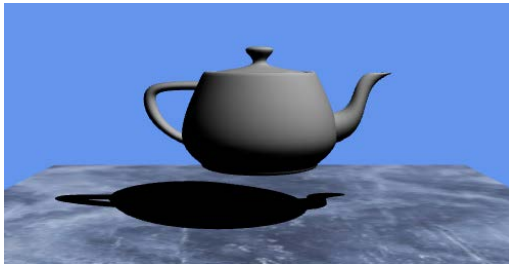
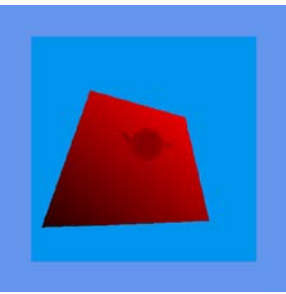


Worksheet 9: Shadow mapping

Reading	Angel: 5.11, 7.5.6, 7.12 WebGL Programming Guide: “Switching Shaders” WebGL Programming Guide: “Display Shadows”
Purpose	The purpose of this exercise is to understand and implement shadow mapping. This includes a deeper understanding of the different coordinate spaces in the pipeline as well as mapping between these coordinate spaces.
Part 1 Scene and projection shadows for reference	<p>The scene is a teapot jumping up and down on a textured ground quad with a point light circling the scene. We can set up this scene by combining Part 3 of Worksheet 5 with Part 3 of Worksheet 8.</p>  <p>The main difficulties are (a) that the ground plane and the loaded object use different shaders and (b) that we need to position the teapot in the scene.</p> <ul style="list-style-type: none"> • Use your code from Part 3 of Worksheet 5 to load and render the teapot model, which is available on File Sharing. Scale the teapot to a quarter of its original size and construct a model matrix for it that translates it by the vector $(0, -1, -3)$. • Insert shaders and the part that initializes and draws the textured ground quad from Part 3 of Worksheet 8. The ground quad and the teapot use different shaders. Consult the section called “Switching Shaders” from the WebGL Programming Guide to render these two objects using their own shaders. It is important to note the use of the function <code>initAttributeVariable</code> in the render function. • Move the teapot up and down over time by modifying the model matrix. Create a button that turns this motion on/off. • Ensure that you can modify the view matrix. Create an extra view matrix that looks at the teapot and the ground directly from above. Use this for debugging purposes, as it is much easier to spot misplaced shadows in this view. • For reference, insert the black projection shadows from Part 3 of Worksheet 8. In this scene, we use a model matrix to move the shadow-casting object. It is important to realize that the model matrix should be applied first (before the shadow projection matrix) when rendering the shadow polygons. • Set the light direction in the teapot shading according to the position of the point light circling the scene. Create a button that switches point light animation on/off.

Worksheet 9: Shadow mapping

<p>Part 2 Shadow mapping</p> 	<p>Projection shadows have several shortcomings. A significant problem is missing self-shadowing. Shadow mapping solves most of these issues (but introduces other problems).</p> <p>Your task is now to replace the projection shadows from Part 1 with shadow mapping. We recommend using the “Display Shadows” section of the WebGL programming guide. This text is available on File Sharing. There are two kinds of coordinate spaces used in this assignment: camera relative and light relative. The following figure illustrates these coordinate spaces and the transformations between them.</p> <pre> graph LR OC(object coords) -- model --> WC(world coords) WC -- view --> CEC(camera eye coords) WC -- view --> LEC(light eye coords) CEC -- projection --> CDC(camera clip coords) LEC -- projection --> LDC(light clip coords) CDC -- "w-divide" --> CND(camera normalized device coords) LDC -- "w-divide" --> LND(light normalized device coords) </pre> <p>The basic steps are:</p> <ul style="list-style-type: none"> • Render the scene from the point of view of the light source. Use a shader that draws fragment depth and use a framebuffer object (fbo) to render directly into a texture. The viewport might need adjustment when using the fbo. Bind the depth texture when drawing the ground plane to inspect the result, and use this inspection to set proper light view and light projection matrices. [Angel 5.11, 7.12] • Use the rendered depth texture in the other shaders to determine whether a fragment is in shadow or fully lit. Multi-texturing is needed for the ground plane to combine shadow and texture mapping. [Angel 5.11, 7.5.6] <p>Make the shadows dark but not pitch black. This can be done by adjusting the visibility factor in the fragment shader.</p>
Part 3	Compare projection shadows to shadow mapping by listing advantages and disadvantages of the two techniques.
Part 4 Optional	<p>Improve the precision of the shadow map using the method described in the “Display Shadows” section of the WebGL Programming Guide.</p> <p>Implement shadow antialiasing by averaging multiple shadow map lookups close to each other instead of using a single lookup. This technique is called “percentage-closer filtering”.¹</p>

¹ See Bunnell, M., and Pellacini, F. Shadow Map Antialiasing. In *GPU Gems*, Chapter 11, Addison-Wesley, 2004.
<https://developer.nvidia.com/gpugems/gpugems/part-ii-lighting-and-shadows/chapter-11-shadow-map-antialiasing>