

Project 3B: Triangles, Shadows and Reflection

Due Nov 11 by 11:59pm **Points** 100 **Submitting** a file upload
Available Oct 24 at 12:45am - Nov 14 at 11:59pm

This assignment was locked Nov 14 at 11:59pm.

Objective

In this project you will expand the ray tracer that you wrote for Project 3A. The major features that you will add are triangles, shadows and reflection. Adding triangles will allow you to create scenes that have more geometric richness. The shading function will be enhanced to have an ambient and a specular term to give more variety to your surfaces. You will also modify your shading algorithm to cast rays at light sources, so that you can introduce shadows into your scene. Finally, you will include recursion to create reflections.

Project Description


You have the following goals for the second part of this project:

1. Create triangles
2. Add ambient and specular terms for shading
3. Enable shadow creation by light sources
4. Allow the creation of surfaces that include reflections

Your results should appear very similar to the examples shown below in Results.

Provided Files

You should use your code from Project 3A as a starting point for part B of this project. You will need to add the new commands to the interpreter, such as those that are used to create triangles.

For this second part of the assignment, your code should work for all of the 10 provided .cli files from part A. In addition, there is one more scene description (.cli) file that has been add for part B. Here is the **NEW SCENE** (<https://gatech.instructure.com/courses/264908/files/36299935?wrap=1>)  (https://gatech.instructure.com/courses/264908/files/36299935/download?download_frd=1) . You should modify your code so that when the user presses the minus key (-), it reads in this new scene file. The new scene file should be placed in the "data" subdirectory along with the other .cli files.

As with part A, you should modify your code in any way you see fit, and comment your code (include

placing your name in the header). Your code should be written in the Python version of Processing. Visit “py.processing.org/reference/” for more information on built in functions. You are **NOT** allowed to use most of the built in processing/openGL functions in this project such as those that draw polygons and spheres. We are not using rasterization for this project! You are, however, encouraged to use the PVector data type and the standard math functions. When in doubt about what library functions are okay to use, please ask the instructor or a TA.

Scene Description Language

Each scene is described by calling several functions that set up and render the scene. You have already created many of these functions for your answer to Part A of the ray tracing assignment. Feel free to define any objects, data structures, and global variables that you want to accomplish this project.

Below are the new functions that you will need to implement for this assignment:

begin

Indicates the start of a triangle description.

end

Indicates the end of a triangle description.

vertex x y z

Specifies the 3D coordinates for one vertex of a triangle. There will always be three such vertex commands between each **begin** / **end** keyword pair.

Although the **surface** command was used in Part A, you only used the diffuse terms. Now you will also make use of the other parameters for more sophisticated shading:

surface dr dg db ar ag ab sr sg sb spec_power k_refl

Specifies a new surface material with diffuse color (**dr, dg, db**). For part B of the ray tracer, you will use the ambient color (**ar, ag, ab**) to create ambient shading effects. You will also use the specular color (**sr, sg, sb**) and the **spec_power** exponent to create highlights on surfaces. Finally, you will shoot reflected rays and incorporate their color if the value of **k_refl** is non-zero.

Commands from Project 3A

In addition to the above commands, your project should still implement the commands that you used for Project 3A (ray tracing spheres). Note that some of the effects of these commands will be enhanced, including adding ambient light, specular highlights, and casting shadows. Here are those

commands from Part A:

background r g b

Sets the background color. If a ray misses all the objects in the scene, the pixel should be given this color.

fov angle

Specifies the field of view (in degrees) for perspective projection.

eye ex ey ez

Set the position of the eye to (ex, ey, ez).

uvw ux uy uz vx vy vz wx wy wz

Specifies the (**u**, **v**, **w**) coordinate frame that helps to define the eye rays.

light x y z r g b

Create a point light source at position (x,y,z) and its color (r, g, b). Your code should allow any number of light sources. For the second part of this assignment, you will cause these lights to cast shadows.

surface dr dg db ar ag ab sr sg sb spec_power k_refl

Specifies a new surface material with diffuse color (**dr**, **dg**, **db**). Ignore the other terms for now, since we will not use them until the second part of this assignment.

sphere radius x y z

Specifies the creation of a sphere with its center at (x, y, z) and with a given radius. The material properties of the surface are given by the last **surface** command that was executed prior to this command.

render

Ray-traces the scene and displays the image in the drawing window.

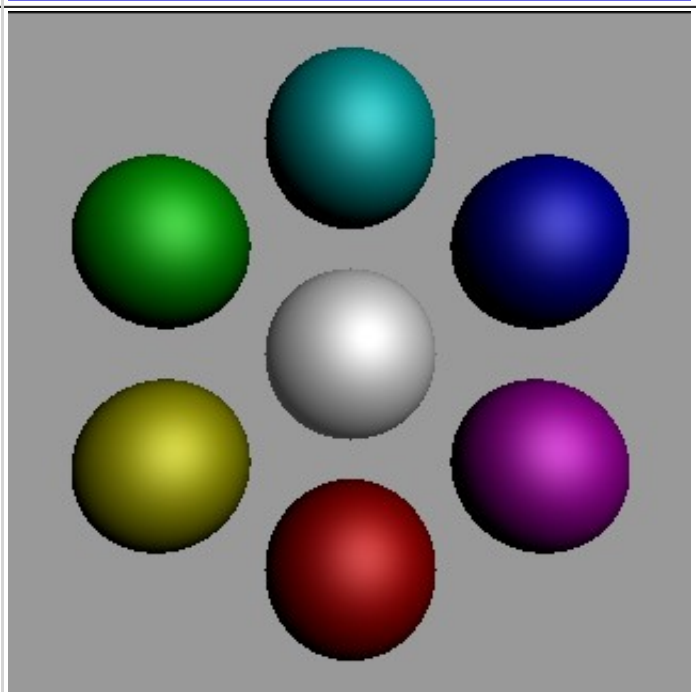
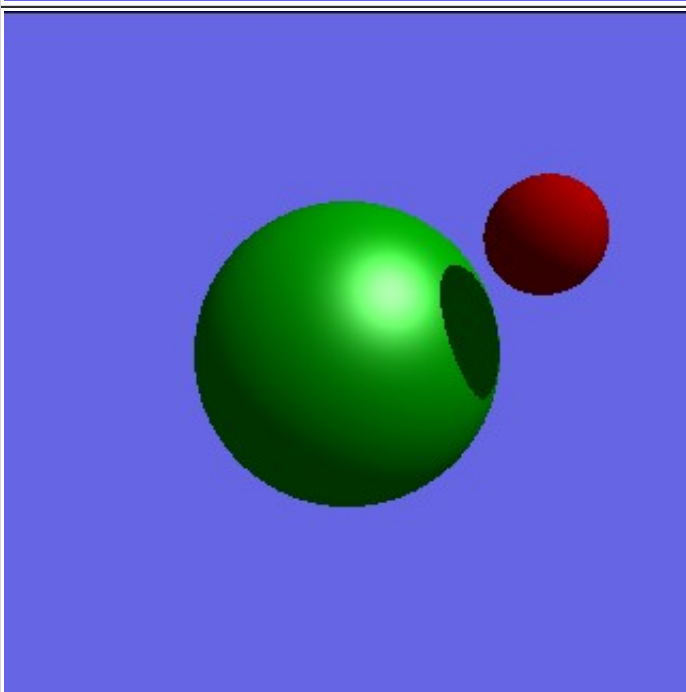
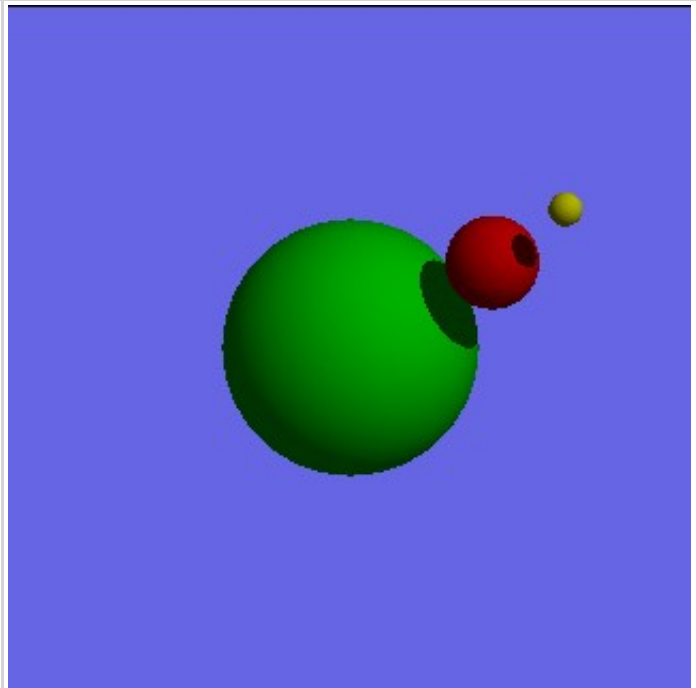
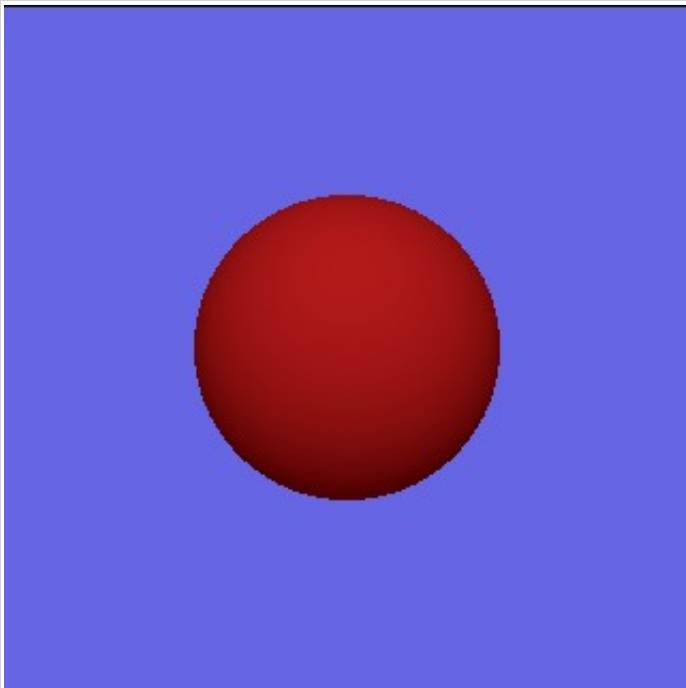
Note on color specification: Each of the red, green, and blue components for the above commands are floating point values in the range of 0.0 to 1.0.

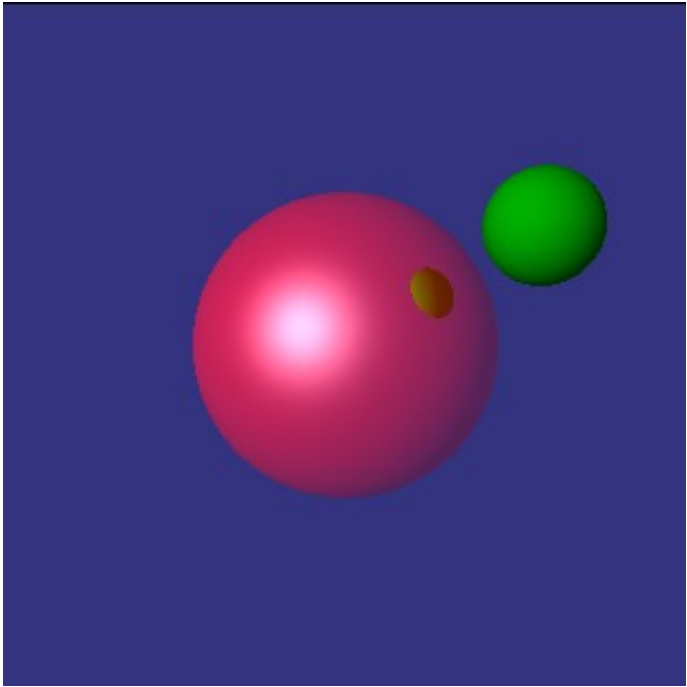
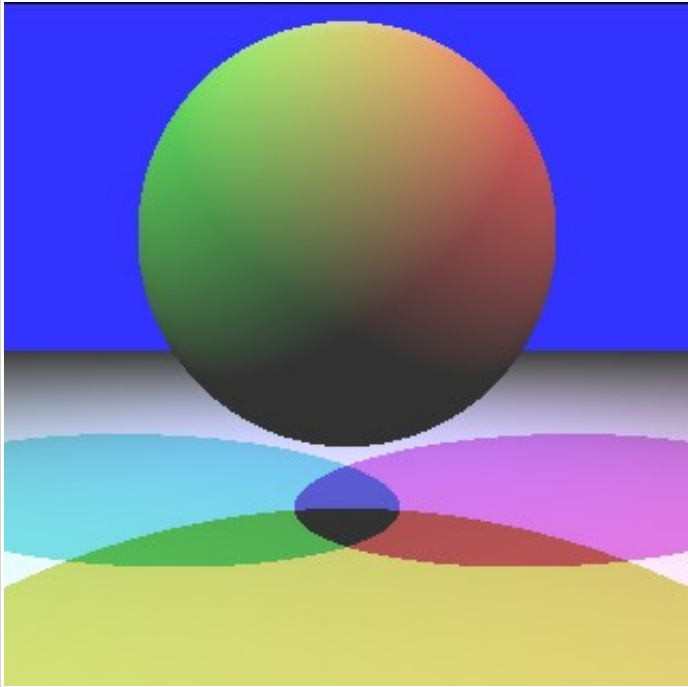
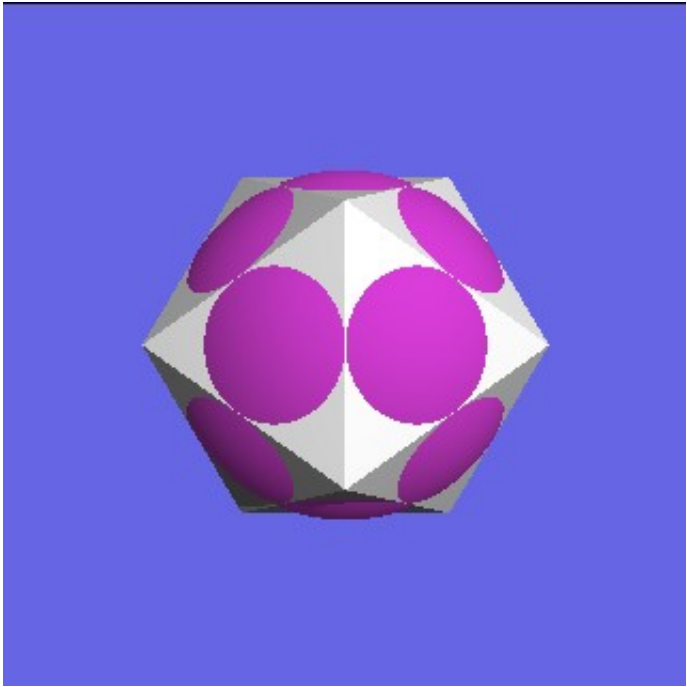
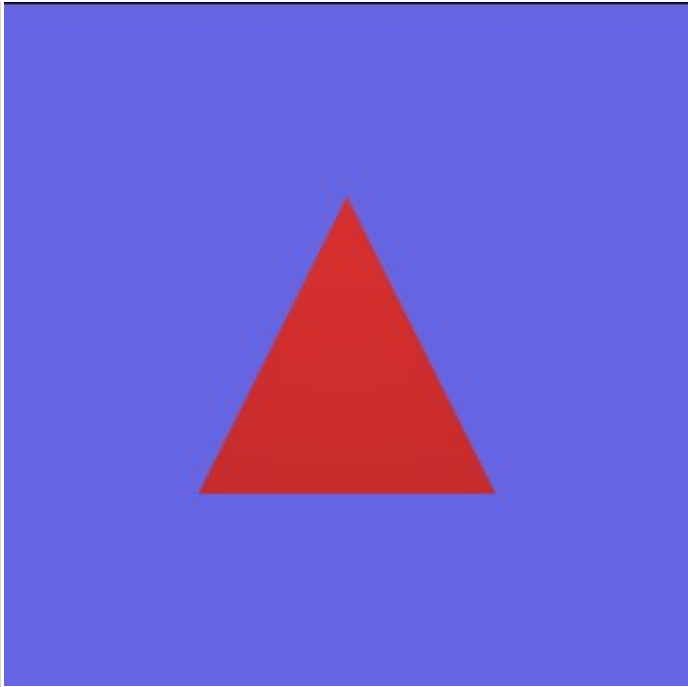
Results

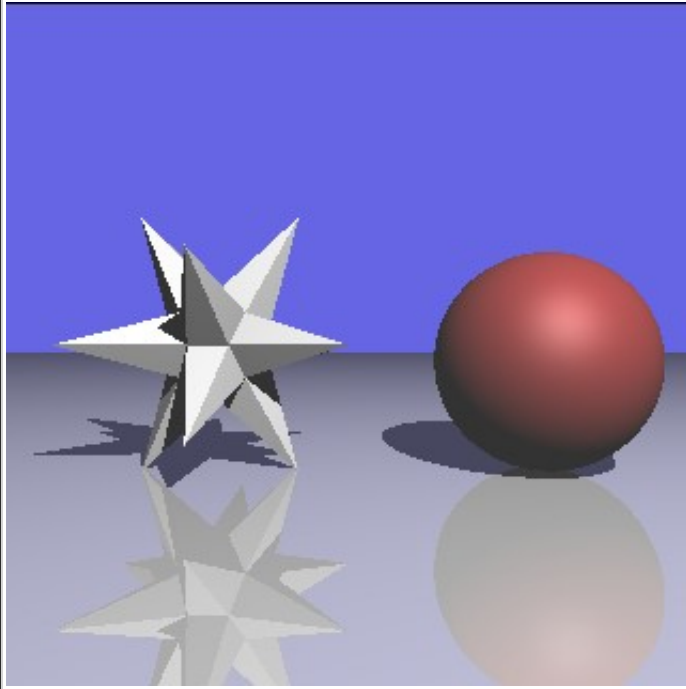
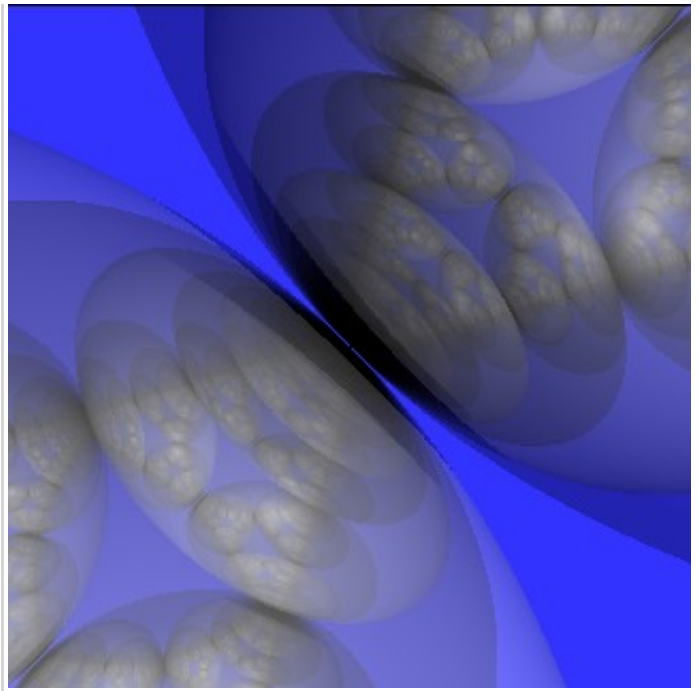
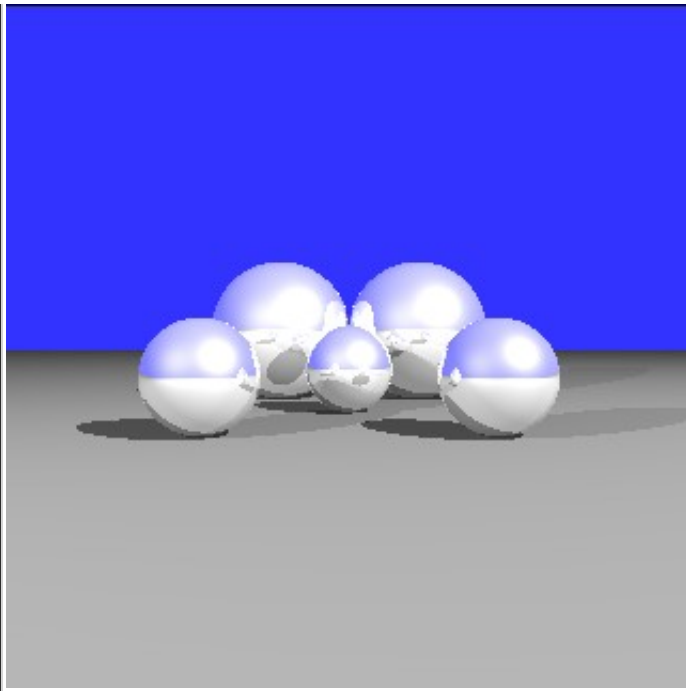
Below are the images that your ray tracer should produce for the eleven different example scenes.

Your code should already be able to read in the first 10 scenes by pressing keys 1-9 and 0. Don't forget to modify your code so that it can also read in the new eleventh scene when you press the minus key (-).

Notice that the objects in the first four images look brighter than they did in part A of the assignment. This is because we are now making use of the ambient and specular terms for surface materials.







Authorship Rules

The code that you turn in entirely your own. You are allowed to talk to other members of the class and to the instructor and the TA's about general implementation of the assignment. It is also fine to seek the help of others for general Processing/Python programming questions. You may not, however, use code that anyone other than yourself has written. The only exception is that you should use the example source code that we provide for this project, and any code that you wrote from part A. Code that is explicitly **not** allowed includes code taken from the Web, github, from books, from the py.processing.org web site, from other assignments, from other students or from any source other

than yourself. You should not show your code to other students. Feel free to seek the help of the instructor and the TA's for suggestions about debugging your code.

Submission

In order to run the source code, it must be in a folder named after the main .pyde file. You should also make sure the eleven .cli files are in the "data" subdirectory within your project folder. When submitting this assignment, leave it in the project folder, zip it and submit via Canvas. Please do not use rar or tar to compress your project folder.