
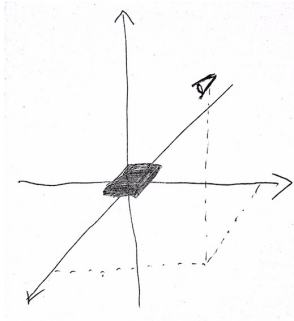
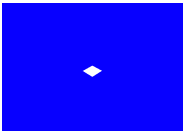
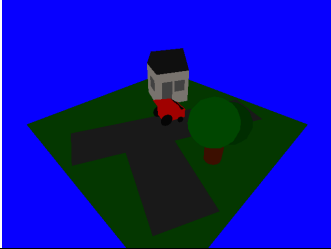
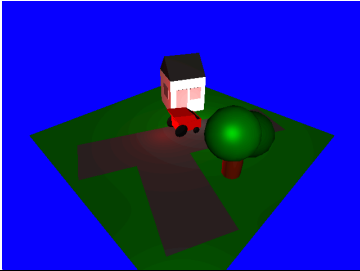


Game Engines: Group Exercise 5-2: 3D Graphics and Scene Graphs

Learning objectives	<ul style="list-style-type: none">• Creating a first (early) version of your own game engine• Using Camera's lookAt function to position camera.• Use perspective projection• Implementing a hierarchical scene graph• Implementing local transform (using translate, rotate, scale)• Implementing global transform (using parent object and local transform) <p>The supplied code can be used as a starting point for your own engine, but it is important to highlight that the code is created for simplicity. It contains many poor design choices, which should be addressed in your own engine (at some point).</p> <p>You are welcome not to use the provided code as a starting point, except for the SceneParser, which must be used for this exercise.</p> <p>When starting the exercise you only see a blue screen with a white line across!</p> 
Exercise 1	<p>The scene is initially composed of a single quad rotated -89 degrees around the x axis (such that its faces upwards)</p> <p>Camera setup</p> <ul style="list-style-type: none">• Change the camera projection such that it uses perspective projection (instead of the default no-projection). Use a field of view of 60 degrees. Set appropriate values for near and far plane (remember both should be positive)• Set the camera position to (10,10,10) and let the camera look at (0,0,0) using (0,1,0) as up vector.• When camera projection has been set, you should see this:  

Exercise 2	<p>Create scene graph</p> <ul style="list-style-type: none"> • Use SceneParser to read the scene file car_house_tree.json. Use the result to create the game objects in the scene (and remove the test object) <ul style="list-style-type: none"> ○ Currently the scene structure only supports three types meshes (plane, cube, sphere). Meshes of the same type must be shared between all game objects that reference it. ○ Use the unlit shader for all meshes. ○ Make sure to set a pointer to the parent game object if this is defined in the GameObjectDescription • Create transformation matrices the GameObject <ul style="list-style-type: none"> ○ The localTransform() should be created based on position, rotation and scale. Note that the rotation is defined in Eulers angels as three distinct rotations using degrees. ○ The globalTransform() should first carry out the local transform and then apply the global transform. <p>When implemented correctly (using the camera settings from exercise 1), you should see the following:</p> 
Exercise 3	<p>Add lights to the scene</p> <ul style="list-style-type: none"> • Add one directional light to the scene with the direction (1,1,1) (towards the sun). The light must have the color white and specularity 20; • Add two point lights to the scene with the following properties <ul style="list-style-type: none"> ○ Position (-1,1,1), color (5,0,0), range 5, specularity 20 ○ Position (0,1,2), color (3,3,3), range 5, specularity 20 ○ These two point lights will work as backlight and front light of the car. Notice that the colors have values more than 1.0 - this is used to simulate high light intensity. <p>When implemented correctly you should see:</p> 
Exercise 4 (Optional)	<p>Camera navigation</p> <ul style="list-style-type: none"> • Allow the user to control the camera (e.g. using AWSD and/or mouse)

Suggestions for improvement (Optional)	<p>The following should be considered (after hand-in of this exercise)</p> <ul style="list-style-type: none">• Represent both Camera and Lights as nodes in the scene graph• Introduce materials, which contains references to shaders and has defined colors and textures.• Update the scene graph file format and parser to reflect your need.• Implement functionality for reloading the scene (e.g. when pressing a button).• Better resource management - make sure resources are only loaded/created once. Make sure that resources are not leaked.• Refactor your solution to a component based scene graph (more on that later)
---	--