Purpose

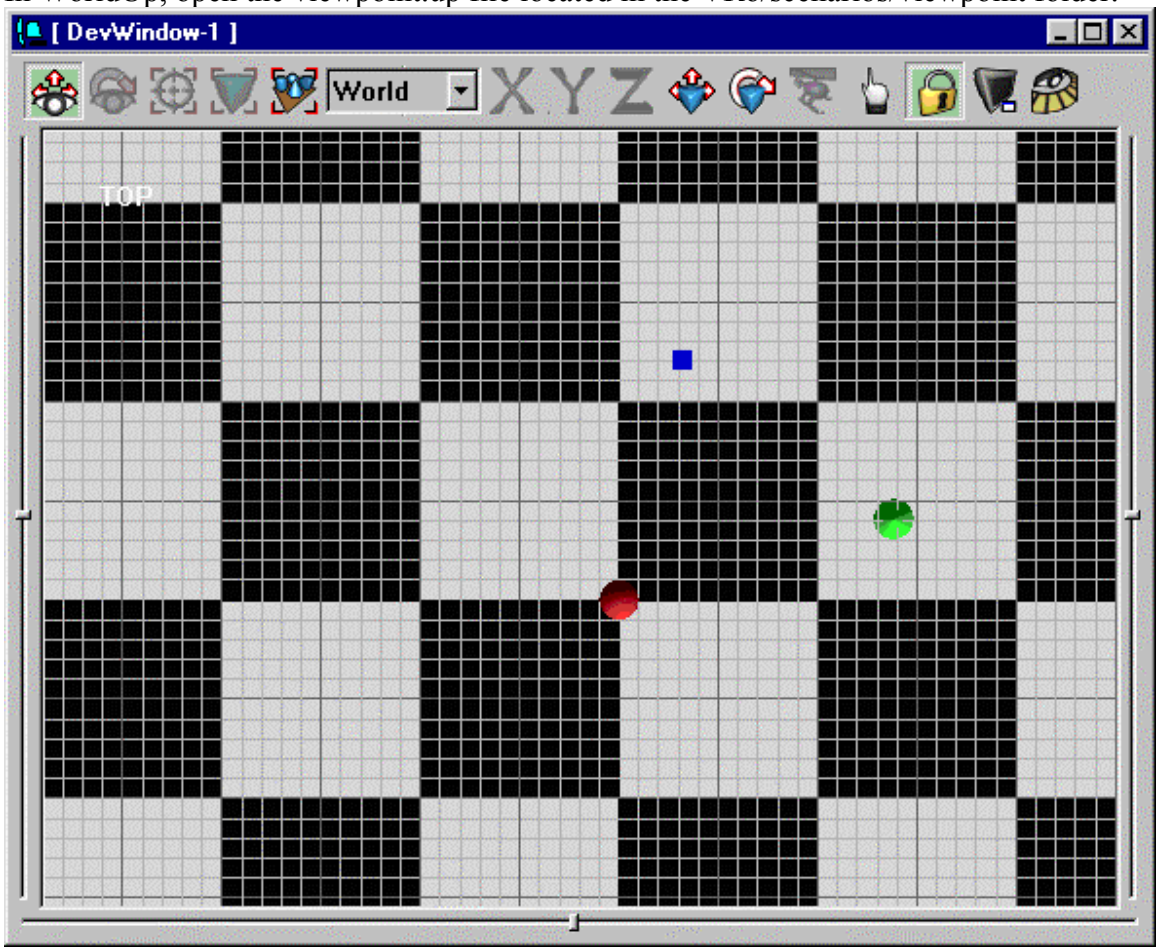
The purpose of this scenario is to show how to create controllable viewpoints.

This scenario hopes to show:

- Create a block subtype to be used as a viewpoint manager.
- Create an object used as a fixed viewpoint.
- Create an object used as a mouse viewpoint.
- Expose objects using the MATLAB Plug-in.
- Load our viewpoints in MATLAB.
- Control our viewpoints in MATLAB with a simple function m-file.
- Load our objects in Simulink.
- Control our viewpoints in Simulink.

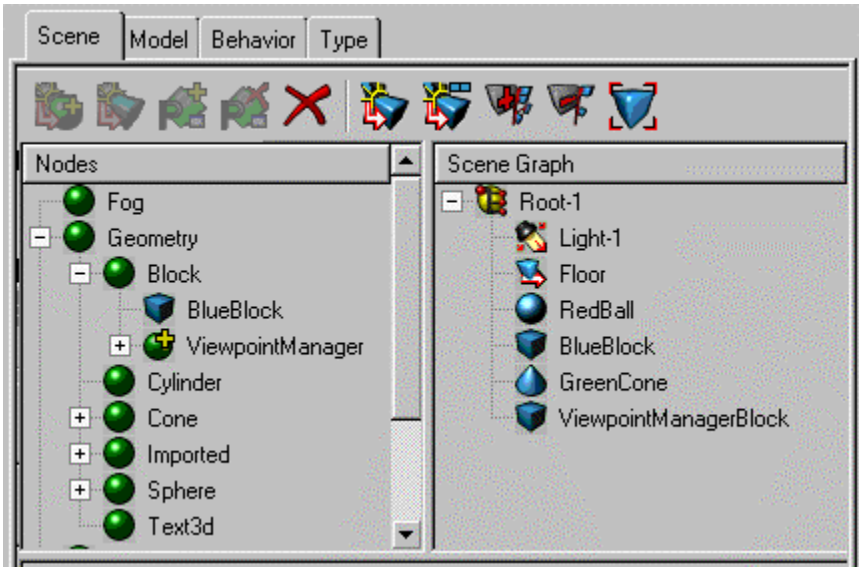
Load sample viewpoint world

In WorldUp, open the viewpoint.up file located in the VRo/scenarios/viewpoint folder.



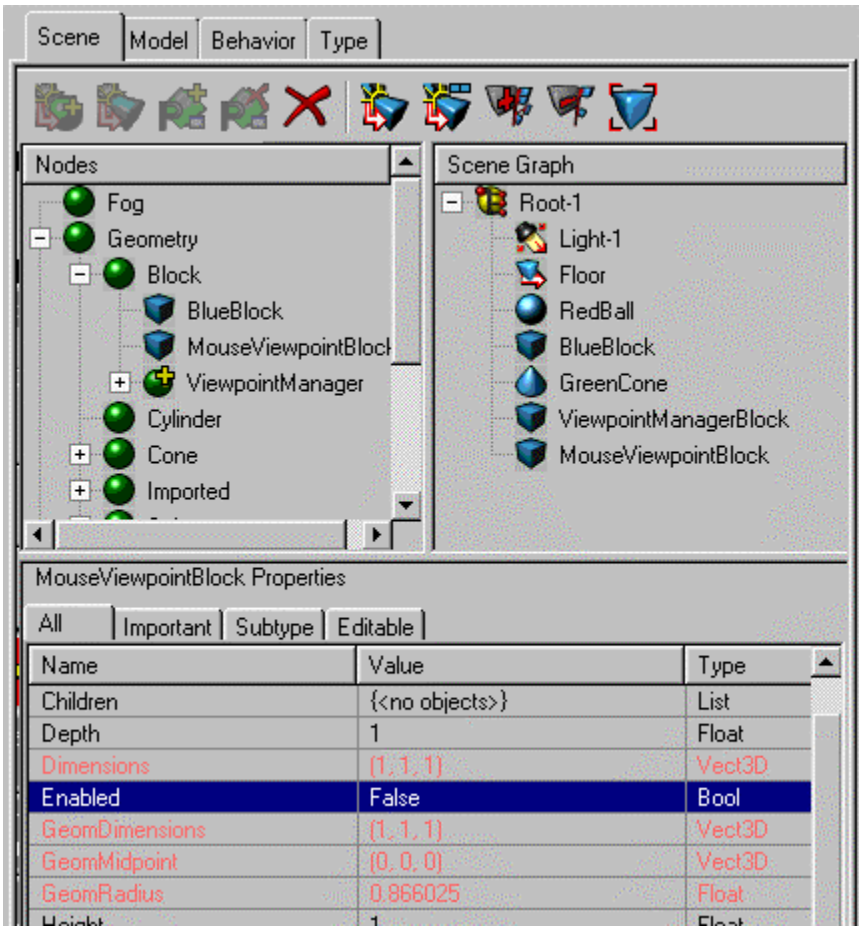
Create object to be used as a viewpoint manager

1. Expand the Block node. You will see the BlueBlock object.
2. Press the Create Subtype  toolbar button.
3. Rename SubBlock to ViewpointManager.
4. Right-click on ViewpointManager and select Add Property.
5. Enter ActiveViewpoint for Property Name.
6. Select Single for Property Type.
7. Enter 1 for Initial Value.
8. Press Done.
9. Drag and drop ViewpointManager subtype to the Scene Graph.
10. Rename to ViewpointManagerBlock.
11. Set Enabled to False.



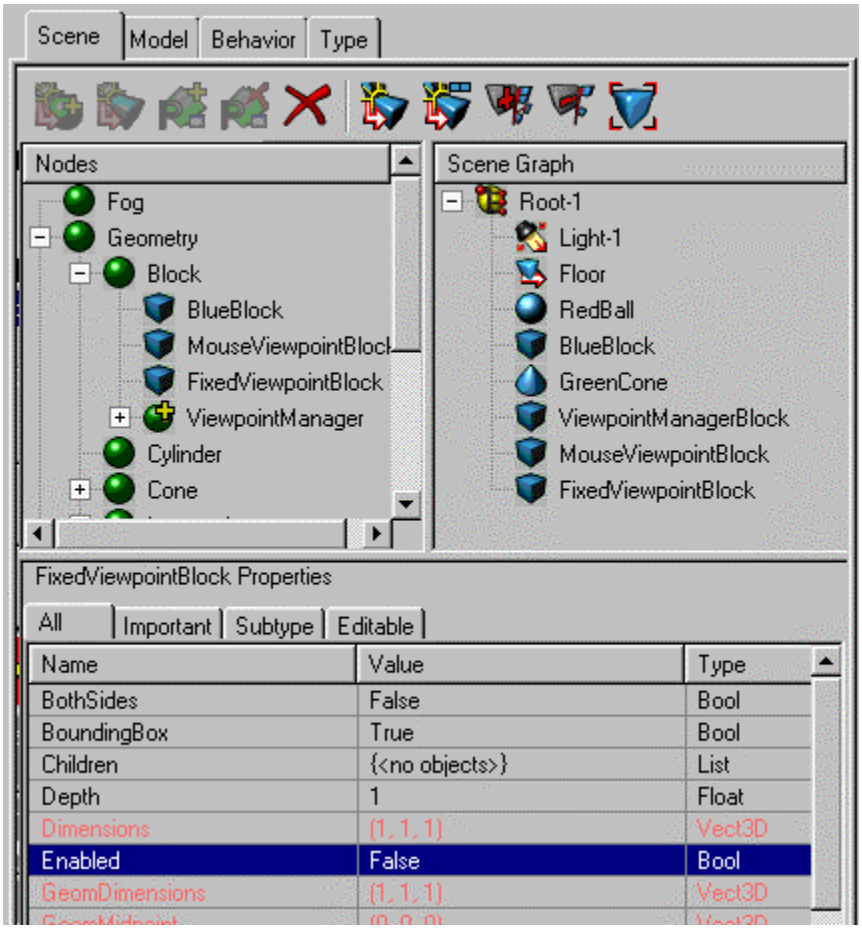
Create object to be used as mouse viewpoint

1. Drag and drop the Block subtype to the Scene Graph.
2. Rename Block-1 to MouseViewpointBlock
3. Set the Stretch property to 20, 20, 20.
4. Set the Enabled property to False.



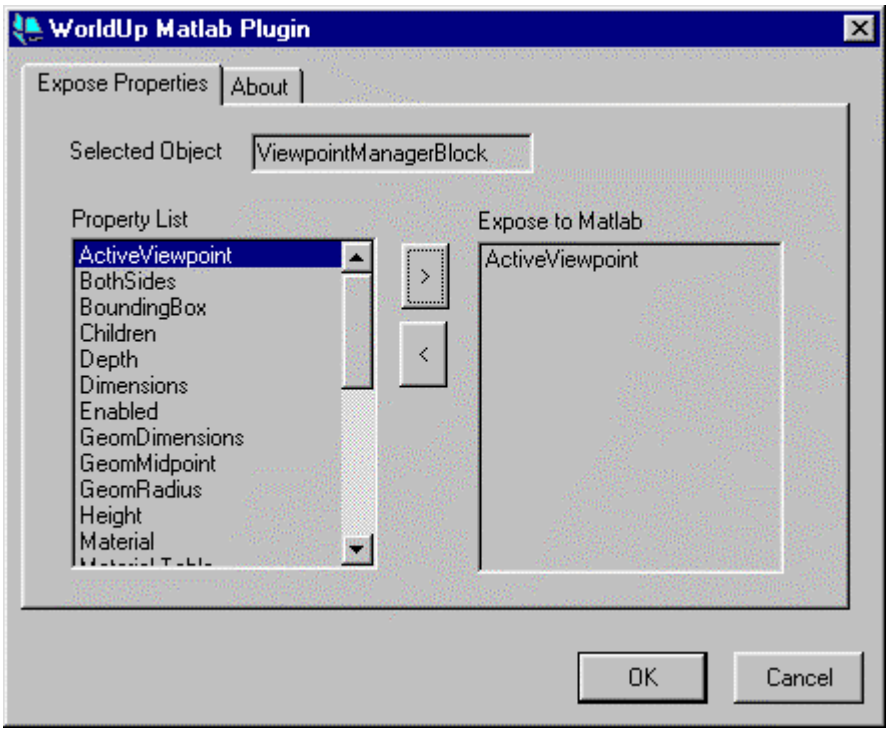
Create object to be used as fixed viewpoint

1. Drag and drop the Block subtype to the Scene Graph.
2. Rename Block-1 to FixedViewpointBlock.
3. Set the Enabled property to False.



Use MATLAB Plug-in to expose objects

1. Select the ViewpointManagerBlock in the Scene Graph.
2. Select File►Plug-in Tools►Expose to Matlab
3. Expose the ActiveViewpoint property.
4. Press OK.
5. Select the MouseViewpointBlock in the Scene Graph.
6. Select File►Plug-in Tools►Expose to Matlab
7. Expose the Translation property.
8. Press OK.
9. Select the FixedViewpointBlock in the Scene Graph.
10. Select File►Plug-in Tools►Expose to Matlab
11. Expose both Translation and Rotation properties.
12. Press OK.



Write task script to control viewpoints

Select File►New Script.

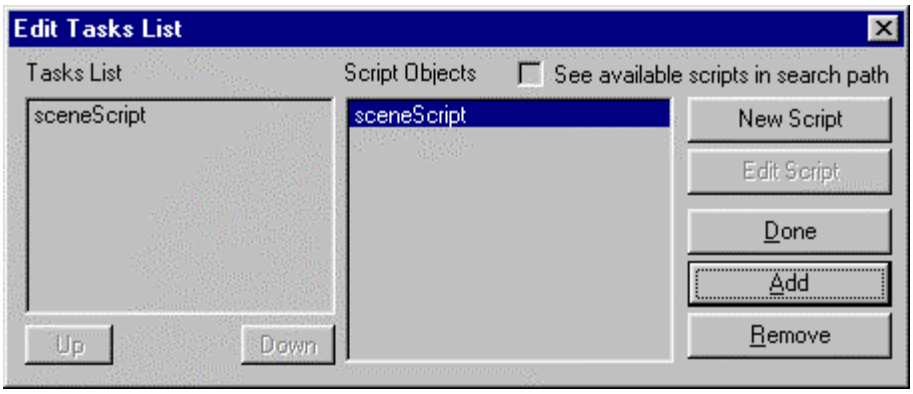
Enter the following:

```
sub task

dim w as window
dim View as viewpoint
dim ViewDirection as vect3d
dim ViewOrientation as orientation

dim ViewpointManagerBlock as viewpointmanager

set w = getfirstwindow()
set View = getviewpoint("Viewpoint-1")

set ViewpointManagerBlock = getviewpointmanager("ViewpointManagerBlock")

if ViewpointManagerBlock.ActiveViewpoint = 1 then
   dim MouseViewpointBlock as geometry
   set MouseViewpointBlock = getgeometry("MouseViewpointBlock")

   w.ZoomToNode MouseViewpointBlock
end if

if ViewpointManagerBlock.ActiveViewpoint = 2 then
   dim FixedViewpointBlock as geometry
   set FixedViewpointBlock = getgeometry("FixedViewpointBlock")

   FixedViewpointBlock.GetGlobalLocation ViewDirection, ViewOrientation
   View.SetPosition ViewDirection
   View.SetOrientation ViewOrientation
end if

end sub
```

Save the task script as **scene.ebs**.

Because this is a task, it needs to be attached to an item in our scene graph. Let's attach it to our Light-1 object.

1. Right-click on Light-1 and select Edit Tasks.
2. Select SceneScript and press Add.
3. Press Done.



Select Simulation►Run to test that our task works. You should be able to fly around the three objects in a spherical fashion. Close the created window when finished.

Load into Matlab

At the MATLAB prompt, pass the path to your viewpoint.up file into `vroload`:

```
my_objects = vroload('c:\mfiles\vro\scenarios\viewpoint\viewoint.up');
```

Let's see what objects we have by using the overloaded `set` command:

```
set(my_objects(1))

         Type: ViewpointManager
         Name: ViewpointManagerBlock
   Properties: ActiveViewpoint (Custom)
```

This tells us `my_objects(1)` is of type ViewpointManager named ViewpointManagerBlock with ActiveViewpoint as the exposed property.

```
set(my_objects(2))

         Type: Block
         Name: MouseViewpointBlock
   Properties: Translation (Vect3D)
```

This tells us `my_objects(2)` is of type Block named MouseViewpointBlock with Translation as the exposed property.

```
set(my_objects(3))

         Type: Block
         Name: FixedViewpointBlock
   Properties: Rotation (Yaw,Pitch,Roll,Order)
               Translation (Vect3D)
```

This tells us `my_objects(3)` is of type Block named FixedViewpointBlock with Rotation and Translation as the exposed properties.

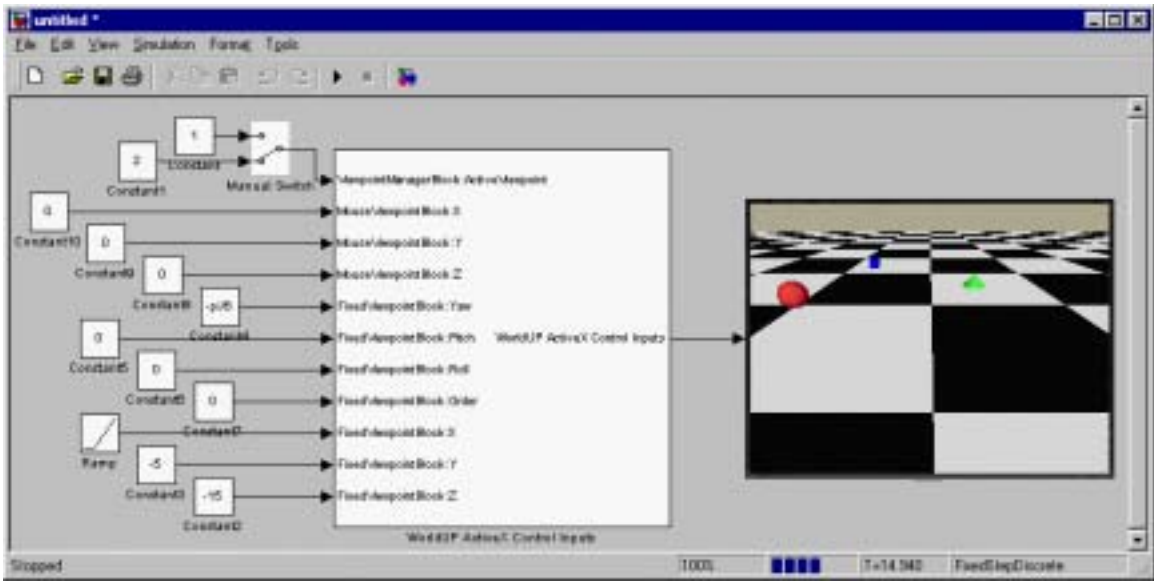
Write m-file to control viewpoints

Here is a simple m-file that controls the various viewpoints in our world:

```
function viewpoint(my_objects)

% assign our objects
ViewpointManagerBlock = my_objects(1);
MouseViewpointBlock = my_objects(2);
FixedViewpointBlock = my_objects(3);

%%% fly around in a circle looking down at our three objects

% compute circular x and z values
TransImagData = 30*exp(i*[0:0.001:pi*2]);
TransXData = real(TransImagData);
TransZData = imag(TransImagData);
TransYData = -15*ones(size(TransImagData));
TransData = [TransXData', TransYData', TransZData'];

% compute rotation values to keep viewpoint looking
% at center.  RADIANS!

% look down at 30 degrees
RotXData = -pi/6*ones(size(TransImagData));

% compute angle to view center of circle
RotYData = angle(TransImagData);

% offset for VR rotational alignments
RotYData = -pi/2 - RotYData;

% construct other vectors
RotZData = zeros(size(TransImagData));
RotOrder = zeros(size(TransImagData));
RotData = [RotXData', RotYData', RotZData', RotOrder'];

% make our fixed viewpoint active
set(ViewpointManagerBlock, 'ActiveViewpoint', 2);

% move our fixed viewpoint
for k = 1:size(TransData,1),
   set(FixedViewpointBlock, 'Translation', TransData(k, :), ...
                            'Rotation', RotData(k, :));
end

% set our mouse viewpoint active above the center of our world
set(ViewpointManagerBlock, 'ActiveViewpoint', 1);
set(MouseViewpointBlock, 'Translation', [0, -15, 0]);
```

Load into Simulink

1. Open Simulink and create a new model window.
2. From the VRo Blockset, drag the WorldUp ActiveX Control into the new model.
3. Double-click on the thin black border surrounding the control to open the mask.
4. In the **World Up File** edit field, enter the full path to the location of viewpoint.up.
5. Press OK.
6. Add some simple sources.



If you notice your simulation finishes too quickly, in Simulation►Parameters, set the Solver Type to Fixed-Step and specify a Fixed Step size of 0.01.

Conclusion

From this scenario we were shown how to:

- Create a block subtype to be used as a viewpoint manager.
- Create an object used as a fixed viewpoint.
- Create an object used as a mouse viewpoint.
- Expose objects using the MATLAB Plug-in.
- Load our viewpoints in MATLAB.
- Control our viewpoints in MATLAB with a simple function m-file.
- Load our objects in Simulink.
- Control our viewpoints in Simulink.

If you have any further questions, do not hesitate to contact Terasoft Support.