Computer Graphics - Assignment 4 - Snake

CS 4830 — Dr. Mihail
Department of Mathematics Computer Science
Valdosta State University

February 11, 2019

1 Introduction

A video game is any software that interacts with a user who can manipulate imagery produced and displayed electronically. This definition is vague and encompasses quite a bit of software. Traditionally, we think of video gaming in the context of human-machine interaction that generates audio-visual feedback and pleasure through rewards (e.g., winning). Thus, the important characteristics of video games are: interaction, visual (and possibly audio) feedback and rewards.

In this assignment you will implement a classic game: Snake. There are many variations of the game, however, the basic idea is that a hungry "snake" moves continuously (visual feedback) in one direction and looks for "food" on finite board. Each time the snake "eats", its body grows and a score increases (reward). As it moves, the player can change its direction to either left or right. The goal of the game is to maximize the score through the number of food items eaten. This goes on until the snake either collides with itself or one of four edges of the world.

You will implement this game in HTML5 and Javascript. The game world is a grid and its size varies according to the game flavor. In this assignment, your game board will be a grid sized 30 rows by 40 columns. The snake is made up of snake elements, discrete blocks of snake matter. When the game starts, the snake has 4 elements, including its head. At each time increment (discussed in the technical recommendations section), the snake advances by one grid block in it's current direction, unless directed by the player. The player cannot stop or pause the snake. At any point in time, there has to be a randomly positioned food item on the board. Once the snake's head is in the same position as the food, it is considered eaten and another shows up randomly on the board. The new food cannot appear on a block currently occupied by the snake's body.

2 Implementation Details

Grid-to-Viewport You are given a skeleton HTML file with a canvas of size 600x800 pixels. The first challenge you have is how to scale your game grid (30x40) to pixel space. There are many ways to do this, it is your choice how to implement the solution. One possibility is to work directly in pixel space and compute the size of one grid element in pixel space (600/30, 800/40) or 20x20 pixels. Another possibility is to use Canvas' transformation stack such that one unit in the Canvas corresponds to one unit in your world grid. Either way is acceptable.

Snake Movement At each time increment, the snake's head will move in the direction it was heading, unless, the player changes it. One possible way to think about the movement of the snake is to consider that each snake element is trailing the element in front of it. Thus, if the head moves, the element behind the head moves into the head's old position, the third element from the head moves in the position previously occupied by the second, and so on.

Data Structures You have to decide how to keep track of the game state. At a minimum, you should keep track of where the snake elements are and where the food is on the grid. When deciding on data structures to use here keep in mind that you have to implement collision detection. There are three types of collisions you have to detect here: when the snake's head collides with food, when the snake's head collides with another part of itself (game is over) and when the snake goes outside of the world grid (game is over).

Interaction The user interacts with the game via keyboard only. The skeleton program implements functions that are called when the user presses a key. The ASCII codes for the arrows are given, so all you have to do is implement the logic. Keep in mind that user interaction in Javascript/HTML is event driven, hence the events can occur independently of the game state or time slices.

Game Loop and Time The skeleton program implements a game loop that calls a function 60 times/second. You may experiment with different frequencies, however, keep in mind that user input and snake movement have to be handled properly, otherwise you introduce bugs. The game has to be fun to play.

Imagery In HTML5/Javascript resources such as images are loaded asynchronously. This creates a few complications for the programmer, as we have to handle the situation where we have to wait for a resource to load. A simple way to do this is to wait indefinitely until all the resources have been loaded. This is risky, because if any one resource fails to load (HTTP is not the most reliable protocol, and is designed to not interrupt when failure of resource loading occurs) then the program hangs indefinitely.

3 Due Date

This assignment is due before midnight on Sunday, February 17th

4 Grading Rubric

- Snake moves properly (20%)
- Snake movement can be controlled properly (20%)
- Snake collision with food causes score increase (20%)
- Score increases with food (10%)
- Snake collision with food causes snake size to increase (10%)
- Snake collision with itself causes game over (10%)
- Snake collision with border of grid causes game over (10%)

5 Bonus

- 20% bonus: distinct levels implemented (e.g., upon successful growth to X snake elements, there can be more than one food item at any time)
- 20% bonus: different types of food cause different outcomes
- 10% bonus: different imagery for snake's head, body and tail