CS 419: Production Rendering Fall 2018
In-class Worksheet 2

Viewing Coordinates and World Coordinates

Suppose we setup a *lookat* style camera with the following parameters:

- Eyepoint is at point **e** in world coordinates
- The viewplane is *d* units away from the eyepoint
- The orthonormal basis for the camera frame is given by the vectors **u**,**v**, and **w**

1. Our viewplane consists of $h_{res} \times v_{res}$ square pixels each with sides of length s. Derive a formula for computing a point (x_v, y_v, z_v) in the center of the pixel at row r and column c. Note that the points coordinates are expressed in view coordinates, not world coordinates.

   $x_v = s(c - h_{res}/2 + 1/2)$
   $y_v = s(r - v_{res}/2 + 1/2)$
   $z_v = -d$

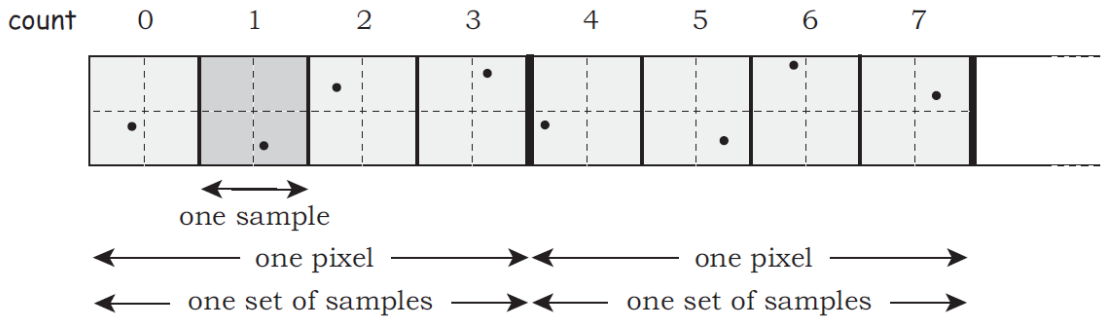
2. What is the ray equation for a ray originating at the eyepoint through the pixel center you computed in part 1? Specify the ray equation in world coordinates.

   With r(t) = o + dt we have
   o = e
   $d = x_v\mathbf{u} + y_v\mathbf{v} - d\mathbf{w}$

A Sampling Framework

In *Ray-Tracing from the Ground Up* the author uses the technique of pre-generating random points in the unit square prior to ray-tracing. These samples are `num_sets` sets of `num_samples` 2D points in [0,1] x [0,1].

These samples are kept in array named samples and a variable count keeps track of how many samples have been used so far.



1. Complete the following code to return a sample point. Note that each pixel will use `num_samples` samples and that the sampler code should jump to a new random set of samples for each new pixel.

```
Sampler::sample_unit_square(void){
if (count % num_samples ==0)

   jump=(rand_int() % num_sets)*num_samples;

return    (samples[jump + count++ % num_samples]);
```

2. Suppose we wish to change our sampling pattern even more by shuffling the indices used in `sample_unit_square`. We create an array called `shuffled_indices` that consists of `num_sets` sets of the integers [0, `num_samples`-1]. Each set is random shuffle of those integers. Complete the following code sample from a random set of points in in samples using the shuffled indices.

```
Sampler::sample_unit_square(void){
if (count % num_samples ==0)

   jump==(rand_int() % num_sets)*num_samples;

          (samples[jump + shuffled_indices[jump + count++ % num_samples]]);
return
```