Competencies

In this project, you will demonstrate your mastery of the following competencies:

- Generate accurate representations of three-dimensional objects using application programming interface libraries and computer graphics development best practices
- Create interactive graphics applications that respond to input devices
- Develop a fully formed three-dimensional project that meets project requirements

Scenario

Recall that you work as a C++ and OpenGL 3D graphics developer for Triangle & Cube Studios. This company designs 3D worlds for clients and customizes them based on the varied needs presented by each particular client.

In this professional landscape, the demand for computational graphics and visualizations is continually growing. Your clients may come from the game industry looking for graphics and animations, the healthcare industry for medical visualizations, the entertainment industry for computer-generated imagery (CGI) and visual effects, business industries for 3D printing to create physical objects for applied real-world problem solving, and much more. When you are assigned one of these types of projects, you become responsible for writing code in OpenGL to create objects, apply texture, apply light, render, and control virtual environments relative to a virtual camera.

Your current project with Triangle & Cube Studios is to recreate a 3D version of a 2D image that you have been given by a client. Your client will later be 3D printing this to use as a preliminary concept for their business, so they only need you to create a simple approximation using a few basic shapes.

Directions

Using the image you selected in a previous milestone, you will be creating 3D objects that represent the components and layout of that image. Although you have already begun to complete some of this work in your other milestones, during this project you will be refining and adding to your earlier submissions before bringing everything together. Note that you will be working on your **3D scene** in Visual Studio but will also submit a written **design decisions** document discussing your approach throughout the process.

3D Objects

1. **Create low-polygon 3D representations of real-world objects**. Make sure you have at least *four* completed objects in your 3D scene. At least one of the objects you create should be made using two or more primitive shapes. Note that the object you completed in a previous milestone can count as one of your four. Utilize organized geometry and ensure that polygons (triangles) on each 3D model are well spaced and connected. To minimize complexity and save 3D modeling time, the polygon count for your objects should not exceed 1,000 triangles. As you work, remember to think in terms of simple shapes and ask yourself what primitive 3D shapes go into making up each object in your scene. *Four* of the following primitive shapes must appear at least once in your creation:
    - Cube
    - Cylinder
    - Plane
    - Pyramid
    - Sphere
    - Torus

2. **Apply accurately projected textures to a 3D model**. You must select *two* objects to texture. Note that you should have already textured one object in a previous milestone. If you use that object here, it will count as one of your two. As you work, the textures you select should be royalty-free images with resolutions of 1024 x 1024 pixels or higher. Please refer to the Sourcing Textures Tutorial, linked in the Supporting Materials section, for guidance on how to locate images that can be used for textures.

3. **Apply lighting to create a polished visualization of 3D models**. You must include a minimum of *two* light sources, and at least

one of them should be colored. Note that the light you worked on in a previous milestone counts as one of your two lights. The light sources you create will need to capture all of the objects in the 3D world you are building, meaning they should be positioned at locations that do not cause parts of the objects to appear dark when moving the camera around them. While we recommend that you include a point light for one of your two lights, you may implement a directional light or spotlight if you choose. As you generate lighting, make sure that any lights are designed in a way that helps curate a final polished presentation. You will need to properly implement all components of the Phong shading model, including the following:

- Ambient
- Diffuse
- Specular

4. **Place objects appropriately, using the X, Y, and Z coordinates, relative to one another in the 3D world**. As you work, be sure to match the photograph you selected as closely as possible by placing the objects in their proper locations. Note that when you first import code for the objects you created in previous weeks, the objects may overlap, as it is likely that they were all initially placed at 0, 0, 0.

Navigation

5. **Apply horizontal, vertical, and depth camera navigation around the 3D scene**. The camera will be traversing the X, Y, and Z axes, and you should ensure it can capture all of the objects in your 3D scene. In a previous milestone, you already created some of this code. It is recommended that you use the code you have already created and then increase the radius of the camera's orbit so it will correctly encompass all of the objects in the world you are building. You may find it easiest to add each object separately and then adjust the orbit radius or position of the camera each time. As you work, we recommend you use the following input devices:

- WASD keys: These keys should be used to control the forward, backward, left, and right motion.
- QE keys: These keys should be used to control the upward and downward movement.

6. **Apply nuanced camera controls to effectively view the 3D objects in the application**. This should allow the orientation of the camera to change even though its location has not moved. You should focus first on pitch and yaw, but careful changes can be made to roll, keeping in mind that you may want the upward direction to stay in the same location. As you work, you will also want to code for adjustments in the speed of the movement so a user will have more control over how they explore the objects in the scene. We recommend you use the following input devices:

- Mouse cursor: This should be used to change the orientation of the camera so it can look up and down or right and left.
- Mouse scroll: This should be used to adjust the speed of the movement, or the speed the camera travels around the scene.

7. **Create perspective and orthographic displays of the 3D world**. Use the tap of a keyboard key to allow a user to change the viewport display of all objects in the scene between orthographic (2D) and perspective (3D) views at will. To accomplish this, you will be switching the function call to retrieve either the perspective or orthographic projection matrix. Note that you will be keeping the camera in the same orientation that you already developed in previous criteria.

Best Practices

8. **Apply coding best practices in formatting, commenting, and functional logic**. To accomplish this, be sure to complete the following:

- Employ formatting best practices by providing program code that is easy to read and follows industry standard code formatting practices, such as indentation and spacing.
- Employ commenting best practices to ensure project source code is briefly and clearly explained using descriptive comments.
- Employ functional coding logic best practices to ensure the program runs as expected. Note that not everything should be written in a single function; it should be well modularized.

Reflection

9. **Justify development choices for your 3D scene**. As you write, think about why you chose your selected objects. Also consider how you were able to program for the required functionality.

10. **Explain how a user can navigate your 3D scene**. As you compose your thoughts, discuss how you set up to control the virtual camera for your 3D scene using different input devices.

11. **Explain the custom functions in your program that you are using to make your code more modular and organized**. Ask yourself, what does the function you developed do and how is it reusable?

What to Submit

To complete this project, you must submit the following:

3D Scene

Submit a completed ZIP folder with all of your code, which may include one or multiple CPP files along with Visual Studio project files. Also make sure the ZIP folder includes an EXE file, because without this your code will not be able to run. Checking for the EXE can be used as a quick reference on the functionality of your code before you submit. Reference the Visual Studio Export Tutorial, linked in the Supporting Materials section, for guidance on how to download the necessary ZIP folder.

Design Decisions

Your written explanation should be submitted as a 2-page Microsoft Word document with 12-point Times New Roman font, double spacing, and one-inch margins. Any sources should be cited according to APA style.

Supporting Materials

The following resource(s) may help support your work on the project:

[Sourcing Textures Tutorial PDF](#)
This guide will explain how to find free, open-source images that you can use to texture 3D objects.

[Visual Studio Export Tutorial PDF](#)
This guide will walk you through how to download all of your work from Visual Studio as a ZIP folder.

Project Rubric

Criteria	Exemplary (100%)	Proficient (85%)	Needs Improvement (55%)	Not Evident (0%)	Value
3D Objects: Representation	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner; areas demonstrating exemplary work may include creating functions that build a single 3D object out of different meshes so it can be easily reused multiple times in the same scene	Creates low-polygon 3D representations of real-world objects	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include using more primitive shapes to create an object that accurately matches the image	Does not attempt criterion	10
3D Objects: Textures	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner; areas demonstrating exemplary work may	Applies accurately projected textures to a 3D model	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include using appropriate-	Does not attempt criterion	10

	include combining more than one texture in a single shader to accomplish a more sophisticated effect		resolution images that are not stretched or applying different texture files to each shape to create the overall object		
3D Objects: Lighting	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner; areas demonstrating exemplary work may include using different types of lights (directional, point light, spotlight) and customizing their color or intensity	Applies lighting to create a polished visualization of 3D models	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include implementing all components of the Phong shading model (ambient, diffuse, and specular)	Does not attempt criterion	10
3D Objects: Organized World	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner; areas demonstrating exemplary work may include making use of the affine transformations (scaling, rotation, and translation) to build a rich 3D scene with a few basic shapes	Places objects appropriately, using the X, Y, and Z coordinates, relative to one another in the 3D world	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include properly aligning objects so they do not unintentionally clip through other objects or making sure the arrangement of objects matches the original image	Does not attempt criterion	10
Navigation: X, Y, and Z Axes Camera Movement	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner; areas demonstrating exemplary work may include creating a speed of translation that is appropriate and/or adjustable at runtime	Applies horizontal, vertical, and depth camera navigation around the 3D scene	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include effective navigation in all three dimensions	Does not attempt criterion	10
Navigation:	Exceeds proficiency	Applies nuanced	Shows progress	Does not attempt	10

Nuanced Camera Controls	in an exceptionally clear, insightful, sophisticated, or creative manner; areas demonstrating exemplary work may include precisely moderating the speed of the movement	camera controls to effectively view the 3D objects in the application	toward proficiency, but with errors or omissions; areas for improvement may include properly aligning the rotation axis with the camera's up or right vectors to produce smooth rotations	criterion	
Navigation: Perspective and Orthographic Displays	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner; areas demonstrating exemplary work may include the ability to dynamically change between the two types of projection while the simulation is running	Creates perspective and orthographic displays of the 3D world	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include ensuring the field of view is not too narrow or too wide, or ensuring the near and far planes are properly defined to contain the whole scene	Does not attempt criterion	10
Best Practices	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner; areas demonstrating exemplary work may include logical encapsulation of code into functions that leads to reusability and efficiency	Applies coding best practices in formatting, commenting, and functional logic	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include adding a description to each of the functions written for the project or encapsulating code in functions as opposed to duplicating it across the program	Does not attempt criterion	5
Reflection: Development Choices	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner; areas demonstrating exemplary work may include a side-by-side comparison of the different	Justifies development choices for the 3D scene	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include adding descriptions of what worked and did not work along with why	Does not attempt criterion	7

	approaches considered during development				
Reflection: Navigation	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner; areas demonstrating exemplary work may include particularly detailed and nuanced explanation of approach to user navigation that is supported by examples from the code	Explains how a user can navigate the 3D scene	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include providing a clear description of the controls and how they are used	Does not attempt criterion	7
Reflection: Custom Functions	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner; areas demonstrating exemplary work may include explaining how these functions could be of benefit to another program or explaining code that was not encapsulated in a custom function but that could have been	Explains the custom functions used to make the code more modular and organized in the program	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include discussing all of the custom functions that were created	Does not attempt criterion	7
Articulation of Response	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner	Clearly conveys meaning with correct grammar, sentence structure, and spelling, demonstrating an understanding of audience and purpose	Shows progress toward proficiency, but with errors in grammar, sentence structure, and spelling, negatively impacting readability	Submission has critical errors in grammar, sentence structure, and spelling, preventing understanding of ideas	4
				Total:	100%