ray_trace()

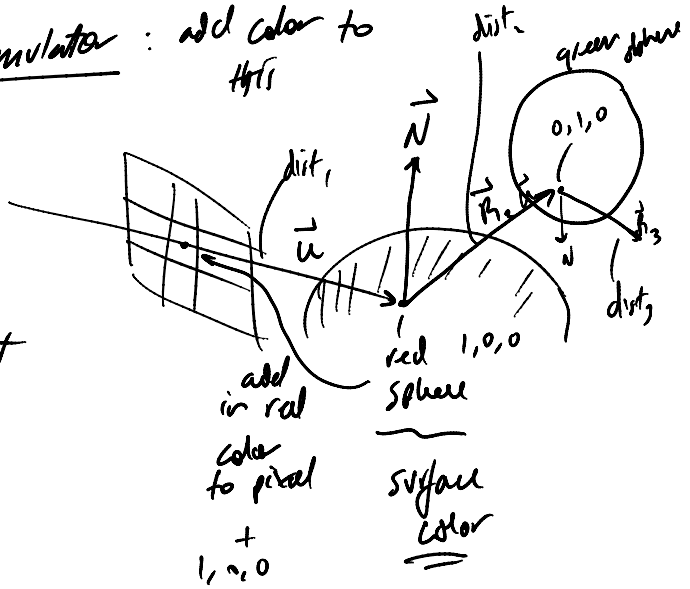
Wednesday, September 29, 2010
9:08 AM

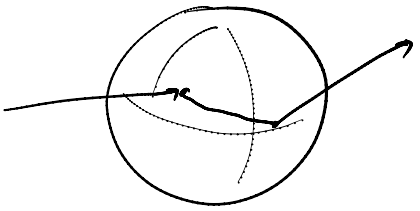
```
ray_trace(model_t *model,   // the world
          vec_t   pos,      // ray pos      } in C++ we'd have a ray object (class)
          vec_t   dir,      // ray dir      }
          drgb_t  color,    // pixel color (output)
          double  raydist,  // distance ray has traveled (accum.)
          object_t *last_hit) // most recent hit
{
    double   dist = 0.0;
    object_t *obj = NULL;
    drgb_t   thiscolor = {0.0, 0.0, 0.0};
    drgb_t   ambient, diffuse, specular;

    if (!(obj = object_find_closest(model->objs, pos, dir, last_hit, &dist)))
        return;

    if (dist > 0) {
        // debugging ...
        raydist += dist;  // accumulate distance traveled by ray
```

Color accumulator : add color to this





check this

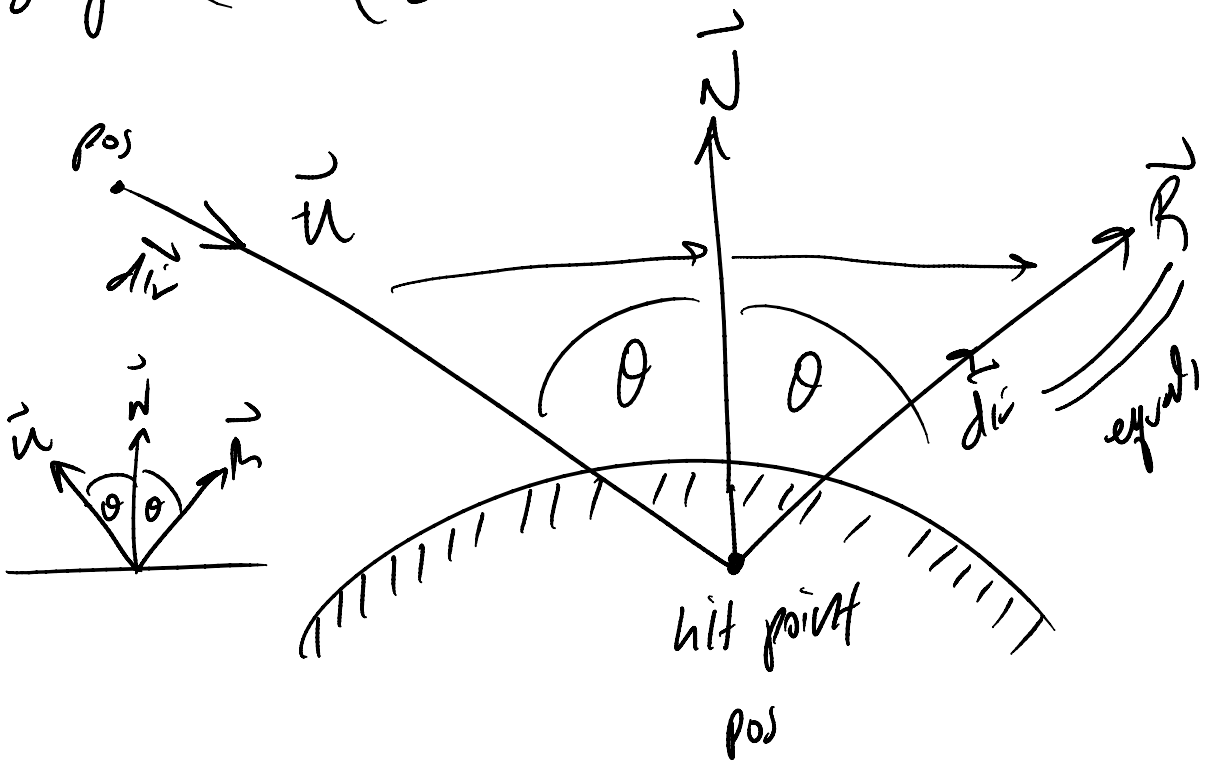
```
// get object material properties
material_getambs(obj->mat, ambient);
                                  ↑
                              this is our
                              surface color

// copy ambient color
pix_copy(ambient, thiscolor);

// scale by ray distance
pix_scale(1.0/raydist, thiscolor, thiscolor);

// add in surface color to (pixel) color
pix_sum(thiscolor, color, color);
    }
}
```

if we were doing recursion, (ray bounces) we'd need to reflect ray at surface (should be in lab 6)



$\vec{R}$ calculated based on "perfect reflector" (mirror)

θ: angle of incidence = angle of reflection

$$\cos\theta = \frac{\vec{u} \cdot \vec{N}}{\quad}$$

$\|\vec{u}\| \quad \|\vec{N}\|$ } can ignore this if $\vec{u}$ & $\vec{N}$ are normalized (both would have length 1)

$$\theta = \cos^{-1}(\vec{u} \cdot \vec{N})$$

$$\vec{R} = 2 * (\vec{u} \cdot \vec{N})\,\vec{N} - \vec{u}$$

↑ scale N by $2(u \cdot N)$

```
vec_reflect(vec_t n, vec_t v, vec_t w)
{
    vec_t u;
```

↑ vector coming in ($\vec{v}$)

↑ vector going out ($\vec{R}$)

```
    vec_scale(2.0 * vec_dot(v,n), n, u);
    vec_diff(u, w, w);
}
```

ray_trace(...)

if(raydist > 1000) return;

vec_reflect(obj_pn, dir, refdir)

ray_trace(..., hitpoint, refdir, ...)