

Bouncing Figures

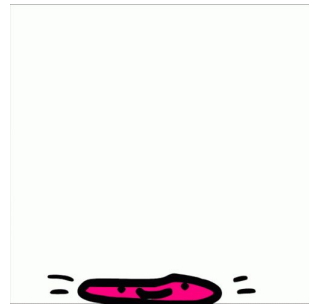
Group 2

Timothy Borunov, Haven Cook, Louis Jimenez-Hernandez, Sergio Rodriguez

Background

Project: Bouncing Figure

- Bounce a 1-million polygon shape at 1 fps
- Deformable upon collision
- User can provide own figure through command line arguments



Features:

- Add 100 fixed (immovable) obstacles to the environment of the bouncing figure
- Add a physics engine to your bouncing shape, including a gravity source and a model for the elasticity of triangle edges
- Bounce around more than one (1-million triangle) figures at the same time. Max of two additional figures.
- Develop this into a game that includes the mouse to affect the motion of the figure and maintains scoring based on object collisions.

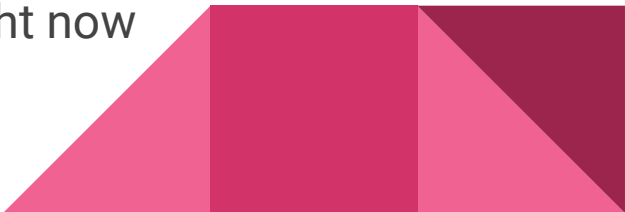


Figures and Obstacles


- Figures done using Triangle Meshes
 - Can have up to 3 1-million triangle figures on screen at a time
- Two types of Obstacles - Immovable and Removable
 - Randomly generated within visible window
 - Fully collidable with figures
 - Interacts with our game



Physics Engine

- Collisions with deformation based on side and axis impacted
 - Acceleration due to gravity and air resistance
 - Implemented as part of each object with modifiable values (Not currently part of the UI, but can be specified per object in code)
 - Modification of triangle mesh with deformations being cached for faster computation.
 - Physics components take $O(1)$ time, but deformation still takes a fraction of $O(V)$ since each vertex position must be modified
 - Could be improved with multi-threading, but none right now
- 

Game

- Interactive Mouse Controls: Utilizing JavaFX's mouse event handling to provide an intuitive user interface, where players can directly influence the velocity and trajectory of objects on the screen. The game employs mathematical functions, to dynamically adjust the camera's perspective, offering a more immersive gaming experience.
 - Scoring Mechanism: Incorporate a scoring system that automatically calculates points for players when their controlled objects collide with specific obstacles.
 - Dynamic Object: After collisions, score-able objects will disappear from the game scene.
- 

UI and QoL (Haven)

- Uses JavaFX 2D graphics libraries for UI, file selection
- Test objects sourced from internet, converted from obj
- Lighting and environment are simplistic JavaFX elements, only there to increase usability
- Command line parser
 - Reads input and sets variables
 - Opens filepath and parses file
 - Runs in $O(n)$ time, where n is number of triangles



Comparison to other projects

- We utilize the JavaFX library which is a pretty lightweight and easy to use choice for 3D graphics.
 - Weaker than other common libraries like OpenGL, and LibGDX
- For physics engine, we decided to use our own algorithms and logic, which gave more control, at the cost of performance
 - Alternative to libraries like JBullet



Demonstration

https://agile.bu.edu/gitlab/ec504/ec504_projects/group2/-/blob/master/demo.myp4





Preguntas?