figure 9-4).
3. To enable the display of aggregate icons, bounding volumes, or both, select Show Aggregates.
4. To display bounding volumes for collapsed aggregates, select Show Bounding Volumes.
5. To restrict the display of bounding volumes to the selected aggregates, select Show Bounding Volumes Only for Selected Aggregates.
6. To display aggregate icons in 3D observer modes, select Show Aggregate Icon.



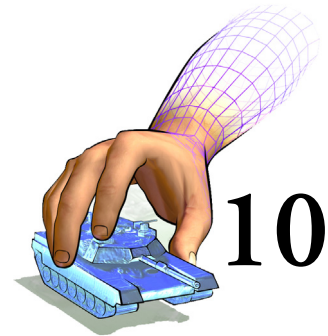
You can also enable and disable aggregates by choosing **Settings** → **Show Aggregates** or by selecting the Show Aggregates icon () on the Display Settings toolbar.

9.4.1. Specifying the Color Scheme for Aggregate Bounding Volumes

Aggregate bounding volumes can be displayed using primary colors or MIL-STD colors.

To specify the color scheme to use for aggregate bounding volumes:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Aggregate Display Settings page (Figure 9-4).
3. In the Bounding Box Color Settings group box, select Use Primary Colors or Use MIL-STD Icon Colors.



The Observer and Observer Modes

This chapter explains how to create, edit, and change observer modes. For details about moving the observer, please see Chapter 11, *Moving the Observer*. For details about attaching the observer to entities and props, please see Chapter 12, *Attaching the Observer*.

Changing the Observer Mode	10-2
Creating and Editing Observer Modes	10-2
Creating an Observer Mode	10-2
Deleting an Observer Mode	10-4
Editing an Observer Mode	10-4
Renaming an Observer Mode	10-4
Adding and Removing Observers	10-5
Communicating Information About the Observer	10-6
Making the Observer Visible to Other Applications	10-6
Viewing and Attaching to Other Observers	10-6
Displaying Models for Remote Observers	10-7

10.1. Changing the Observer Mode

When you change the observer mode, terrain may be displayed differently and the model sets used to display entities and objects may change. For descriptions of the observer modes, please see “[Observer Modes](#),” on page 4-8.

- To change the observer mode, choose **Observer** → **Set Observer Mode** → *mode*, or on the Observer Settings Toolbar, choose an option from the Observer Mode drop-down list.



Figure 10-1. Observer Settings toolbar

10.2. Creating and Editing Observer Modes

VR-Forces comes with 3D and 2D observer modes. They are configured with the display settings, key mappings, and model sets that users most commonly want to use. However, if they do not meet your needs, you can easily edit them or create new observer modes.

10.2.1. Creating an Observer Mode

To create an observer mode:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Observer Mode Settings page ([Figure 10-2](#)).

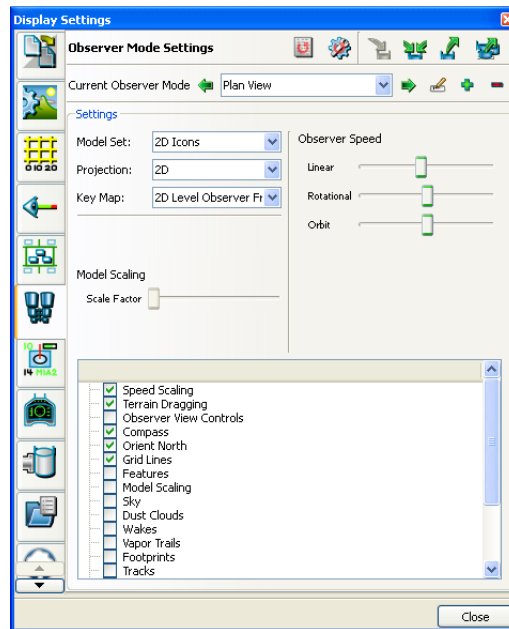


Figure 10-2. Observer Mode Settings page

3. Click the Add icon (+). The Create New Observer Mode dialog box opens (Figure 10-3).

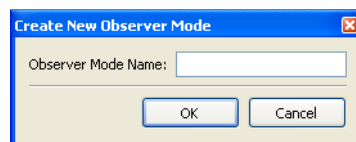


Figure 10-3. Create New Observer Mode dialog box

4. Type a name for the new observer mode.
5. Click OK.
6. In the Settings group box, select a model set.
7. Select a projection (2D or 3D).
8. Select the key map you want to use.
9. Enable or disable the mode-specific settings. The list of settings varies depending on the projection and model set.

10.2.2. Deleting an Observer Mode

To delete an observer mode:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Observer Mode Settings page ([Figure 10-2](#)).
3. In the Current Observer Mode list, select the observer mode that you want to delete.
4. Click the Delete icon (✖).

10.2.3. Editing an Observer Mode

To edit an observer mode:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Observer Mode Settings page ([Figure 10-2](#)).
3. In the Current Observer Mode list, select the observer mode that you want to edit.
4. Edit the attributes of the observer mode as desired.

10.2.4. Renaming an Observer Mode

To rename an observer mode:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Observer Mode Settings page ([Figure 10-2](#)).
3. In the Current Observer Mode list, select the observer mode that you want to rename.
4. Click the Rename icon (✎). The Rename Observer Mode dialog box opens.
5. Type the new name.
6. Click OK.

10.3. Adding and Removing Observers

You can add and remove observers. Having multiple observers lets you assign different observers to different windows or channels. Changing the position and orientation of the observer in one channel does not affect the view of a different observer in a different channel.



If you have multiple observers, the observer listed in the Observer Settings toolbar is the observer assigned to the window that has focus. If you open the Display Configuration Panel and select a different observer in the Channel Attributes list, the data in relevant panels will change to reflect the values for the selected observer. However, as soon as you return focus to a window, the observer selection will change to match that of the window.

You cannot explicitly remove an observer. VR-Forces removes all observers when you close the window on which the observer was created. If you want to save your observers, save the display configuration in which you created them.

To add an observer:

1. Choose **View** → **Display Configuration Panel**. The Display Configuration Panel opens (Figure 10-4).

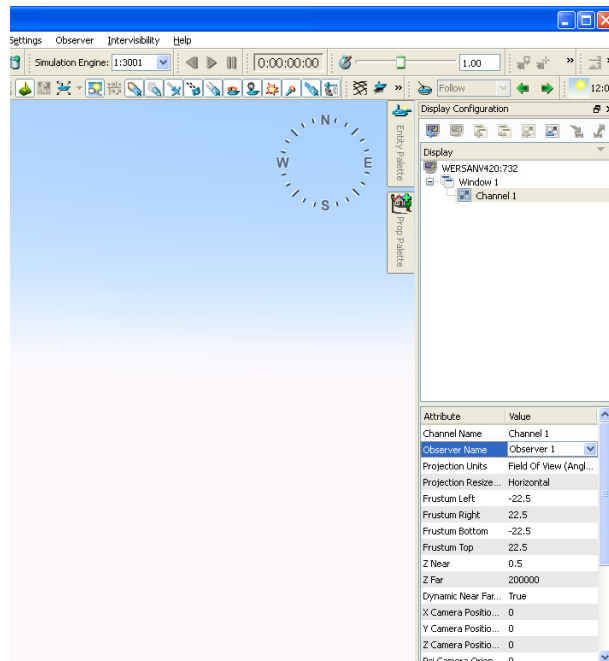


Figure 10-4. Display Configuration Panel

2. Select the channel on which you want to add an observer.

3. Click the Value field for the Observer Name attribute to make it editable.
4. In the drop-down list, select New.
5. Click Commit Changes. The Add Observer dialog box opens.
6. Type a name for the new observer.
7. Click OK. The new observer becomes the selected observer on the Observer Settings toolbar, and is listed in the Observer Name attribute. (You may need to refresh the channel attributes before the observer name is listed.)

10.4. Communicating Information About the Observer

VR-Forces can transmit state updates describing the observer's position and orientation, and can respond to view control messages sent by other applications.

10.4.1. Making the Observer Visible to Other Applications

By default, VR-Forces sends state update messages to make its state information known to other applications. This makes it possible for a plan view display to represent VR-Forces as an icon, or for one VR-Forces application to synchronize its observer movement with that of another VR-Forces application.

- To enable or disable broadcasting of observer information, choose **Settings** → **Publish Observers** (or **Stop Publishing Observers**).

10.4.2. Viewing and Attaching to Other Observers

If an exercise includes stealth applications, and they are sending state updates, VR-Forces can detect them. A remote observer entity is listed in the object hierarchy, so you can attach the local observer to it just as you would attach to any other entity.

Observer icons are displayed in the 2D view, but they are not normally visible in the 3D view, even when they appear in the entity list. If you want to display visual representations of remote observers, map a visual model to the observer's entity enumeration.

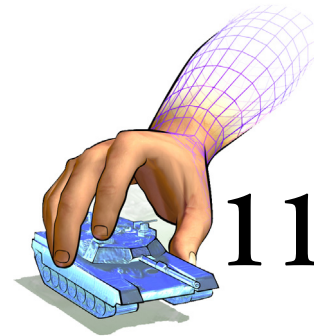
For related information, please see [“Displaying Models for Remote Observers,”](#) on page 10-7.

10.5. Displaying Models for Remote Observers

VR-Forces broadcasts the location and orientation of the observer. By default, it does not display a model for the observer. However, if there are other applications in an exercise that are broadcasting a stealth PDU (such as another VR-Forces), you might want to represent them with a model.

To display a model for remote observers:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Entity Display Settings page ([Figure 7-3](#)).
3. In the Filtering group box, select the Show Other Stealths check box.



Moving the Observer

This chapter explains how to move the observer (eyepoint) in the VR-Forces window.

Moving the Observer	11-3
Using the Compass to View the Observer's Heading	11-4
Editing the Compass's Attributes	11-5
Observer Movement Functions and Keyboard Mappings	11-5
3D Observer Movement Coordinate Systems	11-6
2D Observer Coordinate System	11-6
The Attached Context and Observer Movement (3D Projection Only)	11-7
Moving the Observer from the Keyboard in the 3D View	11-9
Navigation Options	11-11
Orbiting	11-12
Moving the Camera	11-12
Moving the Observer Using the Mouse (3D View)	11-13
Dragging the Terrain	11-13
Changing the Observer's Orientation with the Mouse	11-14
Dragging the View	11-14
Orbiting the Terrain with the Mouse	11-14
Teleporting the Observer	11-14
Moving the Observer in the 2D View	11-14
Moving the Observer from the Keyboard in the 2D View	11-15
Moving the Observer Using the Mouse in the 2D Projection	11-15
Forcing Orientation North	11-16
Enabling and Disabling View Constraints (3D Only)	11-17
Controlling Navigation Speed	11-18
Enabling or Disabling Speed Scaling	11-18

Moving the Observer

Changing the Observer's Movement Speed.....	11-19
Saving and Recalling Views.....	11-20
Recalling a View.....	11-21
Deleting Views.....	11-21
Saving Views to a File.....	11-21
Loading Views.....	11-22
Controlling the Observer from Other Applications.....	11-22
Enabling the Processing of View Control Messages	11-22

11.1. Moving the Observer

When you view a simulation, you usually want to move the observer (eyepoint) to different locations to view the terrain and scene elements being simulated. In VR-Forces, you can move the observer using the keyboard and the mouse. There are two basic contexts in which you can move the observer:

- You can move it independently of any scene element. This is called absolute, or free-fly mode.
- You can attach the observer to an entity, in which case it moves with the entity. (You can also attach the observer to a prop. For details about attaching the observer to an entity or prop, please see Chapter 12, [Attaching the Observer](#).)

Absolute mode and the various attachment types are collectively called attach modes.

Regardless of which attach mode you are using, you can change the position and orientation of the observer. In the 3D view, position is defined by X, Y, and Z coordinates that identify its location in the 3D world. Orientation describes the observer's yaw (heading), pitch, and roll ([Figure 11-1](#)).

In the 2D view, the Z coordinate is fixed. You can change the X and Y coordinates. You can change the observer's yaw, but pitch and roll are fixed.

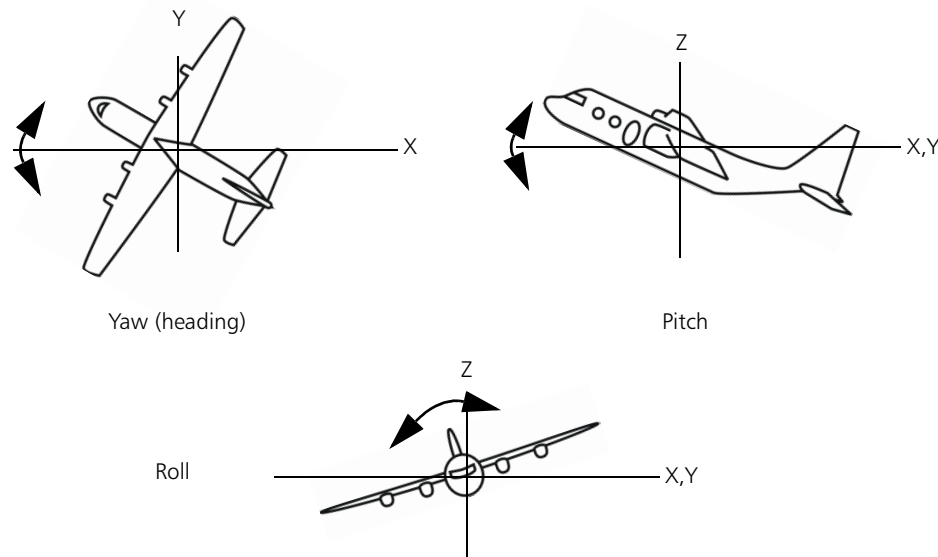


Figure 11-1. Orientation

11.1.1. Using the Compass to View the Observer's Heading

VR-Forces can display a compass that always shows the heading of the observer. The compass is an observer mode setting. In [Figure 11-2](#), the observer's heading is approximately Northwest.

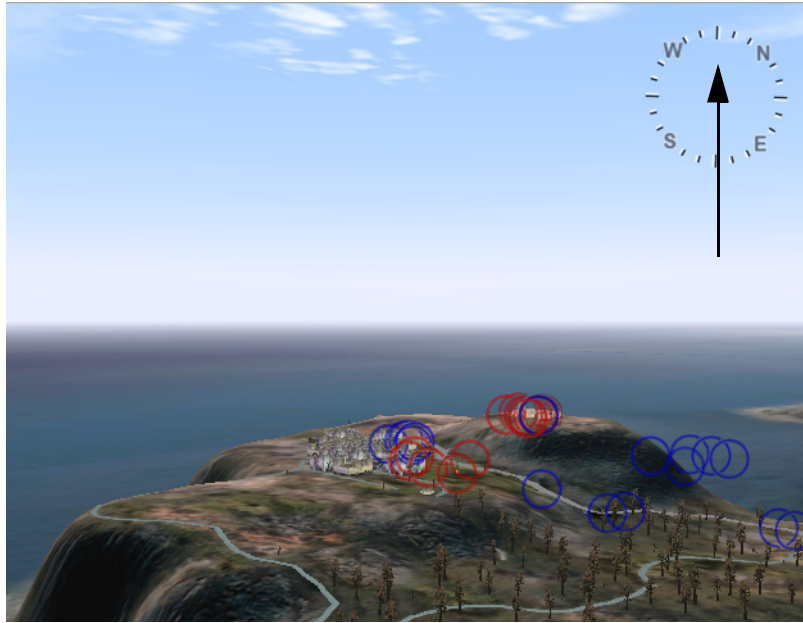


Figure 11-2. Compass

To enable or disable the compass:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Observer Mode Settings page ([Figure 8-6](#)).
3. Select the observer mode for which you want to change the compass setting.
4. In the list of observer settings, select or clear **Compass**.



You can also enable or disable the compass by choosing **Observer** → **Compass** or clicking the Compass icon (☼) on the Observer Settings toolbar.

11.1.2. Editing the Compass's Attributes

You can configure the compass by editing its schema (ChannelCompass) and model definition (smallCompass and largeCompass). By default, all channels have the largeCompass keyword. To change the style of compass used for a channel, change the keyword in the Keywords attribute for the channel. To completely remove the compass, delete the keyword. For details about setting channel attributes, please see Section 10.3, [“Changing a Channel's Attributes”](#), in *VR-Forces Configuration Guide*.

You can change the location of the compass by editing the location attribute. The permissible values are top, topRight, right, bottomRight, bottom, bottomLeft, left, and topLeft.

You can change the color of the compass by editing the darkTickColor and lightTickColor attributes. The color is defined using the OpenGL scheme for specifying RGBA. The first three values are the RGB color values. The fourth value specifies opacity. Each value can be from 0.0 through 1.0.

11.2. Observer Movement Functions and Keyboard Mappings

When you move the observer, you are executing observer movement functions. An observer movement function is a specific way that the observer can move, such as Move Observer Forward or Move Level Observer Forward. (These functions both move the observer forward, but they use different observer coordinate systems. Observer movement coordinate systems are described in [“3D Observer Movement Coordinate Systems,”](#) on page 11-6.)

The various movement functions are mapped to the keys on the keyboard and to mouse buttons and the mouse wheel. If you use the default key mappings for the default observer modes, then most of the time, the observer will behave exactly as you would expect it to. However for some types of movement, such as forward and back, the observer can move differently depending on the view, the observer coordinate system, or the attach mode. Therefore, VR-Forces provides several key mappings that let you choose which type of observer movement to use. You can change these key mappings and create your own key mappings. (For details about editing key maps, please see Appendix E, [Creating and Editing Key Mappings](#), in *VR-Forces Configuration Guide*.) This section describes how observer movement can vary based on the observer coordinate system and the attach mode.

11.2.1. 3D Observer Movement Coordinate Systems

In the 3D view, the observer moves in the context of one of the following coordinate systems:

- ♦ Level Observer Coordinates: Movement is with respect to the terrain. The level observer coordinate system is defined by the current up vector (Z), and a plane normal to that vector into which the observer's X and Y axes are projected. (Essentially, this means that the observer's heading is used while its pitch and roll are considered to be 0.0.)



Movement is always with respect to a plane tangent to the earth's surface. However, in a projected terrain database, such as UTM, Lambert Conformal Conic, or flat earth, the plane is fixed, whereas in a geocentric terrain database the up vector (and hence the tangent plan) changes as the observer's position changes, so movement follows the curvature of the earth.

- ♦ Observer Coordinates: Movement is with respect to the observer's body coordinate system.
- ♦ Entity Level Body Frame Coordinates: Movement is with respect to an entity's body frame, but ignores pitch and roll.
- ♦ Entity Body Frame Coordinates: Movement is with respect to an entity's body frame, including pitch and roll.

The practical difference between observer coordinates and level observer coordinates is that in level observer coordinates, if you are looking down at the terrain, when you move forward and backward, you fly over the terrain, looking down. In observer coordinates, when you move forward and backward, you move down towards the terrain or away from it at whatever angle you are oriented towards it.

11.2.2. 2D Observer Coordinate System

When you use the 2D view, the observer uses a variation of level observer coordinates. Movement is restricted to two dimensions, so there are no movement functions for changing the altitude, pitch, or roll. You can zoom in and out in the 2D view, but this does not change the observer's altitude, it changes the field of view and resolution of the terrain image.

11.2.3. The Attached Context and Observer Movement (3D Projection Only)

The key maps control navigation using a particular coordinate system (observer frame or level observer frame). They also support navigation in alternate coordinate systems. The alternate system varies depending on the attached context of the observer. It represents MÄK's opinion of which alternate coordinate system would be most useful given the attached context.

The Move Attached Context *direction* (forward, back, up, down, left, right) functions move the observer in the context of the attached mode. Movement is modified by attach modes in the following ways:

- ♦ Follow mode: movement is in the entity's level body frame
- ♦ Track mode: movement is in the entity's level body frame
- ♦ Mimic mode: movement is in the entity's body frame
- ♦ Tether mode: movement is in level observer frame
- ♦ Unattached: movement is in level observer coordinates.

Figure 11-3 and Figure 11-4 illustrate the differences between moving the observer in an attached context and without an attached context.

i

The figures in this chapter use binoculars to represent the observer.

Figure 11-3 illustrates movement in entity level body coordinates (Follow and Track modes). The Move Observer Left function (**A** in Observer Frame key map) moves the observer to the left in observer coordinates, that is left relative to the observer's heading. The Move Attached Context Left function (**Shift + A**) moves the observer left in the attached context. In entity level body frame, this means the observer moves left relative to the entity's heading.

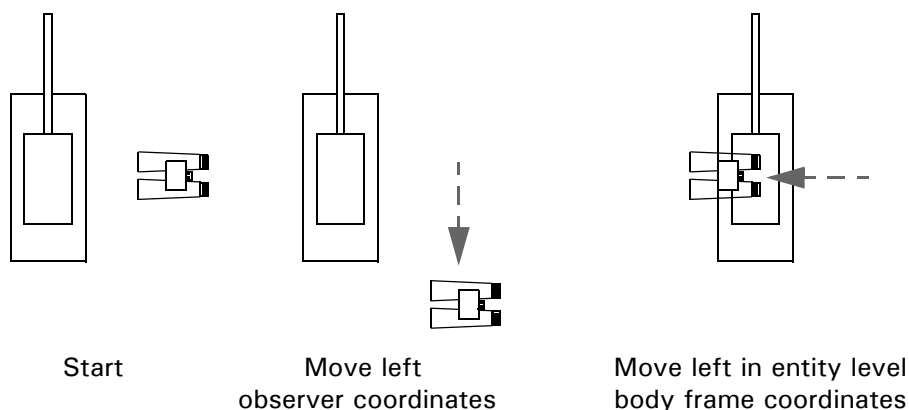


Figure 11-3. Observer movement in entity level body frame coordinates

Figure 11-4 illustrates movement in entity body frame (Mimic mode). The Move Observer Left function moves the observer to the left in observer coordinates, that is left relative to the observer's heading, but ignoring pitch and roll. The Move Attached Context Left function moves the observer left in the attached context. In entity body frame, this means the observer moves left relative to the entity's heading and its pitch and roll.

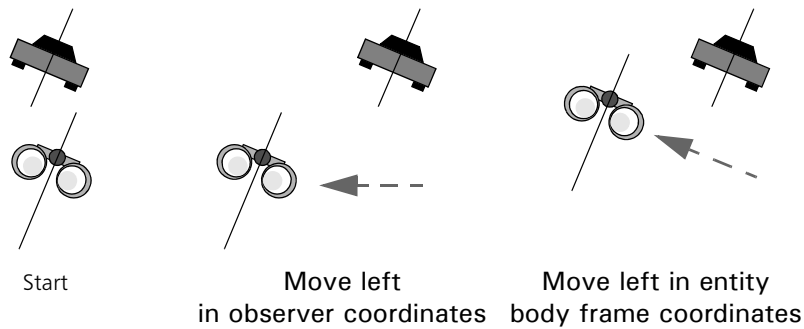


Figure 11-4. Movement in entity body frame coordinates

11.3. Moving the Observer from the Keyboard in the 3D View

VR-Forces maps the observer movement functions to the keyboard using key maps that support the most typical ways that users want to move through the 3D world, either in absolute mode or attached to an entity.



References to movement key combinations use the key mappings provided with VR-Forces. If you change these key mappings, the instructions may no longer be valid.

As described in “[Moving the Observer](#),” on page 11-3, the observer typically moves with respect to the terrain in level observer coordinates or observer coordinates. The corresponding key maps are Level Observer Frame and Observer Frame.

[Figure 11-5](#) illustrates the positional movement keys. [Figure 11-6](#) illustrates the keypad layout for changing orientation and movement speed. For a complete list of movement functions and their keyboard and mouse mappings, please see [VR-Vantage Quick Reference Card](#) or the Key Mapping Editor.



To use the keypad for navigation, NumLock must be on.



Figure 11-5. Primary movement keys, observer coordinates (3D)

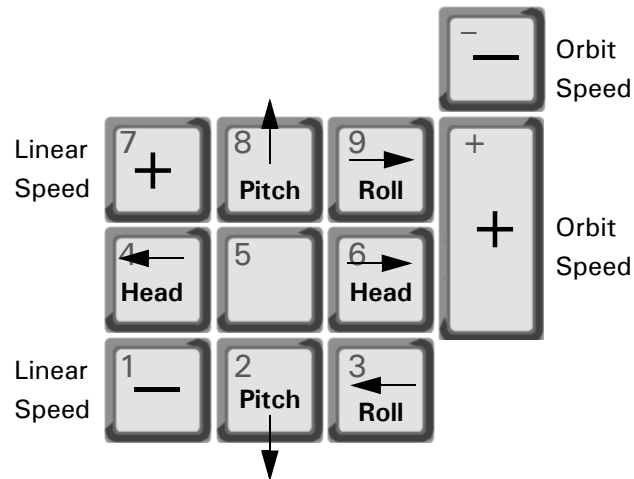


Figure 11-6. Keypad layout, heading, pitch, and roll (3D)

i

If your keyboard does not have a keypad, as on most laptops, the Laptop key map lets you use the arrow keys to control the left, right, up, and down movement mapped to 4, 6, 8, and 2 on keypads.

- To move the observer, press a key combination and hold it for as long as you want the observer to keep moving.

i

You can press more than one key at a time to combine movement functions. For example, pressing the move forward and move left keys at the same time moves the observer diagonally left and forward.

11.3.1. Navigation Options

This section lists the different ways that you can move the observer using the keyboard. For a complete list of movement functions and their keyboard and mouse mappings, please see Chapter B, *VR-Forces Quick Reference Card*. You can:

- ♦ Move forward and backward (observer, level observer, entity, level entity, and attached context coordinates)
- ♦ Move up and down (observer, level observer, entity, level entity, and attached context coordinates)
- ♦ Move left and right (observer, level observer, entity, level entity, and attached context coordinates)
- ♦ Rotate (heading) left and right (observer and level observer coordinates)
- ♦ Rotate (pitch) up and down (observer and level observer coordinates)
- ♦ Rotate (roll) left and right (observer and level observer coordinates)
- ♦ Orbit left and right
- ♦ Orbit up and down
- ♦ Look (move camera) left and right
- ♦ Look (move camera) up and down
- ♦ Tilt (move camera) left and right
- ♦ Reset the observer to the default position and orientation for the attached mode
- ♦ Reset the camera to the default position and orientation.

For details about orbiting, please see “[Orbiting](#),” on page 11-12. For details about moving the camera, please see “[Moving the Camera](#),” on page 11-12.

i

When you view a geocentric terrain and zoom out more than 6 million meters, the view is forced to focus toward the center of the earth. The practical effect of this mode is that the left and right movement keys orbit the eyepoint around the earth, rather than moving the entire earth to the left or right.

11.3.2. Orbiting

You can orbit the observer (revolve left, right, up, and down) around an attached object or the point on the terrain that is at the center of the window. (If the center of the screen is in the sky, orbit does not work.) When you orbit, the observer's heading continuously changes so that it is always facing the point or object around which it orbits. You can change the camera to look at something other than the orbit point while you are orbiting.

To orbit around an object:

1. Attach the observer to the object.
2. Press the right or left arrow keys.

To orbit around a point:

1. If you are attached to an entity, detach (press **z**).
2. Press the right or left arrow keys.

11.3.3. Moving the Camera

By default, the observer always looks in the same direction as its heading. However, just as the turret on a tank can swivel and fire in directions other than its body's heading, the observer can look in a direction other than its heading. The ability to look around is enabled in the camera. You can move the observer's camera up, down, left, right, tilt left, and tilt right. (The default key mappings are **Ctrl** + *Keypad number*, where *number* is 2, 3, 4, 6, 8, or 9.)



To use the keypad for navigation, **NumLock** must be on.



You can tell that the camera is moving, but the heading is not by observing the compass. If you change the heading, the view of the terrain changes and the compass moves. If you move the camera, but keep the heading static, the view of the terrain changes, but the compass does not move.

- To move the camera, press the keys mapped to the camera movement functions.

11.4. Moving the Observer Using the Mouse (3D View)

You can change the observer's position and orientation with the mouse.

Using the mouse, you can:

- ♦ Drag the terrain
- ♦ Drag the view
- ♦ Orbit the terrain
- ♦ Change orientation
- ♦ Teleport (jump).

11.4.1. Dragging the Terrain

Dragging the terrain combines the move forward, backward, left, and right functions.

To drag the terrain:

1. Click a point on the terrain and hold down the left mouse button.
2. Drag the mouse left, right, forward, and backward.



Dragging the terrain is most effective when you are moving large distances, such as several kilometers. To move small distances, use the keyboard.

Disabling Terrain Dragging

- To disable terrain dragging, on the Observer Control Panel, clear the Allow Terrain Dragging check box (Figure 11-7).

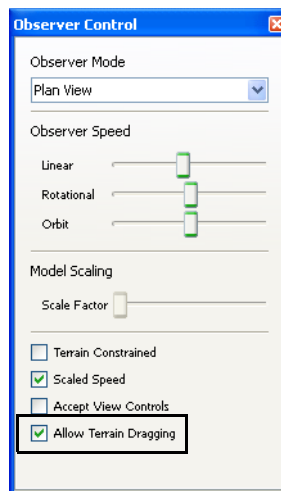


Figure 11-7. Observer Control Panel

11.4.2. Changing the Observer's Orientation with the Mouse

- To change the observer's orientation, hold down the right mouse button and drag the mouse. The observer rotates in the direction you drag the mouse.

11.4.3. Dragging the View

Dragging the view combines the observer up, down, left, and right movement functions. Unlike dragging the terrain, which requires you to click a point on the terrain, you can drag the view from any observer location.

- To drag the view, hold down the middle mouse button (or wheel) and drag the mouse.

11.4.4. Orbiting the Terrain with the Mouse

- To orbit the terrain with the mouse, press **Ctrl** and drag the mouse.

11.4.5. Teleporting the Observer

You can teleport the observer to an entity, a prop, or a place on the terrain. Teleporting moves the observer 90% of the way from its current position to the location or object clicked on.

To teleport the observer:

- **Shift**+click on the terrain or prop that you want to jump to.
- **Ctrl+Shift**+click on the entity that you want to jump to.

11.5. Moving the Observer in the 2D View

In the 2D view, you move the observer using the mouse and keyboard just as you do in the 3D view. Many of the key and button mappings for the 2D view are the same as those in the 3D view. However, the 2D mappings do not support movement that is not meaningful in two dimensions, such as up and down (altitude), pitch, and roll.



When Orient North is enabled, you cannot change the heading of the observer. For information about Orient North, please see [“Forcing Orientation North,”](#) on page 11-16.

11.5.1. Moving the Observer from the Keyboard in the 2D View

Figure 11-8 illustrates the positional movement keys for the 2D view. Figure 11-9 illustrates the keypad layout for changing heading and movement speed. For a complete list of movement functions and their keyboard and mouse mappings, please see [VR-Forces Quick Reference Card](#).



To use the keypad for navigation, NumLock must be on.



Figure 11-8. Primary movement keys, observer coordinates (2D)

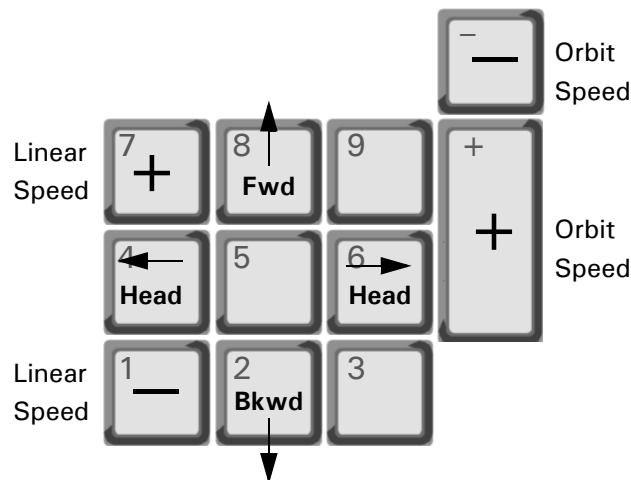


Figure 11-9. Keypad layout, heading, pitch, and roll (2D)

11.5.2. Moving the Observer Using the Mouse in the 2D Projection

You can change the observer's position and heading with the mouse. You can change the observer's position by dragging the terrain, as described in [“Dragging the Terrain,”](#) on page 11-13. You can change heading by orbiting, as described in the next section.

Orbiting the Terrain with the Mouse



When Orient North is enabled, you cannot orbit. For information about Orient North, please see “[Forcing Orientation North](#),” on page 11-16.

- To orbit the terrain with the mouse, press the right mouse button and drag the mouse.

11.6. Forcing Orientation North

In the 2D view, by default the observer’s heading is forced to stay North (up), regardless of the heading of the entity to which it is attached. You can disable this setting, in which case if you attach to an entity, the view of the terrain rotates to match the heading of the attached entity.

[Figure 11-10](#) illustrates the view when the observer is attached to an entity with orientation forced to North or matched to that of the entity.

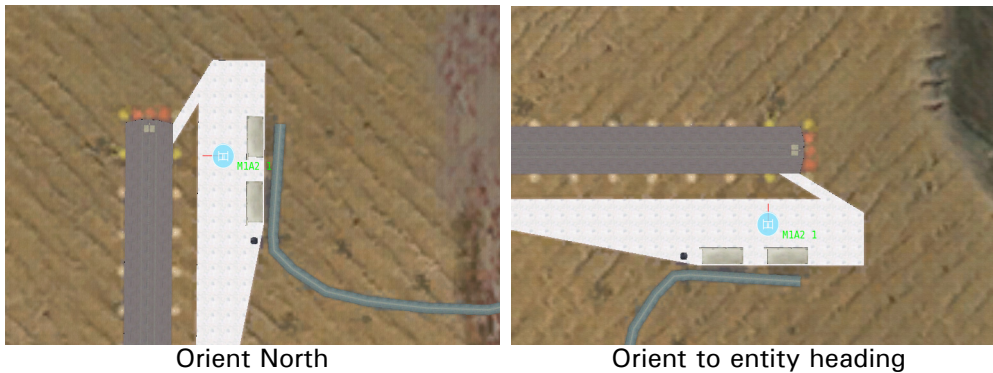


Figure 11-10. Forcing orientation North

To enable or disable a northward orientation:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Observer Mode Settings page ([Figure 8-6](#)).
3. In the Current Observer Mode list, select Plan View.
4. In the list of observer settings, select or clear Orient North.



You can also enable or disable northward orientation by choosing **Observer** → **Orient North** or clicking the Orient North icon () on the Observer Settings toolbar.

11.7. Enabling and Disabling View Constraints (3D Only)

View constraints control how the observer behaves when it approaches solid objects in the view, such as the ground, buildings, and vegetation. When constraints are enabled, the observer has a 1 meter bounding sphere that prevents it from moving through terrain or props.



If view constraints are enabled, you may not be able to move the observer into buildings to view and place entities. Changing the near clipping plane will let you see through walls, but in constrained mode, you will still not be able to move the observer through the walls.

To enable or disable view constraints, use one of the following methods:

- ♦ Press / (slash)
- ♦ Choose **Observer** → **Constrain to Terrain**
- ♦ On the Observer Control Panel (Figure 11-11), select or clear the Terrain Constrained check box
- ♦ Click the Constrain to Terrain icon (🌿) on the Observer Settings toolbar.

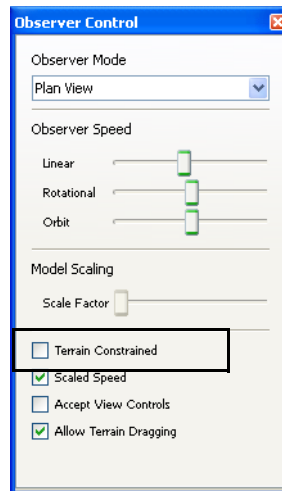


Figure 11-11. Observer Control Panel

11.8. Controlling Navigation Speed

When you navigate through the terrain, there are times you want to move quickly, covering large amounts of terrain, and times you want to move slowly, such as when you are moving down a street or within a building. VR-Forces has the following options for controlling navigation speed:

- ♦ Set the navigation speed manually. The observer always moves at the specified speed.
- ♦ Use speed scaling to automatically move at different speeds under different circumstances. You can change the rate at which speed is scaled.

In either case, the controls for changing speed are the same. Their effect varies depending on whether or not speed scaling is enabled.

11.8.1. Enabling or Disabling Speed Scaling

When speed scaling is enabled, VR-Forces makes the following assumptions:

- ♦ If the observer is unattached, the higher above the terrain it is, the faster you want to move
- ♦ The closer you are to an attached object, the more slowly you want to move.

To enable or disable speed scaling, use any of the following methods:

- ♦ On the Observer Control Panel ([Figure 11-11](#)), select or clear the Scaled Speed check box
- ♦ Press the multiply key (*) (not the asterisk key)
- ♦ Click the Speed Scaling icon () on the Observer Settings toolbar.

11.8.2. Changing the Observer's Movement Speed

You can change the observer's linear, rotational, and orbital speed. If speed scaling is enabled, these controls change the speed gain. If speed scaling is disabled, the controls change the absolute speed.

To change the observer's speed, use any of the following methods:

- ♦ Adjust one of the Observer Speed sliders on the Observer Control Panel ([Figure 11-11](#)) or on the Observer Mode Settings page on the Display Settings dialog box. The default (center of the slider) is 1. The range is 0.1 to 10.
- ♦ Press one of the following keys to change linear speed:
 - **Keypad 7** (faster) or **1** (slower).
 - **Ctrl** + mouse wheel forward or back.
 - Movement key + mouse wheel forward or back.
- ♦ Press **Keypad +** (faster) or **-** (slower) to change orbital speed.



When you change the observer's speed using the keyboard or mouse, if the Observer Control panel or the Observer Mode Settings page is open, you can see the scale slider moving to the new setting.

11.9. Saving and Recalling Views

At any given moment, the state of the observer is called its view. You can save views for later recall. If the observer is not attached to an entity, its view is its position and orientation in the simulated environment. If the observer is attached to an entity, a view saves the following information:

- ♦ Attach mode
- ♦ Positional offset
- ♦ Marking text.



When you save a view in the 2D view, the X and Y values are saved. The zoom level is not saved.

Saved views are managed in the Observer Views Panel (Figure 11-12).



Figure 11-12. Observer Views Panel

Its icons (as indicated by callouts) have the following meanings:

- ♦ Save the current view (1)
- ♦ Delete the selected view (2)
- ♦ Save all views to a file (3)
- ♦ Save the selected view to a file (4)
- ♦ Load a saved views file and append it to the list (5)
- ♦ Load a saved views file and replace the current list (6)
- ♦ Clear the list (7).



Saved views are keyed to the marking text of the object to which the observer is attached.

To save the current view:

1. Choose **View** → **Observer Views Panel**. The Observer Views panel opens ([Figure 11-12](#)).
2. Click the Add icon (+). The current view is added to the list.



Views are remembered only while VR-Forces remains running. To save a view for future recall, you must save it to a file. Please see “[Saving Views to a File](#),” on page 11-21.

11.9.1. Recalling a View

- To recall a view, select it in the Observer Views Panel. The view changes immediately.

11.9.2. Deleting Views

To delete a view:

1. In the Observer Views Panel, select the view you want to delete.
2. Click the Delete icon (-).

Clearing the View List

- To remove all views from the list, in the Observer Views Panel, click the Clear List icon (🗑).

11.9.3. Saving Views to a File

You can save views to a file and load them at a later time. You can save an individual view or all of the views in the list.

To save the selected view to a file:

1. In the Observer Views Panel, click the Save Individual View icon (💾). The Save Selected Observer dialog box opens.
2. Select the file to which you want to save the view, or type the name of a new file.
3. Click OK.

To save all views to a file:

1. In the Observer Views Panel, click the Save All Views icon (💾). The Save Observer Views dialog box opens.
2. Select the file to which you want to save the views, or type the name of a new file.
3. Click OK.

11.9.4. Loading Views

You can load saved views and append them to the list in the Observer Views Panel, or you can replace the list in the panel with the saved views.

To load views and append them to the list of views:

1. In the Observer Views Panel ([Figure 11-12](#)), click the Load Views and Merge icon (). The Open Observer Views dialog box opens.
2. Select the saved views file that you want to load.
3. Click OK.

To load views and replace the list of views:

1. In the Observer Views Panel, click the Load Views and Replace icon (). The Open Observer Views dialog box opens.
2. Select the saved views file that you want to load.
3. Click OK.

11.10. Controlling the Observer from Other Applications

The VR-Forces front-end can respond to view control messages sent by applications such as the MÄK Logger. You can also write VR-Link-based applications to send view control messages to VR-Forces using the classes in the Stealth Control Toolkit or VR-Vantage Control Toolkit. You can use view control messages to:

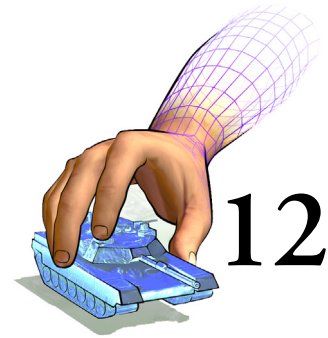
- ♦ Specify the attach mode and attached entity
- ♦ Set the observer's position and orientation.



The view control PDUs are experimental, and are not an official part of DIS. They are of PDU kind 202. VR-Forces responds to move, absolute, tether, mimic, track, tether track, mimic track, follow, orbit, and compass view control PDUs. It does not support the other PDUs in the Stealth Control Toolkit.

11.10.1. Enabling the Processing of View Control Messages

- To enable or disable the processing of view control messages, on the Observer Control Panel, select or clear Accept View Controls.



Attaching the Observer

This chapter describes attachment modes and how to attach the observer to entities and props.

Attaching the Observer to an Entity or Prop	12-2
Attaching to an Articulated Part	12-3
Attaching the Observer in Mimic Track or Tether Track Mode.....	12-4
Filtering the Object Attachment List	12-4
Detaching from an Entity.....	12-4
Hiding the Attached Model (3D Projection Only)	12-4
Attaching to Props.....	12-5
Attach Modes in the 3D View.....	12-5
Absolute Mode	12-5
Follow Mode	12-6
Mimic Mode	12-7
Mimic Track Mode.....	12-8
Tether Mode.....	12-8
Track Mode	12-9
Tether Track Mode	12-10
Attach Modes in the 2D View.....	12-10
Attaching the Observer at Startup	12-11

12.1. Attaching the Observer to an Entity or Prop

You can attach the observer to an entity and follow it as it moves through the simulation. You can also attach the observer to props.

i

This procedure applies to Follow, Tether, Track, and Mimic modes. To attach in Tether Track or Mimic Track mode, please see “Attaching the Observer in Mimic Track or Tether Track Mode,” on page 12-4.

To attach the observer to an entity or prop, do one of the following:

- ♦ Right-click an object in the Objects List Panel (Figure 12-1) or on the terrain and select **Attach mode**, where *mode* is one of the supported attach modes. (For details about attach modes, please see “Attach Modes in the 3D View,” on page 12-5 and “Attach Modes in the 2D View,” on page 12-10.)

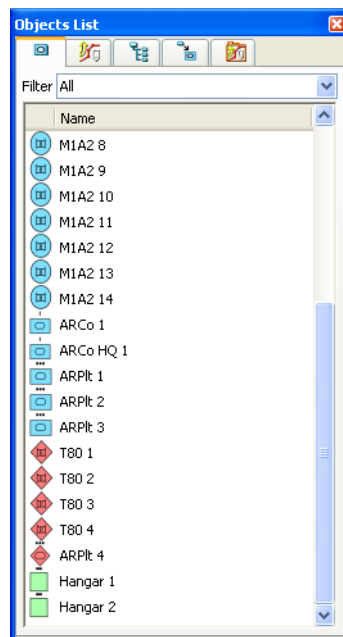


Figure 12-1. Objects List Panel

- ♦ Select an entity or prop. (For details about selecting entities and props, please see “Selecting Entities, Tactical Graphics, and Props,” on page 6-4.) In the Attachments panel (Figure 12-2), click the Attach button (🔗) to attach with the current attachment mode.
- ♦ On the Attachments panel or Attachment toolbar, click the next or previous arrows to attach to the next (or previous) object in the object list.

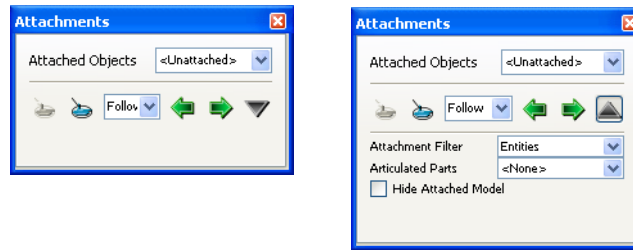


Figure 12-2. Attachments panel

- ♦ Press greater than (>) to attach to the next entity or prop in the object list without regard to the attachment filter. Press less than (<) to attach to the previous object.
- ♦ Press period (.) to attach to the next entity or prop allowed by the attachment filter. Press comma (,) to attach to the previous object.

i

- ♦ When you select the next or previous object, VR-Forces chooses next and previous based on the order in which the objects were discovered. This may or may not represent the order in which they are listed in the Objects List Panel.
 - ♦ When you move the attachment from one object to the next, VR-Forces preserves whatever changes you have made to the observer's view relative to the attached object. So if, for example, you moved the observer so that it is not looking at the attached object, when you attach to the next entity, the observer will not be looking at this object. It will be in the same relative position to the object that you previously established. To quickly return to the default offset for the attachment mode, press the **Spacebar**.
-

12.1.1. Attaching to an Articulated Part

You can attach the observer to an articulated part on an entity, such as a turret. When you attach to an articulated part, the observer moves with the articulated part instead of the main body of the entity.

To attach to an articulated part:

1. Attach the observer to an entity, as described in [“Attaching the Observer to an Entity or Prop,”](#) on page 12-2.
2. On the Attachments panel ([Figure 12-2](#)), click the down arrow. The Attachments panel expands.
3. In the Articulated Parts list, select the part to which you want to attach the observer.

12.1.2. Attaching the Observer in Mimic Track or Tether Track Mode

To attach the observer in tether track and mimic track modes, you must specify a primary attached entity and a secondary entity (the tracked entity).

To attach the observer in Mimic Track mode or Tether Track mode:

1. Right-click the primary entity in the Objects List Panel or in the window. A popup menu is displayed.
2. Choose **Attach Mimic Track** or **Attach Tether Track**. The cursor changes to show an arrow and a Plus sign.
3. Click the secondary entity.

12.1.3. Filtering the Object Attachment List

Although you can attach to entities and props, it is likely that at any given time you will only want to attach to one type of object (usually entities). To reduce the number of objects that you have to iterate through when using the keyboard or Attachments panel arrows to pick the next or previous object, you can filter the attachment list.

To filter the attachment list:

1. On the Attachments panel ([Figure 12-2](#)), click the down arrow. The Attachments panel expands.
2. In the Attachment Filter drop-down list, select the type of scene element that you want to attach to.

12.1.4. Detaching from an Entity

- To detach the observer from the entity to which it is attached, on the Attachments panel ([Figure 12-2](#)), click the Detach icon () or press **z**.

12.1.5. Hiding the Attached Model (3D Projection Only)

You can hide the model for the entity to which the observer is attached.

To hide the attached model:

1. On the Attachments panel ([Figure 12-2](#)), click the down arrow. The Attachments panel expands.
2. Select the Hide Attached Model check box.

12.1.6. Attaching to Props

Since props do not move, attaching the observer to a prop does not result in any observer movement. The principle benefit of attaching to a prop is to let you quickly refocus the observer on a particular location in the scene. If you attach in Track mode, using the keyboard movement keys allows you to orbit around the prop.

12.2. Attach Modes in the 3D View

Attach modes are combinations of position and orientation models and constraints that produce a useful behavior for the observer. Except for absolute mode, attach modes describe how movement of the entity to which the observer is attached affects the position and orientation of the observer.

VR-Forces supports the following attach modes:

- ♦ Absolute: The observer is not attached to an entity; it moves in space independently of all entities.
- ♦ Follow: The observer position and heading changes with the entity's position and heading.
- ♦ Mimic: The observer position and orientation changes with the entity's position and orientation.
- ♦ Tether: The observer's position changes with the entity's position.
- ♦ Track: The observer's heading changes to track the entity's position and heading.
- ♦ Tether Track: The observer's position changes with the entity's position while its heading is focused on a second entity.
- ♦ Mimic Track: The observer's position changes with the entity's position while its orientation is focused on a second entity.

The attach modes are described in detail in the following sections. For details about how to attach the observer to an entity, please see [“Attaching the Observer to an Entity or Prop,”](#) on page 12-2.

12.2.1. Absolute Mode

In Absolute, or free-fly mode, the observer is not attached to an entity. Movement of entities does not affect the observer in any way.

12.2.2. Follow Mode

In Follow mode, the observer maintains its relationship to the attached entity as follows:

- ♦ When the entity's position changes, the observer's position changes to maintain its offset
- ♦ When the entity's heading changes, the observer's heading and position change to maintain the offset
- ♦ When the entity's pitch or roll change, the observer pitch and roll do not change.

Figure 12-3 illustrates the characteristics of Follow mode.

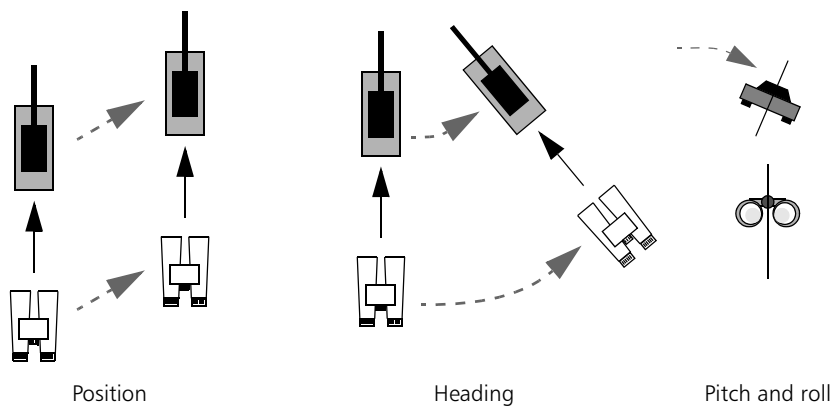


Figure 12-3. Follow mode

12.2.3. Mimic Mode

In Mimic mode, the observer behaves as though it is rigidly connected to the entity. Use Mimic mode to create the effect of riding in a vehicle cockpit, or of following a vehicle from an external location while connected to it by a solid, invisible rod. The observer maintains its relationship to the attached entity is as follows:

- When the entity's position changes, the observer's position changes to maintain its offset
- When the entity's heading changes, the observer's heading and position change to maintain the offset
- When the entity's pitch or roll change, the observer's pitch and roll change and the observer's position changes to maintain its offset.



When cockpit displays are enabled, you cannot change the position or orientation of the observer relative to the entity.

Figure 12-4 illustrates the characteristics of Mimic mode.

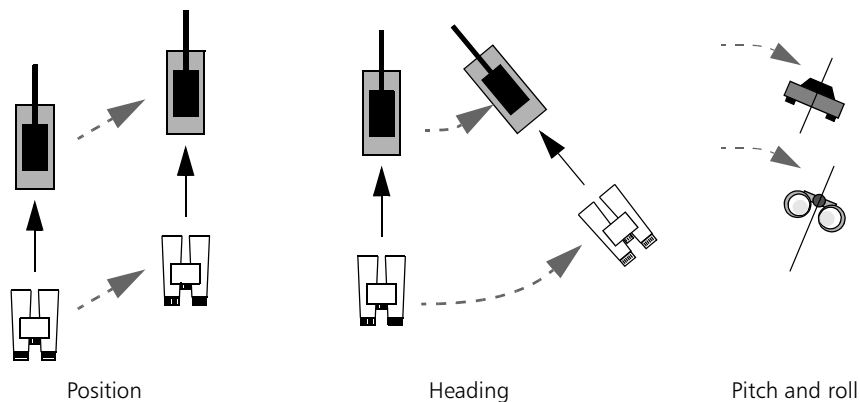


Figure 12-4. Mimic mode

12.2.4. Mimic Track Mode

Mimic Track mode combines Mimic mode and Track mode. The observer's position changes with that of the entity to which it is attached, as in Mimic mode, while its orientation changes to remain focused on a secondary, tracked entity.

Orientation controls are disabled.

12.2.5. Tether Mode

In Tether mode, the observer's maintains a constant offset from the attached entity, but it is not affected by the entity's heading, pitch, or roll, as follows:

- When the attached entity's position changes, the observer's position changes to maintain its offset
- When the attached entity's heading changes, the observer's heading does not change
- When the attached entity's pitch or roll change, the observer's pitch and roll do not change.

Figure 12-5 illustrates the characteristics of Tether mode.

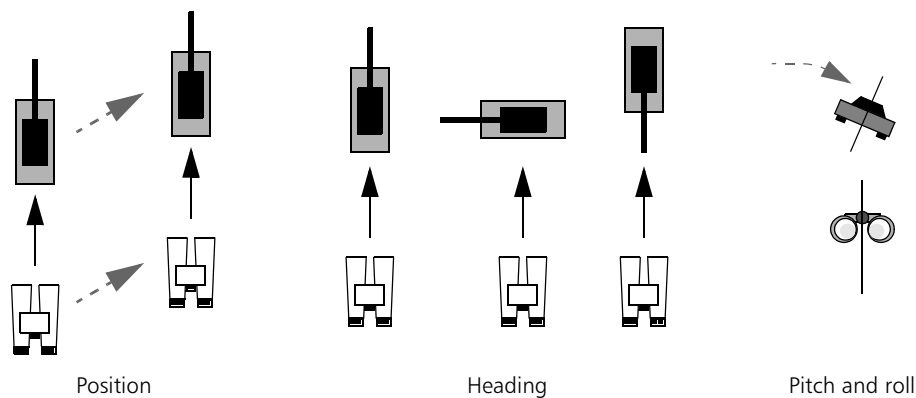


Figure 12-5. Tether mode

12.2.6. Track Mode

In Track mode, the observer's heading changes to track the attached entity's movement, keeping it centered in the field of view. The observer's position does not change. The observer maintains its relationship to the attached entity as follows:

- ♦ When the attached entity's position changes, the observer's heading changes to keep the entity centered in the observer's view; the observer's position does not change
- ♦ When the attached entity's heading changes, the observer's heading does not change
- ♦ When the attached entity's pitch or roll change, the observer's pitch and roll do not change.

Figure 12-6 illustrates the characteristics of Track mode.

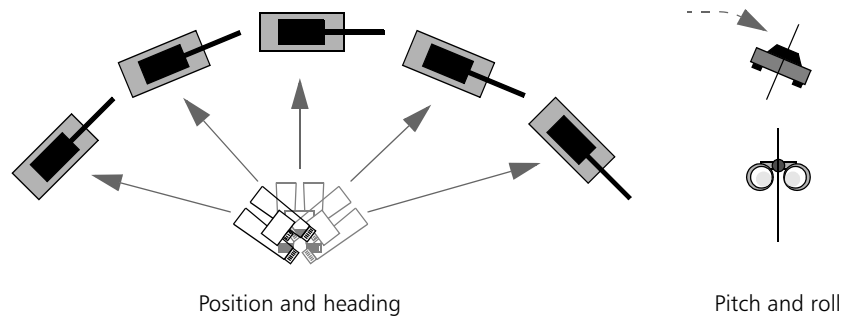


Figure 12-6. Track mode

12.2.7. Tether Track Mode

Tether Track mode combines Tether mode and Track mode. The observer's position changes with that of the entity to which it is attached, as in Tether mode, while its orientation changes to remain focused on a secondary, tracked entity.

Orientation controls are disabled.

12.3. Attach Modes in the 2D View

The 2D view supports Follow mode and Tether mode. In Follow mode, the heading of the observer changes with the entity (if Orient North is disabled); in Tether mode, it does not. You can use the standard movement keys (w-a-s-d) to offset the observer with respect to the attached entity's position. You can use the orbit keys (left arrow, right arrow) to offset the observer's orientation with respect to the entity's orientation (in Follow mode) or the terrain.

If the observer is attached to an entity in the 3D view and you switch to the 2D view, the attach mode is mapped to either Follow or Tether modes, as shown in [Table 12-1](#). If you then switch back to the 3D view the original attach mode is restored.

Table 12-1: 3D to 2D attachment mode mapping

3D Attach Mode	2D Attach Mode
Mimic	Follow
Mimic Track	Follow
Follow	Follow
Tether	Tether
Tether Track	Tether
Track	Tether

12.4. Attaching the Observer at Startup

There may be times when you want to attach the observer to an entity as soon as VR-Forces starts up. For example, suppose you are using VR-Forces as a flight simulator. You might want it to start up and immediately display the cockpit view for the aircraft. VR-Forces has a command-line argument, `--bindObservers`, that lets you bind the observer to an entity using saved views.

To start VR-Forces with the observer bound to an entity:

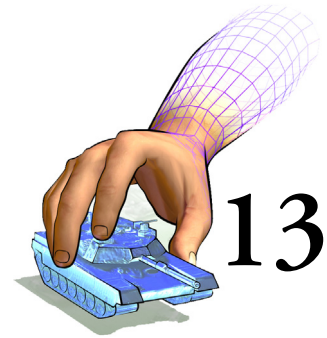
1. Run a live or recorded simulation.
2. Attach the observer to the entity that you want it to be attached to when you start VR-Forces. If you have multiple observers, create a saved view for each observer that you want to attach automatically.
3. Save the views. Views are saved to a file with the extension `.osrx`.
4. Start `vrfGui` with the `--bindObservers` command-line option and use your saved views file as the argument to `--bindObservers`, as follows:

```
vrfGui --bindObservers observer1,...,observern  
      ../appData/settings/vantageIG/saved_views.osrx
```

If the simulation is running, VR-Forces attaches the observer to the entity in the saved view. If the simulation is not running, VR-Forces attaches the observer when the entity is next discovered.



If your display configuration has more than one observer, you can save multiple views and automatically bind the observers to the views. The observers are bound to views on a one-to-one basis in the order that the observers are listed in the command and the saved views are listed in the file. If you only have one observer, it is automatically bound to the first view in the file.



Intervisibility

This chapter explains how to use the intervisibility feature. For an introduction to intervisibility, please see “Intervisibility (Line-of-Sight),” on page 3-34.

Types of Intervisibility.....	13-2
Point-to-Point Intervisibility.....	13-2
Intervisibility Fan	13-3
Entity Intervisibility	13-3
Intervisibility Objects can be Transient or Permanent	13-4
Creating a Transient Intervisibility Object	13-4
Creating a Permanent Intervisibility Object.....	13-5
Editing an Intervisibility Object.....	13-5
Configuring Intervisibility Lines and Fans.....	13-6
Editing an Intervisibility Object’s Properties.....	13-7
Pinning an Intervisibility Object to an Entity	13-8
Deleting Intervisibility Objects	13-8

13.1. Types of Intervisibility

VR-Forces supports the following types of intervisibility:

- ♦ Point-to-point (intervisibility line)
- ♦ Intervisibility fan
- ♦ Entity intervisibility.

The intervisibility fan and entity intervisibility are displayed as a circular area. However, intervisibility calculations are made within a cylindrical area extending above and below the terrain, so that intervisibility calculations can be made for air platform entities and subsurface entities.

In the 2D view, intervisibility lines are fully visible on the terrain. In the 3D view, they may pass through the terrain.

13.1.1. Point-to-Point Intervisibility

Point-to-point intervisibility ([Figure 13-1](#)) shows the line-of-sight between any two points. The portion of the line that is visible from the starting point is green. The portion that is not visible is red. You can change the colors used.



Figure 13-1. Point-to-point intervisibility

13.1.2. Intervisibility Fan

An intervisibility fan (Figure 13-2) shows a series of lines radiating from the center of a circle to its outer edge. Each ray shows intervisibility along its path from the origin to the border. You can configure the distance between rays. (For details, please see “Configuring Intervisibility Lines and Fans,” on page 13-6.)

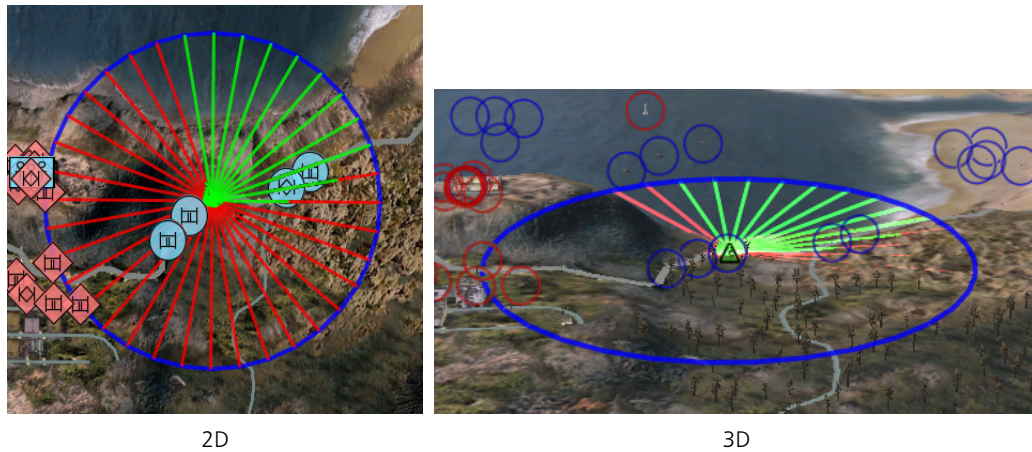


Figure 13-2. Intervisibility fan

13.1.3. Entity Intervisibility

Entity intervisibility (Figure 13-3) shows which entities are visible from the center of a defined area and whether or not the entities can see the center point. If you center the intervisibility circle on a point on the terrain, you can see which entities can see that point. If you lock the circle to an entity, you can see which entities can see that entity and which it can see. (For details, please see “Configuring Intervisibility Lines and Fans,” on page 13-6.)



Figure 13-3. Entity intervisibility

13.2. Intervisibility Objects can be Transient or Permanent

You can configure intervisibility objects to be transient or permanent. (For details, please see “[Configuring Intervisibility Lines and Fans](#),” on page 13-6.) A transient intervisibility object displays intervisibility from the origin point (defined by a mouse click) to wherever you move the cursor. If you click the mouse again, you reset the origin. You cannot use the mouse for other actions until you exit intervisibility mode (by right-clicking). This mode is convenient when you want to quickly check intervisibility, but do not want to keep intervisibility objects in the display.

A permanent intervisibility object stays in the display after you create it. You can resize the object, edit drawing options, and move it around the display. Intervisibility objects are not saved as part of the scenario.

13.2.1. Creating a Transient Intervisibility Object

Transient intervisibility objects do not stay in the display after you quit intervisibility mode.

To display transient intervisibility:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Intervisibility Settings page ([Figure 13-4](#)).

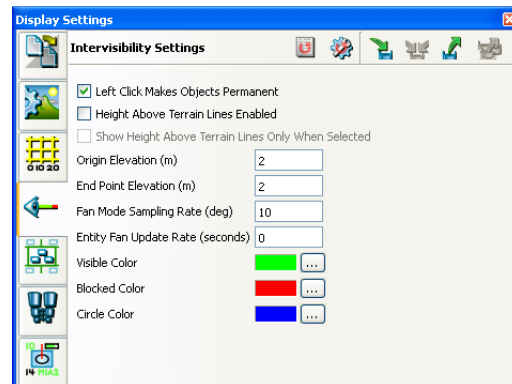


Figure 13-4. Intervisibility Settings page

3. Clear the Left Click Makes Objects Permanent check box.
4. Choose **Intervisibility** → *intervisibility_type*, where *intervisibility_type* is one of the options, or click one of the icons on the Intervisibility toolbar.
5. Click on the terrain to place the origin of the intervisibility object.
6. Move the cursor around the terrain to expand and contract the object. For point-to-point intervisibility, moving the cursor also changes the direction of the line.
7. To change the origin of the intervisibility display, click another location on the terrain.

8. Right-click to cancel intervisibility mode.

13.2.2. Creating a Permanent Intervisibility Object

Permanent intervisibility objects remain in the display after you create them. (However, they are not saved as part of the scenario.)

To create a permanent intervisibility object:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Intervisibility Settings page (Figure 13-4).
3. Select the Left Click Makes Objects Permanent check box.
4. Choose **Intervisibility** → *intervisibility_type*, where *intervisibility_type* is one of the options, or click one of the icons on the Intervisibility toolbar.
5. Click on an entity or on a location on the terrain where you want the origin of the intervisibility object to be located. An intervisibility object is displayed. As you move the cursor closer to the origin, or farther away from it, the circle or line contracts and expands. If you are displaying point-to-point, you can rotate the line around the origin before you specify the end point.
6. Click on the terrain where you want to fix the end point of a line or border of a circle.

13.3. Editing an Intervisibility Object

You can change the following aspects of an intervisibility object:

- ♦ Change the location of either end point of an intervisibility line, to change its length and direction
- ♦ Change the origin of an intervisibility fan, thereby moving the entire fan
- ♦ Change the diameter of an intervisibility fan.



If an intervisibility object is anchored to an entity, you cannot change the origin.

To change the length or diameter of an intervisibility object:

1. Double-click the object's end point or circle. The cursor changes.
2. Drag the endpoint or circle towards or away from the origin. If you are editing an intervisibility line, you can move the end point to any location.
3. Click to set the endpoint.

To move an intervisibility fan or line:

1. Double-click a fan's center point or any place on a line. The cursor changes.
2. Drag the object to a new location.
3. Click to set the location.

13.4. Configuring Intervisibility Lines and Fans

You can configure the following attributes of intervisibility objects:

- The elevation of the origin and end point
- The sampling rate for intervisibility fans
- The update rate for entity intervisibility fans
- Display of height above terrain lines (3D) and height labels (3D and 2D)
- The colors used to display intervisibility objects.

You can change these settings globally and for individual intervisibility objects. When you change colors or the sampling rate, changes affect all existing intervisibility objects. Changes to the origin and end point elevation only affect newly created objects.

To configure intervisibility objects:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Intervisibility Settings page (Figure 13-4).
3. Change the settings described in Table 13-1.

Table 13-1: Intervisibility settings

Option	Description
Enable Height Above Terrain Lines	Enables or disables display of height above terrain lines (3D) and height labels (3D and 2D).
Show Height Above Terrain Lines Only for Selected	If selected, HAT lines are displayed only for the selected intervisibility objects.
Origin Elevation	The height, in meters, above the terrain at the origin point from which to calculate intervisibility.
End Point Elevation	The height, in meters, above the terrain at the end point to use to calculate intervisibility.
Fan Mode Sampling Rate	The distance, in degrees, between each ray in a intervisibility fan. After changing this value, press Enter or click away from the field.
Entity Fan Update Rate	How frequently, in seconds, to update the entity intervisibility fan.

Table 13-1: Intervisibility settings

Option	Description
Visible Color	The colors to use for the portion of an intervisibility line that an entity can see and the portion where visibility is blocked.
Blocked Color	
Circle Color	The color to use for the outer circle of an intervisibility fan.

- To apply the changes to specific intervisibility objects, select the objects and click Apply to Selected.

13.4.1. Editing an Intervisibility Object's Properties

If you want a particular intervisibility object to have different properties than those set on the Intervisibility page, you can edit that object's properties.



If, after editing an intervisibility object's properties, you make a global change in the Intervisibility Settings page, the global setting overrides the individual setting.

To edit an intervisibility object's properties:

- Select the object.
- Choose **Intervisibility** → **Properties**. The Intervisibility Properties dialog box opens (Figure 13-5). It displays the properties that are appropriate for the object.

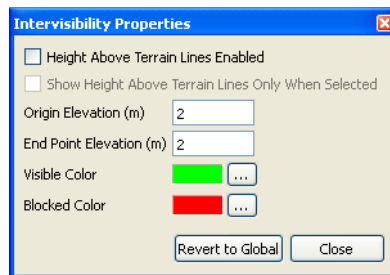


Figure 13-5. Intervisibility Properties dialog box

- Change the properties as desired. They are described in Table 13-1.

13.5. Pinning an Intervisibility Object to an Entity

When you pin an intervisibility object to an entity, the intervisibility object moves with the entity and you know that its origin is centered on the entity.

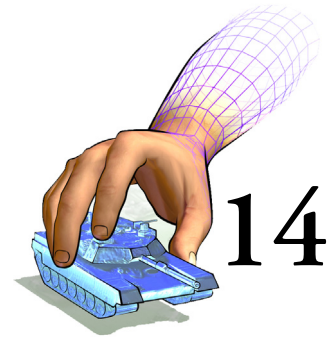
To pin an intervisibility object to an entity:

1. Select the entity to which you want to pin the intervisibility object.
2. Choose **Intervisibility** → *intervisibility_type*, or click one of the icons on the Intervisibility toolbar. The origin is placed on the entity icon or model.
3. Specify the end point for the intervisibility object.

13.6. Deleting Intervisibility Objects

Before you delete an intervisibility object, note the following:

- ♦ If you delete an entity that has an intervisibility object pinned to it, the intervisibility object also gets deleted.
 - ♦ If an intervisibility object is pinned to an entity, you can delete the intervisibility object without affecting the entity by selecting the outer circle or end point and deleting it.
- To delete an intervisibility object, select the object and choose **Intervisibility** → **Delete**



Changing How Terrain is Displayed

This chapter describes how to change the display of terrain to provide different types of useful terrain information.

Introduction	14-2
Configuring Visible Surfaces (GDB Soil Types)	14-2
Coloring Terrain Based on Elevation.....	14-4
Configuring Elevation Color Settings.....	14-5
Displaying Grid Lines	14-6
Configuring Grid Lines	14-7
Displaying Contour Lines	14-8
Configuring Contour Lines	14-9
Displaying Models and Terrain in Wireframe Mode	14-9
Displaying Feature Data.....	14-10
Changing the Colors Used for Feature Data	14-11
Terrain Databases Provided with VR-Forces	14-13

14.1. Introduction

As you create and view simulations, you may need information about the terrain. By changing how terrain is displayed, you can get some of this information. If your terrain does not have texture data or an image overlay, features such as elevation coloring can provide a more realistic look to the terrain.

For information about how to compose terrains, please see Chapter 11, [Composing Terrains](#), in *VR-Forces Configuration Guide*.

14.2. Configuring Visible Surfaces (GDB Soil Types)

If a terrain patch was loaded from a GDB file, you can display or hide GDB soil types. [Figure 14-1](#) shows the *makland.gdb* terrain file. The view on the left shows all soil types. The view on the right has Building, Grass, Ocean, and Rock surfaces hidden.

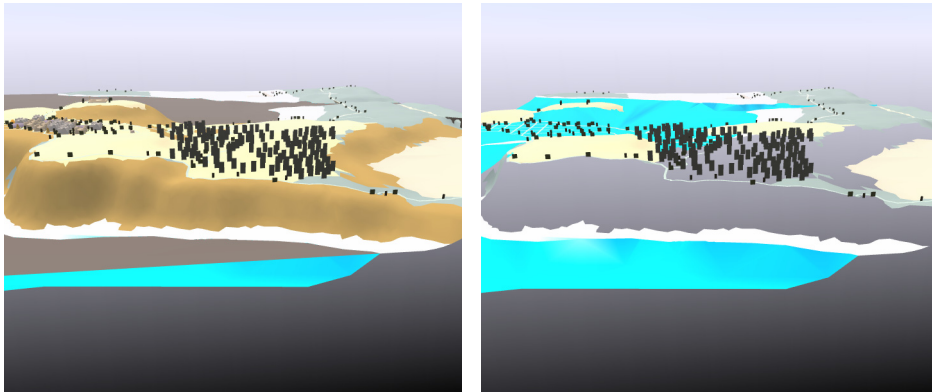


Figure 14-1. Visible surfaces toggled off

To display or hide GDB soil types:

1. Choose **Settings** → **Terrain**. The Terrain Settings dialog box opens.
2. Select the Visible Surfaces page ([Figure 14-2](#)).

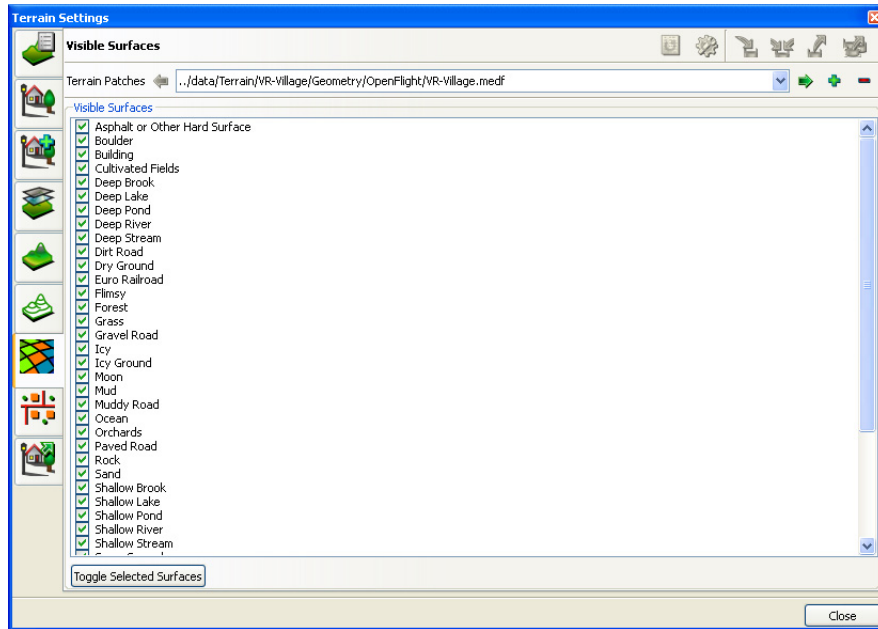


Figure 14-2. Visible Surfaces page

3. In the Terrain Patches list, select the terrain patch that you want to configure.
4. Select the check box for each soil type that you want to be visible.
5. Clear the check box for each soil type that you want to hide.
6. To change the state of multiple soil types at one time:
 - a. Select (highlight) the names of the soil types whose state you want to change.
 - b. Click Toggle Selected Surfaces.

14.3. Coloring Terrain Based on Elevation

You can color the terrain based on elevation. If you have a raster map, the color is superimposed on the image.

To enable or disable coloring of terrain by elevation:

1. Choose **Settings** → **Terrain**. The Terrain Settings dialog box opens.
2. Select the Elevation Color Settings page (Figure 14-3).

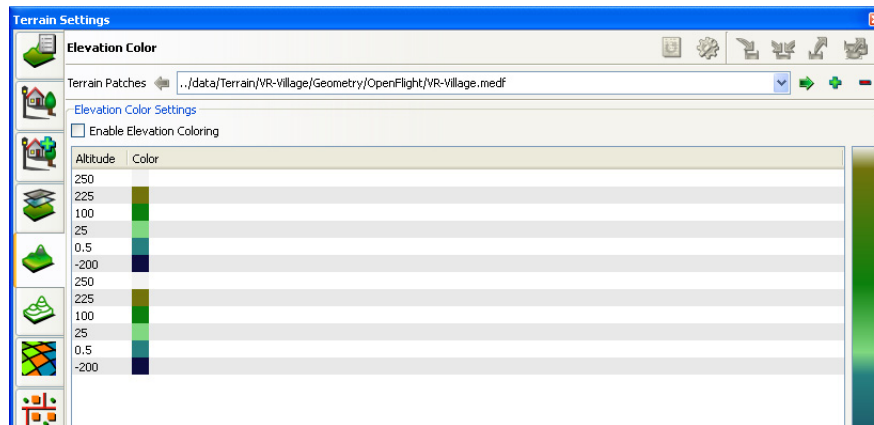


Figure 14-3. Elevation Color Settings page

3. In the Terrain Patches list, select the terrain patch for which you want to enable elevation coloring.
4. Select or clear the Enable Elevation Coloring check box.

14.3.1. Configuring Elevation Color Settings

You can specify the color applied to terrain at a given altitude. You can add altitude points and remove them.

Inserting an Altitude Point

To insert an altitude point:

1. Choose **Settings** → **Terrain**. The Terrain Settings dialog box opens.
2. Select the Elevation Color Settings page ([Figure 14-3](#)).
3. In the Terrain Patches list, select the terrain patch for which you want to insert an altitude point.
4. Select an altitude point in the list that is above or below the altitude point you want to add.
5. Click Insert Point. A new point is added with the same altitude as the one you selected.
6. Double-click the new altitude point to make it editable.
7. Set the altitude point to the altitude that you want.
8. Click away from the altitude point to set it.

Deleting an Altitude Point

To delete an altitude point:

1. Choose **Settings** → **Terrain**. The Terrain Settings dialog box opens.
2. Select the Elevation Color Settings page ([Figure 14-3](#)).
3. In the Terrain Patches list, select the terrain patch for which you want to delete an altitude point.
4. Select the altitude point that you want to delete.
5. Click Delete Point.

Changing the Color of an Altitude Point

To change the color of an altitude point:

1. Choose **Settings** → **Terrain**. The Terrain Settings dialog box opens.
2. Select the Elevation Color Settings page (Figure 14-3).
3. In the Terrain Patches list, select the terrain patch for which you want to change the color of an altitude point.
4. Double-click the color swatch for the altitude point whose color you want to change. A color chooser dialog box opens.
5. Select or specify a new color.
6. Click OK.

14.4. Displaying Grid Lines

VR-Forces can display grid lines in the 2D view. Grid-lines are an observer-specific setting.

To enable or disable grid lines:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Observer Mode Settings page (Figure 8-6).
3. Select the observer mode for which you want to enable or disable grid lines (2D modes only).
4. Select or clear the Display Grid Lines check box.



You can also enable or disable track histories by choosing **Observer** → **Display Grid Lines** or clicking the Display Grid Lines icon () on the Observer Settings toolbar.

14.4.1. Configuring Grid Lines

You can configure the coordinate system used for grid lines, the precision, and spacing. Spacing can be fixed, in meters, or determined by VR-Forces using a specified number of screen pixels. Fixed spacing can result in a very tight grid and unreadable labels as you zoom out. Auto spacing does not have this problem. However, the grid lines will constantly recalculate as you zoom in and out.

To configure grid lines:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Grid Lines Settings page (Figure 14-4).

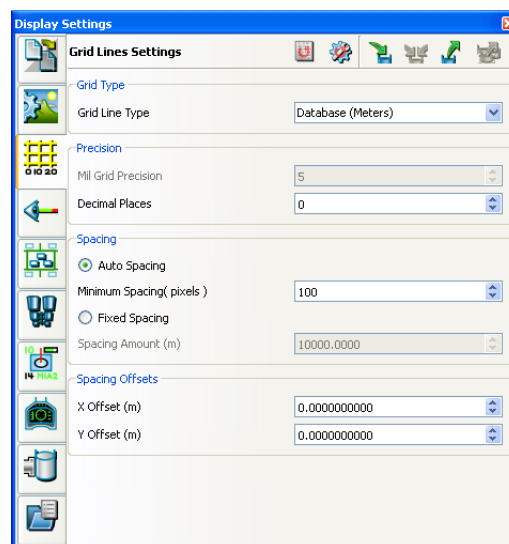


Figure 14-4. Grid Lines Settings page

3. To change the coordinate system used for grid lines, select an option in the Grid Line Type drop-down list. The list displays grid line types that are appropriate for the type of database loaded. (If no database is loaded, the coordinate system is set to Flat Earth.)
4. To specify the precision of the grid line labels, type a value in the appropriate option in the Precision group box.
5. To change the spacing type, select an option in the Spacing group box and specify a value for the chosen spacing type.
6. To specify an offset from the origin, change the values in the Spacing Offsets group box.

14.5. Displaying Contour Lines

You can display contour lines and intermediate contour lines on the terrain. Intermediate contour lines divide the contour line interval into fifths (Figure 14-5). You can configure the thickness of contour lines and the interval between them.



Figure 14-5. Contour lines and intermediate contour lines

To display contour lines:

1. Choose **Settings** → **Terrain**. The Terrain Settings dialog box opens.
2. Select the Contour Line Settings page (Figure 14-6).

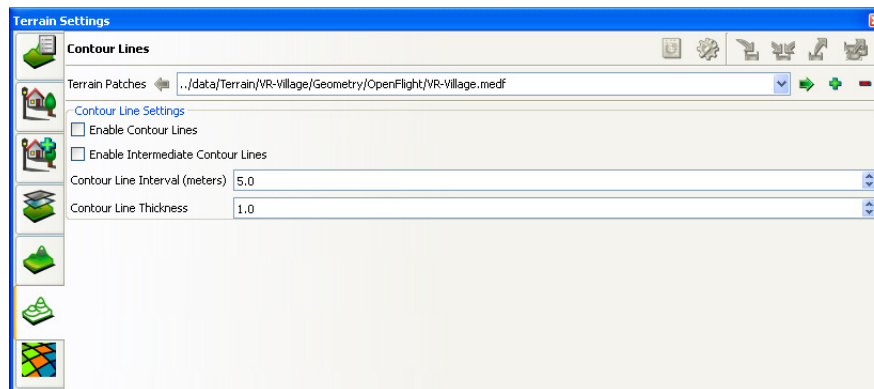


Figure 14-6. Contour Lines page

3. In the Terrain Patches list, select the terrain patch for which you want to enable or disable the display of contour lines.
4. Select Enable Contour Lines.
5. To enable intermediate contour lines, select Enable Intermediate Contour Lines.

14.5.1. Configuring Contour Lines

You can configure the thickness of contour lines and the interval between them.

To configure contour lines:

1. Choose **Settings** → **Terrain**. The Terrain Settings dialog box opens.
2. Select the Contour Line Settings page ([Figure 14-6](#)).
3. In the Terrain Patches list, select the terrain patch for which you want to configure the thickness and interval of contour lines.
4. In the Contour Line Interval text box, specify the distance, in meters, between contour lines.
5. In the Contour Line Thickness text box, specify the thickness, in pixels, for contour lines.

14.6. Displaying Models and Terrain in Wireframe Mode

In wireframe mode, VR-Forces displays the polygons that make up the terrain and objects ([Figure 14-7](#)).



Figure 14-7. Wireframe off and on

To enable or disable wireframe mode:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Render Settings page ([Figure 14-8](#)).

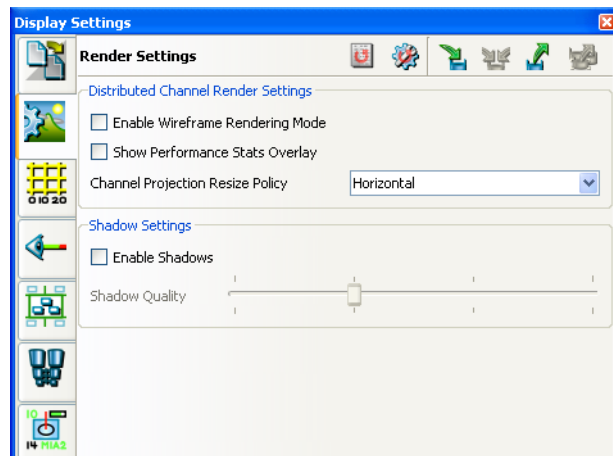


Figure 14-8. Render Settings page

3. Select the Enable Wireframe Rendering Mode check box.



You can also enable and disable wireframe mode by choosing **Settings** → **Wireframe Rendering** or clicking the Wireframe Rendering icon (⌘) on the Display Settings toolbar.

14.7. Displaying Feature Data



Figure 14-9. Feature data

- To display feature data, choose **Observer** → **Features**, or click the Feature Data icon (⌘) on the Observer Settings toolbar.

14.7.1. Changing the Colors Used for Feature Data

If a terrain database contains feature data, you can change the colors used to display it. In [Figure 14-10](#), we have changed the Woodland feature in the Little Pond terrain to be pink. Compare it to the default color ([Figure 14-9](#)).



Figure 14-10. Feature data with color change

To change the color of feature data:

1. Choose **Settings** → **Terrain**. The Terrain Settings dialog box opens.
2. Select the Feature Settings page ([Figure 14-11](#)).

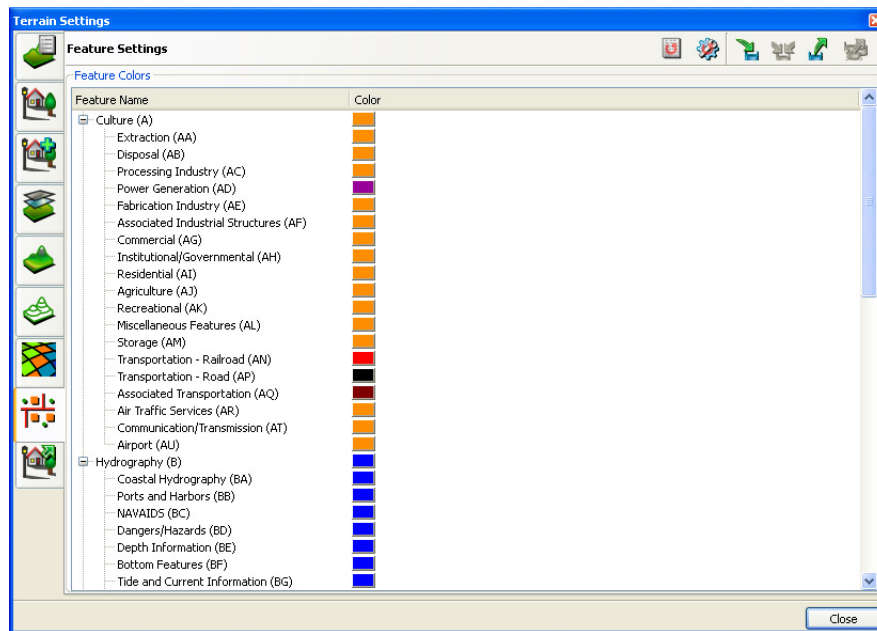


Figure 14-11. Features page

3. Click the color swatch for the feature that you want to change. A color picker dialog box opens.
4. Select the color that you want to use.
5. Click OK.

14.8. Terrain Databases Provided with VR-Forces

Table 14-1: Terrain databases shipped with VR-Forces

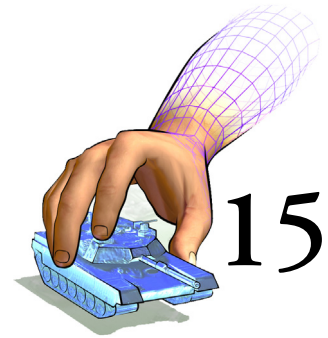
Database	Variation	Description
Berkeley	Berkeley.mtf	UTM. OpenFlight. Feature data. Shows the area around Berkeley, California, USA. It is provided courtesy of Terrex.
Ground	ground-db.mtf	UTM. OpenFlight. A generic relatively flat ground terrain.
HarrisAfghan-Village	HarrisAfghanVillage-FlatEarthMetaFlight.mtf	Paged terrain using flat earth projection. A terrain of Afghanistan with a high resolution inset of a small village. This terrain was created in VR-Vantage from source data. The terrain data is provided courtesy of Harris Inc.
	HarrisAfghanVillageGeocentricMetaFlight.mtf	Paged terrain using geocentric coordinates.
Kabul	Kabul.mtf	UTM. OpenFlight. The area around Kabul, Afghanistan.
Little Pond	LittlePond.mtf	Flat earth. DTED. Shape data. Imagery. A detailed suburban inset in a larger, less detailed terrain covering Boston, Massachusetts, U.S.A. and points west. This terrain was created in VR-Vantage using source data as described in "Composing a Terrain," on page C-2, in <i>VR-Forces Configuration Guide</i>
	LittlePondVR-TheWorldOnline.mtf	Little Pond features inset into VR-TheWorld Online.
Makland	Makland.mtf	UTM. OpenFlight. A fictional terrain created by MÄK. It includes a variety of terrain and feature data, including underwater features. It is placed in the Indian Ocean.
	Makland-geocentric.mtd	A geocentric OpenFlight terrain.
	MaklandLocalWorld.mtd	Geocentric. Low resolution local terrain.
	MaklandNoSpeedtrees.mtf	No SpeedTrees for improved performance.

Table 14-1: Terrain databases shipped with VR-Forces

Database	Variation	Description
Nellis	Nellis.mtf	UTM. An example of a SAF terrain suitable for simulation, but not typically used in 3D imaging. A terrain covering the area around the Nellis Air Force Base, in Nevada, U.S.A.
OpenStreet-MapServer	OpenStreetMapServerOnline.mtf	Connects to a remote terrain server.
SanLuisObispo	SanLuisObispo.mtf	Flat earth. DTED. Imagery. A terrain that covers a portion of California, U.S.A.
	SanLuisObispo-TopoGdal.mtf	Geocentric. San Luis Obispo in topographic coordinates as local streaming terrain using a .earth file.
TerraSim_SampleUrban	TerraSim_SampleUrban.mtf	UTM. An urban database based on Pittsburgh, Pennsylvania, U.S.A., with a highly detailed segment that includes feature data and buildings with multiple floors. It is provided courtesy of TerraSim.
VR-TheWorldOnline	VR-TheWorldOnline.mtf	Lets you quickly connect to the publicly available VR-TheWorld Server for streaming terrain.
	VR-TheWorldOnline-high-ResolutionElevation-withBathymetry.mtf	A configuration that uses higher resolution LODs than VR-TheWorldOnline.mtf. This may result in reduced performance, particularly for aircraft.
VR-Village	VR-Village.mtf	GDB, feature data, OpenFlight terrain. <i>VR-Village</i> is a fictional database created by VT MÄK. It has a highly detailed small town in a larger, sparsely detailed desert environment. The town includes multi-level buildings, underground tunnels, and a road that tunnels through a mountain. The coordinates place it off the west coast of Africa.
	VR-VillageAfghanistanVR-TheWorldOnline.mtf	Geocentric. The high resolution data from VR-Village.mtf is inset in Afghanistan in VR-TheWorld online server.
	VR-VillageNoSpeedTrees.mtf	VR-Village with no SpeedTrees for improved performance.

Table 14-1: Terrain databases shipped with VR-Forces

Database	Variation	Description
World	WorldFlatEarthNoElevation.mtf	Flat earth. Low resolution projected image of the world.
	WorldGeocentric.mtf	Geocentric. Low resolution projection of the world.



Remote Graphics

Remote graphics are graphics that are drawn onto the scene by remote programs that use the VR-Vantage Remote Draw API. This chapter explains how to display them in VR-Forces.

Displaying Remote Graphics	15-2
Viewing a List of Remote Graphics	15-3

15.1. Displaying Remote Graphics

VR-Forces can display graphics sent from other applications using the VR-Vantage Remote Draw API ([Figure 15-1](#)).

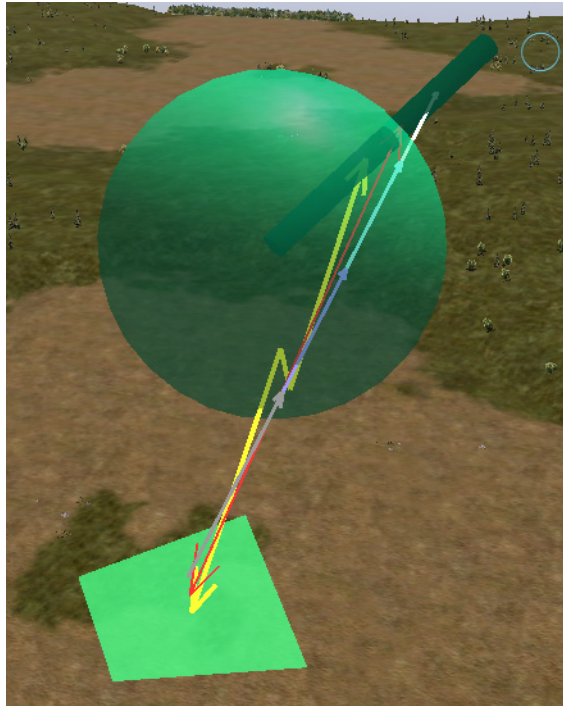


Figure 15-1. Remote graphics

To enable or disable the display of remote graphics:

1. Choose **Settings** → **Display**. The Display Settings dialog box opens.
2. Select the Tactical Graphics Display Settings page ([Figure 15-2](#)).

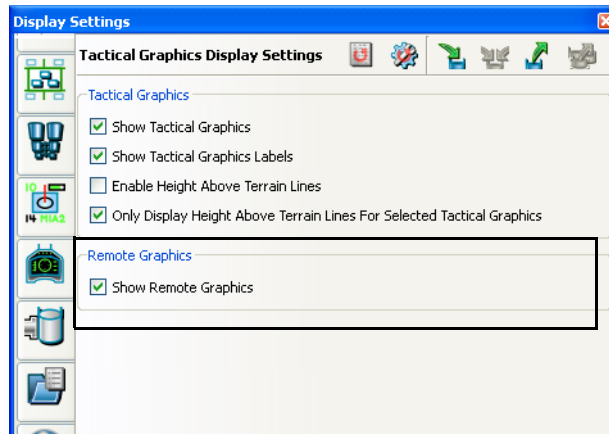


Figure 15-2. Tactical Graphics Display Settings page

3. Select or clear the Show Remote Graphics check box.



You can also enable and disable remote graphics by choosing **Settings** → **Remote Graphics** or by clicking the Remote Graphics icon (🎮) on the Display Settings toolbar.

15.1.1. Viewing a List of Remote Graphics

Remote graphics can be organized into sets. There is no fixed organization. Labeling of sets is entirely at the discretion of the programmer who develops the application that generates the graphics. The Remote Graphics panel displays remote graphics in a hierarchy by set.

The VR-Vantage Remote Draw API lets developers send data about objects as attribute:value pairs, for example Speed:30 KM. The attributes are organized by type using attribute group templates. You can enable and disable the display of attribute information by group and by individual attribute. Figure 15-3 illustrates the attributes for the Entity and Packet attribute group templates displayed in (Figure 15-4).

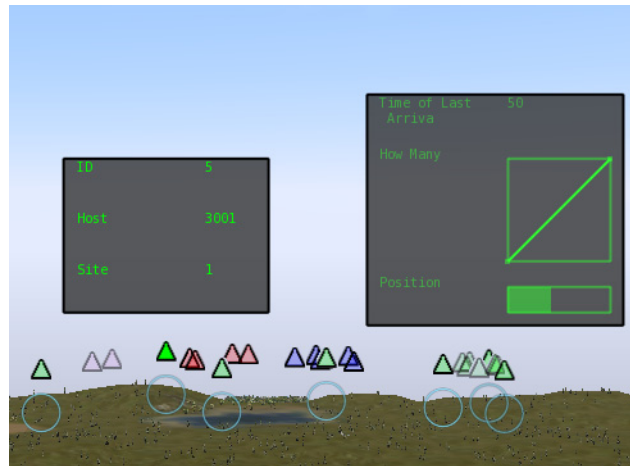


Figure 15-3. Remote graphics attribute group display

To view a list of remote graphics:

1. Choose **View** → **Remote Graphics Panel**. The Remote Graphics panel is displayed (Figure 15-4). The Remote Graphics Sets tab lists remote graphics by set.

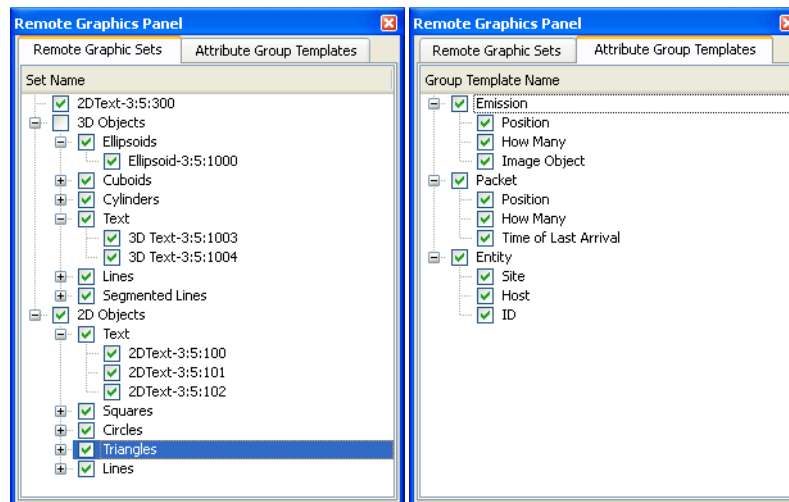
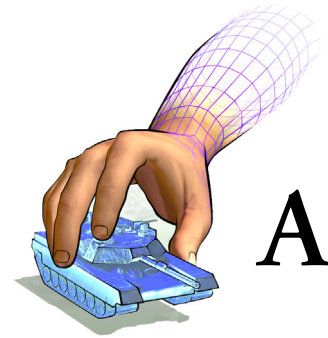


Figure 15-4. Remote Graphics panel

2. To hide or display a remote graphic set or individual graphic, check or clear its check box in the Remote Graphics Sets list.
3. To view Attribute Group Templates, select the Attribute Group Templates panel. To see the attribute display, mouse over an object for which display data is available.



Common GUI Controls and Behaviors

This appendix provides details about using the various panels and toolbars in the front-end.

Dockable Panels and Toolbars	A-2
Docking and Undocking a Panel	A-2
Docking and Undocking a Toolbar	A-3
Displaying and Hiding Panels and Toolbars	A-3
Restoring the Layout of Toolbars and Panels	A-4
Using Tear-Off Menus	A-5
Viewing the Window in Full Screen Mode	A-5
Using Context-Sensitive Menus	A-6
Common VR-Forces Icons	A-6

A.1. Dockable Panels and Toolbars

VR-Forces has floating, dockable panels for many of its important functions (Figure A-1). It also has toolbars that let you quickly access the commands available on the menus.

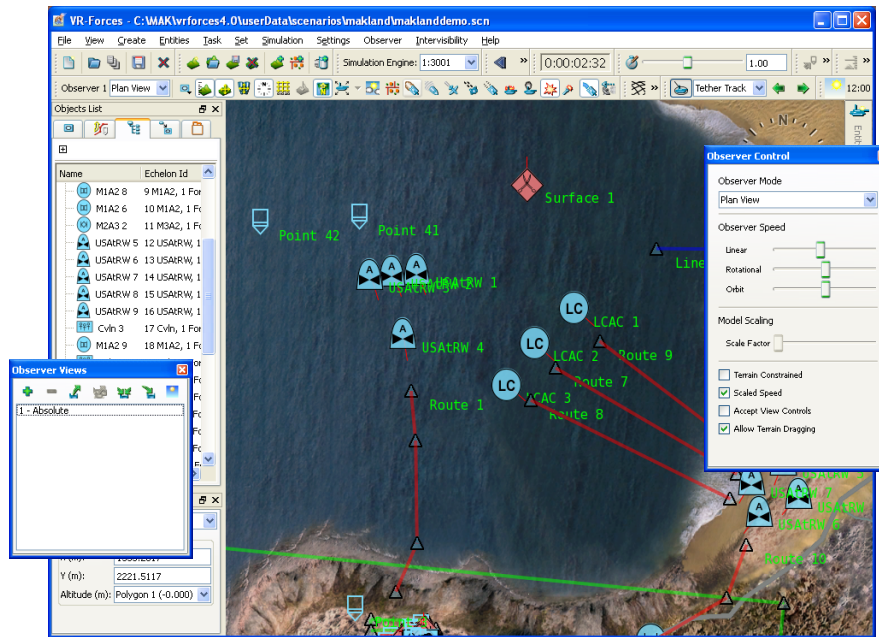


Figure A-1. VR-Forces window and undocked panels

A.1.1. Docking and Undocking a Panel

To undock a panel:

1. Place the cursor over the panel's title.
 2. Press and hold the left mouse button.
 3. Drag the panel away from the window.
- To dock a floating panel, double-click its title bar.

A.1.2. Docking and Undocking a Toolbar

To undock a toolbar:

1. Place the cursor over the handle.



Figure A-2. Toolbar handle

2. Press and hold the left mouse button.
 3. Drag the toolbar away from the window.
- To dock a toolbar, drag it back to the toolbar strip. The other toolbars will move to create a space to drop in the toolbar (Figure A-3).

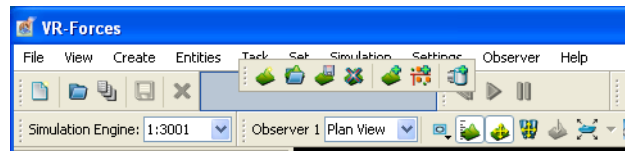


Figure A-3. Docking a toolbar

A.1.3. Displaying and Hiding Panels and Toolbars

You can hide any of the panels or toolbars.

- To hide or view a panel or toolbar, select it on the View menu.

A.1.4. Restoring the Layout of Toolbars and Panels

VR-Forces saves the layout of toolbars and panels when you shut down. You can restore the factory layout or the last saved layout.

To restore the layout of toolbars and panels:

1. Choose **Settings** → **Application**. The Application Settings dialog box opens.
2. Select the GUI Layout page (Figure A-4).

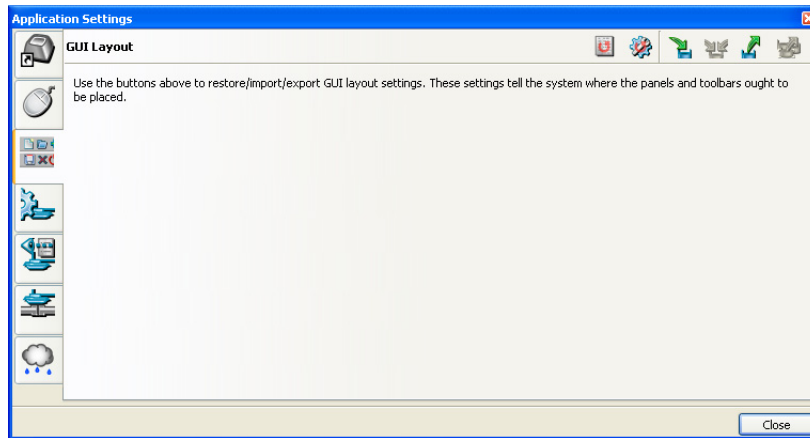


Figure A-4. GUI Layout page

3. Click the settings icon for the operation you want to perform. (For a summary of common settings icons, please see “[Managing VR-Forces Settings](#),” on page 5-19.)

A.2. Using Tear-Off Menus

You can “tear-off” some menus and leave them displayed on the desktop.

To tear off a menu:

1. Click a menu title to display the submenu.
2. Move the cursor over the grab bar (dashed line) at the top of the menu to highlight it (Figure A-5).
3. Click the grab bar. The menu is displayed as a toolbar that you can leave open and reposition.

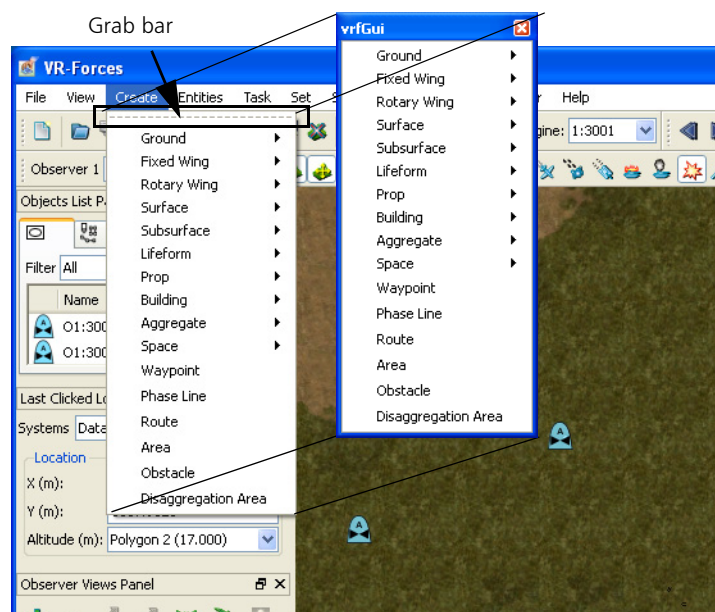


Figure A-5. Tearing off a menu

A.3. Viewing the Window in Full Screen Mode

You can display VR-Forces so that it covers the entire screen and does not show the title bar or border. This mode is intended for use in demonstrations. When menus and panels are hidden, you can still use all keyboard controls, and you can still click entities in the window to attach the observer to them.

You can configure VR-Forces to start in full screen mode with the `--fullScreen` command-line option.

- To view VR-Forces in full screen mode, choose **View** → **Show Full Screen**, or press **Ctrl+Enter**.
- To exit full screen mode, press **Ctrl+Enter**.

A.4. Using Context-Sensitive Menus

VR-Forces has context sensitive menus that you can display by clicking the right mouse button.

If you right-click an entity, tactical graphic, or other scene object, the menu displays the actions that you can take.

If you right-click the menu bar, you can select the commands available on the View menu.

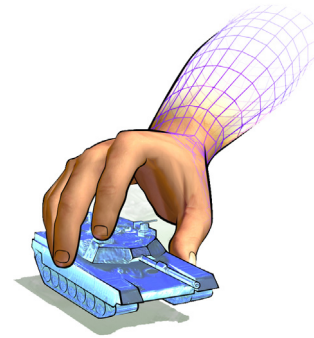
A.5. Common VR-Forces Icons

VR-Forces dialog boxes have a common set of icons (Figure A-6). The callouts in (Figure A-6) correspond to the numbers in the list of icon descriptions.



Figure A-6. Common icons

- ♦ Add an item (definition, model, and so on) (1).
- ♦ Remove the selected item (2).
- ♦ Copy the selected item and add the copied version to the list (3).
- ♦ Select a filter of some type (4).
- ♦ Rename the selected item (5).
- ♦ Add from a list (6).
- ♦ Move up in the list (7).
- ♦ Move down in the list (8).
- ♦ Move upward in the hierarchy (to the parent level) (9).
- ♦ Move downward in the hierarchy (become a child) (10).
- ♦ Copy the selected object to the clipboard (11).
- ♦ Paste an object that has been copied to the clipboard (12).
- ♦ Add attribute (13).
- ♦ Remove attribute (14).
- ♦ Search (15).



MÄK Products Glossary

This glossary defines terms used in MÄK product documentation.

absolute dead reckoning

A method of [dead-reckoning](#) in which an application uses the time stamps contained within state updates if the sender is using absolute time stamping. 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.

AMO

See [Application Management Object](#).

Application Management Object

An Application Management Object (AMO) is a standard TENA object that is supposed to be used by every TENA application. It gives other applications information about the local application. The VR-Link exercise connection, by default, publishes an AMO object, updates it at a specified rate (the rate is part of the AMO's state repository; the default is 30 seconds), and updates the number of objects published and proxies (TENA reflected objects) held.

API

Application Programming Interface.

articulated part

A part of an entity that is capable of movement relative to the entity, such as a turret or gun.

attached part

An object that is attached to another object, such as a rocket attached to a jet.

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.

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 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.

Data Distribution Management

The set of HLA 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.

DDM

[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

[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](#).

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 MÄK 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.

execution

The TENA term for one or more interacting simulation applications.

Execution Manager

An Execution Manager governs a TENA execution for applications joining, resigning, or changing subscriptions to that execution.

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 MÄK products are VR-Link applications.)

exercise ID

A numeric identification for a DIS simulation exercise.

FED file

federate

A connection to the [RTI](#). Typically a single simulation application can be thought of as a federate.

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, defines additional modular data content for a federation execution, usually extensions to an existing FOM. (Also supported by the HLA 1.3 and HLA 1516 versions of the MÄK 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.

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 through 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 MÄK 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.

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 PDUs are sent to the network regardless of whether or not 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](#).

interaction

A [message](#) describing a simulation event. An interaction describes an event, it does not update an object's state.

Logger control PDUs

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 an RTI. A federate communicates with the HLA network through the LRC.

Logical Range Object Model

The data definition file for TENA exercises. It contains the set of object and message definitions that may be used in the exercise.

LRC

See [Local RTI Component](#).

LROM

See [Logical Range Object Model](#).

MÄK Technologies Lisp

An adaptation of the Lisp language used in configuration files for MÄK 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 HLA standard MOM. It is a subset of the FOM 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 RTI. The standard MIM can be replaced with a user-supplied MIM containing the standard MIM plus extensions.

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](#).

Network Naming Service

Acts as a registry for TENA Execution Managers.

NNS

See [Network Naming Service](#).

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 RTI 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 RTI, 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 MÄK 3D applications, the point of view into the scene. (Called the eyepoint in MÄK Stealth 6.x and earlier.)

PDU

See [Protocol Data Unit](#).

Protocol Data Unit

A unit of data message (packet) that is passed on a network between DIS simulation applications.

proxy

In TENA, a reflected object ([Stateful Distributed Object](#)).

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.

range resource

A participant in a TENA execution (similar in concept to the HLA federate).

Realtime Platform Reference FOM

An [HLA](#) reference [FOM](#) based on the [DIS](#) protocol.

recording

A 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 [FOM](#) 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.

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

The initialization data required by the RTI for operation.

Data required by an RTI during initialization, independent of the FOM being used. RID data is usually dependent on a specific implementation of the RTI.

RPR FOM

See [Realtime 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.

SDO

See [Stateful Distributed Object](#).

servant

A published object (SDO) in TENA.

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.

Stateful Distributed Object

An object in a TENA exercise.

tape

An alternative term for a Logger recording. Not a physical magnetic tape.

TENA

See [Test and Training Enabling Architecture](#).

TENA Middleware

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.

Test and Training Enabling Architecture

Test and Training Enabling Architecture (TENA) is an interoperability architecture used to pass data in the form of object updates and messages (like HLA objects and interactions) from one application to another.

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.

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

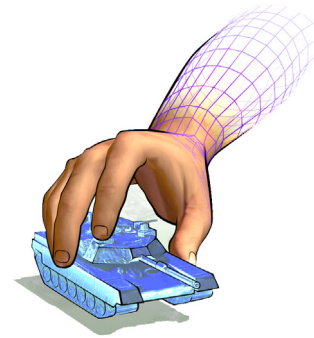
See [Universal Transverse Mercator](#).

view control messages

A set of programmatic messages that let you control the VR-Vantage remotely.

view floor

In MÄK Stealth 6.x and earlier, a minimum height above the terrain in Absolute or Track mode, below which the observer is not allowed to go. The view floor is measured relative to the terrain surface.



Index

- command-line option 5-25, 5-30, 5-32
- app command-line option 5-32
- appDataDir command-line option 5-25
- appIdRange command-line option 5-28
- appNumber command-line option 5-27, 5-28, 5-31
- autoConnect command-line option 5-25
- bindObservers command-line option 5-25
- configFilename command-line option 5-32
- configureConnections command-line option 5-32
- daemon command-line option 5-28
- dataDir command-line option 5-25
- destAddrString command-line option 5-27, 5-31
- deviceAddress command-line option 5-27, 5-31
- diGuyAnimationsFile command-line option 5-28
- diGuyCharacerDataFile command-line option 5-28
- dis command-line option 5-25
- disPort command-line option 5-27, 5-31
- dispSetting command-line option 5-25
- disVersion command-line option 5-27, 5-31, 5-38
- doNotCheckPluginVersions command-line option 5-25
- doNotLoadVrfPlugins command-line option 5-26, 5-28
- entDispSetting command-line option 5-26
- entTypeMap command-line option 5-26
- envSetting command-line option 5-26
- execname command-line option 5-27, 5-31
- exerciseId command-line option 5-27, 5-31
- federateName command-line option 5-27, 5-30
- fedFileName command-line option 5-27, 5-30
- fomMapperInitData command-line option 5-27, 5-30
- fomMapperLib command-line option 5-27, 5-30
- frontEndPID command-line option 5-28
- fullScreen command-line option 5-26, A-5
- fullyLoadTerrain command-line option 5-28
- help command-line option 5-26
- hla13 command-line option 5-26
- hla1516 command-line option 5-26
- ignore_rest command-line option 5-25, 5-30
- loadObservers command-line option 5-26
- loadPlugi command-line option 5-26
- loadPlugin command-line option 5-29, 5-34
- locale command-line option 5-26
- logFileName command-line option 5-26
- mcastTtl command-line option 5-27, 5-31
- multicastAddresses command-line option 5-27, 5-31
- noGui command-line option 5-26
- noPluginDirectory command-line option 5-29
- notifyLevel command-line option 5-26, 5-29
- observerMode command-line option 5-26
- outputLogFile command-line option 5-29
- physicalWorldFile command-line option 5-29
- plugin command-line option 5-26
- recvBufferSize command-line option 5-27, 5-31
- rprFomRevision command-line option 5-27, 5-30
- rprFomVersion command-line option 5-27, 5-30
- scenarioFile command-line option 5-26
- sendBufferSize command-line option 5-27, 5-31
- settingsFile command-line option, vrfSim 5-28
- showConsole command-line option 5-26
- simulationModelSet command-line option 5-30

- siteId command-line option 5-27
- suppressSelfReflect command-line option 5-31
- targetFrameRate command-line option 5-30
- timeManagement command-line option 5-16, 5-30, 5-37
- unplug command-line option 5-26
- useAdvisories command-line option 5-27, 5-30
- useAsyncIO command-line option 5-27, 5-31, 5-38
- useGeographicDdm command-line option 5-27, 5-31
- useIpv6 command-line option 5-31
- userDataDir command-line option 5-26
- version command-line option 5-27, 5-30, 5-32
- vsync command-line option 5-27
- withDisplayEngine command-line option 5-27

Numerics

- 2D
 - observer, attachment behavior 12-10
 - observer coordinate system 11-6
 - window 5-11
- 2D icon, label 7-3
- 2d icon, changing size 7-9
- 2D Icon model set 7-10
- 2D projection, switching to 3D 5-11
- 3D
 - observer coordinate systems 11-6
 - window 5-11
- 3D Colorized Models model set 7-10
 - color, changing 7-13
- 3D model, licensing 1-17
- 3D Models model set 7-10
- 3D projection, switching to 2D 5-11
- 3DS 4-6
- 3DS file 3-29

A

- A command-line option 5-27, 5-31, 5-37
- a command-line option 5-27, 5-28, 5-31, 5-32, 5-33
- abandoning, plan 3-26
- Absolute mode 12-5
- acceleration, displaying 7-3
- actuator 3-17, 3-18
- adding
 - feature layer 3-30
 - observer 10-5

- adding (continued)
 - plug-in 5-40
 - configuration 5-42
- aggregate
 - bounding volume, color 9-8
 - closing formation 3-23
 - displaying by level 9-4
 - displaying ghosted entities 9-5
 - echelon ID 3-21
 - expanding and collapsing 9-2
 - formation 3-22
 - movement 3-22
 - name length 3-13
 - organization concepts 3-21
 - reorganizing 3-22, 3-23
 - selecting 9-2
 - task behavior 3-23
- Aggregate Display Settings page 9-6, 9-8
- Aggregate Information dialog box 6-10
- agility, terrain 1-9
- altitude 7-2
 - entity, displaying 7-14
 - switching from 2D to 3D 5-11
 - vertex, displaying 6-9
- altitude point
 - changing color 14-6
 - deleting 14-5
 - inserting 14-5
- Always Join Session with Open Database 5-14
- Always Join Session with Session Database 5-14
- animation, DI-Guy 5-28
- appearance, DI-Guy 5-28
- appending, saved views to view list 11-22
- application
 - data, specifying location of 5-25
 - ID 3-5
 - number 5-27, 5-31
- application data, specifying location of 5-25
- application number 5-28, 5-32
 - default 5-33
 - specifying at startup 5-33
- Application Settings dialog box 5-6
- architecture, component 3-17
- area 3-24, 3-26
 - disaggregation 3-24
 - terrain page-in 3-24
- articulated part, attaching to 12-3
- Ask to Join Session 5-14
- assertOnBlockingTerrainCalls parameter 3-33

- assigning
 - echelon ID 3-16
 - session ID 5-7
- asynchronous I/O 5-38
- attach 12-5
 - observer, articulated part 12-3
- attach mode 1-10, 11-3, 11-7, 12-5
 - 2D behavior 12-10
 - Absolute 12-5
 - Follow 12-6
 - Mimic 12-7
 - Mimic Track 12-8
 - movement, attached context 11-7
 - prop, affect of 12-5
 - Tether 12-8
 - Track 12-9
- attached, context 11-7
- attaching observer
 - to entity or prop 12-2
 - to object, with keyboard 12-3
- attachment
 - list, filtering 12-4
 - order 12-3
 - preservation of view changes 12-3
- Attachment PanelAttachments panel, filtering
 - attachment list 12-4
- Attachment toolbarPanel 12-2
- attribute, shadowCaster 8-13
- automatic, running of scenarios 3-11
- Automatically Join Session 5-14

B

- B command-line option 5-25, 5-28
- back-end 3-3
 - dropping from exercise 3-8
 - enabling time management 5-16, 5-37
 - late joining 3-8
 - mapping objects to 3-10
 - missing 3-8
 - multiple 3-3, 3-8
 - remapping 3-8
 - sending messages 3-4
 - starting as daemon 5-28
 - synchronizing 3-38
- batch file 3-11
 - command-line option 5-28
- behavior, entity 3-19
- berkeley*, database 14-13
- B-HAVE Module for VR-Forces 1-13

- bitmap 3-30
 - location of 2-5
- BMP 3-30
- body coordinates 11-3
- border, icon 7-8
- bounding box, selecting objects 6-5
- bounding volume
 - aggregate, color 9-8
- Brains for Human Activity in Virtual Environ-
ments 1-13
- broadcast, stealth 10-7
- broadcasting 10-6
- buffer, size 5-27, 5-31

C

- C command-line option 5-25
- c command-line option 5-28
- camera
 - moving 11-12
 - ways to movemoving, options 11-11
- category, echelon 3-15
- changing
 - color
 - altitude point 14-6
 - force 7-13
 - gain 11-19
 - icon size 7-9
 - intervisibility object 13-5
 - model set 7-10
 - observer mode 10-2
- channel
 - keyword, compass 11-5
 - observer 10-5
- character, DI-Guy 5-28
- checking
 - version, plug-in 5-25
- clamping, entities 8-2
- clearing
 - object console 7-17
 - view list 11-21
- clock, modes 3-36
- closing
 - all windows 5-21
 - application 5-21
 - formation 3-23
- cloud, disabling 8-14
- cockpit display
 - attaching at startup 12-11
 - element definition 4-4

- cockpit display (continued)
 - enabling or disabling 8-12
- collapsing, aggregate 9-2
- color
 - aggregate bounding volume 9-8
 - altitude point
 - changing 14-6
 - deleting 14-5
 - inserting 14-5
 - compass 11-5
 - entity label 7-2
 - force, changing 7-13
 - intervisibility, configuring 13-6
 - terrain, configuring 14-5
- coloring, terrain 14-4
- command line
 - loading terrain 5-26
 - options 5-25
- command-line option 5-25
 - 5-25, 5-30, 5-32
 - A 5-27, 5-31, 5-37
 - a 5-27, 5-28, 5-31, 5-32, 5-33
 - app 5-32
 - appDataDir 5-25
 - appIdRange 5-28
 - appNumber 5-27, 5-28, 5-31
 - autoConnect 5-25
 - B 5-25, 5-28
 - bindObservers 5-25
 - C 5-25
 - c 5-28
 - configFilename 5-32
 - configureConnections 5-32
 - daemon 5-28
 - dataDir 5-25
 - destAddrString 5-27, 5-31
 - deviceAddress 5-27, 5-31
 - diGuyAnimationsFile 5-28
 - diGuyCharacerDataFile 5-28
 - dis 5-25
 - disPort 5-27, 5-31
 - dispSetting 5-25
 - disVersion 5-27, 5-31, 5-38
 - doNotCheckPluginVersions 5-25
 - doNotLoadVrfPlugins 5-26, 5-28
 - E 5-26
 - entDispSetting 5-26
 - entTypeMap 5-26
 - envSetting 5-26
 - execname 5-27, 5-31

- command-line option 5-25 (continued)
 - exerciseId 5-27, 5-31
 - f 5-27, 5-30, 5-36
 - federateName 5-27, 5-30
 - fedFileName 5-27, 5-30
 - fomMapperInitData 5-27, 5-30
 - fomMapperLib 5-27, 5-30
 - frontEndPID 5-28
 - fullScreen 5-26, A-5
 - fullyLoadTerrain 5-28
 - G 5-26, 5-33
 - h 5-26, 5-28, 5-32
 - help 5-26
 - hla13 5-26
 - hla1516 5-26
 - i 5-7, 5-28
 - ignore_rest 5-25, 5-30
 - L 5-26, 5-29
 - l 5-26
 - loadObservers 5-26
 - loadPlugi 5-26
 - loadPlugin 5-29, 5-34
 - locale 5-26
 - logFileName 5-26
 - mcastTtl 5-27, 5-31
 - multicastAddresses 5-27, 5-31
 - N 5-27, 5-30, 5-35
 - n 5-26, 5-29, 5-34
 - noGui 5-26
 - noPluginDirectory 5-29
 - notifyLevel 5-26, 5-29
 - observerMode 5-26
 - outputLogFile 5-29
 - P 5-27, 5-31, 5-37
 - physicalWorldFile 5-29
 - plugin 5-26
 - q 5-29
 - r 5-30
 - recvBufferSize 5-27, 5-31
 - rprFomRevision 5-27, 5-30
 - rprFomVersion 5-27, 5-30
 - S 5-37
 - s 5-30, 5-32, 5-33
 - scenarioFile 5-26
 - sendBufferSize 5-27, 5-31
 - settingsFile 5-28
 - showConsole 5-26
 - simulationModelSet 5-30
 - siteId 5-27
 - suppressSelfReflect 5-31

- command-line option 5-25 (continued)
 - T 5-30
 - targetFrameRate 5-30
 - timeManagement 5-16, 5-30, 5-37
 - unplug 5-26
 - useAdvisories 5-27, 5-30
 - useAsyncIO 5-27, 5-31, 5-38
 - useGeographicDdm 5-27, 5-31
 - useIpv6 5-31
 - userDataDir 5-26
 - v 5-27, 5-30, 5-32
 - version 5-27, 5-30, 5-32
 - vsync 5-27
 - W 5-27
 - withDisplayEngine 5-27
 - x 5-27, 5-31, 5-35
 - DIS 5-38
 - Z 5-32
 - communication, component 3-17
 - compass
 - color 11-5
 - configuring 11-5
 - location 11-5
 - model definition 11-5
 - observer 11-4
 - component
 - actuator 3-18
 - architecture 3-17
 - communication among 3-17
 - controller 3-18
 - sensor 3-18
 - composing, terrain 3-29
 - concurrent task 3-26
 - configuration
 - entity display 5-26
 - environment 5-26
 - plug-in
 - adding 5-42
 - deleting 5-42
 - configuration file
 - location of 2-5
 - RTI 2-10
 - configuring
 - compass 11-5
 - contour lines 14-9
 - display 5-25
 - federate name 5-35
 - GDB soil type 14-2
 - grid lines 14-7
 - ground clamping 8-3
 - configuring (continued)
 - intervisibility objects 13-6
 - props 3-30
 - sessions 5-14
 - shadows 8-14
 - system for MÄK RTI 2-10
 - terrain color settings 14-5
 - time management 5-15
 - configuring orientation clamping 8-3
 - connecting, to network automatically 5-25
 - console
 - messages sent to 5-34
 - object 7-16
 - constrained, time 3-38
 - constraint 12-5
 - view 11-17
 - context-sensitive menu A-6
 - contour line
 - displaying 14-8
 - spacing 14-9
 - thickness 14-9
 - Contour Line Settings page 14-8, 14-9
 - control, remote 1-11
 - control object 3-24
 - concepts 3-24
 - list of for scenario 3-10
 - name 3-24
 - selecting 6-4
 - control objects, selecting 6-5
 - controller 3-17
 - component 3-18
 - making decisions and performing tasks 3-18
 - controlling
 - messages 5-34
 - remote 11-22
 - conventions, manual xix
 - coordinate system 1-9
 - 3D 11-6
 - coordinates 11-3
 - creating
 - inset view 8-6
 - observer mode 10-2
 - CTDB file 3-29
- D**
- d command-line option, vrfSim 5-28
 - daemon, starting back-end as 5-28
 - data
 - directory 2-5

- data (continued)
 - elevation 3-29
 - port 3-17
 - synchronizing 5-21
- data directory, specifying location of 5-25
- database 3-30
 - berkeley* 14-13
 - ground-db* 14-13
 - Kabul* 14-13
 - large 3-32
 - LittlePond* 14-13
 - Makland* 14-13
 - nellis* 14-14
 - paging 3-32
 - SanLuisObispo* 14-14
 - saving as GDB 5-18
 - See also terrain database
 - terrain 1-9, 3-28
 - in scenario file 3-9
 - loading 5-17
 - supported 1-9
 - TerraSim_SampleUrban* 14-14
 - VR-TheWorldOnline 14-14
 - VR-Village* 14-14
- dead-reckoning 8-4
- decal effects 8-7
 - enabling and disabling 8-7
 - models 8-8
- decision making 3-18
- decreasing speed of navigation 11-18
- default settings, restoring 5-19
- definition, element 4-4
- Delete button 11-21
- deleting
 - altitude point 14-5
 - intervisibility object 13-8
 - messages from object console 7-17
 - observer mode 10-4
 - plug-in
 - configuration 5-42
 - XML file 5-43
 - view 11-21
- demonstration, mode A-5
- deselecting
 - entity 6-6
 - object 6-6
- designator 3-15
- destination address 5-27, 5-31
- detaching, observer from object 12-4
- detonation, graphic 8-10
- Device address 5-27, 5-31
- dialog box
 - Aggregate Information 6-10
 - Application Settings 5-6
 - Display Settings 5-19, 7-5, 7-9, 7-13, 9-6, 9-8
 - Entity Display Settings page 7-4, 7-14, 8-2, 8-3, 10-7
 - Observer Mode Settings page 8-9, 8-11, 13-4, 13-5, 13-6, 14-6
 - Render Settings page 8-13, 8-14
 - Tactical Graphics Display Settings page 6-8, 6-9, 15-2
- Entity Information 6-10, 7-15
- Environmental Object Information 6-10
- License Setup 2-7
- Session Startup 5-5, 5-6
- Stealth Information 6-10
- DI-Guy 1-14
 - character data file 5-28
 - configuring animations and appearances 5-28
 - license 1-16
- directory
 - data 2-5
 - installed 2-5
- DIS 1-18
 - aggregate name length 3-13
 - command-line options 5-37
 - enumerations document 4-7
 - exercise ID 5-27, 5-31
 - IPV6 5-31
 - port 5-27, 5-31
 - port number 5-37
 - self reflection 5-31
 - specifying broadcast modes 5-37
 - starting in 5-25
 - startup tutorial 5-7
- DIS Enumeration Document 3-11
- disabling
 - cockpit displays 8-12
 - decal effects 8-7
 - entity labels 7-5
 - environment 8-14
 - sensor volumes 8-11
 - shadows 8-13
 - speed scaling 11-18
 - track histories for all entities 8-9
 - trailing effects 8-7
 - wireframe mode 14-9
- disaggregation area 3-24

- display
 - dragging 11-14
 - loading configuration 5-25
 - navigating 11-3
 - display engine 4-2
 - Display Settings
 - dialog box, Observer Mode Settings page 8-9, 8-11, 13-4, 13-5, 13-6, 14-6
 - toolbar 7-7
 - Display Settings dialog box 5-19, 7-5, 7-9, 7-13, 9-6, 9-8
 - Entity Display Settings page 7-4, 7-14, 8-2, 8-3, 10-7
 - Render Settings page 8-13, 8-14
 - Tactical Graphics Display Settings page 6-8, 6-9, 15-2
 - displaying
 - contour lines 14-8
 - entity information 7-2
 - full screen A-5
 - grid line 14-6
 - height-above-terrain lines 6-9, 7-14
 - intervisibility object 13-5
 - label
 - 2D 7-6
 - entity 7-3
 - levels of aggregation in Echelon View 9-4
 - objects 6-7, 6-8
 - other MAK 3D GUIviewers 10-6
 - panel A-3
 - resources 7-15
 - sensor volumes 8-11
 - stealth models 10-7
 - toolbars A-3
 - track histories 8-9
 - trailing effects 8-7
 - transient intervisibility 13-4
 - vertex labels 6-9
 - view 11-21
 - Distributed Interactive Simulation 1-18
 - DLL
 - plug-in, removing 5-41
 - specifying for plug-in 5-41
 - DLLs 5-39
 - dockable panel A-2
 - docking, panel A-2
 - documentation, location of 2-5
 - dragging
 - terrain 11-13
 - view 11-14
 - dropping from exercise 3-8
 - DTED file 3-29
 - dust cloud 8-7
 - dynamic linked library 5-39
- ## E
- E command-line option 5-26
 - echelon ID 3-13, 3-14, 3-15
 - aggregate member 3-21
 - assignment of 3-16
 - Echelon View 6-2, 9-4
 - expanding and collapsing aggregates 9-4
 - ghosted aggregates 9-5
 - levels of aggregation 9-4
 - editing
 - intervisibility object 13-5, 13-7
 - observer mode 10-4
 - effects
 - model 4-6
 - trailing 8-7
 - electromagnetic emissions 8-11
 - element definition 4-4
 - common icons A-6
 - elevation
 - coloring terrain by 14-4
 - data 3-29
 - intervisibility point, configuring 13-6
 - Embarkation View 6-2
 - embedded window 4-2
 - emission, electromagnetic 8-11
 - emitter system 8-11
 - enabling
 - cockpit displays 8-12
 - decal effects 8-7
 - entity labels 7-5
 - sensor volumes 8-11
 - shadows 8-13
 - speed scaling 11-18
 - time management 5-15
 - in back-end 5-16, 5-37
 - track histories for all entities 8-9
 - trailing effects 8-7
 - wireframe mode 14-9
 - entity
 - altitude, displaying 7-14
 - attaching observer to 12-2, 12-5
 - behaviors 3-19
 - body frame coordinate system 11-6
 - clamping to ground 8-2

- entity (continued)
 - concepts 3-12
 - console message 7-16
 - display settings 5-26
 - element definition 4-4
 - filtering, attachment list 12-4
 - ghosted 9-5
 - ground clamping 8-3
 - icon, changing size 7-9
 - identifiers 3-13
 - inset view 8-6
 - intervisibility 13-3
 - label 3-14
 - 2D 7-3, 7-6
 - 3D 7-2
 - color 7-2
 - enabling and disabling 7-5
 - pinning to display 7-5
 - level body frame coordinate system 11-6
 - local and remote 3-12
 - model 4-6
 - name 7-2
 - length restrictions 3-13
 - notification level, object console 7-16
 - organization 3-15
 - pinning intervisibility object to 13-8
 - primary 12-8, 12-10
 - resource monitoring 7-15
 - scaling, 3D model 7-10
 - secondary 12-8, 12-10
 - selecting 6-4, 6-5
 - in Objects List Panel 6-6
 - sending console messages to log file 7-17
 - shadows 8-13
 - state 3-18
 - storing list of 3-10
 - task 3-19
 - track history 8-9
 - trailing effects 8-7
 - type 4-7
 - enumeration 4-4, 7-2
 - unselecting 6-6
 - views of 6-2
 - VR-Forces 3-12
 - Entity Display Settings page 7-4, 7-5, 7-9, 7-13, 7-14, 8-2, 8-3, 10-7
 - Entity Editor 1-14
 - Entity Information dialog box 6-10, 7-15
 - Entity List, filtering 6-6
 - entity type mapping, loading 5-26
 - Entity View 6-2
 - enumeration, document 4-7
 - Enumeration and Bit Encoded Values for Use with Protocols for DIS Applications 3-11
 - environment
 - disabling 8-14
 - loading configuration 5-26
 - environment variable
 - MAKLMGRD_LICENSE_FILE 2-8
 - RTI_CONFIG 2-10
 - Environmental Object Information dialog box 6-10
 - Environmentals List, filtering 6-6
 - Environmentals View 6-2
 - event 3-18
 - exaggerating, 3D models 7-10
 - executable, installation locations 2-5
 - execution name, HLA, specifying 5-35
 - exercise
 - clock 3-35
 - time 3-9, 3-35
 - exercise ID, DIS 5-27, 5-31
 - exiting
 - application 5-21
 - full screen window A-5
 - expanded label 7-2
 - pinning to display 7-5
 - expanding, aggregate 9-2
 - explosion 8-10
 - exporting
 - settings 5-19
 - terrain to GDB 5-18
 - external communications effects server, configuring VR-Forces for 1-13
 - extracting, prop from terrain 3-30
 - eyepoint. *See* observer
- F**
- f command-line option 5-27, 5-30, 5-36
 - factory settings, restoring 5-19
 - fan, intervisibility 13-3
 - FDD file 2-10
 - feature 1-9
 - feature layer, adding 3-30
 - FED file 2-10
 - name 5-27, 5-30
 - specifying 5-36
 - federate

federate (continued)
 name 5-27, 5-30
 specifying 5-35
 time managed 3-38
 federation execution, specifying name 5-35
 federation time 3-38
 file
 batch 3-11
 FED 2-10, 5-36
 hostility relationships 3-10
 installed 2-5
 log 5-44
 object map 3-10
 order of battle 3-10
 plan 3-10
 RID 2-10, 5-35
 rid.mtl 5-35
 scenario 3-9
 vrfgui.log 5-44
 vrfsim.log 5-44
 filtering
 object attachment list 12-4
 object lists 6-6
 fire, line 8-10
 fixed-frame 5-15
 Fixed-Frame Best-Effort 3-36
 Fixed-Frame Run-To-Complete 3-36
 fixed-frame-run-to-complete 5-15
 flat earth coordinate system 11-6
 FLEXlm 2-6
 floating, panel A-2
 FLT file 3-29, 4-6
 mapping to models 4-7
 Follow mode 11-7, 12-5, 12-6
 FOM 2-9
 FOM Mapper
 initialization data 5-27, 5-30
 library name 5-27, 5-30
 specifying at startup 5-35
 footprint 8-7
 force 3-15, 7-2
 formation 3-22
 closing 3-23
 frame advance mode 3-9
 frame rate 5-30
 exercise clock 3-36
 repeatability 3-37
 frame time, interval 3-9
 frame-mode parameter 5-16
 time management 5-15

free-fly mode 12-5
 front-end 3-3
 coordinating multiple 3-7
 joining session at startup 5-14
 multiple 3-3
 process ID 5-28
 sending messages 3-4
 full screen modewindow, specifying 5-26
 full screen window 4-2, A-5

G

-G command-line option 5-26, 5-33
 gain, changing 11-19
 GDB 3-31
 saving terrain as 5-18
 GDB terrain 3-29
 configuring soil type 14-2
 shapefile 3-30
 GEO TIFF 3-30
 geocentric coordinate system 11-6
 ghosted aggregate 9-5
 GIF 3-30
 GL Studio 1-14, 8-12
 license 1-15
 global setting 5-21
 graphical object 3-24
 showing and hiding 6-7, 6-8
 graphical user interface 3-3
 graphics, fire and detonation 8-10
 graphics mode, wireframe 14-9
 grid line
 configuring 14-7
 displaying 14-6
 ground clamping 8-2
 configuring 8-3
 Ground Clamping Cutoff Distance Scale 8-3
ground-db, database 14-13
 GUI 3-3
 localizing 2-11

H

-h command-line option 5-26, 5-28, 5-32
 HarrisAfghanVillage, terrain 14-13
 HAT. See height-above-terrain
 heading 7-2, 11-3
 indicator 7-3
 observer 11-4
 height, above terrain 11-17

- height-above-terrain line 6-7, 7-14
 - displaying 6-9
- help 5-28, 5-32
- hiding
 - objects 6-7, 6-8
 - panel A-3
 - symbol decorations 7-7
 - toolbars A-3
 - vertex labels 6-9
- hierarchy
 - displaying levels of aggregation 9-4
 - entity 3-15
- High Level Architecture 1-18
- HLA 1-18
 - aggregate name length 3-13
 - command line options 5-35
 - federation execution name, specifying 5-35
 - specifying federation execution 5-35
 - starting in 5-26
 - startup tutorial 5-7
 - time management 3-38
- hostility relationships file 3-10
- hostname, license server 2-7
- hst* file 3-10

I

- i command-line option 5-7, 5-28
- icon
 - 2D, displaying label 7-6
 - borders 7-8
 - common for model definitions A-6
 - level of nesting 9-5
 - location of 2-5
 - settings 5-19
 - size, changing 7-9
 - toolbars 5-9
- ID
 - echelon 3-14, 3-15
 - object 3-14
 - site 5-30, 5-32
- identifier, site and application 3-5
- identifying, entities 3-13
- image
 - format, supported 3-30
- importing, settings 5-19
- increasing speed of navigation 11-18
- independent mode, tutorial 5-7
- independent setting, global and observer 5-21
- independent task 3-26

- indicator 7-5
 - heading 7-3
- information, entity label 7-2
- inserting, altitude point 14-5
- inset view 8-6
- installation 2-2
- installing, RTI 2-10
- interaction 3-38
- intermediate contour line 14-8
- interval, contour line 14-9
- intervisibility 3-34
 - entity 13-2, 13-3
 - fan 13-2, 13-3
 - moving 13-6
- grid 13-2
- object
 - configuring 13-6
 - deleting 13-8
 - displaying 13-5
 - editing 13-5, 13-7
 - transient and permanent 13-4
- pinning object to entity 13-8
- point-to-point 13-2
- types 13-2

Intervisibility Settings page 13-4, 13-5, 13-6

IPv6

- DIS 5-31

J

- joining exercise, late 3-8
- JPEG 3-30
- JPG 3-30
- jumping, observer 11-14

K

- Kabul*, database 14-13
- key mapping 11-10
- keyboard
 - attaching to objects from 12-3
 - changing speed 11-19
 - control A-5
 - inset views 8-6
 - mappings 11-5, 11-9
 - movement functions 11-5
 - moving observer 11-9
- keyword
 - largeCompass 11-5
 - smallCompass 11-5

Kynapse 1-13

L

-L command-line option 5-26, 5-29

-l command-line option 5-26

label 3-13

2D

displaying 7-6

showing and hiding 7-7

entity 3-14

2D 7-3

3D 7-2

enabling and disabling 7-5

pinning to display 7-5

Lambert Conformal Conic coordinate system 11-6

language

specifying 5-26

specifying for GUI 5-33

largeCompass, keyword 11-5

late joiner 3-8

latitude 7-2

layer

feature, adding 3-30

length

entity name 3-13

intervisibility object 13-5

level, notification for messages 5-34

level observer 11-6

library, dynamic 5-39

license

3D models 1-17

DI-Guy 1-16

GL Studio 1-15

OpenSceneGraph 1-17

osgEarth 1-17

SilverLining 1-15

SpeedTree 1-16

License Manager 2-6

license server, hostname 2-7

License Setup dialog box 2-7

lighting, disabling 8-14

line

fire 8-10

height-above-terrain 6-7, 7-14

line-of-sight 3-34

Linux, installing on 2-3

LittlePond terrain database 14-13

loading

display configuration 5-25

loading (continued)

entity display settings 5-26

entity type mapping 5-26

environment configuration 5-26

plug-in 5-26, 5-34

plug-ins 5-26, 5-28, 5-39

preventing 5-26

saved views 11-22

scenario 5-26, 5-29

settings 5-19

terrain

from command line 5-26

paged 5-28

terrain database 5-17

local

entity 3-12

object 3-12

locale, specifying 5-26

localization, specifying 5-26

localizing, GUI 2-11

location

compass 11-5

jumping to 11-14

log file 5-29, 5-44

messages sent to 5-34

sending entity messages 7-17

specifying 5-26

vrfSim 5-29

longitude 7-2

LWO file 3-29

M

MÄK RTI, configuring 2-10

MÄK Terrain Database Tool 3-31

MÄK Terrain Format 3-31

making decisions 3-18

Makland, database 14-13

MAKLMGRD_LICENSE_FILE, environment

variable 2-8

managing, plug-ins 5-39

manual

conventions xix

location of 2-5

mapping

keyboard 11-10

model 4-4

to entity type 4-7

to remote observer 10-6

movement to keyboard 11-9

- marking text 7-2
 - MEDF file 3-29
 - menu
 - context-sensitive A-6
 - Settings 5-19
 - tearing off A-5
 - message
 - console 5-29
 - entity 7-16
 - notification level 5-26
 - sending between front-end and back-end 3-4
 - sending entity to log file 7-17
 - sent 3-18
 - session, configuring 5-14
 - setting levels 5-34
 - state update 10-6
 - time stamp ordered, TSO 3-38
 - view control 11-22
 - messages 5-29
 - middle-click menu A-6
 - MIL-STD 2525B colors 7-13
 - MIL-STD color, aggregate 9-8
 - Mimic mode 11-7, 12-5, 12-7
 - Mimic Track mode 12-8
 - mode
 - Absolute 12-5
 - batch 3-11
 - Follow 12-6
 - Mimic 12-7
 - Mimic Track 12-8
 - observer 4-3
 - setting 5-26
 - specifying 5-37
 - Tether 12-8
 - Tether Track 12-10
 - Track 12-9
 - view 11-7, 12-5
 - model 4-6
 - 3D, scaling 7-12
 - 3D entity, exaggerating 7-10
 - decal effects 8-8
 - mapping 4-4
 - common icons A-6
 - to FLT files 4-7
 - to remote observer 10-6
 - name 7-2
 - physical 3-18
 - stealth 10-7
 - trailing effects 8-8
 - types 4-6
 - model 4-6 (continued)
 - vehicle 1-4
 - model definition 4-4, 4-7
 - compass 11-5
 - model set 7-10
 - changing 7-10
 - model sets 7-10
 - modeling, 3D objects 4-4
 - monitoring, entity resources 7-15
 - mouse
 - changing speed 11-19
 - drag terrain 11-13
 - drag view 11-14
 - moving observer 11-13, 11-14, 11-15
 - orbiting terrain 11-14, 11-16
 - teleporting observer 11-14
 - movement
 - aggregate 3-22
 - speed scaling 11-18
 - moving
 - camera 11-12
 - intervisibility fan 13-6
 - observer 11-3, 11-11
 - mouse with 11-13, 11-14, 11-15
 - MTD file 3-29
 - MTL file 3-29
 - MTP file 3-29
 - multicast mode, specifying 5-37
 - multicast time-to-live 5-37
 - multiple
 - back-ends 3-3, 3-8
 - front-ends 3-3
 - coordinating 3-7
 - VR-Forces processes 3-3
 - munition, display of path and detonation 8-10
 - mutually exclusive task 3-26
- ## N
- N command-line option 5-27, 5-30, 5-35
 - n command-line option 5-26, 5-29, 5-34
 - name 3-13
 - control object 3-24
 - displaying 7-3
 - entity 3-13, 7-2
 - navigating, view 11-3
 - navigation
 - attached context 11-7
 - options to 11-11
 - speed scaling 11-18

nellis, database 14-14
 network
 connecting to, automatically 5-25
 new, window 5-11
 notification level 5-29
 setting 5-34
 setting for entity console 7-16
 specifying 5-26
 notify level, entity object console 7-16

O

OBJ 4-6
 OBJ file 3-29
 object
 attaching observer to 12-2
 concepts 3-11
 control 3-24
 detaching observer from 12-4
 exaggerating models 7-10
 filtering list of 6-6
 ID 3-13, 3-14
 intervisibility 13-4
 editing 13-7
 pinning to entity 13-8
 list of 3-10
 local 3-12
 model 4-6
 modeling 4-4
 orbiting 11-12
 overlay 3-24
 remote 3-12
 selecting 6-4
 in Objects List Panel 6-6
 showing and hiding 6-8
 unselecting 6-6
 views of 6-2
 VR-Forces 3-12
 object console 7-16
 clearing 7-17
 setting notify level 7-16
 object map file 3-10
 missing back-ends 3-8
 object parameter database 3-11
objectConsoleNotifyLevel parameter 7-16
 objects list, using 12-2
 Objects List Panel 6-2, 6-4, 6-6, 9-4
 selecting object in 6-6
 observer 11-3
 adding 10-5
 observer 11-3 (continued)
 attaching
 articulated part 12-3
 to entity at startup 12-11
 attaching to, entity or prop 12-2
 attachment
 in 2D 12-10
 order 12-3
 camera 11-12
 compass 11-4
 coordinate system
 2D 11-6
 3D 11-6
 detaching from object 12-4
 dragging
 terrain 11-13
 view 11-14
 heading 11-4
 inset view, in 8-6
 mode
 setting 5-21
 setting at startup 5-26
 switching projection 5-11
 movement 12-5
 speed 11-19
 movement functions 11-5
 moving
 keyboard with 11-9
 mouse with 11-13, 11-15
 orbiting 11-12
 orientation 11-14
 saving view 11-20
 teleporting 11-14
 view, attachment change 12-3
 ways to movemoving, options 11-11
 Observer Control Panel 11-17, 11-22
 observer mode 4-3
 changing 10-2
 creating 10-2
 deleting 10-4
 editing 10-4
 renaming 10-4
 Observer Mode Settings page 8-9, 8-11, 14-6
 Observer Modes Editor page 7-12, 10-2, 10-4
 Observer Settings toolbar 7-10, 7-12
 Observer Views Panel 11-20
 loading saved views 11-22
 saving views 11-21
 obstacle 3-24, 3-26
 OPD Editor 1-14

- OpenFlight 4-6
 - format 1-4
- OpenFlight file 3-29
- opening, inset view 8-6
- OpenSceneGraph 1-3, 1-14
 - license 1-17
- OpenStreetMapServer, terrain 14-14
- option
 - command-line 5-25
 - navigation 11-11
- orbit, observer 11-12
- orbiting
 - terrain, with mouse 11-14, 11-16
- order of battle 3-10
 - file 3-10
- organization
 - aggregate 3-21
 - entity of 3-15
- orientation 11-3
 - clamping 8-3
 - displaying 7-3
 - observer 11-13, 11-14, 11-15, 12-5
- OSG file 3-29
- osgEarth 1-14
 - license 1-17
- Out-the-Window observer mode 5-21
- overlay, concepts 3-24
- overlay object 3-24
 - selecting 6-4
- Overlays view 6-2

P

- P command-line option 5-27, 5-31, 5-37
- page
 - Resource Information 7-15
 - Session Options 5-6
- paged terrain, loading 5-28
- paging, terrain 3-32
- panel
 - displaying and hiding A-3
 - dockable A-2
 - docking A-2
 - floating A-2
 - undocking A-2
- parameter
 - `assertOnBlockingTerrainCalls` 3-33
 - `frame-mode` 5-16
 - time management 5-15
 - `objectConsoleNotifyLevel` 7-16

- parameter (continued)
 - providing information to components 3-17
 - `RTI_timeMgmt` 5-15
 - `RTI_useRtiExec` 5-15
 - `runInTimeManagementMode` 5-16
 - `tick-period-uses-real-time` 3-37
- patch, terrain 3-29
- path, specifying for RTI 2-10
- pause status 3-9
- PDU
 - sending 10-6
 - view control 11-22
- performance
 - asynchronous I/O 5-38
 - frame rate 5-30
 - HAT lines 6-9, 7-14
 - shadows 8-14
- permanent intervisibility object 13-4
- phase line 3-24, 3-25
- physical model 3-18
- pitch 11-3
- plan
 - abandoning 3-26
 - concepts 3-27
 - file 3-10
- Plan View window 5-11
- plug-in
 - adding 5-40
 - configuration
 - adding 5-42
 - deleting 5-42
 - DLL
 - removing 5-41
 - specifying 5-41
 - list, viewing 5-43
 - loading 5-26, 5-28, 5-39
 - managing 5-39
 - prevent loading 5-26
 - version, checking 5-25
 - XML file 5-40
 - deleting 5-43
- plug-in loading 5-34
- Plugins Editor 5-39, 5-40, 5-41, 5-42, 5-43
- PNG 3-30
- point 3-25
 - orbiting around 11-12
- point-to-point communications, specifying 5-37
- point-to-point intervisibility 13-2
- polygon 3-26, 3-32
 - not going through 11-17

popup menu A-6
 port 3-17
 default 5-37
 group 3-17
 number, specifying at startup 5-37
 specifying, at startup 5-37
 position 11-3
 observer 11-13, 11-15, 12-5
 Presagis 1-17
 primary attached entity 12-8, 12-10
 primary color, aggregate 9-8
 process
 object mappings to 3-10
 VR-Forces 3-3
 process ID, front-end 5-28
 product
 technical support xviii
 upgrades xviii
 projected terrain database 11-6
 projection, switching 2D to 3D 5-11
 projections, supported 1-9
 prop
 attaching observer to 12-2
 attaching to 12-5
 configuring 3-30
 extracting, from terrain 3-30
 filtering attachment list 12-4
 properties
 intervisibility object, editing 13-7

Q

-q command-line option 5-29
 Qt Linguist 2-11

R

-r command-line option 5-30
 random number seed 3-9
 range, ground clamping 8-3
 ray, intervisibility 13-3
 Real DB 1-17
 real time, running faster or slower than 3-37
 recalling
 view 11-21
 view state 11-20
 receive buffer size 5-27, 5-31
 regulating, time 3-38
 rejoining exercise 3-8
 remapping, back-ends 3-8

remote
 control, of the observer 11-22
 entity 3-12
 object 3-12
 stealth, displaying model 10-7
 remote control 1-11
 removing, view 11-21
 renaming, observer mode 10-4
 Render Settings page 8-13, 8-14, 14-9
 reorganizing
 aggregate, versus closing formation 3-23
 concepts for aggregates 3-22
 repeatability 3-37
 replacing, saved view list 11-22
 replicating, settings and data 5-21
 resizable window 4-2
 resource, monitoring entity 7-15
 Resource Information, page 7-15
 restoring, view 11-21
 revision, RPR FOM 5-27, 5-30
 revolving, observer 11-12
 RGB 3-30
 RID file 2-10, 5-35
rid.mtl file 2-10, 5-35
 right-click menu A-6
 roll 11-3
 rotation
 path, smoothing 8-4
 route 3-24, 3-26
 altitude, displaying 6-9
 RPR FOM 1-18, 4-7, 5-35
 revision 5-27, 5-30
 version 5-27, 5-30
 RTI 2-9
 1.3 and 1516 1-18
 configuring for time management 5-15
 definition of 2-9
 RTI_CONFIG environment variable 2-10
 RTI_timeMgmt parameter 5-15
 RTI_useRtiExec parameter 5-15
 rtiexec 5-15
 run mode, starting back-end in 5-30
 runInTimeManagementMode parameter 5-16

S

-S command-line option 5-37
 -s command-line option 5-30, 5-32, 5-33
 sampling rate
 intervisibility, configuring 13-6

SanLuisObispo terrain database 14-14

saving

- scenario 3-8

- settings 5-19, 5-21

- terrain as GDB 5-18

- view 11-20, 11-21

 - to file 11-21

Scale Factor slider 7-12

Scaled Speed check box 11-18

scaling

- 3D models 7-10

- model, 3D 7-12

scenario 3-9

- file 3-8, 3-9

- loading 5-26, 5-29

 - at startup 5-29

- order of battle 3-10

- saving 3-8

Scenario Merge tool 1-14

schema 4-4

SDK, uninstalling 2-4

secondary, window 5-11

secondary attached entity 12-8, 12-10

selecting

- aggregates 9-2

- entities 6-5

- objects 6-4

 - in Objects List Panel 6-6

- observer 10-5

selection box 6-5

Selection Group View 6-2

self reflection, DIS 5-31

send buffer size 5-27, 5-31

sending, messages 3-18

sensor 3-17, 3-18

sensor volume, displaying 8-11

server, terrain 3-32, 3-33

session 3-6

- ID

 - assigning 5-7

 - setting at startup 5-28

- joining at startup 5-14

- message, configuring 5-14

Session Options, page 5-6

Session Startup dialog box 5-5, 5-6

set data request 3-28

set statement 3-28

- See also. set data request

setting

- color, aggregate bounding volume 9-8

setting (continued)

- mode, observer 5-26

- notification level 5-34

 - entity 7-16

settings

- factory and default 5-19

- icons 5-19

- loading 5-19

- saving 5-19

- synchronizing 5-21

- user 5-21

Settings menu 5-19

shadow

- configuring 8-14

- enabling and disabling 8-13

shadowCaster attribute 8-13

shape terrain data 3-29

Show Database Correlation Warning Dialogs 5-14

Show Session Terrain Change Prompts 5-14

showing

- objects 6-7, 6-8

- symbol decoration 7-7

- vertex labels 6-9

SHP file 3-29

SilverLining 1-14

- license 1-15

SimthetiQ 1-17

simulated world, navigating 11-3

simulation

- repeating 3-37

- running, automatically 3-11

- time 3-9, 3-35

Simulation Control toolbar 5-9

simulation engine 3-3

simulation model set, specifying on command line
5-30

Simulation Selection toolbar 5-9

simulation time 3-38

Simulation Time Scale, toolbar 5-9

Simulation Time toolbar 5-9

site ID 3-5, 5-27, 5-30, 5-32

- specifying 5-30

 - at startup 5-33

smallCompass, keyword 11-5

smoke, trail 8-7

smooth period 8-4

smoothing 8-4

SO 5-39

soil type, configuring for GDB 14-2

spacing, contour line 14-9

- specifying
 - application number 5-28, 5-32
 - FED file 5-36
 - federate name 5-35
 - HLA federation execution 5-35
 - language 5-26
 - log file 5-26
 - notification level 5-26
 - point-to-point and multicast modes 5-37
 - port, at startup 5-37
 - simulation model set 5-30
 - site ID 5-30
- speed 7-2
 - gain 11-19
 - scaling 11-18
 - enabling and disabling 11-18
- SpeedTree 1-14
 - license 1-16
- startup
 - attaching observer 12-11
 - option 5-25
 - tutorial 5-7
- state updates, sending 10-6
- status, entity 7-15
- Stealth, window 5-11
- Stealth Control Toolkit 11-22
- Stealth Information dialog box 6-10
- Stealth observer mode 5-21
- streaming, terrain 3-33
- support, technical xviii
- surface entity, wake 8-7
- surface, clamping to 8-2
- symbol, location of 2-5
- symbol decorations, showing and hiding 7-7
- synchronizing
 - back-ends 3-38
 - data and settings 5-21
- T**
 - T command-line option 5-30
 - tactical graphics 3-24
 - element definition 4-4
 - HAT lines 6-9
 - selecting 6-5
 - showing and hiding 6-7, 6-8
 - Tactical Graphics Display Settings page 6-8, 6-9, 15-2
 - task
 - aggregate concepts 3-23
 - task (continued)
 - concepts 3-26
 - concurrent 3-26
 - entity 3-19
 - independent 3-26
 - mutually exclusive 3-26
 - performing 3-18
 - TDB Tool 1-14, 3-31
 - tearing off menu A-5
 - technical support xviii
 - teleporting observer 11-14
 - terminating, application 5-21
 - terrain
 - agility 1-9
 - agility and composability 3-29
 - altitude point
 - changing color 14-6
 - deleting 14-5
 - inserting 14-5
 - coloring by elevation 14-4
 - database 1-9, 3-28
 - in scenario file 3-9
 - dragging with mouse 11-13
 - extracting props from 3-30
 - formats, supported 3-29
 - GDB, soil type 14-2
 - HarrisAfghanVillage 14-13
 - large 3-32
 - loading from command line 5-26
 - OpenStreetMapServer 14-14
 - orbiting, with mouse 11-14, 11-16
 - paged, loading 5-28
 - paging 3-32
 - patch 3-29
 - server 3-33
 - shape 3-29
 - streaming 3-33
 - wireframe mode 14-9
 - terrain database 3-30
 - loading 5-17
 - loading at startup 5-30
 - saving as GDB 5-18
 - terrain page-in area 3-24
 - terrain server 3-32
 - Terrapage file 3-29
 - TerraSim 1-17
 - TerraSim_SampleUrban*, database 14-14
 - Terrex 1-17
 - Tether mode 11-7, 12-5, 12-8
 - Tether Track mode 12-10

- thickness, contour line 14-9
- tick-period-uses-real-time parameter 3-37
- TIFF 3-30
- time
 - multiplier 3-9
 - simulation and exercise 3-35
- time constrained 3-38
- time management
 - concepts 3-38
 - configuring 5-15
 - enabling 5-15
 - for back-end 5-16, 5-37
 - RTI requirements 5-15
- time regulating 3-38
- time stamp ordered message 3-38
- tire track 8-7
- toolbar
 - Display Settings 7-7
 - displaying and hiding A-3
 - icon 5-9
 - Objects 6-6, 9-4
 - Observer Settings 7-10, 7-12
- toolbarpanel, Attachment 12-2
- track history
 - displaying 8-9
 - enabling and disabling for all entities 8-9
- Track mode 11-7, 12-5, 12-9
- tracking
 - entity, resources 7-15
- trail, model 8-8
- trailing effects 8-7
 - enabling and disabling 8-7
 - model 8-8
- trajectory history. See track history
- trajectory smoothing 8-4
- transient intervisibility
 - displaying 13-4
 - object 13-4
- translating, GUI text 2-11
- tread track 8-7
- triangle, vertex label 6-9
- turret, attaching to 12-3
- tutorial, starting VR-Forces 5-7
- TXP file 3-29

U

- unconstrained 3-38
- undocking, panel A-2
- uninstalling VR-Forces 2-4

- Universal Transverse Mercator 3-30
- unselecting
 - entity 6-6
 - object 6-6
- update, messages 10-6
- update rate
 - intervisibility, configuring 13-6
- upgrades xviii
- user, settings 5-21
- user data, specifying directory 5-26
- UTM 3-30
- UTM coordinate system 11-6

V

- v command-line option 5-27, 5-30, 5-32
- vapor, trail 8-7
- Variable-Frame Run-To-Complete 3-36
- vehicle, model 1-4
- velocity, displaying 7-3
- version
 - displaying 5-27
 - plug-in, checking 5-25
 - RPR FOM 5-27, 5-30
- vertex
 - altitude, displaying 6-9
 - showing and hiding 6-9
- view
 - clearing list 11-21
 - deleting 11-21
 - dragging with mouse 11-14
 - inset 8-6
 - loading saved 11-22
 - navigating 11-3
 - recalling 11-20
 - restoring 11-21
 - saving 11-20, 11-21
- view constraint 11-17
- view control message 11-22
 - enabling processing of message 11-22
 - script 11-22
- view floor 11-17
- view mode, Tether Track 12-10
- viewing
 - other MAK 3D GUIviewers 10-6
 - plug-in, data 5-43
- visible surface, configuring 14-2
- Visible Surfaces page 14-2
- vrfGui* 3-3
- vrfGui.log* file 5-44

- vrfMsgDump 1-14
- VR-Forces
 - entity or object 3-12
 - uninstalling 2-4
- VR-Forces window 5-9, 5-17
- vrfSim, log file 5-29
- vrfSim* 3-3
 - daemon, starting as 5-28
- vrfSim.log* file 5-44
- VR-TheWorldOnline, database 14-14
- VR-Vantage Control Toolkit 11-22
- VR-Village*, database 14-14
- vsynch 5-27

W

- W command-line option 5-27
- wake 8-7
- wall, not walking through 11-17
- wall-clock time 5-15
- waypoint 3-24, 3-25
 - altitude, displaying 6-9
- window
 - new 5-11
 - observer 10-5
 - secondary 5-11
 - selecting entities in 6-5
 - types 4-2
- Windows, installing on 2-2
- wireframe mode 14-9
- world coordinates 11-3

X-Y-Z

- x command-line option 5-27, 5-31, 5-35, 5-38
- XML, RTI configuration file 2-10
- XML file
 - plug-in 5-40
 - deleting 5-43
- yaw 11-3
- Z command-line option 5-32
- zoom, switching from 3D to 2D 5-11

VR-Forces Quick Reference Card

vrfSim Command-Line Options

Options	Description	Options	Description
-a <i>app_number</i>	Application number.	-n <i>notify_level</i>	The level for application messages.
-A <i>dest_addr</i>	Address for point-to-point mode. (DIS)	--noPluginDirectory	Do not load MTL files in <i>./plugins</i> .
--appIdRange <i>low-high</i>	Range of application IDs for VR-Forces objects. (DIS).	--outputLogFile <i>file</i>	The name of the log file for messages.
-B <i>file</i>	Runs the specified batch file.	-P <i>port</i>	The port number for DIS.
-c <i>PID</i>	The process ID for the front-end.	--physicalWorldFile <i>file</i>	Physical world file to load.
-d <i>file</i>	An alternate to <i>vrfSimSettings.mtl</i> .	-q	Do not display a console.
--daemon	Runs vrfSim as a daemon. (Linux)	-r	Start the back-end in run mode (if -L and -T are also specified).
--deviceAddress <i>address</i>	Device address to connect to network if there are multiple ethernet cards. (DIS)	--recvBufferSize <i>size</i>	The receive buffer size. (DIS)
--diGuyAnimations-File <i>file_name</i>	Additional DI-Guy animations file.	--rprFomRevision <i>revision</i>	The RPR FOM 2, draft 17 revision to use. (HLA)
--diGuyCharacer-Data-File <i>file_name</i>	Additional DI-Guy character data file.	--rprFomVersion <i>version</i>	The version of the RPR FOM to use. (HLA)
--disVersion	The DIS version to use.	-S <i>multicast_addr</i>	The address for multicast mode. (DIS)
--doNotLoadVrfPlugins	Do not load plug-ins.	-s <i>site_ID</i>	The site ID.
-f <i>FOM_mapper</i>	The FOM Mapper to use. (HLA)	--simulationModelSet	Default SMS for the back-end.
-F <i>FED_file</i>	The FED file to use. (HLA)	--sendBufferSize <i>size</i>	The send buffer size. (DIS)
--fomMapperInitData <i>data</i>	Initialization data for a FOM Mapper, if required.	--suppressSelfReflect	Represses self reflection of DIS exercise connection.
--fullyLoadTerrain	Load all pages of paged terrain.	-T <i>terrain</i>	The terrain database to load. Mutually exclusive with -L.
-h	Displays help information in a console window.	--targetFrameRate <i>rate</i>	Specifies frame rate above which vrfSim yields CPU to other processes.
-I	Use asynchronous I/O. (DIS)	--timeManagement	Enables time management.
-i <i>session_ID</i>	The session ID.	--useAdvisories	Enable HLA advisory messages.
--ignore_rest	Ignore any options after this one.	--useGeographicDdm	Use geographic DDM. Supported only for MÄK RTI.
--ignoreAdvisories	Ignore HLA advisory messages.	--useAsyncIO	Use asynchronous I/O. (DIS)
--loadPlugin <i>file</i>	Plug-in file to load.	--useIpv6	Exercise connection uses Ipv6. (DIS)
--logFrameRateStatistics	Enable logging of frame rate statistics.	-v <i>version</i>	Specifies version information.
-L <i>scenario</i>	Load the specified scenario.	-x <i>execution_name</i>	The HLA federation execution name.
--mcastTtl	The multicast time-to-live. (DIS)	-x <i>exercise_ID</i>	The DIS exercise ID.

vrfSim Command-Line Options

Options	Description	Options	Description
-N <i>federate_name</i>	The federate name to use. (HLA)		

vrfGui Command-Line Options

Option	Description
(-- --ignore_rest)	Ignore all command-line options after this one.
--appDataDir <i>directory</i>	Specifies the location of application data.
-B --bindObservers <i>observer1</i> , <i>observern</i> , <i>saved_views_file</i>	Assigns the views in the specified saved view file to the observers in the order that the observers are listed in the command.
(-C --autoConnect) { <i>filename</i> <i>connector_name</i> }	Automatically connect to the network using the specified connector.
--dataDir <i>directory</i>	Specifies the location of the data directory (terrains and so on).
--dis	Use the DIS protocol and enable DIS-specific options.
--dispSetting <i>filename</i>	Specifies the display configuration to load.
--entDispSetting <i>filename</i>	Specifies the entity display setting to load.
--envSetting <i>filename</i>	Specifies the environment configuration file to load.
(-E --entTypeMap) <i>filename</i>	Load the specified entity type mapping file. The file extension determines the type of mapping.
--fullScreen	Start in full screen mode.
(-G --locale) <i>language</i>	Specifies the locale (for localization).
(-h --help)	Displays command-line usage information.
--hla13	Use HLA 1.3 and enable HLA-specific options.
--hla1516	Use HLA 1516 protocol and enable HLA-specific options.
(-l --logFileName) <i>filename</i>	Specifies the log file name.
(-L --loadObservers <i>filename1</i> <i>filenamen</i>)	Loads the saved observers in the specified files.
(-n --notifyLevel) <i>notify_level</i>	Specifies the notification level for application messages. Range: 0-4.
--noGui	Start without the graphical user interface.
--plugin <i>filename</i>	Load the specified plug-in. Can be used multiple times.
--showConsole	Display a console window.
(<i>terrain_MTF_file</i> <i>scene_MSF_file</i>)	Specifies a terrain (MTF) or scene (MSF) file to load.
--unplug <i>filename</i>	Do not load the specified plug-in. Can be used multiple times.
--userDataDir <i>directory</i>	Specifies the user data directory.
(-v --version)	Display version information and exit.
--vsync	Start with vsynched windows.
-W --withDisplayEngine <i>port@netAddress;displayConfigFilenam</i> <i>e1 n</i>	Connect to a display engine running in subordinate mode on another computer.
HLA Only	
(-f --fomMapperLib) <i>libname</i>	FOM Mapper library name.
(-F --fedFileName) <i>fedfile_name</i>	FED file name.
--fomMapperInitData <i>data</i>	FOM Mapper initialization data.

vrfGui Command-Line Options

Option	Description
(-N --federateName) <i>federate_name</i>	Federate name. Default: MAK IG HLA 1.3 Driver or MAK IG HLA 1516 Driver.
--rprFomRevision	RPR FOM revision
--rprFomVersion <i>version_number</i>	RPR FOM version. (0.5, 0.7, 0.8, 1.0, 2.0006, 2.0014, or 2.0017)
--useAdvisories	Enables or disables HLA advisory messages. Default: true.
--useGeographicDdm	Specifies use of geographic DDM if you are using the MÄK RTI. Default: false.
(-x --execName) <i>exec_name</i>	Execution name. Default: VR-Link.
DIS Only	
(-a --appNumber) <i>number</i>	Application number.
(-A --destAddrString) <i>address</i>	Destination address.
--deviceAddress <i>address</i>	Specifies the address of the network card to use for UDP traffic.
--disVersion	The DIS version. Default: 5.
--mcastTtl	Specifies the multicast time-to-live. Default: -1.
--multicastAddresses <i>address1 ... addressn</i>	Specifies one or more multicast addresses.
(-P --disPort) <i>port</i>	DIS port.
--recvBufferSize <i>size</i>	Receive buffer size
(-s --siteId <i>ID</i>	Site ID.
--sendBufferSize <i>size</i>	Send buffer size. Default: -1.
--useAsyncIO	Use asynchronous I/O. Default: False.
(-x --exerciseId) <i>ID</i>	Exercise ID.

vrfLauncher Command-Line Options

Option	Description
-a <i>application_number</i>	Specifies the application number.
--app vrfSim	Start Launcher for back-end only.
--configureConnections	Open Simulation Connections Configuration dialog box for creating and editing connection configurations.
--configFilename <i>filename</i>	Specifies the connection configurations file to use. Default: <i>./config/connectionsSettings.xml</i> .
-h	Displays help information in a console window.
-s <i>site_id</i>	Specifies the site ID. This option applies only to the front-end in combined mode.
-v	Displays version information.
-Z vrfSim	Start Launcher in combined mode. (Case sensitive; must be capital Z.)
--	Ignore all of the command-line options that are listed after this argument. This lets you quickly limit the arguments applied if you want to edit a script or batch file without losing the original command line.

VR-Forces Mouse Button Mappings

Navigation Action	Mouse Action	Context
Attach to entity.	Shift + left-click	Entity
Attach to prop.	Ctrl + Shift + left-click	Prop
Deselect all.	Left-click	Terrain or sky
Drag heading and pitch.	Right Drag	
Drag terrain.	Left-click + drag	Terrain or prop
Object remains selected, deselect all others.	Left-click	Selected entity or prop
Orbit drag terrain.	Ctrl + left-click + drag	Terrain or prop
Select entity or prop.	Ctrl + left-click	Unselected entity or prop
Select object and unselect all others.	Left-click	Unselected entity or prop
Teleport to location.	Shift + left-click	Terrain
Teleport to prop.	Shift + left-click	Prop
Teleport to entity.	Ctrl + Shift + left-click	Entity
Unselect.	Ctrl + left-click	Selected entity or prop
Drag the view.	Middle mouse button drag	
Mouse wheel while observer is not moving		
Scroll forward.	Mouse wheel forward	
Scroll back.	Mouse wheel back	
Mouse wheel while moving observer		
Increment linear speed.	Mouse wheel forward	
Decrement linear speed.	Mouse wheel back	
Increment linear speed.	Ctrl + mouse wheel backward	
Decrement linear speed.	Ctrl + mouse wheel forward	
Scroll level forward.	Shift + mouse wheel forward	
Scroll level backward.	Shift + mouse wheel backward	

VR-Forces Keyboard Mappings

K = numeric keypad (K8 means press Keypad 8). To use the numeric keypad for navigation, NumLock must be on.

Observer Movement	Observer	Level Observer	Laptop	2D Level Observer
Move forward.	w	unmapped	w	unmapped
Move back.	s	unmapped	s	unmapped
Move left.	a	unmapped	a	unmapped
Move right.	d	unmapped	d	unmapped
Move up.	q, Ctrl + q	unmapped	q, Ctrl + q	unmapped
Move down.	e, Ctrl + e	unmapped	e, Ctrl + e	unmapped
Move forward level observer.	unmapped	w	unmapped	w
Move back level observer.	unmapped	s	unmapped	s
Move left level observer.	unmapped	a	unmapped	a
Move right level observer.	unmapped	d	unmapped	d
Move up level observer.	unmapped	q, Ctrl + q	unmapped	unmapped
Move down level observer.	unmapped	e, Ctrl + e	unmapped	unmapped
Move attached context forward.	Shift + w	Shift + w	Shift + w	Shift + w
Move attached context back.	Shift + s	Shift + s	Shift + s	Shift + s
Move attached context left.	Shift + a	Shift + a	Shift + a	Shift + a
Move attached context right.	Shift + d	Shift + d	Shift + d	Shift + d
Move attached context up.	Shift + q	Shift + q	Shift + q	unmapped
Move attached context down.	Shift + e	Shift + e	Shift + e	unmapped
Level yaw left.	K 4	K 4	←	←
Level yaw right.	K 6	K 6	→	→
Level pitch up.	Shift + j	Shift + j	Shift + j	unmapped
Level pitch down.	Shift + k	Shift + k	Shift + k	unmapped
Level roll right.	Shift + i	Shift + i	Shift + i	unmapped
Level roll left.	Shift + m	Shift + m	Shift + m	unmapped
Yaw left.	Shift + h	Shift + h	Shift + h	Shift + h
Yaw right.	Shift + l	Shift + l	Shift + l	Shift + l
Pitch up.	K 8	K 8	↑	unmapped
Pitch down.	K 2	K 2	↓	unmapped
Roll right.	K 9	K 9	K 9	unmapped
Roll left.	K 3	K 3	K 3	unmapped
Orbit right.	Ctrl + d, →	Ctrl + d, →	Ctrl + d	unmapped
Orbit left.	Ctrl + a, ←	Ctrl + a, ←	Ctrl + a	unmapped
Orbit up.	Ctrl + w, ↑	Ctrl + w, ↑	Ctrl + w	unmapped
Orbit down.	Ctrl + s, ↓	Ctrl + s, ↓	Ctrl + s	unmapped
Look up.	Ctrl + K 8	Ctrl + K 8	Ctrl + K 8	Ctrl + K 8
Look down.	Ctrl + K 2	Ctrl + K 2	Ctrl + K 2	Ctrl + K 2

Observer Movement	Observer	Level Observer	Laptop	2D Level Observer
Look left.	Ctrl + K 4	Ctrl + K 4	Ctrl + K 4	Ctrl + K 4
Look right.	Ctrl + K 6	Ctrl + K 6	Ctrl + K 6	Ctrl + K 6
Look tilt left.	Ctrl + K 3	Ctrl + K 3	Ctrl + K 3	Ctrl + K 3
Look tilt right.	Ctrl + K 9	Ctrl + K 9	Ctrl + K 9	Ctrl + K 9
Zoom In	unmapped	unmapped	unmapped	e
Zoom Out	unmapped	unmapped	unmapped	q
Suppress constrain to terrain.	x	x	x	x
Reset observer:	Space	Space	Space	Space
In absolute mode, go to the center of the scene.				
If attached, reset the attach position and orientation.				
Level observer.	K 5	K 5	Shift + Space + ↓	K 5
Toggle Force North Orientation	unmapped	unmapped	unmapped	unmapped
Toggle Orient To Target Constraint	unmapped	unmapped	unmapped	unmapped
Reset the camera.	K 0	K 0		K 0
Detach.	z	z	z	z
Attach to the next entity allowed by the attachment filter.	Period (.)	Period (.)	Period (.)	Period (.)
Attach to next entity or prop.	>	>	>	>
Attach to the previous entity allowed by the attachment filter.	Comma (,)	Comma (,)	Comma (,)	Comma (,)
Attach to previous entity or prop.	<	<	<	<
Next attach mode.	Shift + z	Shift + z	Shift + z	Shift + z
Increase linear speed.	K 7	K 7	K 7	K 7
Decrease linear speed.	K 1	K 1	K 1	K 1
Increase orbit speed.	K +	K +	K +	K +
Decrease orbit speed.	K -	K -	K -	K -
Toggle speed scaling on or off.	K *	K *	K *	K *
Toggle constrain to terrain on or off.	/	/	/	/
Clear the current movement mode. This key is an escape mechanism if keys are not behaving as you expect. This is typically due to a loss of the window context.	c	c	c	c
Toggle view controls on or off.	Shift + c	Shift + c		Shift + c
Toggle entity labels on or off.	unmapped	unmapped	unmapped	unmapped



Link - Simulate - Visualize