asg7

Monday, November 29, 2010
8:54 AM

```
                                        rgb_t<double>&
// prototypes
void  ray_trace (model_t&, vec_t&, drgb_t&, double, object_t*);
int   main(...);

int main(int argc, char *argv[])
{
    model_t   model;  // the world
                      { camera
                        list_t<material_t *>
                        list_t<object_t *>
    std::ifstream model_ifs;
                  ( model_ifs("filename"),

    :

    model_ifs.open( argv[1], std::ifstream::in);
    model_ifs >> model;
    model_ifs.close();

    std::cerr << model;

        int     i, x, y;
        int     w = model.get_pixel_w(),
                h = "     "   _h();
        double  wx, wy, wz = 0.0;
        double  ww = model.get_world_w(),
                wh = "     "   _h();

        vec_t   pos, pix, dir;
rgb_t<double>
        drgb_t  color;
        rgb_t   iclr;
```

#ifndef vchar
typedef vector<char> vchar

~~rgb_t~~ iclr;

rgb_t<uchar>

#ifndef uchar
typedef unsigned char uchar
#endif

uchar *imgloc, *img = NULL;

std::cout << "P6␣" << w << "␣" << h << "␣" << 255 << std::endl;

(␣ = space)

for (y = h-1; y >= 0; y--) {
  for (x = 0; x < w; x++) {
    ux = (double)x / (double)(w-1) * wcw;
    wy = " y " (h-1) * wh;

    pos = model.getviewpos();

    pix[0] = wx; pix[1] = wy; pix[2] = wz;

    dir = pix - pos; // or (pix - pos).normal()

    // zero out color
    color[0] = 0.0; color[1] = 0.0; color[2] = 0.0
    (or for(---)
      color = rgb_t<double>(0.0,0.0,0.0);

    ray_trace(model, pos, dir, color, 0.0, NULL);
                                  ray dist   dir; last hit

    color = 255.0 * color;

```
        icolor = color;   (or for(i=0; i<3; i++) ...)
        for(i=0; i<3; i++) std::cout.put(icolor[i]);
    }
}
return 0;
} // like #563
```

ray-trace very similar to #563 just in C++.