Worksheet 3

If we compare rendered images to photographs, flatly coloured surfaces with no visual detail is one of the first giveaways that an image is a rendering. To obtain realism in rendered images, it is important to add visual detail. One way to add visual detail is to use texturing. This set of exercises is about how to do texturing in ray tracing.

Learning Objectives

- Use texture mapping (mapping an image to a surface) to heighten the level of visual detail.
- Compute texture coordinates using inverse mapping.
- Use bilinear interpolation for texture magnification filtering.
- Use stratified jitter sampling for texture minification filtering.

Texture Mapping

Texture mapping is in four steps: (1) loading the texture image, (2) computing texture coordinates for the object to which the texture should be mapped, (3) looking up a colour in the texture image for a given set of texture coordinates, and (4) using the texture colour in a shader.

1. Load and render a texture image file using different addressing and filtering modes.

   1.a Create a function that loads a texture image file without colour space conversion. Once loaded, store the image data in a texture created on the GPU. Use image plane coordinates (`uv`) to render images of the texture with clamp-to-edge addressing and nearest filtering. (Comment out the regular loop in the main function of the fragment shader for this part, but save it for later parts.).

   1.b Implement repeat addressing and linear filtering and create an interface (e.g. selection menus) for selecting clamp-to-edge vs. repeat addressing and nearest vs. linear filtering.

2. Map the loaded texture to the plane of the default scene. This means that the regular main loop should be uncommented. Shift texture look-up to the `intersect_scene` function, where the colours of objects are retrieved. For the look-up, add texture coordinates (`texcoords`) to your hit info struct (`HitInfo`). Use the following as an orthonormal basis of the plane in the default scene:

   ```
   struct Onb {
     tangent: vec3f,
     binormal: vec3f,
     normal: vec3f,
   };
   const plane_onb = Onb(vec3f(-1.0, 0.0, 0.0), vec3f(0.0, 0.0, 1.0), vec3f(0.0, 1.0, 0.0));
   ```

   and change your `intersect_plane` function to take an `Onb` variable as argument instead of the surface normal. Compute texture coordinates when finding an intersection with a plane using the inverse mapping based on the tangent and the binormal of the plane. Use a texture scaling factor of 0.2 to scale the texture coordinates. As with the former colour of the plane, use 90% of the colour for the diffuse reflectance of the plane and 10% for the ambient term. Make an interface that enables the user to switch texturing on/off.

3. Do stratified jitter sampling for anti-aliasing of your ray traced images. Implement an interface in your HTML page for incrementing and decrementing the pixel subdivision level. Use a JavaScript function (`compute_jitters`) to compute an array of vectors from the pixel centre to a jitter sampled position for each sub-pixel. Call this function whenever the pixel subdivision level is changed and store the array in a pre-allocated storage buffer. Modify the main function `main_fs` so that a ray is cast through each sub-pixel. For each sub-pixel, use the jitter vectors in the storage buffer to modify the image space coordinates used for generating the ray. Accumulate the result from each sub-pixel and divide by the number of sub-pixels.

4. Create an interface for dividing the texture scaling factor by a factor from 1 to 10. This corresponds to magnify the texture by a factor of 1 to 10. Make sure your webpage interface supports changing:
   - texturing on/off (when using the base colour shader too)
   - texture address mode
   - texture filtering mode
   - texture scaling factor
   - pixel subdivision level
   - gamma (for gamma correction)

   Inspect the default scene using different settings and describe how texture aliasing is affected by pixel subdivision level and nearest versus linear filtering. The former is most easily explored using the base colour shader. The latter is most easily explored using a magnified texture.

Reading Material

The curriculum for Worksheet 3 is (41 pages)

B Chapter 11. *Texture Mapping*.

B Section 13.4.1. *Antialiasing* (from 4th edition of the text book).