Comment and self-evaluation:
This lab is similar to lab 8. We have to change the Makefile to lead to new location in /pool/class…
I’ve done all the lab successfully. Therefore, I would evaluate myself 20/20 for the completed lab work.

Part 1: Check intersection Point between Ray and Plane.
Code:
//intersect.cpp
...
using namespace std;

bool intersect_ray_plane(const Plane &plane, const Ray& ray, double &t)
{
    if ( ray.origin == plane.p0 ) {
        t = 0;
        return true;
    }
    Vec3 op = ray.origin - plane.p0;
    double opn = dot ( op, plane.n );
    double dn = dot ( ray.direction, plane.n );

    if ( fabs ( dn ) < 0.000001 )
        return false;
    t = -opn / dn;
    if ( t < 0.0 )
        return false;
    return true;
}
//test0.cpp

bool intersect_ray_plane(const Plane &plane, const Ray &ray, double &t);

int main()
{
    Ray r;

    Plane plane ( Point3(3, 4, 0), Vec3(0, -1, 1)); // change the Plane here
    r = Ray ( Point3(2, 1, 0), Vec3(-1, 2, 0)); // change the Ray here

    // Plane plane ( Point3(3, 3, 3), Vec3(0, 1, 1)); // change the Plane here
    // r = Ray ( Point3(2, 2, 2), Vec3(-1, -1, -1)); // change the Ray here

    cout << r.direction << endl;

    double t;

    bool it = intersect_ray_plane ( plane, r, t );

    if ( !it )
        cout << "Miss!" << endl;
    else {
        cout << "Hit!" << endl;
        cout << "Intersection point is: " << endl;
        cout << "t = " << t;
        Point3 hit_point;
        // hit_point = ( r.direction * t ) + r.origin;
        hit_point = r.getPoint( t );
        cout << " hit point = " << hit_point << endl;
    }

    return 1;
}

...
Running program:
With the set of
//Plane plane ( Point3(3, 3, 3), Vec3 (0, 1, 1)); // change the Plane here
// r = Ray ( Point3 (2, 2, 2), Vec3 (-1, -1, -1)); // change the Ray here

With original set (left side) and on the right side is set of
Plane plane ( Point3(3, 4, 0), Vec3 (0, -1, 1)); // change the Plane here
r = Ray ( Point3 (2, 1, 0), Vec3 (-1, 2, 0)); // change the Ray here
Sketch on paper for the picture on right site:
\[ t = (-1, 2, 0). \]

Because the Ray and the Plane Normal vector are going 2 different way, therefore they can’t intersect each other.
Part 2: Change image and compile textures again:

First, edit a little in Makefile as lab8.

Code:

```cpp
//tracecone1.cpp

void SetupRayTraceWorld()
{
    // Initialize Pixel Array and main viewpoint
    SetUpMainView();  // in RayTraceData.cpp, declaring variable 'MainView' and creating new instance also
    *MainView
        // CameraView class defined in Graphics/CameraView.h
        // pixels = new PixelArray(640,480);  // Array of pixels
    pixels = new PixelArray(600,600);  // Array of pixels
    MainView->SetScreenPixelSize( *pixels );

    float color[] = {0.1, 0.8, 0.1};
    SceneDescription scene;
    scene.SetBackGroundColor( color );  // at RaytraceMgr/SceneDescription.h
    // Initialize Array of Materials
    //SetUpMaterials();

    // Initialize Array of Lights
    SetUpLights();
    SetUpMaterials();

    // Initialize array of viewable objects
    // SetUpViewableObjects();
    const int MAX_OBJECTS = 8;
    ViewableCone *vcone = new ViewableCone;
    vcone->SetCenterAxis ( 0.0, 1.0, 0.0 );
    vcone->SetHeight ( 5.0 );
    //Texture
    TextureCheckered* txchecked = new TextureCheckered();
    txchecked->SetMaterial1(MatArray[3]);
    txchecked->SetWidths(0.125,0.125);
    vcone->TextureMap(txchecked);
    ViewObj[0] = vcone;

    ViewableSphere* vs = new ViewableSphere;
    vs->SetCenter(-6.5,1.5, 0);
    vs->SetRadius(4.0);

    // First, edit a little in Makefile as lab8.
    // in RayTraceData.cpp, declaring variable 'MainView' and creating new instance also
    *MainView
```
vs->SetMaterial(MatArray[1]);

TextureRgbImage *tRgb = new TextureRgbImage ( "th.bmp" ); // change picture here.
vs->TextureMap( tRgb );
ViewObj[1] = vs;

NumObjects = 2;

assert( NumObjects <= MAX_OBJECTS );
}

Running program:

Description:
Right side is original picture.
Left side is the picture on the sphere when press ‘g’