Super Archer Final Report

Project Name: Super Archer

Group Members:

Rohith Venkatesh:
- Email: rvenkatesh2025@g.ucla.edu
- UID: 305798435

Rishi Padmanabhan:
- Email: rpadmanabhan20@g.ucla.edu
- UID: 105685533

Beomjoo Kim:
- Email: globalman96@g.ucla.edu
- UID: 605128830

Theme of Our Project

All of us enjoy playing Game Pigeon archery, and we decided to create a spinoff of the game based on inspiration from other video games, including Assassin's Creed where the user is allowed to control the arrow after it is launched up to the target.

Our Game

The game involves a user shooting an arrow and attempting to hit a target. Between the shooting location and the target, there will be multiple moving obstacles that try to prevent the user from successfully hitting the target. If the user hits one of the obstacles or the ground, or passes the target, the user loses a life. If the user hits the target, the user will gain a certain amount of points based on the collision location on the target.

If the user hits the edge, 1 point will be added to the score. If the user hits the third ring from the bullseye, 2 points will be added to the score. If the user hits the second ring from the bullseye, 3 points will be added to the score, and finally, if the user hits the bullseye, 4 points will be added to the score. Each time the user hits the target, the obstacles will move slightly faster, essentially creating a concept of infinite levels (until the number of lives reaches 0).

In addition, the user can also obtain power-ups. One power-up is the “life-up”. It is a purple cube in our game. If the user manages to hit the purple cube with the arrow, the number of lives left for the user will go up by 1. Of course, if the user collides with an obstacle after collecting this power-up, their life count will go down 1, so it’s as if they just restarted the same round. The second power-up in our game is the “shield power-up”. This is represented by a white cube in our game. If the user manages to hit the white cube with their arrow, they will be immune to hitting obstacles for the duration of that round. That means the user can just aim for the bullseye irrespective of the moving obstacles! Even if the user collides with an obstacle, the round will not end because they have the shield. The power-up lasts for the entirety of the round.

Our power-ups spawn in random locations each round, so there is an element of mystery in the game. The user cannot simply predict the locations of the power-ups or know their locations beforehand. It's all random to keep the user guessing! Of course, it's possible to obtain both power-ups in the same round. In addition, if the user goes past the power-up without obtaining it, the power-up will disappear from view. The user will just see what is currently in front of the arrow.

The user will have unlimited attempts until they lose all of their lives to maximize their score. After that, the game will end. The game uses the point system as an incentive for the user to hit the target (providing a level of interactivity).

How to Play

The round begins with the arrow starting in a horizontal position. At the start of the game, there is no gravity acting on the arrow. Press 'c' to shoot the arrow, and then the fun begins! You should utilize the keys I-J-K-L to move the arrow up, left, down, and right respectively. If you do not press any of these keys, the arrow will move downward, as it is being acted on by a constant gravitational force. You must resist this gravitational force by utilizing the keyboard controls presented above (it's as if you are providing a resistive force that acts against the gravity to propel the arrow and prevent it from hitting the ground). Try your best to avoid the moving obstacles. Time your pressing of 'c' accordingly! If you do run out of lives, the game will end, but you can always restart by pressing 'b' on your keyboard. That will take you back to the beginning of the game, and you can try beating your previous score!

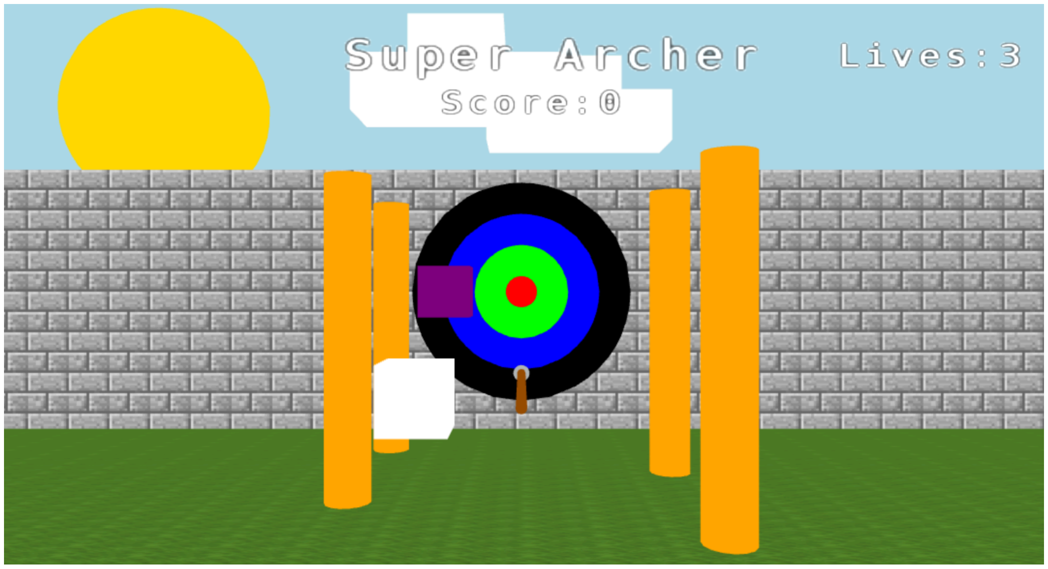

Figure 1: *Screenshot of starting position, including text displaying current score, and lives left*

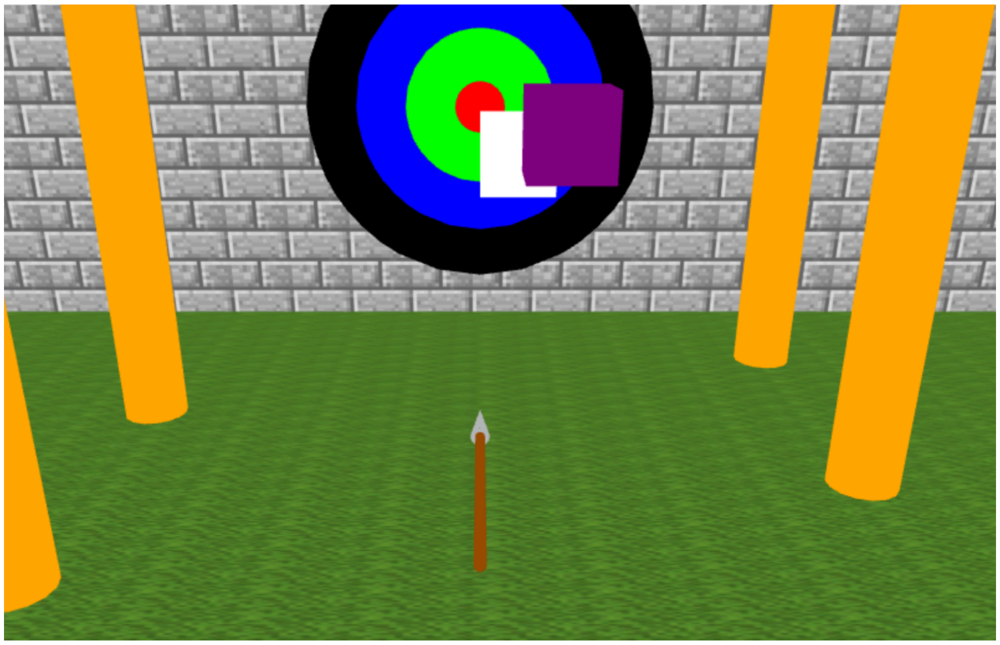

Figure 2: *Screenshot showing gameplay when arrow is in motion*

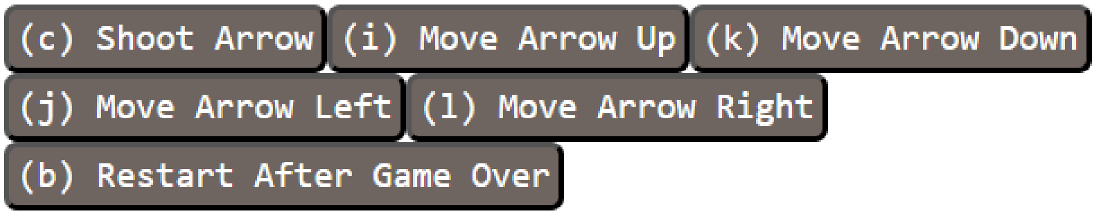

Figure 3: *Screenshot showing buttons implemented*

Course Topics Applied

- **Collision Detection:** We utilized collision detection in our project for detecting when the arrow hits an obstacle, the ground, a power-up, or the wall behind the target.
- **Anti-Aliasing Techniques:** We used anti-aliasing techniques to improve the accuracy of our collision detection.
- **Bump Mapping:** We utilized bump mapping to add textures to objects and make them appear more realistic in our game scene.
- **Lighting and Shading:** We used lighting and shading in our project to make our game scene more visually appealing. We specifically utilized ambient lighting and Phong shading.
- **Change of Basis:** We utilized our change of basis for accurate arrow movement and interaction within our game. When the game begins, the user can see the POV of the actual arrow, which is essentially a change of basis (recalculating the coordinates based on a new origin and set of bases).

- **Geometric Transformations:** We utilized rotation, translation, scaling, and inverse matrix transformations to model the objects in our scene.
- **Painter's Algorithm:** We utilized Painter's algorithm to paint our target in our game. We analyzed each of the target sections and first scan-converted the one with the largest depth, followed by the next largest, etc. to get a good-looking image of the target in our game.

Advanced Features

- **Collision Detection:** As mentioned above, we used collision detection for the arrow hitting obstacles, the target, the ground, power-ups in the scene, and the wall behind the target in the case the arrow missed the target completely.
- **Bump Mapping:** We utilized bump mapping to make some of our objects look more realistic in the scene, specifically our background wall and grass.
- **Physics-Based Simulation:** In our game, we had a constant gravitational force acting on the arrow in the form of a torque towards the ground. This is a torque due to gravity. The user would use the keyboard controls to propel the arrow and resist this force.

Challenges Faced and How We Overcame Them

- **Collision Detection:** We had an issue with getting our collision detection to become as accurate as we wanted it to become. We thought for a while and then figured out that we had an aliasing issue. We weren't sampling enough points on the arrow for the collision detection. Thus, we refined our collision-detection algorithm to sample more points on the arrow, and we were able to resolve this issue.
- **Implementing Gravity:** This was a big challenge for us at first. Initially, the arrow would just move down with a horizontal orientation rather than just rotating downward if no force acted on it other than gravity. We spent a lot of time drawing out diagrams of vectors and then we were able to implement this via a rotation (applying the gravitational force as a torque on the arrow so it rotated towards the ground and moved in a more realistic manner)
- **Consistent Target Visibility:** Initially, we had an issue where if the user missed a power-up and they moved the arrow to the same level as the power-up, the power-up would obstruct the camera view and the user would not be able to see the target for a bit. We were able to resolve this by adding an extra check for the arrow going past the power-up. If the arrow were to move-past the power-up (if the user missed it and now the arrow is past it), the power-up object would disappear, making it easy for the user to see the target at all times (note that the power-up also disappears upon the user hitting it with their arrow to obtain the power-up).

Lessons Learned

1. **Aliasing is a big problem in Computer Graphics.**
    - At first when we were introduced to this concept, we didn't really think it would be that big of a deal in Computer Graphics, but during our process of implementing our project, we realized how important it was to address aliasing and how prevalent it was in the world of Computer Graphics.
2. **Incremental development and testing is super important.**
    - During the project, we realized how important it was to test features individually as we developed them to ensure proper game functionality in the end. Early on in the project, we often moved along too quickly without thoroughly testing our implementation of the features we just implemented, and that gave us problems in later implementations of more features.
    - We realized how important it was to thoroughly test each feature as we developed it to make sure that it fully works before moving on to the next thing.

Thoughts on the Peer Review Process For Reviewing, Feedback, TM Evaluations

- Overall, we really enjoyed the process of peer-reviewing. It allowed us to have the opportunity to see what our classmates were doing and learn from their projects along with providing them with good feedback for how they can improve their project and what was already looking really good. It was also really nice to be able to inquire about implementing certain features that you may want to include in your project as well.
- The feedback process was also really nice. It allowed us to gain an additional perspective on what was working well and what we could do better. We liked the idea of not only getting feedback from instructors, but also from classmates as well.

If We Had More Time

1. **Implemented more power-ups**
    - With the limited amount of time we had to brainstorm, we were only able to come up with and implement two power-ups in our game, but if we had more time, we could have added extra power-ups as features, such as an arrow speed-boost.
2. **Implemented more types of obstacles**
    - We only had time to implement obstacles moving from side to side. If we had more time, we could have added obstacles that moved up and down, along with differing obstacle shapes.
    - This could've added an extra complexity for the user of the game (even though the game is already pretty hard!)
3. **Incorporated A Moving Target**
    - We only had time to implement a stationary archery target with collision detection, but if we had more time, we could have implemented a moving target as the user hit the target more and more times (making it tougher each time)
4. **Added different "Maps"**
    - It would have been great to implement different backgrounds/sceneries for each level to provide a degree of differentiation. It would have made it more fun for the user and interesting to progress through the levels.

Other Suggestions

1. **The Midway Demo and Final Demo Presentations being Asynchronous**
   - Although we liked the idea of having these demos as part of the project, we felt that they took away two discussion sections that we would have liked to have to address questions about course topics and help prepare more for the final exam.
2. **A Discussion Section Going Over The Basics of Git**
   - During the development stage of our project, we had a lot of issues related to Git, and I feel like a quick refresher on the knowledge of Git in the discussion sections (maybe 10-15 minutes or so) would have really been helpful for the purposes of the team project.