VRchitect V1.1 Online Help File For the IBM PC

Contents

Introduction

VRchitect is a virtual world editor that allows you to alter most aspects of your existing Spectre VR games and also helps you to create entirely new games.

This application follows the conventions of the standard Windows interface.

Getting Started

One of Spectre VR's major advances beyond the original Spectre is that games are now stored separately from the Spectre application in Matrix files. Spectre uses the information in the Matrix files to generate each game grid with which players will be challenged.

To start, select Load from the File menu. From here, you can choose which Matrix file to open in order to edit the corresponding Spectre VR game. In the future, you will also save your work from this pull-down menu by selecting the Save or Save as option.

Select Exit from the File menu to leave VRchitect.

Warning: Always keep a backup copy of your Matrix files in case something goes wrong!

Editing Matrix files

The Matrix Editor provides quick access to the game levels in the same order in which they will be played. Each game level consists of a game grid layout, a set of enemies that roam the game grid, and a set of parameters that define how the enemies behave.

The Game Difficulty corresponds to the difficulty level that you set in the Spectre Options window.

The Level scrollbar controls which game level you are currently viewing. For instance, if you are looking at Game Difficulty: Medium and Level 5, then what you see displayed in the Matrix Editor is the fifth game grid that players will face when playing at the intermediate difficulty level.

The Game Grid pull-down list specifies which game grid layout to use for the selected level. Each grid is designated by a unique number .

The Enemy Set pull-down list specifies which set of enemies to fight against on the selected level. Each set is designated by a unique number.

The Enemy Data pull-down list specifies which set of characteristics to use for the enemies on the selected level. Each set is designated by a unique number.

The Song pull-down list specifies which song to use on the selected level. Each song is designated by a unique number and can also have a name. If you do not want to play a song on the selected level, select the 0 None item.

The Activate pull-down list specifies which special weapon to activate before the selected level is actually played.

Clicking an Edit button allows you to edit the corresponding selected information.

Clicking a New button allows you to create a new set of information that you can later select and use.

Clicking a Kill button permanently removes the corresponding selected item from the matrix file.

The three buttons below the level scrollbar allow you to add and delete levels from the currently selected Game Difficulty.

Clicking the New Lives button calls up a dialog window in which you can set the scores required for the first ten extra tanks and the score increment required for each additional extra tank thereafter.

Clicking the Weapons button calls up a dialog window in which you can change parameters for each type of weapon.

Editing Weapon Data

The Weapons Editor allows you to specify how the various weapons function.
Note: The robots are also using these weapons! If you are too liberal with the damage done parameter, the game will end very quickly when the robots shoot back!
The menu contains a list of all the weapons.

When you select an item from the drop-down list, the corresponding values will be displayed below.

The Parameters

Damage points done: How much damage is done when you or a robot gets hit. Corresponds to the Damage points needed to kill in the Enemy Editor.

Amount of ammo needed: How much ammo you use up when you use this weapon. (The robots never worry about this parameter! Now you can get even with them! Set it to zero!)

Editing New Lives Values

The Scoring Editor allows you to specify when you get extra tanks.

The Parameters

Score required for n 'th extra tank: How high your score has to go before you get the n th extra tank.

Score increment required for each additional tank: How many extra points you have to accumulate before you get your 11th, 12th, etc. extra tank.

Editing Game Grids

When you open a game grid, VRchitect will display it as it would be seen in Spectre from the bird's eye view above the game grid. The objects on the game grid are called obstacles. You may place new obstacles on the game grid by selecting one from the 15 obstacle palette on the lower right-hand side of the grid editor window. When you select an obstacle from the palette, your cursor will change to a symbol of that obstacle ONLY while the mouse is over the main grid edit window (on the left-hand side of the screen). Simply click with the left mouse button on the main edit grid to 'stamp' down new obstacles. The 15 types of obstacles are, going in order down each 5 button column on the obstacle palette:

flag
column
random
cyber mud
down transport

cube
wall
acid
teleport
pusher

pyramid
windmill
ice
up transport
flat wall

You will remain in the same mode (for example, 'add new flag' mode) until you explicitly choose a new mode by clicking on the appropriate button on the right-hand side of the grid edit window.

If you choose wall or flat wall from the obstacle palette, you will get the 'line draw' mode instead of the customary 'stamp' mode. While in line draw mode, click and drag with the left mouse button to create multiple wall segments. When you first press the left mouse button and hold it down, a 'line' will follow your mouse movements. Determine where you want the walls to go and let go of the left mouse button. VRchitect will fill in the line with the wall or flat wall obstacles.

Obstacles are removed from the game grid by selecting the Kill button, which changes the shape of your cursor into a 'Ghostbusters/Don't Smoke' sign when you move the cursor over the edit game grid. Simply click on obstacles with the left mouse button to remove them from the game grid. Again, you will remain in kill mode until you choose another mode.

Each obstacle has a rotation or 'heading' attribute used by Spectre when placing the object in the 3D world. The rotation of an object (which must be between 0 and 360 degrees) can be changed by entering select mode, which is accomplished by clicking on the Select button directly to the left of the Kill button. Once in select mode, simple click on any obstacle with the left mouse button, which will cause the rotation dialog window to appear. Enter the rotation factor and hit the Ok button to save the results, or the Cancel button to abort.

The view of the grid can be changed at any time by using the zoom and scrolling features found in the upper right-hand area of the grid edit window, directly under the 'radar' display. Hit the '+' button to zoom in, the '-' button to zoom out, or the arrow buttons to move around. Notice that the black rectangle on the radar screen shows you exactly where you are in relation to the entire game grid. If the black rectangle hits the sides of the radar box, notice that it can not go any further. You must scroll the other way. Also, notice that you also can not zoom out any further if the black rectangle is against the edge. First move over and then zoom out.

When you are ready to save your changes to the game grid, select the Done button, found on the lower right-hand side on the grid edit window. A dialog will appear asking if you want to save your changes. Choose the Yes button to permanently save your edits, or the No button to loose you changes.

The obstacles for each game grid are stored in an ordered list. When Spectre builds the actual game grid, it automatically converts the first few obstacles in this list into flags. VRchitect takes advantage of this by collecting all of the flags that you place and placing them at the beginning of the list. This is only partially effective, however, because Spectre uses different numbers of flags on different levels. Thus, you may observe one of two events:

1) Spectre wanted fewer flags than you placed. All the other obstacles that you placed will still be on the game grid. 2) Spectre wanted more flags than you placed. The required extra number of flags comes from converting some of your other obstacles into flags.

Thus, in case 2, don't be surprised when your artistic arrangement of obstacles has a gaping hole in it.

An Advantageous Way of Rearranging the Game Grids

Stone Henge type structures and mazes confuse the robots. When you hide inside one of these, the robots have a very hard time finding their way in. You can then blast them through a hole before they find you. The more exits you have from the place the better. If a robot finds its way in, you probably won't have time to turn and shoot it before it blows you away. Mazes can screw you up, too. If the walls are too close together, you will have a very hard time moving around. A good ruleofthumb for creating tight mazes is that you need at least space for one cube between adjacent walls in order to allow you tank to move through.   More space is, of course, safer.

Spectre does not allow more than 100 obstacles on a game grid because more would slow down the game too much.   To be safe, place at least 10 or 15 flags so that Spectre won't remove any of your other obstacles.   (See Placing Flags above)

Editing Enemy Sets

The Enemies Editor allows you to specify which enemies you have to fight against.

The menu contains a list of all the possible enemies.   The items marked with a '*' are the ones that will appear on the game grid.   When you select an item from the menu, the corresponding values will be displayed below.   If the item is not marked, (i.e. no data exists for it yet) it will be displayed as all zeros.

The buttons below the menu works as follows:

The Kill button will remove the enemy from the set when clicked.   The '*' in front of the enemy name will be removed.

The Add button will add the enemy to the set when clicked.   The initial values will be set to 1s.

Note: With more than 4 types of enemies on a game grid, game speed can slow down considerably as Spectre pauses between levels.

The Parameters

<u>Maximum # at start of level</u>:   At most this many of the selected type will be created when the level starts.   Note that the sum of all maximum values for a given enemy set must be less than or equal to 8.

<u>Minimum # at start of level</u>:   At least this many of the selected type will be created when the level starts.

<u>Regeneration multiplier</u>:   Each one of the robots that exists at the start of play will have to be destroyed this many times before it stops regenerating.

As an example, assume you are dealing with Super Robots, and you specify min 3, max 5, multiplier 2.   When you enter the level, Spectre generates a random number between 3 and 5 inclusive.   Let's say it picks 4.

There will then be 4 Super Robots on the game grid when you start playing. The multiplier then tells you that you will have to face a total of 2*4=8 Super Robots.

Editing Enemy Data

The Enemy Editor allows you to specify the parameters of all the different enemies you have to fight against, even if some of them are not used on the game grid.

The top pull-down list contains a list of all the possible enemies.

When you select an item from the list, the corresponding values will be displayed below.

The Parameters

<u>Damage points needed to kill</u>: This is how much armor the robot has. Note that weapons can cause more than 1 point of damage. (Refer to section on Weapons Editor)

<u>Damage points from collision</u>: This is how much damage the robot does to you if you collide with it. (This is especially useful for Slicers since this is their normal mode of attack.)

<u>Attack range</u>: The lower this number, the closer you have to get to the robot before it starts following you.

<u>Firing range</u>: The lower this number, the closer the robot has to get to you before it starts shooting at you.

<u>Firing arc</u>: The lower this number, the more accurate the robot is at shooting you. The higher this number, the more the robot will shoot, even if you are not directly in the line of fire.

<u>Firing interval</u>: The lower this number, the more often the robot shoots.

<u>Moving speed</u>: How fast the robot can pursue you. Note that you can't go faster than 8.

<u>Turning speed</u>: How fast the robot can turn around.

<u>Scoring value</u>: How many points you get for blowing it up.
<u>Bonus Scoring value</u>: How many bonus points you get for blowing it up.

Weapon type: The weapon that the robot uses to shoot at you. Note: These are the same weapons that you use!

Fun Games to play with Spectre

The obvious possibilities are such combinations as six players in a network game.

Less obvious, but even more exciting is six players in a network game with each player having a totally different arrangement of obstacles. When a player tries to hide behind one of his/her obstacles, you can drive right through it!!

Another possibility is for two players to play against each other on one computer. To explain: One person uses the keyboard and one person uses the mouse. One person tries to commit suicide while the other tries to score points and progress to higher levels. Neither person really has control of the tank, which results in chaos.

You could also arrange for twelve people to play a network game: two at each computer as described above!

My congratulations to Sam, Steve, and Howard for writing such a flexible game! The possibilities for fun and addiction seem almost endless...especially when coupled with the power of VRchitect!

Credits

VRchitect was programmed and adapted to Windows by **Dana Dominiak** with many thanks to the following:

Pierre Maloka, Spectre VR's DOS/Windows programmer, for support routines, testing, suggestions, and much help in general, **Thom Grace**, graphics professor from Illinois Institute of Technology, **Kitric Kerns**, and **Peep Marie Dominiak** for moral support, and **Erin Dominiak** for caffeine. And remember, loosing your driver's license is just God's way of saying, "Booga, Booga".

Original Credits from Macintosh Version:

VRchitect for Macintosh was written and designed by **John Lindal** with many thanks to the following:

Sam Schillace for Spectre and for finding the original version of VRchitect and seeing its potential, **Moses Ma** without whom VRchitect would merely have wasted my time and disk space, **Howard Shere** for Spectre VR and the encouragement,

Steve Newman for Spectre awesome graphics and for giving Sam Internet access, **Craig Fryar** for helping with the contract (seriously!)

Special thanks to the Spectre players everywhere who sent me suggestions, comments and encouragement regarding VRchitect.

Disclaimer

References to particular products by this documentation do not constitute endorsement of these products or their creators by either the author or Velocity Development Corporation. The author just happens to have used these particular products while testing VRchitect. There may be better programs available for the purposes cited.