Odds and ends: plane_load_attributes, argc, argv, camera_t, pointer to pointer

Friday, September 17, 2010
8:56 AM

```
void vec_xform(mtx_t m, vec_t v1, vec_t v2)
{
    int i;

    for (i=0; i<3; i++) v2[i] = vec_dot(m[i], v1);
}
```

BUG!! compiles & runs, (not a syntactic bug)

but seems if when calling

vec_xform(m2, v4, v4)

we have a semantic bug ⇒ code produces wrong results (1 of 6). Solution to bug left as an exercise for the reader

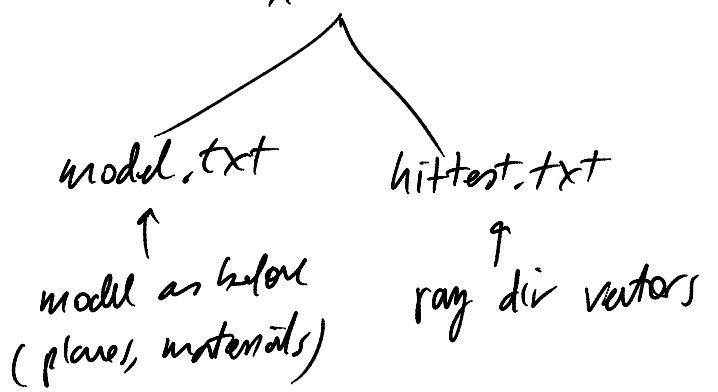
```
void plane_init(FILE *in, list_t *objs,
                list_t * *mats, int attrmax)
{
    object_t    *obj;

    // init object
    object_init(in, objs, mats);
    assert((obj = (object_t *)list_get_data(objs)) != NULL);
    assert(obj->cookie == OBJ_COOKIE);
    // load attributes
    plane_load_attributes(in, obj);
}
void plane_load_attributes(FILE *in, object_t *obj)
{
    plane_t     *pln;
    char        c, attrname[NAME_LEN];
    int         count;

    // alloc & init plane object (setting up pln->)
    ...
```

add this prototype to plane.h

moved from plane_init

Friday, September 17, 2010
9:19 AM

For lab 5, use 2 arguments in main program




```
int main(int argc, char *argv[])
{
    FILE *model_file = NULL, *hits_file = NULL;

    if (argc != 3) {
        fprintf(stderr, "Usage: %s <model_file> <hit_file>\n", argv[0]);
        exit(1);
    }

    if ((model_file = fopen(argv[1], "r")) == NULL) {
        fprintf(stderr, "Error opening input file: %s\n", argv[1]);
        exit(1);
    }

    model = model_init(model_file);

    fclose(model_file); // DON'T FORGET THIS
```

name of file

read mode

taken place of stdin but works the same way

---

same story for hits file

```
if((hits_file = fopen(argv[2],"r")) == NULL) {
                ;
}
while (fscanf(hits_file, "%lf %lf %lf", &dir[0], &dir[1], &dir[2]) == 3) {
                ;

}
fclose(hits_file);
return(0);
} // end of main()
```

Friday, September 17, 2010
9:30 AM

Camera model

{ camera info
  position
  screen dim (640 480)
  world dim (8 6)
  name
  cookie

to be stored in our model obj
read from model.txt

Makefile:

```
OBJS = \
pixel.o \
camera.o \
material.o \
object.o \
plane.o \
model.o
```

---

camera.h

```
#define CAM_COOKIE 49495923

typedef struct camera_type
{
    int     cookie;
    char    name[NAME_LEN];
```

```
    int     pixel_dim[2];    // screen coords (image w x h)
    double  world_dim[2];    // screen coords in 3-space
    vec_t   view_point;      // cam pos in 3-space
}
```

pointer to pointer !!

```
void   camera_init(FILE*, camera_t**, int);
void   camera_local_attributes(FILE*, camera_t*);
void   camera_print(camera_t*, FILE*);
char*  camera_getname(camera_t*);
```

---

model.h

```
typedef struct model_type
{
    camera_t *cam;
    list_t   *mats;
    list_t   *objs;
} model_t;
    :
```

Friday, September 17, 2010
9:41 AM

camera.c

pointer to pointer

```
void camera_init(FILE *in, camera_t **cam, int attr_mask)
{
  // malloc camera_t struct, use memset() to zero out
  *cam = (camera_t *) malloc(sizeof(camera_t));
  memset((void *) *cam, 0, sizeof(camera_t));
  (*cam)->cookie = CAM_COOKIE;
```

↑ dereference pointer

```
  camera_load_attributes(in, *cam);
}
```

```
main()
{
    int x = 3, y = 4;
    swap(x, y);
    printf("x = %d, y = %d\n", x, y);
}
```

x [3]
y [4]

```
void swap(int a, int b)
{
```

a [3] 4

```
void swap(int a, int b)
{
    int tmp = a;
    a = b;
    b = tmp;
}

{
    int x = 3, y = 4;
    swap(&x, &y);
    printf(...)
}

int swap(int *a, int *b)
{
    int tmp = *a;
    *a = *b;
    *b = tmp;
}
```