Final Project

Publish | Edit | ⋮

GPR-400 Advanced Real-Time Rendering
Instructor: Daniel S. Buckstein
Final Project

Summary:
The primary takeaways of this course are, in no particular order or rank, ***portfolio*** and ***engineering***. The final project is for you to demonstrate all you have learned in this course from a game programmer's lens and produce a portfolio-worthy project.

Submission:
Start your work immediately by ensuring your coursework repository is set up, and public. Create a new main branch for this assignment. ***Please work in pairs (see team sign-up). Begin your submission immediately, copy the following into the text box, decide on your repository branch name for this assignment and fill in the information below.*** **The submission box locks at the specified deadline; be proactive and don't miss it. Late submissions, even by a minute, will not be accepted.**

Copy, edit and submit the following text once as a team (you do not need the headings, please just provide your info as shown in the examples here):

1. ***Names of contributors***: Write the names of the contributors of this assignment.
   e.g. **Dan Buckstein**
2. ***A link to your public repository online***: Grab the clone link from your working repository, which should end with "*.git*".
   e.g. **https://github.com/dbucksteinccorg/graphics-coursework.git** (note: not a real link)
3. ***The name of the branch that will hold the completed assignment***: Create a new branch for this project, submit only the name of this branch.
   e.g. **final-main**

4. ***A link to the read-me and user instructions for this project***: Ensure your repository includes a read-me that summarizes how to use your finished product.
   e.g. **https://github.com/dbucksteinccorg/graphics-coursework/blob/final-main/final-readme.pdf** (note: not a real link)
5. ***A link to your video (see below) that can be viewed in a web browser***: Ensure your video is public or shared with your instructor.
   e.g. YouTube link: **https://www.youtube.com/watch?v=OqOyxQVs8lY** (note: "Attack on Game Development" by Will Gordon & Connor Breen)

Finally, please submit a ***10-minute max*** demo video of your project. Use the screen and audio capture software of your choice, e.g. Google Meet, to capture a demo of your project as if it were in-class. This should include at least the following, in enough detail to give a thorough idea of what you have created (hint: this is something you could potentially send to an employer so definitely show off your professionalism and don't minimize it):

- Show the final result of the project and any features implemented, with a voice over explaining what the user is doing.
- Show and explain any relevant contributions implemented in code and explain their purpose in the context of the assignment and course.
- Show and explain any systems source code implemented, i.e. in framework or application, and explain the purpose of the systems; this includes changes to existing source.
- **DO NOT AIM FOR PERFECTION, JUST GET THE POINT ACROSS**. Please mind the assignment rubric to make sure you have demonstrated enough to cover each category.
- **Please submit a link to a video visible in a web browser, e.g. YouTube or Google Drive.**
- **Instead of waiting until last-minute for the video to upload, create a folder or links document on Drive that is accessible to your instructor. Submit the link to this folder as part of your official submission, and copy your video file/link there when it is ready.**

Instructions:
For the final project, you will design and implement a portfolio-worthy tech demo, game or tool that demonstrates and integrates all of the course subject matter in some meaningful way. Consider the following example base designs for your project, which are in no way the limit of what you can accomplish:

- Implement a new graphics framework or renderer from scratch, or extend an existing framework with new features.
- Implement a novel algorithm based on some paradigm of real-time rendering.
- Implement a research tool or experimental framework to help with some advancement in real-time technology.

All components of the project must be functional and impressive; this is something you would want to show off at an interview... hence, this project is tied with a technical interview! Find a way to demonstrate the requirements as part of the technical interview (which involves a presentation). Implementing the bare minimum will earn you up to 80% on the project; go above and beyond for the remaining 20%.

Requirements:
Design requirements: These count towards the organizational part of the project:

- You must have a thoroughly organized repository with branching, and evidence of the codebase evolving over time and absolutely not slapped together in the last week.
- Your repository must include a read-me and any design documents, such as UML diagrams and/or technical specifications, that are started early in the project and clearly referred to throughout development. These documents are strictly focused on the subject matter of this course; e.g. if you are working on the shaders for your Capstone, I don't care about game design documents or networking, I care about the graphics engineering and shaders.

Project requirements: These count towards the implementation part of the project:

- The project should be designed in a way that facilitates some research and development, such as an experiment or advancement in the field. It is a tool for completing some sort of performance or user study. Consider open sourcing it for others to use to reproduce your experiment.
- The project is tied with the research paper and conference seminar.
  - **These requirements may be updated and clarified over time to best reflect the progress of the course and lessons**.

Presentation requirements: These count towards the demonstration part of the project:

- In your video, you must thoroughly demonstrate the outcome of your project and explain how all of the above implementation requirements are met. Walk through and summarize the main architectural contributions in code, and the critical shaders.
- You should not explain every little thing that you implemented; this is a 10-minute summary so please figure out how to get to the point and share only what needs to be shared, and thoroughly demonstrates your prowess as an engineer in this area.

Points	10
Submitting	a text entry box

Due	For	Available from	Until
-	Everyone	-	-

GraphicsAnimation-Master-Range-x2

Criteria	Ratings			Pts
IMPLEMENTATION: Architecture & Design Practical knowledge of C/C++/API/framework programming, engineering and architecture within the provided framework or engine.	**2 to >1.0 pts** **Full points** Strong evidence of efficient and functional C/C++/API/framework code implemented for this assignment; architecture, design and structure are largely both efficient and functional.	**1 to >0.0 pts** **Half points** Mild evidence of efficient and functional C/C++/API/framework code implemented for this assignment; architecture, design and structure are largely either efficient or functional.	**0 pts** **Zero points** Weak evidence of efficient and functional C/C++/API/framework code implemented for this assignment; architecture, design and structure are largely neither efficient nor functional.	2 pts
IMPLEMENTATION: Content & Material Practical knowledge of content relevant to the discipline and course (e.g. shaders and effects for graphics, animation algorithms and techniques, etc.).	**2 to >1.0 pts** **Full points** Strong evidence of efficient and functional course- and discipline-specific algorithms and techniques implemented for this assignment; discipline-relevant algorithms and techniques are largely both efficient and functional.	**1 to >0.0 pts** **Half points** Mild evidence of efficient and functional course- and discipline-specific algorithms and techniques implemented for this assignment; discipline-relevant algorithms and techniques are largely either efficient or functional.	**0 pts** **Zero points** Weak evidence of efficient and functional course- and discipline-specific algorithms and techniques implemented for this assignment; discipline-relevant algorithms and techniques are largely neither efficient nor functional.	2 pts
DEMONSTRATION: Presentation & Walkthrough Live presentation and walkthrough of code, implementation, contributions, etc.	**2 to >1.0 pts** **Full points** Strong evidence of accuracy and confidence in a live walkthrough of code discussing requirements and high-level contributions; walkthrough is largely both accurate and confident.	**1 to >0.0 pts** **Half points** Mild evidence of accuracy and confidence in a live walkthrough of code discussing requirements and high-level contributions; walkthrough is largely either accurate or confident.	**0 pts** **Zero points** Weak evidence of accuracy and confidence in a live walkthrough of code discussing requirements and high-level contributions; walkthrough is largely neither accurate nor confident.	2 pts
DEMONSTRATION: Product & Output Live showing and explanation of final working implementation, product and/or outputs.	**2 to >1.0 pts** **Full points** Strong evidence of correct and stable final product that runs as expected; end result is largely both correct and stable.	**1 to >0.0 pts** **Half points** Mild evidence of correct and stable final product that runs as expected; end result is largely either correct or stable.	**0 pts** **Zero points** Weak evidence of correct and stable final product that runs as expected; end result is largely neither correct nor stable.	2 pts

Criteria	Ratings			Pts
ORGANIZATION: Documentation & Management Overall developer communication practices, such as thorough documentation and use of version control.	**2 to >1.0 pts** **Full points** Strong evidence of thorough code documentation and commenting, and consistent organization and management with version control; project is largely both documented and organized.	**1 to >0.0 pts** **Half points** Mild evidence of thorough code documentation and commenting, and consistent organization and management with version control; project is largely either documented or organized.	**0 pts** **Zero points** Weak evidence of thorough code documentation and commenting, and consistent organization and management with version control; project is largely neither documented nor organized.	2 pts
BONUSES Bonus points may be awarded for extra credit contributions.	**0 pts** **Points awarded** If score is positive, points were awarded for extra credit contributions (see comments).		**0 pts** **Zero points**	0 pts
PENALTIES Penalty points may be deducted for coding standard violations.	**0 pts** **Points deducted** If score is negative, points were deducted for coding standard violations (see comments).		**0 pts** **Zero points**	0 pts
				Total Points: 10