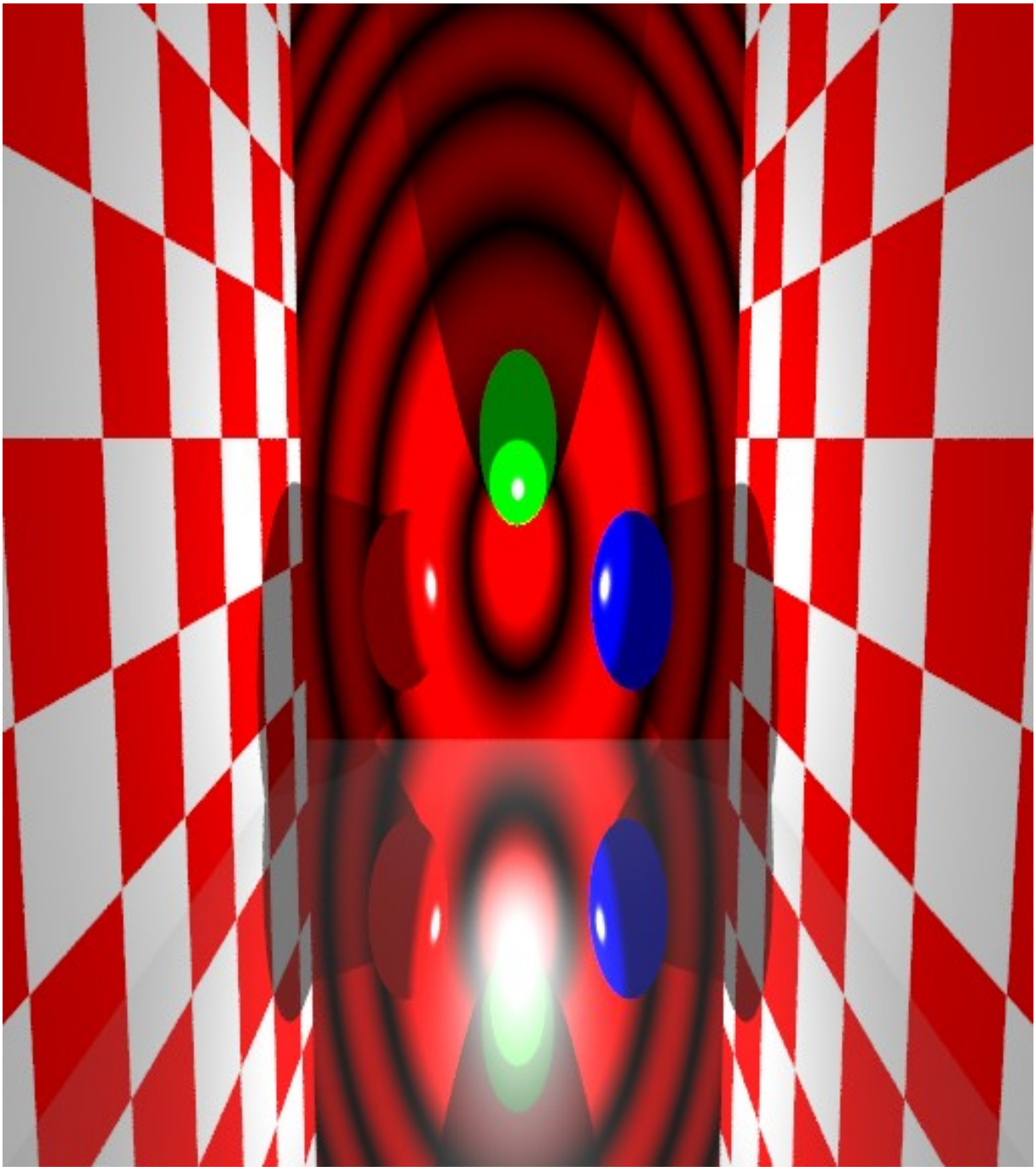
CSE 3431- Assignment 4
Ray Tracing Assignment



Matthew Conte – 207333230 cs243082
Victoria Kirnos – 206009310 cs231167

December 3, 2007

CSE 3431 A4 Instruction Manual

Begin the program by typing "art" and a command prompt will be presented awaiting your commands.

COMMANDS:

```
scale sx sy sz
rotate axis angle
translate tx ty tz
shear axis1 axis2 amount
initTM
pushTM
popTM

camera fromX fromY fromZ  atX atY atZ [upX upY upZ]
perspective width height distance
lens focusDistance lensRadius

light x y z intensity [radius]
pop_light

material red green blue Ka Kd Ks n [Kr Kt indexOfRefraction [textureID]]
background red green blue

sphere
plane
cube

trace x y fileName [numSamples]

include filename

quit
```

Transformations and Objects

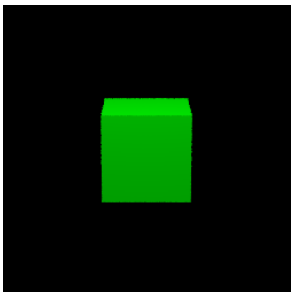
The first seven commands deal with the current transformation matrix (CTM). The axii in rotate and shear can be either *x, y* or *z*, indicating the x-axis, y-axis and z-axis. In the shear command, you shear along axis1 as a function of axis2. Rotations are in degrees counter-clockwise.

There are three primitives: *sphere* (a unit radius sphere centered at the origin), *plane* (y=0), and *cube* (-1 <= x, y, z <= 1). Every time a primitive is specified, it is placed in a linked-list of primitives, with each node including the CTM and the current surface description as well. This linked-list stores the objects in the scene.

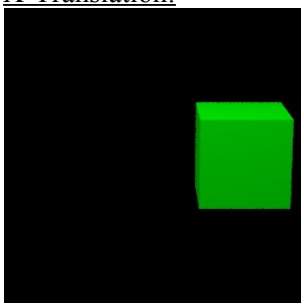
The following examples are objects about the origin with the camera at co-ordinates (0, 5, 20).

Translation: Green cube before and after translation (by 1)

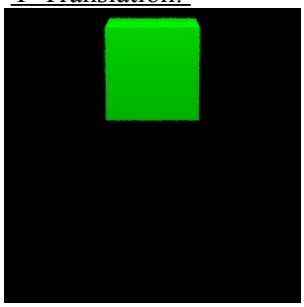
Before:



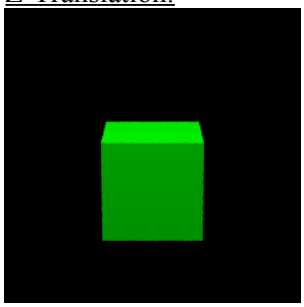
X-Translation:



Y-Translation:

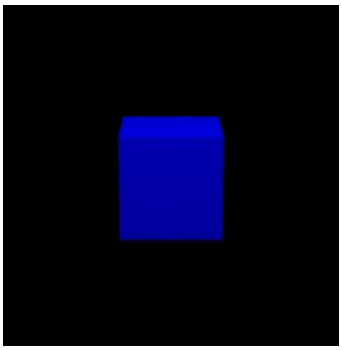


Z-Translation:

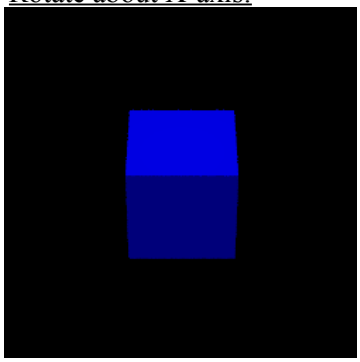


Rotate: Blue cube before and after rotation (25 degrees)

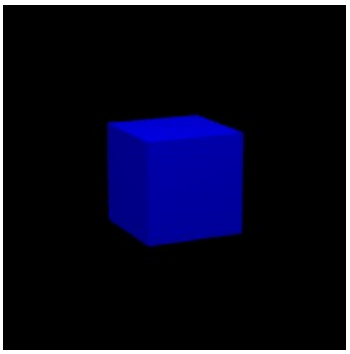
Before:



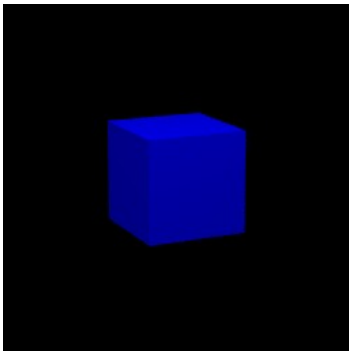
Rotate about X-axis:



Rotate about Y-axis:

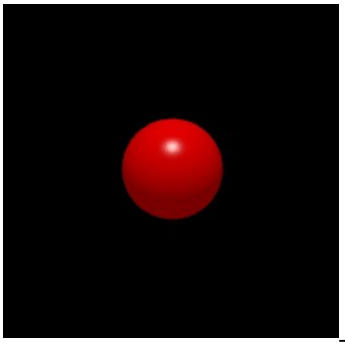


Rotate about Z-axis

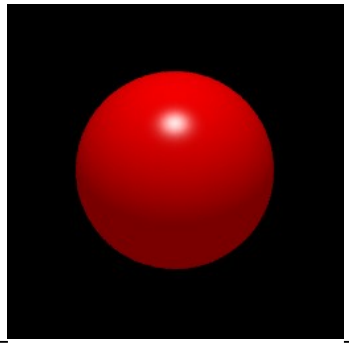


Scale: Red Sphere before and after scaling

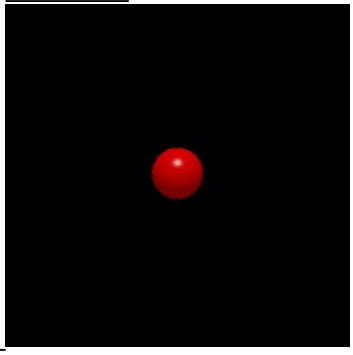
Before:



Scale 2x

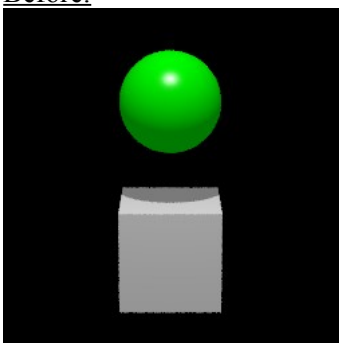


Scale 1/2x



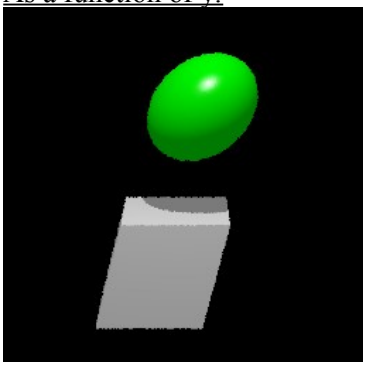
Shear: green sphere and white cube (by a factor of 0.25)

Before:

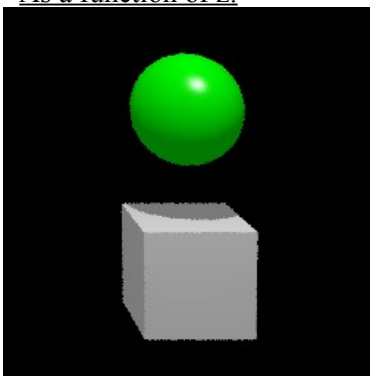


Shear along x-axis:

As a function of y:

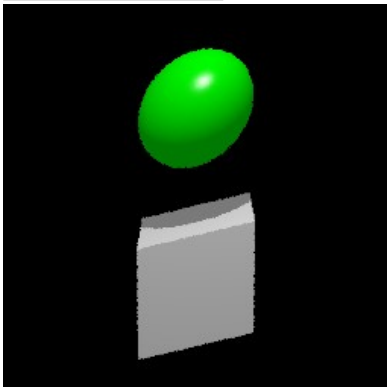


As a function of z:

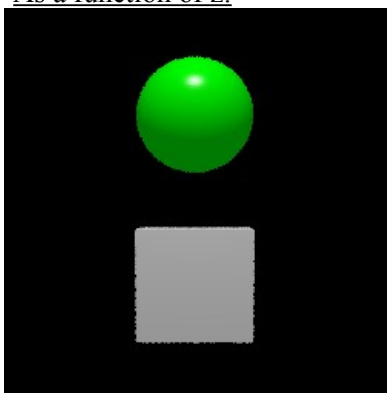


Shear along y-axis

As a function of x:

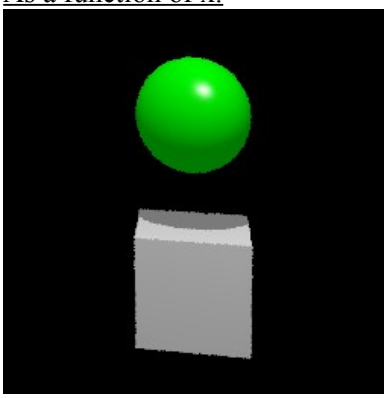


As a function of z:

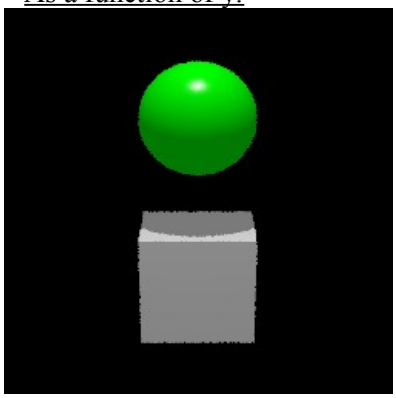


Shear along z-axis

As a function of x:



As a function of y:



View Volume

Camera and *perspective* define view volume. The *camera* command deals with the position and orientation of the camera. The startup position is (0, 0, 10) looking at the origin. The optional up vector is defaulted to (0,1,0). The *perspective* command allows you to specify what portion of the projection plane appears on the screen. The default is 1, 1, 1 for *perspective* if none is specified.

Light

The *light* command puts a light source at the indicated position with the indicated intensity, and optionally, radius. You can have multiple lights in a scene. There is no light if none is specified.

Material

The *material* command modifies the "current" material description. The default is (1, 1, 1), 0.2, 0.6, 0.7, 50.0, 0.0, 0.0, 1.0 0.

The respective fields are for material, aside from RGB:

Ka = coefficient of ambience
Kd = coefficient of diffusion
Ks = coefficient of specularity
Kr = coefficient of reflectivity
Kt = coefficient of transivity.
(above are all greater than 0 and less than 1)
n = shininess
indexOfRefraction = index of refraction
textureID = ID of which texture mapping to use (0 = none, 1 = checkerboard, 2 = Zone Plate)

Background

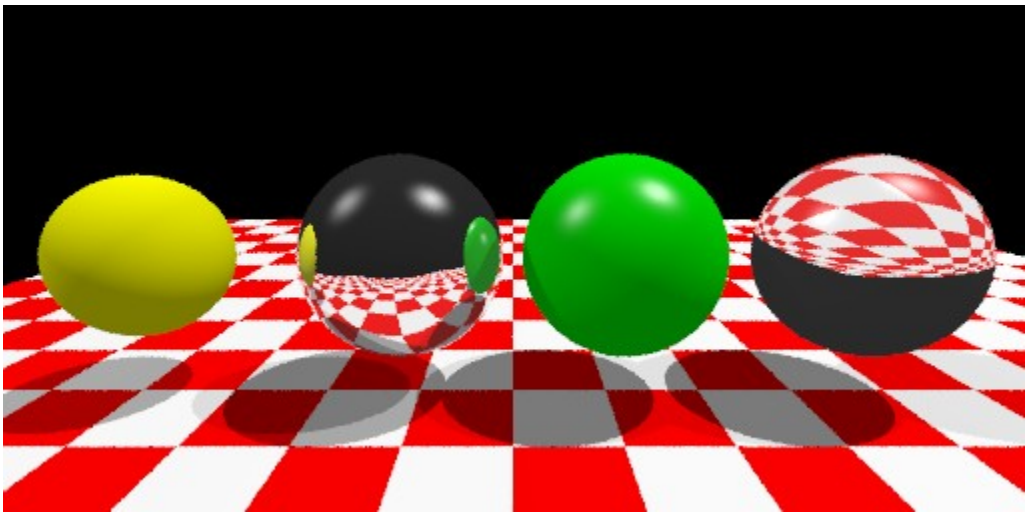
The *background* command sets the background color. The default is (0, 0, 0) (black).

Here's an example displaying various material spheres on top of a cube of a checkerboard texture mapping with two light sources. Using the following sequence of commands:

```
% 3 spheres + cube and 2 lights over cube

% generate checkered cube
translate 1 0 -2.5
scale 10 10 1
material 1 0 0  .2 .8 0 1   0 0 1   1
cube

% yellow diffuse sphere
initTM
translate -3.7 0 0
rotate x -160
scale .8 .8 .8
material 1 1 0 .2 .8 0 1
sphere

% mirrored sphere
initTM
translate -1.1 0 0
material 1 1 1 .01 .01 .9 50 0.9 0 1
sphere

% green specular sphere
initTM
translate 1.1 0 0
material 0 1 0 .2 .6 .8 50
sphere

% glass ball
initTM
translate 3.5 0 0
material 1 1 1 .01 .01 .9 50   0.0 0.9 1.5
sphere

light 50 0 100 10000
light -100 0 100 7000

camera 0 -5 1.3  0 0 0 0 0 1
perspective 2 1 1


trace 512 256 MYPIC2.tiff 4
```

From Left to Right we have a:

Diffuse Yellow Sphere, Mirrored Sphere, Green Specular Sphere, Glass Sphere



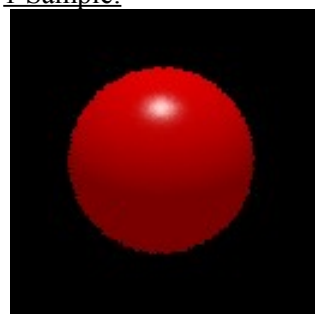
Trace

The command *trace* generates a raster image of the current scene and stores it in *fileName*. It stores it in PPM file format (basically, a header and the raw pixels, 3 bytes at a time). The x and y values indicate the resolution of the raster image. Good values for x and y are 128, 256 or 512. An optional *numSamples* indicates how many samples to shoot per pixel. Default of numSamples is 1 if none is specified.

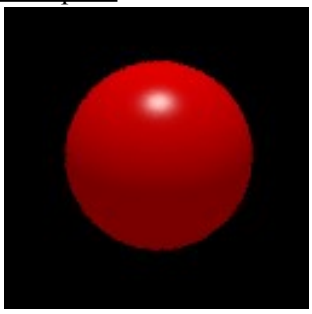
This particular program uses depth bounded ray tracing of length 5 and randomized sampling about any location within the designated pixel during rasterization.

Here's an instance of the same image under different number of samples.

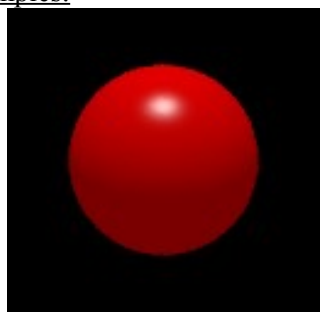
1 Sample:



4 Samples:



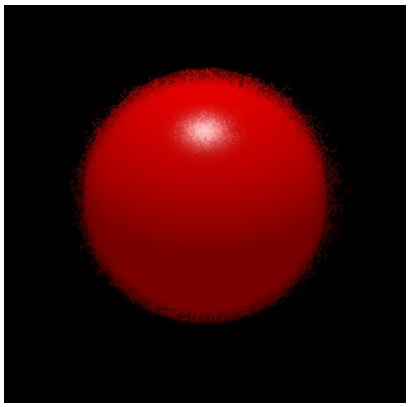
8 Samples:



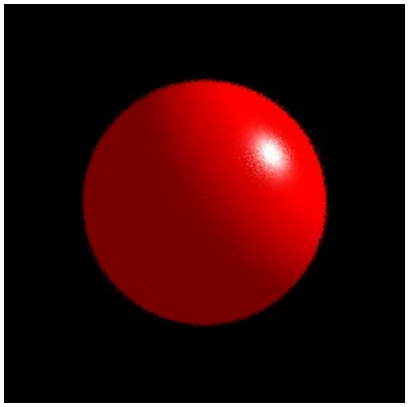
Lens

The *lens* command allows you to specify the focusing distance and the radius of the lens. The *lens* function takes as a parameter one ray. This ray is shot from six random spots on the lens to intersect the object. These six rays are stored in an array and returned to be later used for computing the color of the sample.

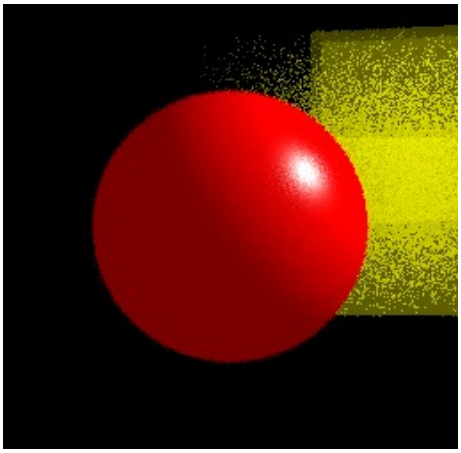
Lens with one object out of focus:



Lens with one object in focus:



Lens with two objects the red sphere in focus and the yellow cube out of focus:



By default, *lensRadius* = 0.0, and *focusDistance* = 1.0 if none is specified.

Include

The command *include* allows you to read in and interpret commands from a file.

For instance, given that you had a file named example.script that contains this:

```
material 1 0 0  .2 .6 .7 50
sphere
light 0 10 5 100
camera 0 0 10 0 0 0
perspective 1 1 3
trace 256 256 MYPIC.tiff 4
```

the command:
include example.script
will store the 4 sample sphere from **Trace** into MYPIC.tiff.