2. Requirements, Constraints, And Standards

2.1 Requirements & Constraints

The primary requirements involve many baseline expectations for functional software, primarily with performance and soft design requirements. For the rendering engine, the primary requirements are to meet the requirements specified by the game design team in order to support the operation of the game in general. The requirements are broken into various sectors, including functional, physical, and user experience.

Rendering Engine

Functional requirements

- Renders non-euclidean geometry
- Render vectors of data into a laptop
- Should update at least 30 fps (**constraint**)

Resource requirements

- Resources should not exceed that of a laptop

Physical requirements

- A graphics processing unit is needed to do the graphics processing.

Aesthetic requirement

- The code we write should be readable
- The code should be commented

User experiential requirements

- A good interface should be provided with good documentation

Interface requirements

- The interface should be something a game engine should be able to interact with

Other requirements

- Easy to install

Game Design

Functional requirements

- Easy to understand yet interesting game mechanic
- Smooth gameplay

Resource requirements

- Game Engine
- Initial internet connection to install

Physical requirements

- System capable of running the game
- Laptop/PC with display
- Keyboard
- Controller
- Mouse

Aesthetic requirements

- Aesthetically pleasing art styles and music/sounds that mesh well

User experiential requirements

- Intuitive UI
- Easy to use controls
- Interesting and easy-to-follow story

Economic/market requirements

- Easy to monitor and control distribution
- A demand for cool games

Interface requirements

- Keyboard & Mouse/ Controller usage to send inputs to the system
- Laptop/PC with display

Other requirements

- We need to have some deliverables by the end of Senior Design II.
- This project needs to be able to show some non-euclidean rendering by the end of CPRE 492
- This project needs to be able to show some game design and game functionality on a rendering engine of our own by the end of CPRE 492

2.2 Engineering Standards

Engineering standards are essential because they are the guiding principles for developing a sound solution. They are what allow products to be safe, reliable, and consistent. They help establish good practices, guidelines, and requirements, allowing engineers to develop solutions that meet expectations. Without standards, users would not have any expectations of what to expect from different companies, and there would be no consistency between different companies and components. Standards promote uniformity, which allows for ease of use by the consumer.

Standard 1 Standard for Video Games Vocabulary

This standard goal is to unify the vocabulary used in the video game industry. It aims to remove ambiguity in terms of the use of artificial intelligence can be used within the industry. It precedes how characters speak and how industry professionals develop new material. Terms will be defined based on existing literature and the community's consensus. This standard is relevant to our project because we are developing a video game, so we should adhere to those standards.

Standard 2 Quality Assurance - IEEE 730.1-1995

This standard's goal is to put in place good practices when it comes to the development and maintenance of software. It is a standard focusing on where failures could occur and how failure could impact the safety of its users. It makes sure that plans are in place in case of failure. This seems relevant to our project as we should put into place some plan in case our rendering engine fails, or our project has failed.

Standard 3 Software cycles

This standard goal establishes a common framework for the software life cycle. It also defines terminology the industry should use when referencing software life cycles. It aims to help provide reasonable standards for developing software regarding the timeline. It also wants to provide context for software products regarding how it was developed. This is an international standard that also provides processes to help define and improve upon existing life cycle systems in organizations. This is an essential standard for us to consider because we should set into place a good framework for how our software lives.

The three standards above seem relevant to our project but may not be the most relevant. Standard one is important to our project. We are developing a video game and to produce a high-quality video game, we should adhere to the standard. When we document how our video game works and when we implement communication in the game, we should adhere to the vocabulary standard defined by this standard. This standard may not be relevant to the main structure of our project, but it is relevant when it comes to the quality of the product we produce. Standard two may have less of an impact on our project overall. While it is essential to have a defined plan for failure to help the users when our software fails, Our software is not high risk. We are not developing something that will put our users in high risks environments or developing something that handles sensitive information. The standard is important, but it probably should not be the first thing to consider. Standard 3 is vital in the later stages of our project and will become more important if this project continues outside of senior design. Setting up a framework for the software life cycle is not important if you don't have working software. For this to be important, the software needs to be in a working-ish state. Our software is not planned to be working until 2nd semester, so this is not a consideration until then. However, once we have a working software, a good framework is important so the game can be maintained properly.

Other standards that we looked at were Software testing, which we will consider once we get to the testing stage of our project. We also looked into standard revolving rendering and if there were any standards on how to use/create/maintain an engine in software like OpenGL. Because we were also going to create an interface for this engine, we looked at standards regarding computer graphics and computer graphics interface, which defined how functionality should be set in the interface.

We are going to make some modifications to our project design to incorporate these standards. Given the standards that we look at, our design documents about low-level features might not change that much. However, the design details of our final deliverable and how we will express our project to the users will change. We will have to give more detail and attention to the vocabulary we use in documentation and gameplay for our video game. We will also need to change our design for how we will manage our video game once it is working. We need to include the framework we will use for the life cycle and make sure it makes sense with our other requirements for our project. We may also want to change our design in how we go about testing in order to make sure we meet the standard for quality assurance.