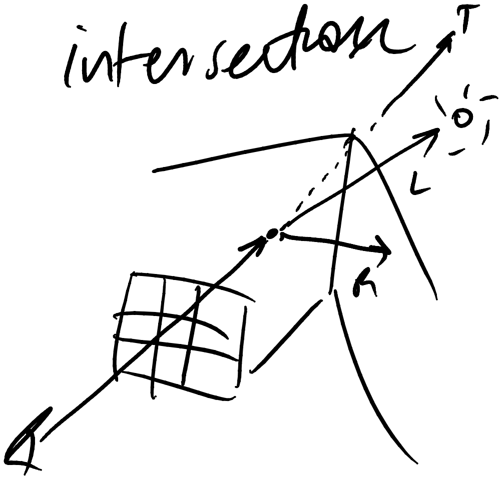
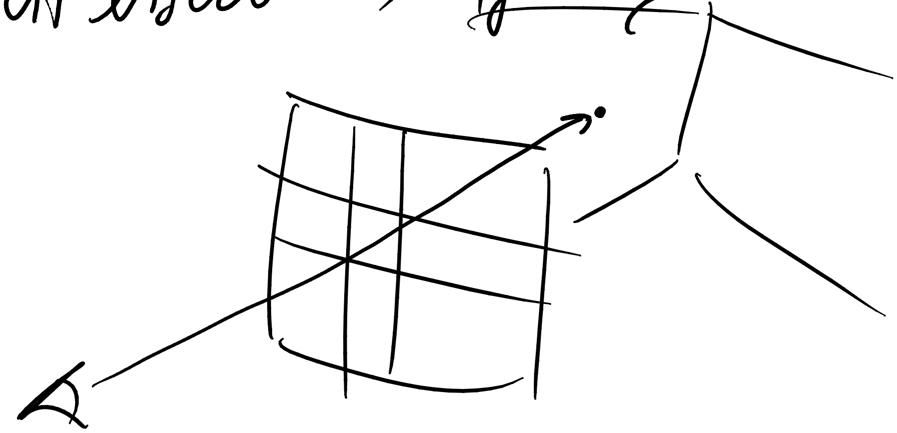
Basic ray tracer (ray caster)

Monday, September 27, 2010
9:16 AM

rays spawned at surface intersection

casting one ray (per pixel) into scene & returning color (r, g, b) at surface of ray/obj intersection, if any





recursive

Monday, September 27, 2010
9:22 AM

Asg 3: all the code that you need should now be available except for the "driver" code in main.c

main:

1. open & parse (read) model.txt as input via argv[1]
   (don't forget to close input file)
2. print model file to stderr
3. output PPM image to stdout

```
fprintf(stdout, "P6 %d %d 255\n", w, h);
```

decimals, or ints not doubles

image header

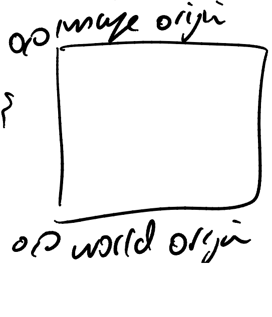
PPM magic number 'P' '6' (six)

```
int w = model->cam->pixel_dim[0];
    h =   "      "      "     [1];

int x,y; // pixel coords

for(y=0; y<h; y++) // Aug. 27 notes
```

```
for(y=h-1; y>=0; y--) {
  for(x=0; x<w; x++) {
```



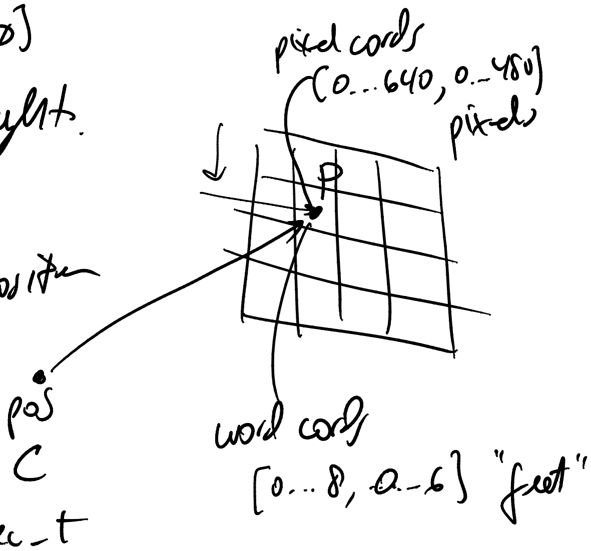
$wx = \underbrace{(double)x / (double)(w-1) * }_{}\boxed{ww};$

conversion of 640x480 image coords into world coords — see Aug. 27 notes & Westall's notes

world width from camera

cam → world_dim[0]

same for world height.



pos : camera position

dir = P − C

vec_t vec_t vec_t

vec_diff (pos, pix, dir)

$\quad\quad\quad\quad$ C $\quad$ P

vec_unit (dir, dir) // normalize dir.

Now we have pos, dir — base, dir for ray.

↓

these get sent to object_find_closest()

drgb_t color; // double color at pixel

irgb_t icolor; // unsigned char color

//zero out color at pixel (for output)

color[0] = 0.0 ; color[1] = 0.0; color[2]=0.0,

ray_trace(model, pos, dir, color, 0.0, NULL);

ray distance

ray_trace fills this in

object last hit by ray

double color is in range [0,1]

need to scale up to [0,255]

pix_scale(255.0, color, color)

//convert to irgb_t

for (i=0; i<3; i++) icolor[i] = (unsigned char) color[i];

0xrrggbb

8 bits

24 bits ≡ 3 bytes

that's what we want print for each pixel

fwrite(icolor, sizeof(irgb_t), 1, stdout);

one of these

} //end for

} //end for

return(0)

}