COMP330 Assignment 1 Report

Name: Benjamin Rendell
Student ID: 44655010

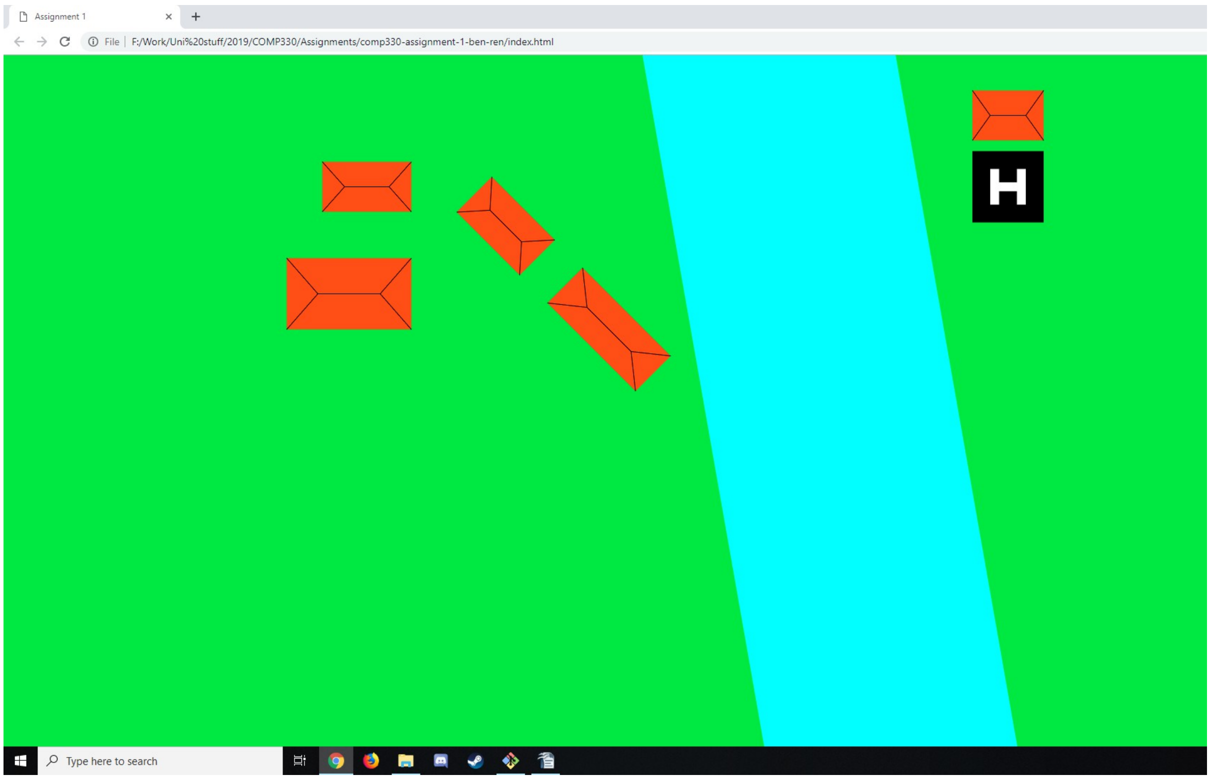
Features implemented in this assignment:

Feature	**Mark**	**Check if used**
Static 2D world Buildings, river, helipad	20%	Check
Moving helicopter with keyboard control	20%	Check
Helicopter with spinning tandem rotors	20%	Check
Rescuing people	5%	
Resizing the canvas, maintaining aspect ratio	5%	
Control helicopter with the mouse	10%	
Camera mounted on the helicopter	10%	
Minimap	10%	
Curved Rivers	10%	Check
Rain particle effect	10%	Check
Heads up display	10%	Check
TOTAL (max 100%)		

On the following pages you should indicate where each of the above features appear in your game, using screenshots and filenames/line-numbers to indicate where it occurs in your project.

You will not get marks for a feature if your marker cannot easily locate it within your world.

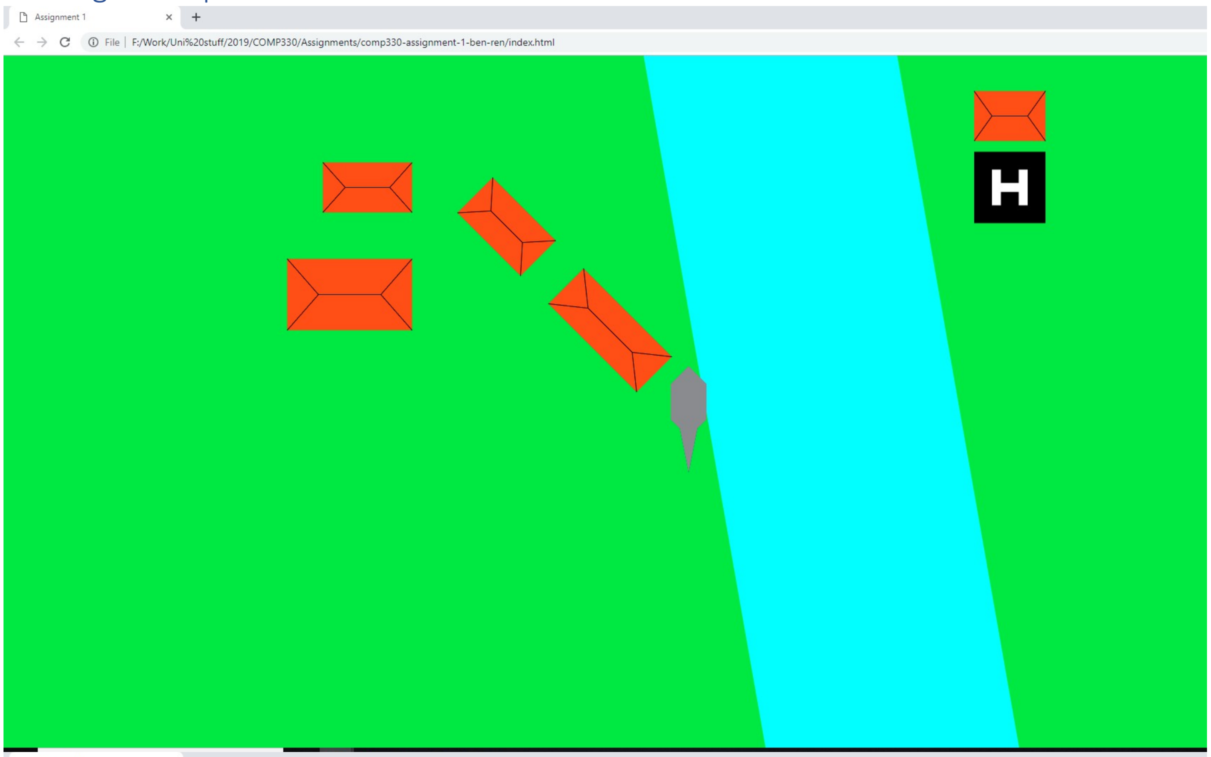
Static Terrain

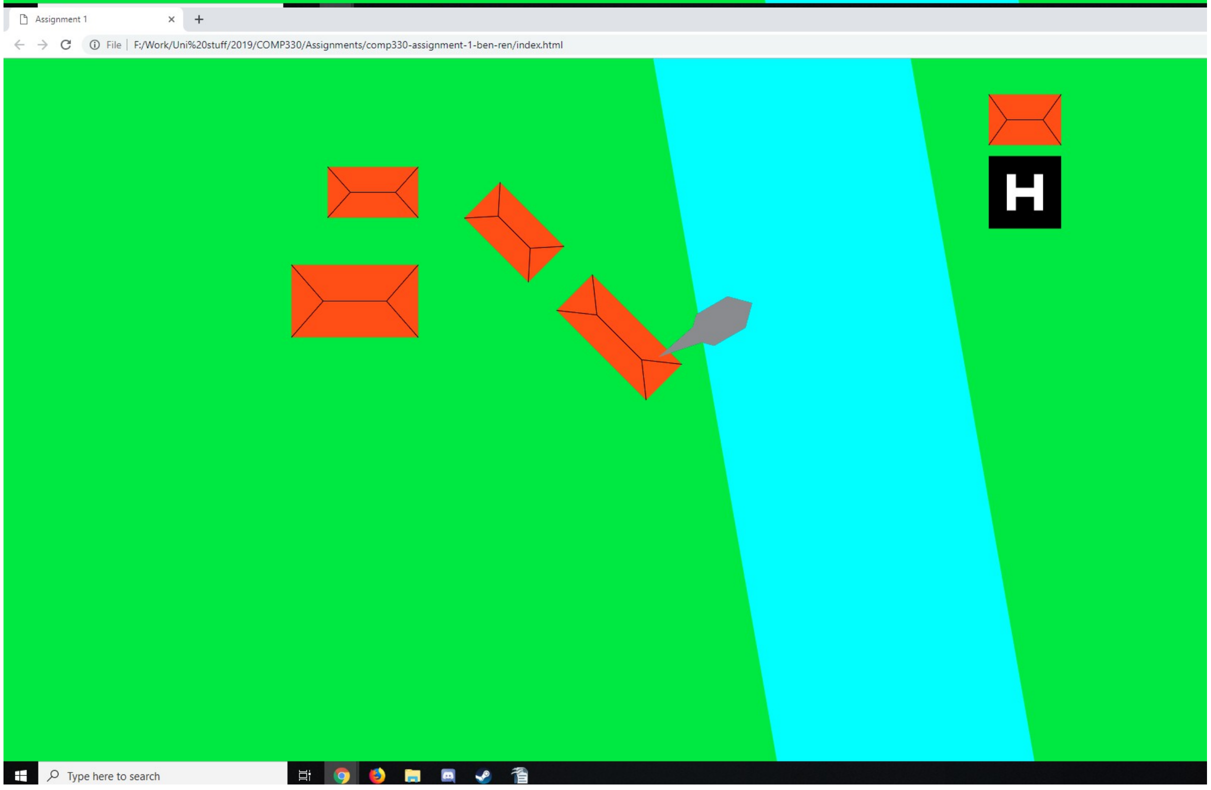

Assignment 1
File | F:/Work/Uni%20stuff/2019/COMP330/Assignments/comp330-assignment-1-ben-ren/index.html
H
Type here to search

Implemented in:

- game.js: 8-117 webgl & js code for drawing fragment and vertex shaders in the browser. Additional browser functionality is also added from week 5 practicals.
- game.js: 121-162 implementation and rendering of terrain
- terrain.js: 10-22 instantiation of colours and objects
- terrain.js: 24-50 transformation and rendering of objects
- box.js: 19-37 drawing the primitive box using triangles
- box.js: 40-52 conditional for drawing rooves drawn using lines
- box.js 55-64 conditional for drawing helipad drawn using triangles

Moving Helicopter and Controls

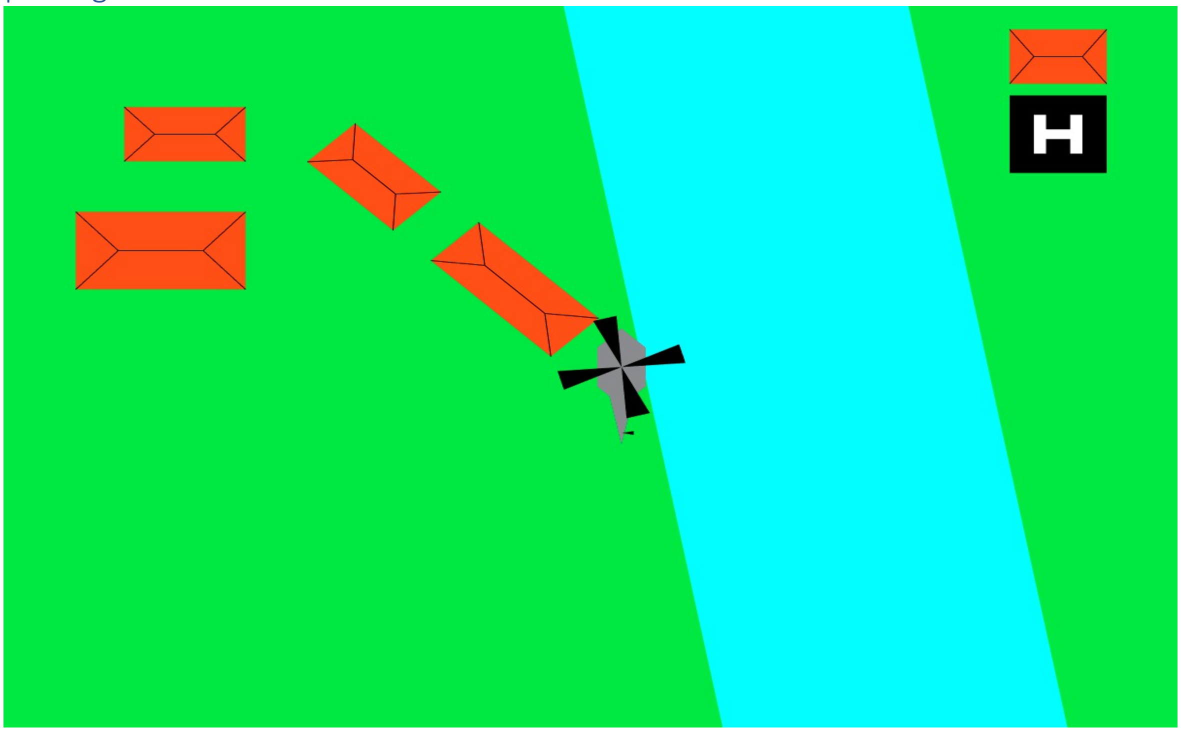

Assignment 1
File | F:/Work/Uni%20stuff/2019/COMP330/Assignments/comp330-assignment-1-ben-ren/index.html
H


Assignment 1
File | F:/Work/Uni%20stuff/2019/COMP330/Assignments/comp330-assignment-1-ben-ren/index.html
H
Type here to search

Implemented in:

- game.js: 121-162 implementation, updating and rendering of helicopter as chopper
- input.js: 12-48 altered input.js to register keypresses
- helicopter.js: 19-42 updates and controls helicopter position using cosine and sine based off of the helicopters current rotation.
- helicopter.js: 45-74 draws helicopter body

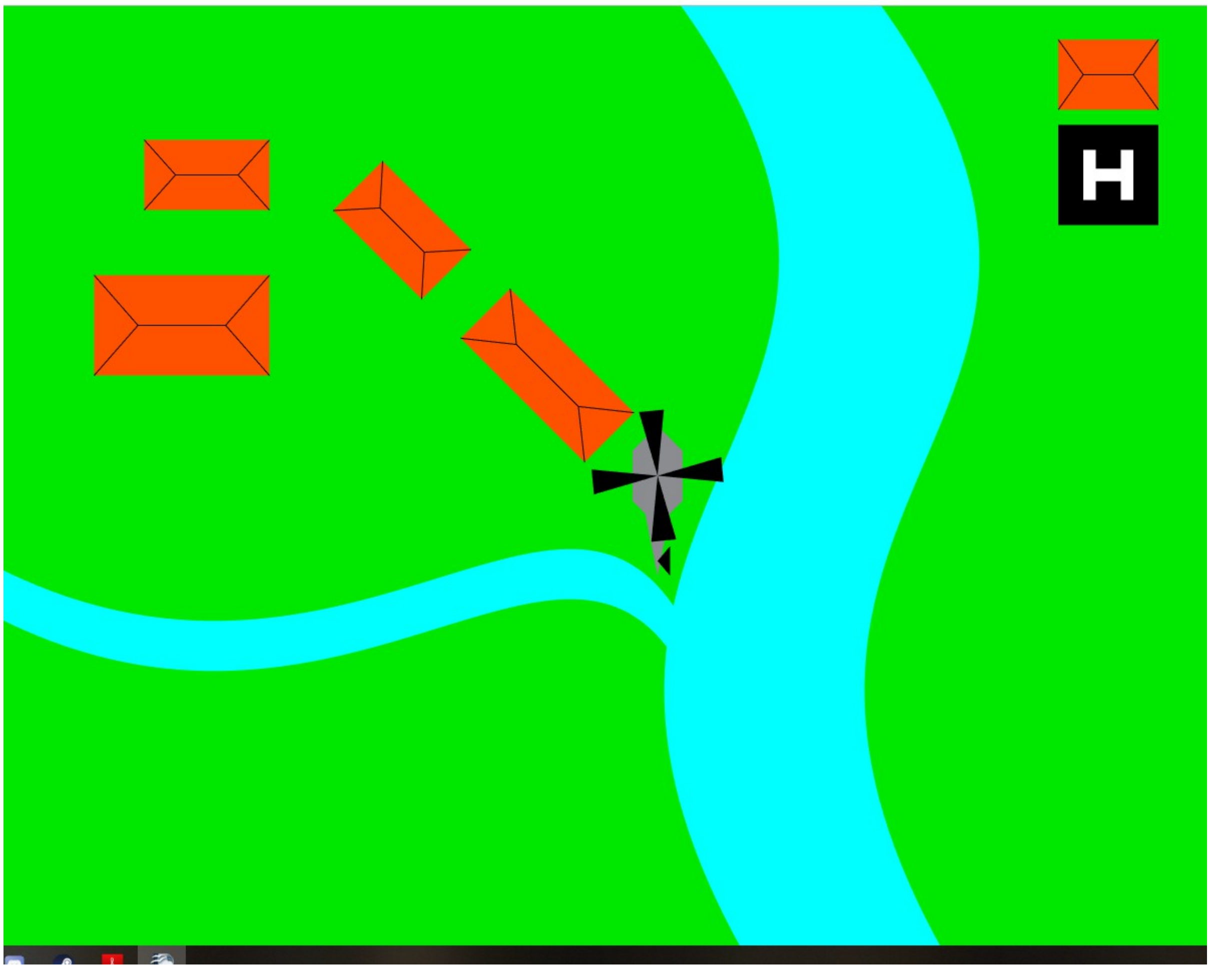
Spinning Rotors



Implemented in:

- game.js: 121-162 implementation of children
- helicopter.js: 41-42 added updateChild functions to the parent's update function
- helicopter.js: 72-74 added children rendering to parent
- helicopter.js: 78-112 added mainRotor child class
- helicopter.js: 114-146 added backRotor child class
- helicopter.js: 88-111 render mainRotor
- helicopter.js: 125-145 render backRotor
- helicopter.js: 83-86 rotate mainRotor in updateChild function
- helicopter.js: 120-123 provided illusion of backRotor rotating by scaling between 0 & 1 using cosine in updateChild function.

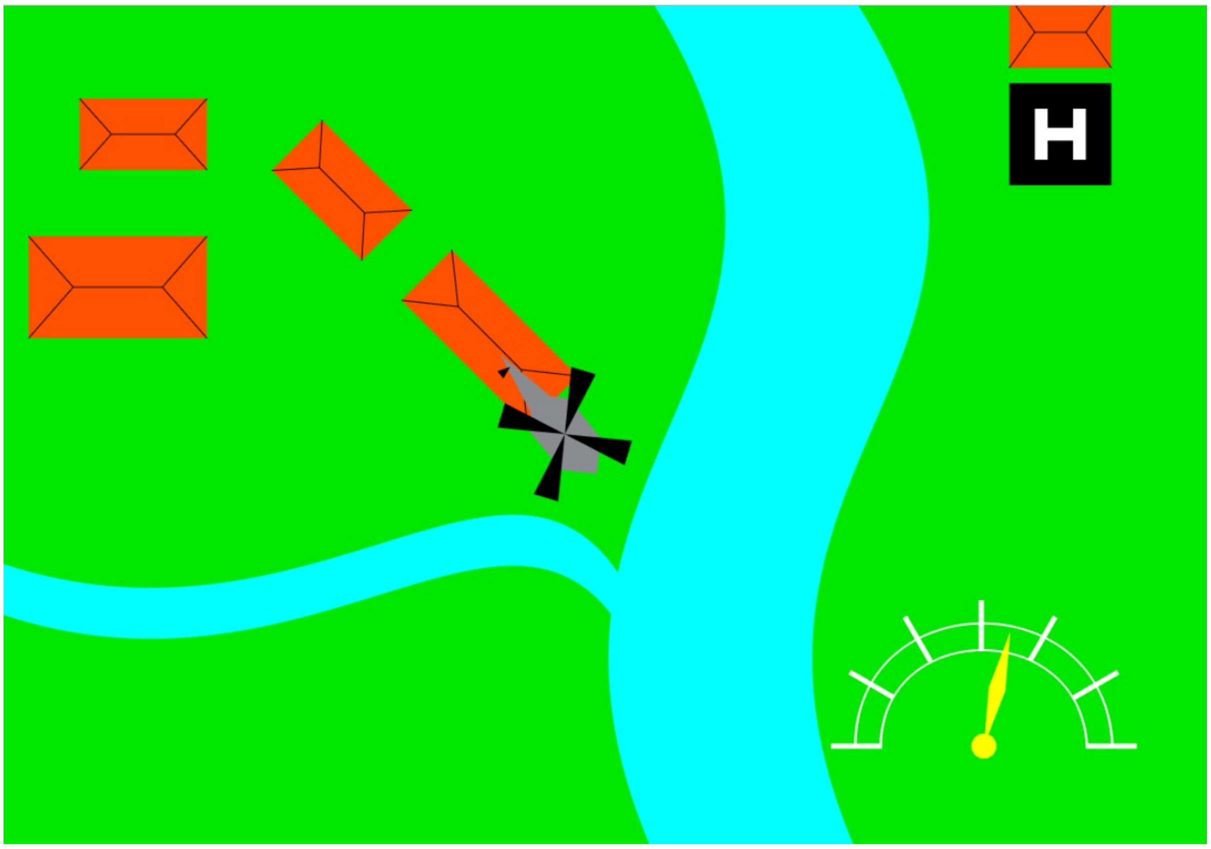
Curved Rivers



Implemented in:

- terrain.js: 21-90 implementation of control point arrays & bezier objects as rivers. Renders the rivers.
- bezier.js: 52-67 function that calculates bezier curve using a 2D array of input control points
- bezier.js: 33-44 applies each calculated x, y point and the curves offset into an array for drawing triangle strips.
- bezier.js: 46-50 draws the bezier points into a triangle strip.

Heads up Display

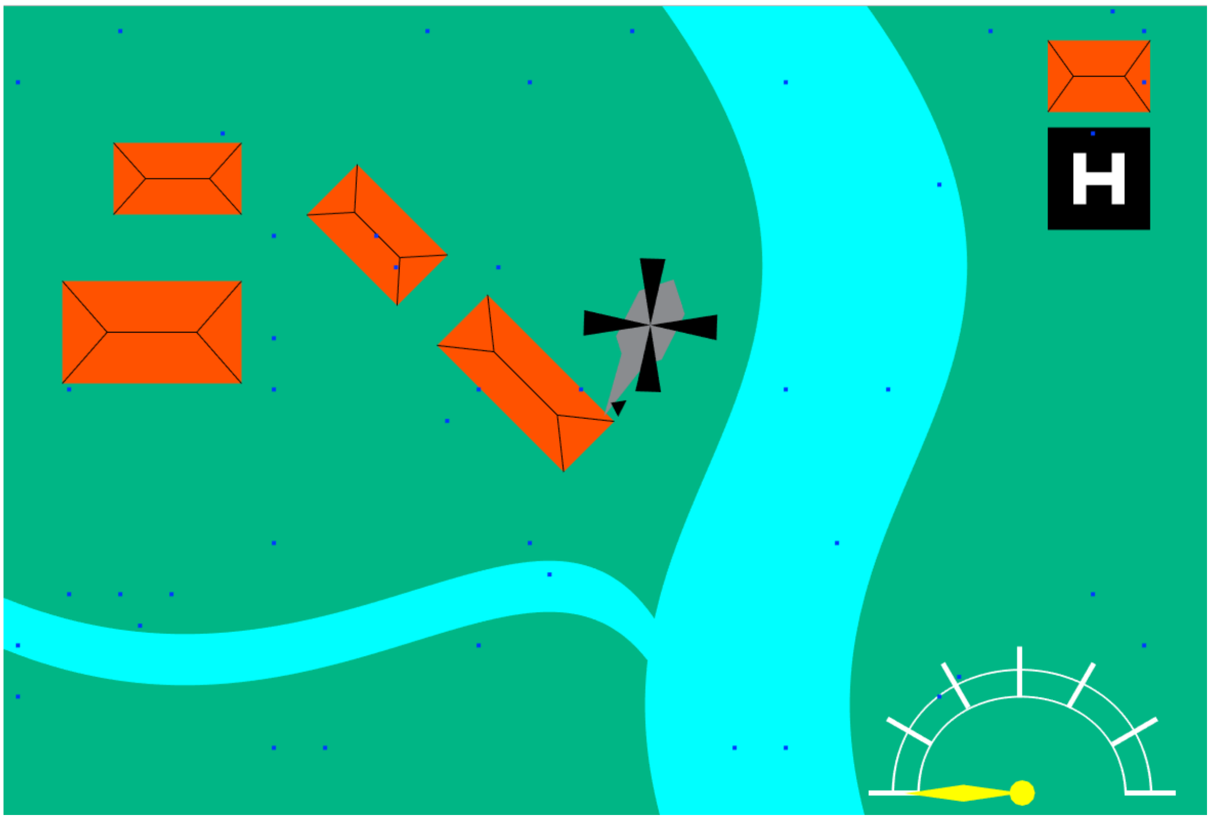


Implemented in:

- game.js: 121-162 instantiates, renders and updates hud & needle objects.
- helicopter.js: 9-39 added acceleration to helicopter to be accessed from the hud.
- HUD.js: 6-124 HUD class
- HUD.js: 126-173 Needle class as child of HUD class.
  - Note: childing the needle to the hud may not be necessary.
- HUD.js: 77-111 draws arcs using bezier curves
- HUD.js: 27-122 transform and render ticks and needle
- HUD.js: 133-141 update needle rotation based on helicopter movement and acceleration.
- HUD.js: 144-172 draws the needle
- circle.js: 19-26 sets the points for the needles circle
- circle.js: 29-45 draws the circle using a triangle fan

Note: circle.js code was obtained from the week 5 practical and modified for the program.

Rain particles



Implemented in:

- game.js: 27 changed PointSize value in shader to render visible points
- game.js: 121-162 instantiate, update and render rain & other_rain
- points.js: 3-15 construct positions array and reset_pos
- points.js: 17-25 moves particle positions and deletes points from random positions
- points.js: 26-30 resets particles to imitate additional rain falling
- points.js: 31-33 repopulates positions array
- points.js: 37-56 renders the points
- points.js: 62-66 getRandomInt creates the random integer values used for repopulating the array & for randomising the particle translation & position.