Brick Breaker Detection

Marko Vukotić

Introduction

We will be using written code to determine number of hits with a ball to a wall. Game functions like this: There is a moving pad on the bottom of the screen that bounces the ball. Ball can bounce of bricks and left, right and upper wall. Goal of the game is to break as many bricks as possible by bouncing the ball with the moving pad.

This is one frame of brick breaker game.

Data

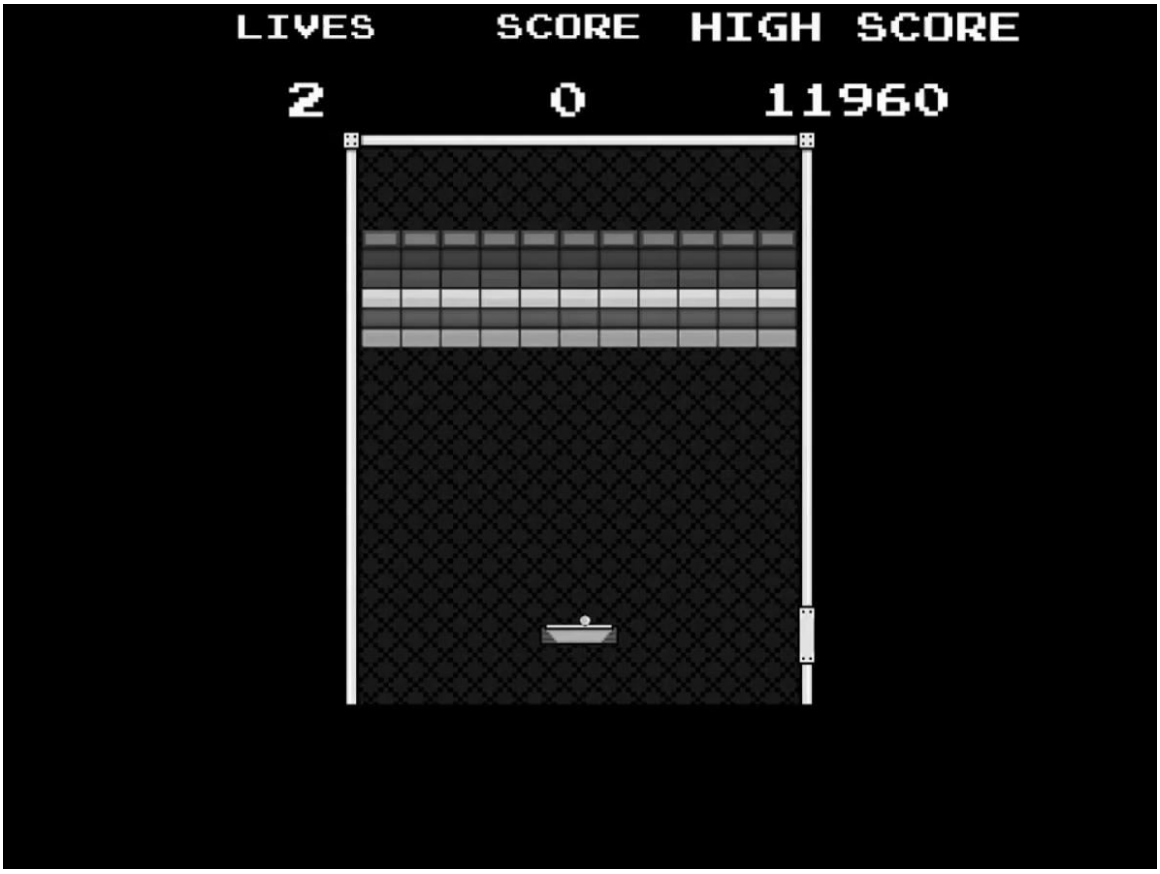
Dataset contains ten clips from the game (gameplay) in which the ball is moved and hit with the moving pad. We also have the txt file containing number of wall hits in each video.

Goal

Goal of project is to count with a program as many hits as we can. Goal is to reach the numbers from res.txt file. We also have the goal to achieve MAE lower than three.

Solution

First step that I did was to cut the videos into frames and then use frames for later analysis.



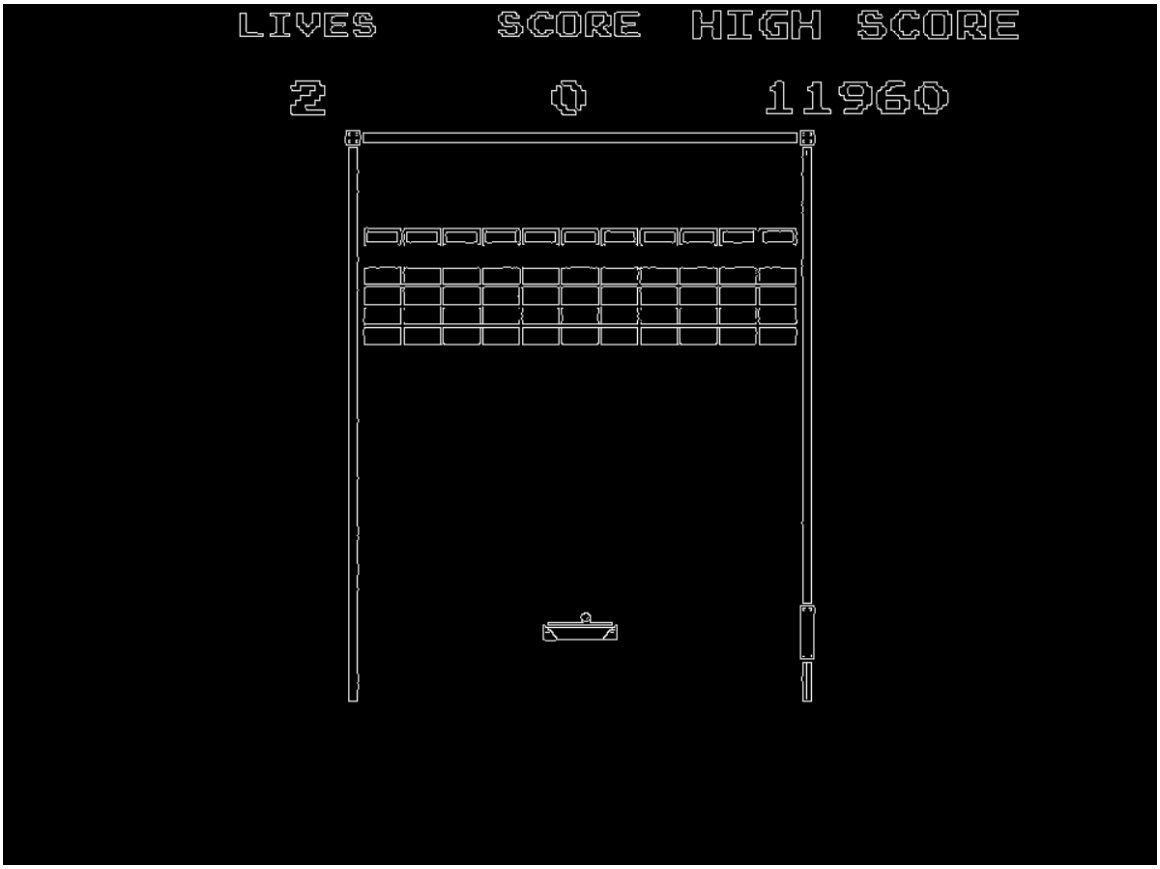
Each frame will be gray after looping.

After that we need to find walls. Walls have same exact position in each frame so we did this only once.

Then we look for the balls in the picture, this has to be done frame by frame since the ball is moving in game.

Methods used

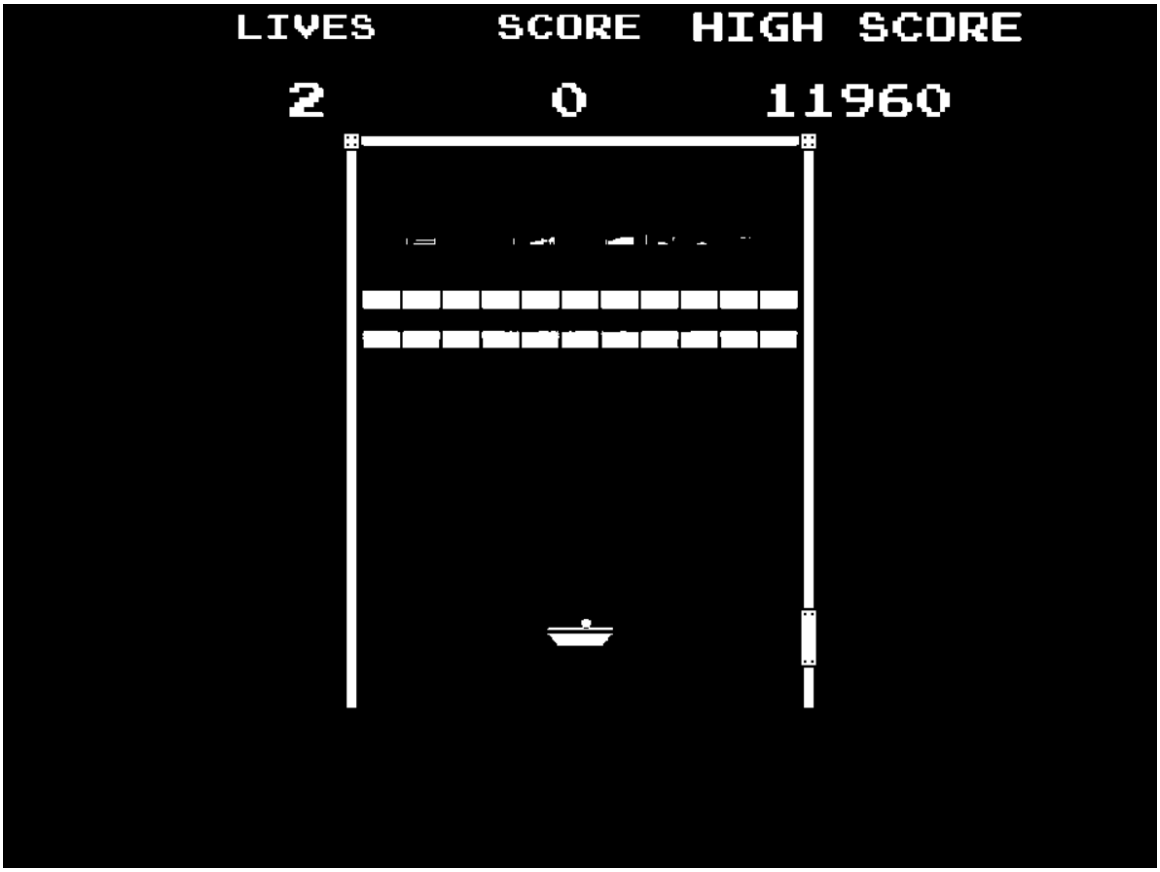
Canny edge: We are using canny edge to get binary image with edges only.



Frame after canny edge.

Hough transform: This method is used to get all lines on the frame. Then we filter the lines by looking for the furthest line to the left and right.

Threshold: Threshold is used to get binary image. Then we will be able to extract balls from the picture.



Binary picture after threshold. (Note, the ball is easily accessible here with findContours()).

The code

Code goes like this:

- We extract frames from the videos with method get_frames()
- We get game edges with Hough lines and their coordinates
- Then we iterate trough frames finding all balls in those frames
- Then we count how close is ball to the wall and if it's close we take it as a hit
- We write results into dictionary for each video
- We convert dictionary to the list
- We read res.txt getting true results and then compare it to our list from dictionary and calculating mean absolute error with those lists

End goal

End goal is to count as many hits as possible and get MAE lower than three.