PES UNIVERSITY

Bengaluru – 560085

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Session: Aug– Dec 2017

Computer Graphics & Visualization

UE15CS321

5th semester Elective-2

2015 - 2019

Project Report

STAR WARS

Team :

1. Rutha Prasad 01FB15ECS367

Table Of Contents:

1.0 Introduction:

1.1 From Arcades, Consoles and PCs to Web-browsers, Mobiles and Virtual Reality, Gaming has transcended across various platforms since the 1980s and across various dimensions of 2D and 3D.

As graphics and peripheral technologies have developed radically, development of Modern Games is easier, more efficient and more competitive.

Although, the gaming industry also faces a lot of issues such as; the modern games which are upcoming need better graphic cards; most games require lots of storage and end up burdening disk-ware; nowadays the shift to the new paradigm of online-gaming demands better interfaces and net support.

Many softwares such as Unity, DirectX, Stencyl, Panda3D are used as gaming softwares. iOS, Android and Windows also develop their own inherent gaming IDE based. OpenGL, WebGL, Javascript GL,HTML5 Canvas and other graphic libraries provided by Programming languages are also used extensively for basic implementations.

Gaming is one of the booming software industries as demand of entertainment and recreation has only increased.

1.2 A 2D game is developed in this project to understand how graphics and gaming concepts work together.

The game is a single level game with a single player mode. A score has to be reached and the highest score is recorded. A spaceship which is 2D object is controlled by the user and must shoot down enemy ships before it is shot down, to win the game. It is similar to the 90s point-and-shoot arcade games.

It is implemented in offline HTML. The basic code will be implemented using Java Script and HTML5 Canvas to build game strategy and user functions. The Java Script DOM will handle the graphic interactions, models and frame functions. Styling and layout is managed using HTML5 and CSS.

2.0 Literature Survey:

2.1 This game was inspired by the 90s arcade game, Galaga. *Galaga* is one of the most commercially and critically successful games from the golden age of arcade video games. The arcade version of it has been ported to many consoles, and it has had several sequels. These games were usually designed with Flash/Java/DHTML and run directly in web-browsers.

2.2 Arcade racing games have a simplified physics engine and do not require much learning time when compared with racing simulators. Similar games like Space Invaders, Chicken Invaders were developed on OpenGL for PC platforms.

2.3 The game in this project is developed in HTML and Javascript and based on the Web Browser platform. It works offline and does not need internet to function, but uses WebGL which is a web-based GL, as a basis for implementation of the JavaScript graphics functions.

2.4 This is advantageous as we do not need to configure our workstation hardware to support WebGL or OpenGL.

2.5 It is disadvantageous in the sense that it does not offer the various functionalities and graphic speeds offered by OpenGL/WebGL and other such platform independent GLs.

2.6 Java Script DOM, used in this project offers a wide range of object oriented programming for game development. This increases the level of development and experimentation as implementation of code is very easy.Thus implementing your game idea is made easy with JS DOM scripting.

2.7 HTML5 canvas offers 2D graphic speeds and methods that are the best in web-based gaming platforms. Although Canvas has a better developed programming interface for users, WebGL is still preferred for serious online-game development and 3D gaming.

2.8 The features of graphics-and-gaming inter-dependency we can see from this game are:

- Configuring optimal fps required for smooth real-time gaming
- Rendering models and objects to Viewport using inbuilt drawing algorithms
- Decreasing required raster scans such that graphic speeds can increase.

2.9 Features offered by Javascript DOM and HTML Canvas:

- When a web page is loaded, the browser creates a Document Object Model of the page. With the HTML DOM, JavaScript can access and change all the elements of an HTML document.
- The HTML DOM is a standard object model and programming interface for HTML. It defines:
  - The HTML elements as objects
  - The properties of all HTML elements
  - The methods to access all HTML elements
  - The events for all HTML elements
- The Image object represents an HTML <img> element. The Image object also supports the standard properties and events.
- The HTML <canvas> element is used to draw graphics on a web page.
- The HTML <canvas> element is used to draw graphics, on the fly, via JavaScript.
- The <canvas> element is only a container for graphics. You must use JavaScript to actually draw the graphics.
- Canvas offers several methods for drawing paths, boxes, circles, text, and adding images.
  - canvas.getContext("2d");//initializes a 2D canvas
  - ctx.moveTo(0,0);//moves origin
  - ctx.lineTo(200,100);//starts line from point
  - ctx.stroke();//draws a line
  - ctx.beginPath();//begins a line drawing algorithm
  - ctx.arc(95,50,40,0,2*Math.PI);//draws an arc
  - ctx.font = "30px Arial";//can set font style
  - ctx.fillText("Hello World",10,50);//can print out characters as graphics
  - ctx.createLinearGradient(0,0,200,0);//polygon filling algorithm
  - grd.addColorStop(0,"red");//different colours
  - grd.addColorStop(1,"white");
  - ctx.fillStyle = grd;//offer to choose solid fill or only outline
  - ctx.fillRect(10,10,150,80);//renders a rectangle
  - ctx.createRadialGradient(75,50,5,90,60,100);//can create radial gradients
  - ctx.drawImage(img,10,10);//can render an image onto the canvas in x,y

3.0 Methodology:

3.1 Softwares used for game development :

- HTML5
- CSS
- Java Script DOM
- Java Script GL

3.2 Game Design consists of 5 unique parts:

- Part 1 – Setting up the structure of the game and panning a background
- Part 2 – Create the player controlled ship and it's properties (move and shoot)
- Part 3 – Create the enemy ships
- Part 4 – Collision detection
- Part 5 – Audio and final touches

3.3 Data Structures used:

- Javascript objects (defined using DOM) are Enemy,Ship,Bullets,EnemyBullets. They act as data structures holding all information related to the models that are rendered on the screen. Such as x-y coordinates, speed of change in coordinates, width, height, etc.
- A Quad-Tree is used to split the viewport into a matrix of viewports, which are again divided into another set of matrixes. This is a recursive data structure (like a Tree data structures with roots, and children nodes). It helps in choosing which part of the viewport needs to be scanned for the detection of collision, instead of scanning the whole viewport. This increases graphic speed.
- Other peripheral data structures such as repositories hold the set of images and audio samples used in constructing the "world" of the game.

3.4 Main Algorithms:

- Game_init : initializes the models and viewport and renders all the animated models and objects onto the screen and starts the game.
- Draw: each data structure object has a prototype of this function. It is used to draw the model/object onto the canvas and display it according to the viewport.

- Quad-Tree: is a background function that runs continuously to detect any collision between the ship object and bullet objects. The functions runs on each fps.
- Move: function that defines the movement of the graphic object within the canvas such that it is bounded by the viewport dimensions. It keeps track of keyboard entries such that movement can be controlled by user/gamer.
- Animate: optimizes the Draw function by defining the fps.

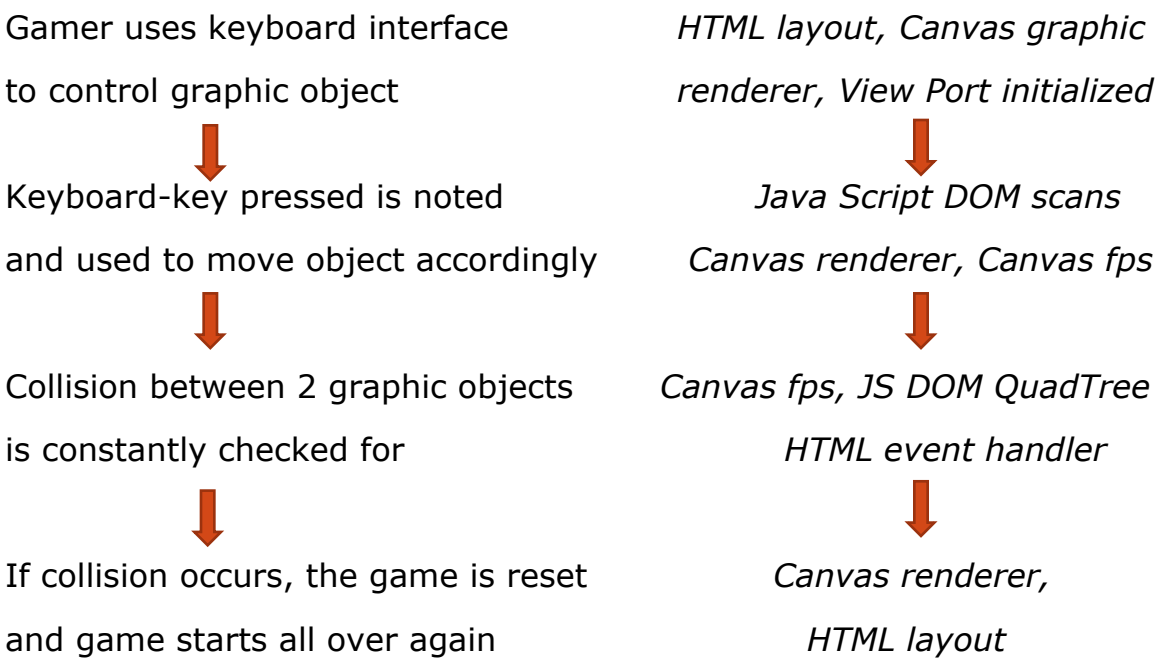
3.5 Front-end:

- Basic CSS styling and HTML tags are used to set the basic layout of the gaming environment, defined in the game.html file.
- <canvas>,, , window.innerWidth, position, z-index, etc are some of the HTML tags and CSS fields used.

3.6 Back-End:

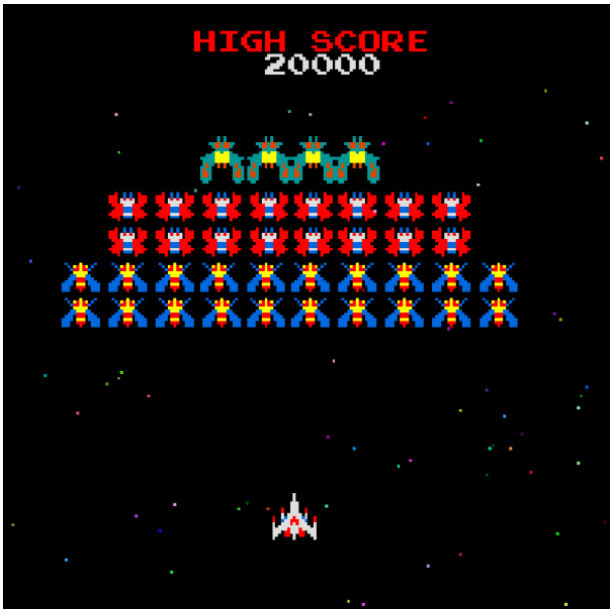
- Gaming implemntation and DOM functions are defined in the the javascript file code.js and imported into the HTML file.
- Javascript uses functions of the canvas element such as drawImage(), newImage(),contect(),.getObjects() ,etc are some of the DOM methods and GL methods used.

3.7 Flow Diagram:

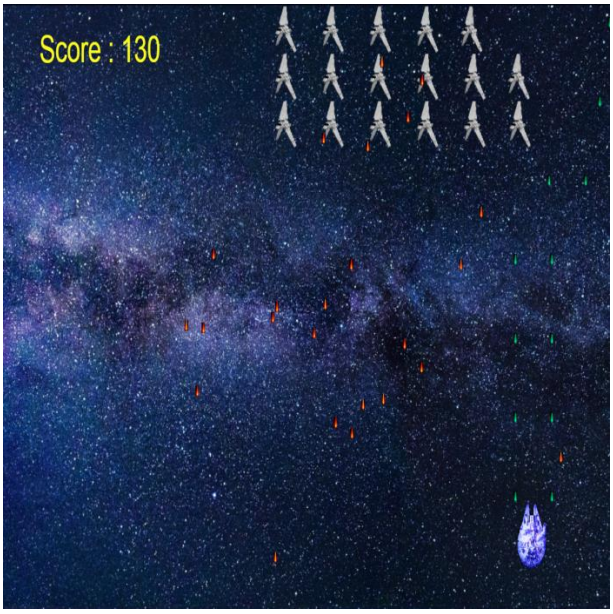


4.0 Results:

4.1 Goal:



4.2 Achieved:



5.0 Conclusion:

5.1 Goals set for this project included :

1. Develop HTML5 game using only native JavaScriptGL functionality to see the capabilities of JavaScript and the Canvas element.
2. Code must be web optimized and address real issues of fps optimization and overloading GPU.
3. Game must maintain an average frame rate between 50-60 FPS.
4. The Game must load and be ready to play in less than 1 minute.

In conclusion all these goals were achieved.

5.2 Even though 3D graphics cannot be achieved with these tools, they offer a huge variety of 2D tools.JavaScript GL and HTML5 Canvas , help in rendering web graphics with very less effort and boost web-based graphic operations and industries. This is also without overloading the workstation and allowing portability of the programs.

This can also be connected to the current trend of shifting from console/offline gaming to online gaming.

6.0 Bibliography:

https://www.w3schools.com/html/html5_canvas.asp

https://www.w3schools.com/html/html_media.asp

http://devmag.org.za/2009/04/13/basic-collision-detection-in-2d-part-1/

https://en.wikipedia.org/wiki/Collision_detection#Video_games

https://en.wikipedia.org/wiki/Space_partitioning

https://gamedevelopment.tutsplus.com/tutorials/quick-tip-use-quadtrees-to-detect-likely-collisions-in-2d-space--gamedev-374

https://en.wikipedia.org/wiki/Quadtree

https://www.slideshare.net/ernesto.jimenez/5-tips-for-your-html5-games

https://www.html5rocks.com/en/tutorials/canvas/performance/#toc-mul-canvas

http://devmag.org.za/2009/04/13/basic-collision-detection-in-2d-part-1/