# 

# 

Project 2: WebGL2 Video Game Bullet-Blitz

Blaise E. Allen

Harrisburg University of Science and Technology

CISC 498: Project 2

August 14, 2023

Professor Mina Gabriel

Table of Contents

**Introduction**3-4

**Methodologies** 4-6

**Knowledge Obtained**6-10

[**Design Document & Game Concept Document**](#_Toc72917242) 10-11

**Prototype Development & Evaluation**11-14

**Future Work**14-16

**Program Goals & Core Competencies**16-17

**Conclusion**17-18

**References**

**Introduction**

In an ever-evolving landscape where the digital realm intertwines seamlessly with the realms of imagination, a new era of creative expression and immersive storytelling has emerged. At the forefront of this evolution stands the world of video games, a dynamic domain where technology and artistry coalesce to craft experiences that transcend the boundaries of reality. As a fervent enthusiast of computer graphics, my journey unfolded with an unwavering fascination for the intricate mechanisms that breathe life into pixels and polygons, driving me to embark on a voyage of exploration and discovery.

The genesis of this project was not a casual decision, but rather a deliberate choice borne from a deep-seated passion for delving into the intricacies of graphics development. Guided by an insatiable curiosity, I set out to uncover the potential of WebGL2—a powerful and versatile technology that empowers developers to create breathtaking 3D graphics right within web browsers. The allure of this endeavor lay not only in its technical complexity but also in its potential to transcend conventional boundaries and deliver captivating experiences directly to users' screens.

This research paper serves as a chronicle of my odyssey through the realm of WebGL2, a technology that empowers creators to craft interactive and visually stunning worlds. Within its virtual corridors, I explored concepts that would form the bedrock of my creative aspirations, from fundamental graphics theories to the intricate interplay of light and shadow. This voyage encompassed both the theoretical and practical aspects of computer graphics, leading me to venture beyond the boundaries of traditional platforms and delve into the enchanting world of web-based gaming.

As we journey through the pages of this paper, we will navigate the depths of my aspirations and the pathways I forged in my quest to master the art of game development. From the inception of my project, where the seeds of curiosity took root, to the complexities of understanding JavaScript, WebGL2, and their nuanced modules, each step brought me closer to my vision—a vision that encompasses not only a fully-realized game but also a deeper understanding of the intricate dance between technology and creativity.

This is more than a journey through code and concepts; it is a testament to the power of curiosity, the allure of innovation, and the transformative potential of technology. Join me as we embark on this expedition through the realm of WebGL2, where each line of code and every polygon formed led me closer to the realization of a dream—an immersive world of dynamic graphics and engaging gameplay, waiting to be unveiled on the canvas of the web.

**Methodologies**

This section outlines the methodologies employed in the execution of the project, detailing the original scope, its evolution, and the core concepts investigated within the facets of web development, computer graphics, and game design.

The project's inception was guided by a comprehensive plan to develop a web-based first-person shooter game, supported by an intuitive platform, with the overarching goal of enhancing skills in computer graphics, web development, and game design. Key technologies selected for this endeavor included HTML, JavaScript, WebGL2, CSS, Bootstrap, MongoDB, and Node.js.

The initial project scope encompassed the creation of a fully functional 1v1 shooter game embedded within a web platform, complete with realistic graphics, user authentication, matchmaking, and multiplayer functionalities. The anticipated deliverables consisted of a comprehensive software design document, prototyped programs, literature review notes, a research paper, poster, and a mock-up of the final implementation.

However, as the project unfolded and a deeper understanding of the complexities emerged, a re-evaluation of the scope became necessary. The original ambition, while commendable, proved intricate given the limited timeframe of the semester. This prompted a strategic shift towards a more structured approach, with an emphasis on creating a robust design document and refined prototypes.

The foundational methodologies applied in this project were underpinned by an in-depth exploration of crucial concepts in web development and computer graphics. HTML, CSS, and JavaScript formed the backbone of the proof of concept, with a focus on establishing the foundational graphics knowledge required to implement a fully functional prototype using WebGL2.

WebGL2, the linchpin of the graphics pipeline, was meticulously studied. This involved writing shader source code, compiling and linking shaders into programs, configuring buffers, and harnessing global variables like attributes and uniforms. The intricate world of coordinate normalization, projection, and transformation was extensively examined, enabling the creation of intricate 2D objects, as well as the scaling, translation, and rotation of entities within the game environment.

Game logic and interaction were dissected, including the execution of functions and animations based on user interface events and time-based triggers. Concepts like object detection, enabled by WebGL2 attributes and JavaScript global variables, were fundamental in implementing responsive and engaging gameplay mechanics. This foundational knowledge would not only cater to the project's immediate objectives but also establish a solid groundwork for future iterations and more ambitious undertakings.

In summary, the methodologies adopted for this project were driven by a meticulous plan, an understanding of scope, and an exploration of pivotal concepts within web development, computer graphics, and game design. The shift in scope was motivated by the project's inherent complexity and the pursuit of a more attainable set of deliverables, culminating in a comprehensive design document, prototypes, and foundational research materials.

**Knowledge Obtained**

At the inception of this project, the pivotal role of HTML and JavaScript was undeniable in crafting dynamic web applications. However, an evolution in scope for the final deliverables led to a profound shift in focus, directing attention towards a deeper exploration of WebGL and its intricate components. While HTML and JavaScript set the stage for interactivity, the project's overarching goal expanded to unravel the core principles underpinning graphics programming and to manifest a proof of concept that resonates with the realm of graphics applications.

The Shader and GPU integration segment serves as the crux of translating intricate code into visually captivating graphics. This exploration begins with the creation and orchestration of shaders, extending to the symbiotic interaction between these shaders and the GPU for rendering.

Shaders are the virtuoso composers of the graphics programming orchestra. Vertex shaders conduct transformations, precisely altering vertex positions within a scene, while fragment shaders define the color and appearance of pixels. These scripts form the heart of WebGL, imbuing static data with life and imbuing objects with motion and vibrancy. Attributes, buffers, varying’s, and uniforms act as the conduits for data flow, facilitating communication between the CPU and GPU. Attributes, varying’s and buffers typically represent characteristics of vertices, which connect the CPU to vertex shaders.

As the conductor orchestrates a symphony, the GPU processes data with unmatched parallelism. Each vertex, containing a bundle of attributes, undergoes a series of transformations guided by the vertex shader. The GPU processes data in parallel for each vertex, orchestrating rasterizations in accordance with the values declared within the fragment shader as well as the drawing method specified. Varying’s act as the dancers in this synchronized choreography, gracefully transferring data between vertex and fragment shaders. They allow smooth interpolation of values, granting continuity and realism to rendered objects. As vertices transition through shaders, varying’s ensure a seamless flow of data, crafting a harmonious visual experience.

Upon the completion of vertex shader transformations, the spotlight shifts to fragment shaders. Here, individual fragments (pixels) are colored and shaded, employing complex algorithms and techniques. The fragment shader harmonizes lighting, texture mapping, and color blending to produce a coherent visual output, painting each pixel with stunning precision.

Delving into the fundamentals of WebGL uncovered the significance of attributes, buffers, and varying’s. Attributes define the characteristics of vertices, buffers store these attribute data, and varying’s facilitate seamless communication between the vertex and fragment shaders. This comprehension is essential to understanding how to provide data to the shaders or set up state for the shaders to perform additional operations on, and more generally for your GPU to rasterize various objects based on the state you have provided.

The mastery of uniforms and textures proved indispensable in enhancing visual realism and complexity. Uniforms serve as global constants across vertex or fragment shaders. They provide a means to universally modify scenes, for example, declaring an array of vectors that store RGB color data, or for performing transformation functions on vertex data. Textures, however, introduced a new dimension, enabling the incorporation of images and facilitating advanced rendering techniques, texture will be discussed in greater detail in subsequent paragraphs.

The concept of transformation, coupled with matrix math, breathes life into static objects. Matrix transformations, achieved through multiplication, enabled a seamless orchestration of translation, rotation, and scaling of objects. The utilization of these concepts along with a fundamental understanding of game mechanics, allows a programmer to create both basic and complex animations.

Textures emerged as a potent tool for image processing, granting the power to imbue graphics with intricate details. Texture coordinates facilitated precise mapping, while pixel manipulation enabled image alteration. Understanding textures and pixels manipulation offers greater capabilities in producing visually enhanced graphics, through the application of such principles image processing techniques and the manifestation of vibrant visual compositions can be achieved.

The concept of objects and bind points stands as a cornerstone for unleashing the full power of WebGL. Diverse entities like buffers, textures, render buffers, samplers, queries, vertex arrays, framebuffers, and transform feedback provide distinctive functionalities and attributes that manipulate the state within a WebGL program. However, realizing the true potential of these objects necessitates binding them to specific WebGL bind points or targets. This act expands the capabilities of these objects when coupled with a particular bind point, empowering programmers to manipulate and transform the state within these object containers more effectively. A practical example within the WebGL realm involves the amalgamation of vertex, fragment, texture, and shading data into a single buffer. Yet, the organizational value of this data would remain untapped without binding the buffer to a designated WebGL target such as ARRAY\_BUFFER. This binding action unlocks methods like vertex attribute pointers, which instruct shaders on how to extract diverse attributes from the buffer during program execution. This optimization and organization of data are pivotal for generating objects in a graphics application, enhancing overall efficiency and performance.

Delving deeper into optimization, a paramount consideration in graphics applications, especially within the realm of web-based games implemented using JavaScript and WebGL, is essential. The efficiency of your code is crucial for delivering precise real-time renderings to diverse systems over varying networks. However, an important tradeoff looms: precision versus memory. This tradeoff is exemplified when choosing the precision level while declaring varying’s or uniforms in a fragment shader. Opting for higher precision yields more accurate renderings, but at the cost of increased memory usage. This could potentially lead to glitches or unclear renderings when executing a game over the internet, where real-time transformations and rendering occur across a wide geographic range.

Various strategies exist to bolster the efficiency of a WebGL program or graphics application. For instance, storing data in a consolidated buffer reduces the number of buffers an attribute must pass to the vertex shader. Strategic choices in precision when declaring different data types within a single buffer, along with storing only essential unique data within a particular object for manipulation and rasterization, contribute to optimization. While such methods may involve tradeoffs, like the need for at least two buffers in the case of using ELEMENT\_ARRAY\_BUFFER, they effectively curtail memory usage by eliminating superfluous data. By employing 8-bit integers to represent index values rather than 32-bit floats and incorporating a minimal number of additional attributes, developers can strike a balance between data efficiency and performance.

Ultimately, developers must navigate these optimization principles while determining their application's implementation. The tradeoffs involved underscore the need for meticulous planning and consideration, particularly when conceptualizing the functionality of an application within the dynamic world of graphics programming.

**Game Concept & Game Design Overview**

The "Bullet Blitz" project is to be a dynamic and intense first-person shooter (FPS) game developed using JavaScript and WebGL2. The game emphasizes fast-paced multiplayer battles while delivering an immersive gaming experience. The project comprises two essential documents: the Game Design Document (Iteration 01) and the Game Concept Document (Iteration 01).

The Game Design Document provides a comprehensive framework for the game's development. It outlines the game's purpose, overview, scope, and key components. The game concept is introduced, targeting FPS enthusiasts who enjoy intense multiplayer combat. The gameplay mechanics, such as the core gameplay loop and player controls, are meticulously detailed. The document delves into player progression, graphics, technical specifications, multiplayer capabilities, and user interface design. The emphasis is on optimizing gameplay, ensuring fluid animations, and creating a seamless user experience. Additionally, the document underscores the importance of real-time synchronization for multiplayer modes and the need for cross-platform compatibility.

On the other hand, the Game Concept Document paints a broader picture of the project. It introduces the game as a fast-paced FPS with detailed graphics and enhanced gameplay elements. The document outlines gameplay mechanics, including the first-person perspective, diverse weapon loadouts, and fast-paced combat mechanics. The multiplayer functionality is highlighted, showcasing the ability to invite friends or engage in matchmaking for competitive battles. The document emphasizes technical considerations, such as leveraging WebGL2 for graphics rendering, optimizing networking for smooth multiplayer experiences, and ensuring cross-platform compatibility. It also discusses the monetization model, which offers a free-to-play experience with the option to purchase cosmetic enhancements.

Together, these two documents serve as the foundation of the research project. The Game Design Document provides a comprehensive guide for the development process, while the Game Concept Document presents an overarching vision for the game. These documents not only guide the current iteration of the project but also provide a roadmap for future iterations, ensuring a consistent focus on delivering an intense and immersive FPS experience for players seeking competitive online battles.

**Prototype Development & Evaluation**

The proof of concept serves as a pivotal milestone within this research project, acting as a tangible embodiment of my objectives and aspirations for the FPS game "Bullet Blitz." The current state of the proof of concept takes the form of a 2D gun range environment, strategically designed to encapsulate key gameplay mechanics that align with the overarching goal of the project. Within this simulated gun range scene, a dynamic target undergoes continuous translation across the screen, injecting an element of movement and challenge that mirrors the action-packed nature of the final game.

Drawing upon the foundational knowledge acquired through the study of WebGL, I've deftly implemented core functionalities that lay the groundwork for the forthcoming phases of game development. This careful integration of essential components illustrates my deliberate approach towards creating an immersive and engaging gaming experience. By effectively utilizing WebGL, I have introduced critical features that mirror the core principles underlying "Bullet Blitz."

Crucially, the proof of concept includes meticulously designed user interface (UI) events, elevating the level of interaction and engagement. Upon hovering the mouse over the canvas element, a visual optic is seamlessly rendered, providing players with a precise aiming mechanism akin to what they will encounter within the final game. This intuitive and responsive UI element not only showcases the seamless integration of graphics but also underscores the strategic importance of user experience in enhancing gameplay.

Furthermore, the proof of concept encompasses a fundamental shooting mechanism. Upon clicking the mouse, a sophisticated function records the precise position of the cursor and employs WebGL techniques to ascertain whether the shot successfully hits the moving target. This interplay of user input and dynamic scene elements accurately captures the essence of marksmanship and precision that defines the core gameplay of "Bullet Blitz."

The implementation of this rudimentary shooting functionality signifies a significant step towards realizing the complex mechanics required for the final game. By translating my understanding of WebGL into practical coding, I've successfully laid the foundation for vital gameplay mechanics. This deliberate alignment of core functions within the proof of concept with the overarching objectives of "Bullet Blitz" sets a strong precedent for the future iterations of the project. It exemplifies my commitment to gradually build upon these fundamental elements, ultimately culminating in a fully immersive and exhilarating FPS gaming experience that resonates with enthusiasts of the genre.

* **Evaluation of Proof of Concept**
  + Optimization Issue**:**
    - Currently using four buffers with 32-bit floats, leading to higher memory usage.
    - Unintentional data redundancy: data optimization is needed.
    - Utilize alternative WebGL bind points (e.g., ELEMENT\_ARRAY\_BUFFER) to reduce redundancy.
    - Optimize memory usage by using 8-bit integers for index values and 32-bit floats for repetitive vertex data.
    - Utilize attributes, vertex-attribute pointers, and efficient draw calls for improved memory usage and data organization.
  + Callback Function Complexity**:**
    - Caller and callback functions used to create circles for target object.
    - Callback hell scenario in JavaScript due to multiple nested functions.
    - Unable to get return value from original call to the caller function.
    - Created additional checks within callback function to execute main() and start the game.
    - Created scoping issues affecting user sights and aim, hindering core gameplay mechanics.
  + Collision Detection Issue:
    - Collision detection check for target shooting is imperfect.
    - Checking distance between click point and target origin compared to target border.
    - Not foolproof; occasional buggy return values for target hit detection.
    - Need to refine collision detection logic to ensure accurate and consistent target hit detection.

**Future Work**

To address the identified issues in the current proof of concept and to advance towards the final implementation of "Bullet Blitz," several key steps will be taken. These steps involve further optimization, expanding knowledge of WebGL concepts, and preparing the technical groundwork for the multiplayer aspect of the game. The following outlines the plan for addressing the existing issues and enhancing the overall development process:

* Reevaluate buffer usage: Reduce the number of buffers and utilize different bind points for improved data organization.
* Implement more efficient data structures: Explore ways to minimize data redundancy and enhance memory efficiency.
* Utilize optimized data types: Incorporate 8-bit integers for index values and 32-bit floats for vertex data storage to optimize memory usage.
* Apply advanced WebGL methods: Utilize attributes, vertex-attribute pointers, and optimized draw calls for enhanced performance and data management.
* Restructure callback functions: Redesign the callback functions to minimize nested structures and avoid callback hell scenarios.
* Implement Promises or async/await: Utilize modern JavaScript features to simplify asynchronous code and ensure proper function execution and data handling.
* Focus on scoping and visibility: Reevaluate variable scopes to prevent unintended scoping issues affecting core gameplay mechanics.
* Rethink collision detection logic: Develop a more robust and accurate collision detection algorithm to ensure consistent and reliable hit detection.
* Experiment with different approaches: Explore alternative methods such as ray-casting or bounding volume hierarchies to improve collision accuracy.
* 3D Coordinate Projection and Transformation: Study and implement advanced 3D projection techniques and transformations to bring depth and realism to the game's visuals.
* Lighting and Shading: Explore various lighting models, shading techniques, and materials to create realistic lighting effects on objects within the game world.
* Image Processing Prototypes: Develop prototypes that leverage image processing algorithms to enhance visual quality, including effects like bloom, depth of field, and post-processing filters.
* Learn and implement UDP networking: Study networking concepts and JavaScript APIs, such as Node.js, to establish a reliable UDP-based communication system for seamless player interactions in multiplayer matches.
* Develop matchmaking and lobby systems: Create prototypes to manage player connections, matchmaking, and lobby interactions, ensuring smooth player experiences during online play.

In conclusion, the outlined future work aims to address the current limitations of the proof of concept and propel the development of "Bullet Blitz" towards its final implementation. By optimizing data storage, refining complex functions, enhancing collision detection, delving deeper into WebGL concepts, and preparing for multiplayer functionality, the goal is to create an engaging and visually stunning FPS game that offers intense multiplayer battles and a rich gameplay experience. Through these efforts, the core principles of WebGL, advanced graphics rendering, and networking will come together to realize the full potential of "Bullet Blitz" as a captivating and competitive online gaming experience.

**Core Competencies**

The facets of my research project revolving around the development of the WebGL2-based video game "Bullet Blitz" eloquently illustrate how I, an aspiring computer science graduate, encompass Harrisburg University's Core Competencies. Each phase of my endeavor signifies the integration of these competencies into my work. The project's inception was marked by critical thinking and problem-solving as I carefully assessed the game's scope and complexities, adapting the plan for attainable objectives. My technical and quantitative literacy was demonstrated through the mastery of JavaScript, WebGL2, and associated libraries, enabling the implementation of core gameplay mechanics and shooting functionality. Clear communication was exemplified through comprehensive design documents, adeptly conveying the game's concept, mechanics, and technical intricacies.

Furthermore, my information literacy skills were evident as I engaged in thorough research to comprehend WebGL2 fundamentals, shader programming, graphics pipelines, and networking protocols, thus laying the groundwork for a functional prototype. Ethical considerations were paramount as I adhered to coding standards, respected intellectual property rights, and employed legitimate resources to advance my understanding. My commitment to teamwork and cultural competency shone through collaboration with peers and mentors to surmount challenges collectively, emphasizing adaptability in a diverse environment.

The optimization endeavors underscored my global awareness and social responsibility by enhancing the game's accessibility and efficiency for a worldwide audience. The implementation of multiplayer functionality underscored my cultural competency by catering to diverse player preferences and advocating for inclusivity in gaming experiences. Lastly, my project's dynamic nature showcased lifelong learning and adaptability, evidenced by my willingness to modify methodologies and goals in response to evolving challenges. In essence, my exploration of WebGL2 and the creation of "Bullet Blitz" reflect the interplay of these competencies, reflecting not only a deep comprehension of technical concepts but also a holistic readiness for the dynamic landscape of computer science and game development.

**Conclusion**

In conclusion, this research paper has documented my immersive journey into the realm of WebGL2 and its application in the development of the dynamic first-person shooter game, "Bullet Blitz." This endeavor reflects my passion for graphics development and my commitment to mastering the intricate interplay of technology and creativity. As I delved into the depths of WebGL2, I unraveled its core principles, from shaders and GPU integration to optimization techniques and advanced graphics concepts. The paper outlined the methodologies I employed, the challenges I encountered, and the solutions I devised to create a proof of concept that embodies the essence of the envisioned game.

Looking ahead, my future endeavors as a computer scientist involve expanding beyond the scope of this research project. I plan to address the current limitations and optimize the proof of concept by reevaluating buffer usage, implementing more efficient data structures, and applying advanced WebGL methods. Furthermore, I intend to enhance scoping and visibility, rethink collision detection logic, and experiment with different approaches for improved gameplay mechanics and visual realism. Embracing modern JavaScript features like Promises or async/await will simplify asynchronous code and streamline function execution.

Beyond refining the proof of concept, my future path involves exploring 3D coordinate projection, advanced lighting and shading techniques, and image processing prototypes to elevate the visual quality of the game. Learning and implementing UDP networking, developing matchmaking and lobby systems, and studying Node.js will lay the foundation for a seamless multiplayer experience in "Bullet Blitz." These ambitious steps aim to bring the game closer to its final implementation, delivering an engaging and captivating FPS experience for players.

Throughout this journey, I have demonstrated Harrisburg University's Core Competencies. Critical thinking, problem-solving, technical literacy, and communication were evident in the development process. Information literacy was showcased through extensive research, and ethical considerations were upheld through coding standards and intellectual property respect. Collaboration and cultural competency were highlighted in teamwork, while optimization efforts reflected global awareness and social responsibility. The project's dynamic nature exemplified lifelong learning and adaptability, qualities that will continue to guide me as I navigate the evolving landscape of computer science and game development.

**References**

*Webgl2 fundamentals*. WebGL2 Fundamentals. (n.d.). https://webgl2fundamentals.org/