1979 Atari Game: Asteroids

Skylan Recana
CSE | BS Computer Science Student
University of Minnesota Twin Cities
13th of January 2021

1 Introduction

Atari Game is a classic 1979 asteroids game where the rocket must destroy the asteroids and the saucers in the asteroid belt. The rocket can thrust, stop, shoot, and maneuver around the asteroid belt. The difficulty increases as the level progresses. The rocket must avoid being hit and shot by saucers and asteroids to beat the best score. The practice project enhances the ability to code in HTML and CSS, but mostly JavaScript while delivering the Atari Game successfully.

2 Explanation

2.1 Creation of HTML

Set the meta charset to “UTF-8”, add the title of the webpage, and use CSS to change the background color of the website as well as to modify the margins. The body contains the canvas and the index.js file.

2.2 Creation of Canvas element (30-34 js)

The Canvas for the website can be created inside the index.html file. This is done by using ¡canvas¿ while customizing its width and height and defining the id for reference inside the index.js file. The canvas can be accessed in index.js by using getElementById() and the context of the canvas is in 2D, which can be obtained by using getContext().

2.3 Creation of Space (356-358 js)

The canvas will be used to create the space for the game in a loop to create an animation, where each frame uses different black rectangles to visualize the space while preserving the smooth transition of moving objects in space.

2.4 Creation of Key Bindings (157 and 178 js)

The functions keyDown() and keyUp() are needed when the player pushes and releases a button respectively. The function keyDown() features the ability to shoot lasers when spacebar key is pressed and maneuver the rocket by using the up, left, and right arrow keys (down key is not applicable when running rockets). On the other hand, the function keyUp() determines if such actions in different keys are needed to be modified once the function keyDown() is pressed again.

2.5 Creation of Initial Game (219 js)

The function newGame() is assigned when the game is over, when a new level is needed, or when the program just started. In every call, the function calls a new rocket, stores the high score in the disk, and calls a new level.

2.6 Creation of Triangular Rocket (241 js)

The rocket is an object that composes of x and y-coordinates; rocket size and angle; blink number and blink time for respawns; Boolean if the rocket can shoot, is dead, and can thrust; array of lasers, the thrust coordinates, and the rotation of the rocket.

2.7 Illustration of Triangular Rocket (120 js)

The rocket will be drawn using the context obtained from the canvas. It uses methods moveTo and lineTo to move the points while using trigonometry to draw the rocket.

2.8 Creation of Asteroids (198 js)

The asteroids will have the x and y-coordinates and velocities as well as the angle, vertex, and offsets to create asteroids with different vertices and shapes.

2.9 Creation of Asteroid Belt (59 js)

The asteroids will be spawned inside the creation of the asteroid belt. The number of asteroids being spawned depends on the number of levels the player has successfully played. The asteroids will be placed far from the ship initially to ensure that the rocket will not die at the start of the game.

2.10 Creation of Level (235 js)

The level calls the createAsteroidBelt() to create a new set of asteroids with updated number and speed. It also provides a text to show what level the player is currently in at the start of each level.

2.11 Destruction of Asteroids (75 js)

The asteroids will be destroyed when they are hit or shot by the rocket. If not small enough, the asteroids when destroyed will create two small asteroids from the ruined asteroid with different starting positions and velocities. The score will be calculated when an asteroid is hit or shot; the smaller the asteroid, the larger the score to be obtained. This will also remove the asteroids in the space once they cannot be divided into halves anymore. A sound will be played when an asteroid is hit or shot.

2.12 Explosion of Rocket (146 js)

The explodeShip() function calculates the explosion time of the rocket. When the rocket explodes, a sound will play.

2.13 Execution of Shooting Lasers (262 js)

The lasers to be shot are created depending on the length of the rocket's magazine. The laser objects will be added in magazine with positions and velocities calculated where they will spawn on the nose of the ship.

2.14 Execution of Music Background (282 js)

The function composes of low and high beats, where the beats are playing alternately, while the tempo increases slightly when there are less asteroids floating in the space.

2.15 Execution of Sound (327 js)

The function composes of the source, the max streams, and the volume. The Sound() function is the improved Audio when playing the sound simultaneously depending on how many lasers are being shot. It uses a list to store each sound and play them when play method is called.

2.16 Execution of Update (349 js)

The function updates the screen and is responsible for the animation as well as the update on the variables needed for each object. It also composes of the creation of thruster as well as the blinking, explosion, bounding, illustration of asteroids, illustration of lasers, game text, game lives, game score and high score, collision, and edge loops. The function also calls several functions defined within the scope.

3 Future Work

Add gradient colors on asteroids, change the structure of the rocket, add more vertices on asteroids, change the speed of the asteroids, change the background color, add saucers, add powerups.