Jake Schott

Prof. Ong

CSS 385 Intro to Game Development

30 October 2025

**Performance Considerations One-Pager**

**Consideration #1: Handling a Large World Space**

One consideration that must be taken into account with my game proposal is how to handle the relatively large world space. Floating point errors can ensue if you are not careful about how your positional data is stored, which if left unchecked, could lead to flickering or rendering issues. There is no way around needing to navigate a large world space in my game, so some solution will need to be a focus. For the models, the Unity Engine has a feature which allows for 3D models to be swapped out for less-detailed versions when far away, to reduce the number of triangles that must be rendered in a given frame. From the player’s perspective, the model looks the same, but because of the distance, it is actually a much less-detailed version of the same model.

**Consideration #2: Handling a Detailed Bridge Environment**

For the players to be immersed in the ship environment, there must be extensive detailing throughout the bridge, including screens and controls. The controls would ideally be intricate and highly-detailed, and because of the large quantity of interactable controls, there will need to be careful thought placed into how the controls are ordered, modeled, designed, and programmed to ensure that they are not too taxing on the CPU and GPU of players. Similar to the world space consideration, the use of model detail switching will be key to preventing GPU overload.

**Consideration #3: Handling a Large Quantity of UI Elements**

The interior of the ship requires a large number of in-game “screens” which display critical animations using graphics and animations. Each screen will likely have to be implemented as an in-world UI element which could be relatively expensive from a computational standpoint. Each screen will have to be designed strategically as to limit its expense when rendered, potentially through compressed graphics or other means. Canvas rendering is different from model rendering, so further research will be necessary before the project can progress. UI elements can be made more efficiently by scaling and compressing the graphics, especially when the player is not close to the UI element, which from a distance looks about the same as it would up close.

**Summary**

To ensure the game is performative, I must ensure that the models are not overly-detailed and the UI elements are strategically used as to not overload the GPU. This should be attainable as long as the game is designed and planned to utilize these strategies early on. There are built-in features within the Unity engine which allow these to be a focus without having to create external plug-ins, programs, or scripts which will take away my focus from developing other aspects of the game.