Intro to CS [Sec 003] Final Project Proposal
Joy Zheng & John Guo
November 18

Tentative Title: Asteroids

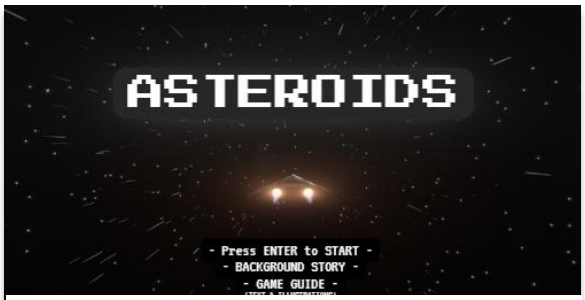
I. Game Concept & Visuals

Our project's core gameplay loop centers on a player-controlled spaceship that operates in a 2D asteroid over the course of multiple waves. The primary objective is to destroy all asteroids using projectiles, being missiles, while at the same time avoiding any collisions with asteroids or their fragments. During the process, the player gradually accumulates points, which serves as the metric of their performance.

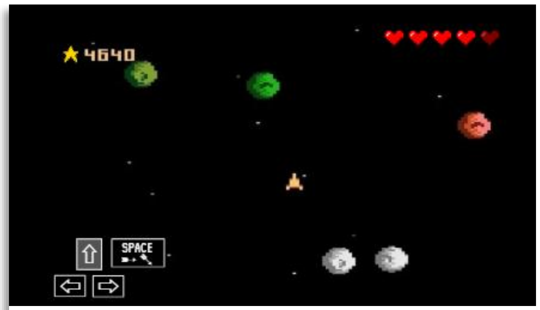
Each destroyed asteroid awards a set number of points, and victory of every individual wave is achieved when all asteroids are cleared from the screen. The player completes the game after they beat 10 unique procedurally generated levels. All game entities, including the spaceship, projectiles, and asteroids, utilize screen wrapping, meaning that they will reappear on the opposite side when they move out of bounds.

We will include hint buttons on the intro screen that explains the controls, such as pressing <u>LEFT</u>/<u>RIGHT</u> arrows to rotate, <u>UP</u> arrow to thrust, and <u>SPACE</u> to shoot. The intro screen will also show a pixel mock-up of the ship surrounded by asteroids, plus options such as "Background Story" When the player wins or loses, the game will transition to an end screen showing the final score and options to restart or quit, satisfying the requirement for clear instructions and the ability to restart the game.

This proposal draws on the inspiration of the classic arcade game with the same title, originally published by Atari in 1979 on their arcade machines. Here is a modern adaptation of
the game, playable online: https://www.arkadium.com/games/atari-asteroids/



1 Demo Intro Screen



2 Demo Main Game

II. Core Gameplay & Mechanics

Upon examining the features of similar existing games, we identified a mechanism that can make the game substantially more engaging over the repetitive loop of gameplay, which is inertia. The game implements a simple physics engine to simulate the effects of inertia in different outer space settings, which constantly changes as the player progresses over waves. The player's ship will maintain its state through a velocity vector (vx, vy) and a rotation angle, instead of reacting directly and instantly to player's inputs, thus creating a sense of uncertainty to make the game more engaging.

- Pressing <u>UP</u> adds thrust in the direction of the angle, changing the velocity; when no thrust is applied, the ship continues drifting.
- We will use circle-based collision detection for bullet-asteroid and ship–asteroid interactions.

- Asteroids are represented as circles for collision detection but displayed as irregular shapes, with textures.

Asteroids are also subject to the impacts of inertia physics, regarding their movement patterns as well as during splitting upon any hit.

- When a bullet collides with a larger asteroid, it is destroyed and generates two smaller asteroids with different velocities. The replacement gives an initial velocity to the new asteroids.
- Different waves generate asteroids in different directions (from the margins of the screen), with a hybrid procedure that combines scripted directions with randomly determined patterns.
- Each asteroid size gives a different score and awards the same number of coins: large = 10, medium = 20, and small = 30. The score and the number of coins is displayed in the top corner in pixel-style white text.

Each of the 10 waves takes place in unique environments set in the outer space, with different background elements, ambience, and inertia settings.

- The player wins a singular wave when all asteroids are cleared from the screen. When one wave is cleared, a new wave with more or faster asteroids appears, increasing difficulty.
- As the player progresses into the second wave, they will unlock a shop system, which allows them to gradually acquire a monetary currency of coins (different from the score, which ought not to be spend) to afford props and modifiers to enhance their abilities.
- Player will gradually unlock more options of purchasable items as they clear out more waves. For instance, spend 1000 coins to acquire an additional heart, 2000 to thrust more efficiently, or 3000 to but a new "skin" appearance for the spaceship.
- If the ship collides with any asteroid, the player loses one of their initial 5 hearts. When all the hearts are deducted, the player loses the game, and a "Game Over" screen will be shown with the score and a prompt to click to restart.

We will define at least four main classes: Ship, Asteroid, Bullet, and Shop. The main game loop will manage lists of asteroids and bullets, update them each frame, and handle drawing and collisions. Keyboard-based inputs (e.g., LEFT/RIGHT/UP/SPACE, Press ENTER to Start, B for the shop menu) will control the main game, and mouse-based inputs (e.g., Select menu options and purchase items in the shop menu) will navigate between pages. Simple sound effects will be included (e.g., thrust noise, shooting sound, explosion when an asteroid breaks) using imported sound files.

III. **Responsibilities**

1. Joy's responsibilities:

- The main game loop that controls the flow of the game, and handle keyboard inputs.
- Define and program the Ship Class, including its properties of position, velocity, angle, etc.
- An inertia-based physics engine for the ship's movement.
- Visual and audio assets to create and utilize in the game, such as the ship, asteroids of various sizes, and the projectiles.

.

2. John's responsibilities:

- Asteroid, Bullet, and Shop classes programming. Including properties and the complete life cycle of asteroids, from spawning to interactions.
- Implement the global circle-based collision detection system, as well as consequences of different collisions.
- "Game Over" state, and the system of restarting the game after one session ends.
- Score system and win condition, as well as a difficulty system that determines the waves of asteroids.