Assignment 3

Batuhan Faik Derinbay derinbay18@itu.edu.tr
150180705

Preparation of the Homework

Please download and extract the web version of the game, then run it using Mozilla Firefox. To run, you can use the `'firefox <path-of-game>/saturdaynightfilter_web/index.html'`. Also, make sure that the game is running on your primary monitor, without any obstructions (file explorer, image viewer etc.) on top and it is NOT in fullscreen mode. The provided scripts will put the game in fullscreen mode.

If you have any problems with the dependencies on Linux operating systems you can do the following:

- `cd /path/to/homework/source/files`
- `python3 -m venv .`
- `pip install -r requirements.txt`

For more detailed instructions, please refer to the this link (`https://packaging.python.org/guides/installing-using-pip-and-virtual-environments/`).

Part 1: Sobel Filter

In this part, screenshot of the all shapes page is taken, passed through Sobel edge detector and saved as `output_sobel.png`. The output of the program is given in Figure 1.

You can run the script using the `'python part1.py'` command.

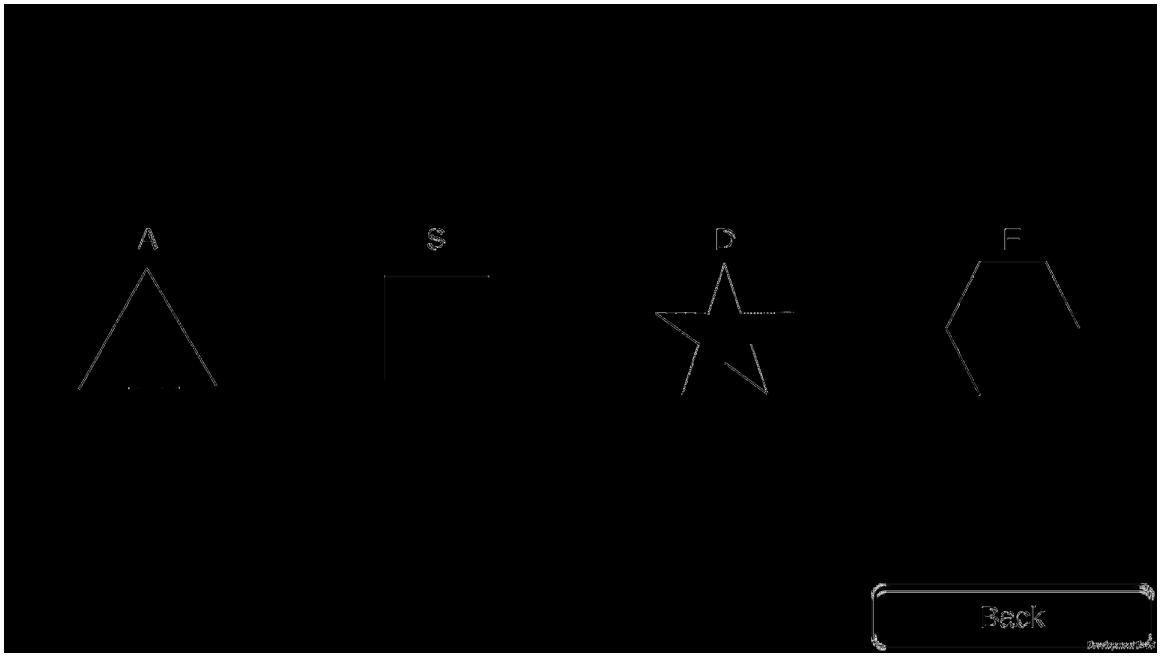

Figure 1: Sobel Filter Applied All Shapes Page

Part 2: Canny Edge Detector

In this part, screenshot of the all shapes page is taken, passed through Canny edge detector and saved as `output_canny.png`. Once the edges are detected using the Canny edge detector, retrieved edges are used to find the contours of the shapes and saved as `output_contours.png`. The output of the program is given in Figures 2-3

You can run the script using the `'python part2.py'` command.

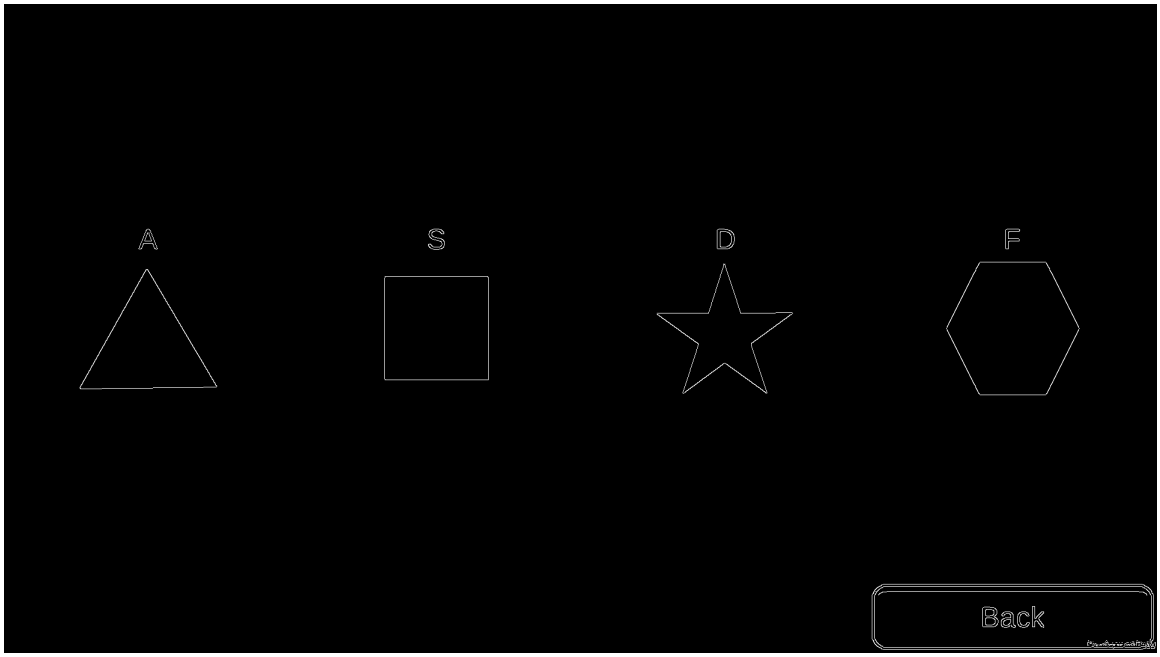

Figure 2: Canny Edge Detector Applied All Shapes Page

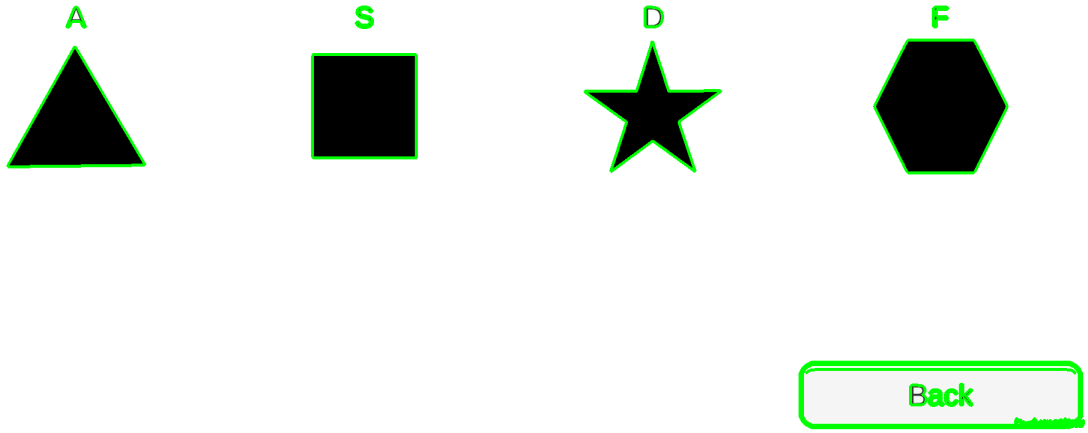

Figure 3: Contours of All Shapes After Applying Canny Edge Detector

Part 3: Minimum Eigenvalue Corner Detector

In this part, screenshot of the game is taken, passed through a minimum eigenvalue corner detector algorithm and saved as `output_corners.png`. The output of the program is given in Figure 4.

The corner detector algorithm can be examined in `common/detector.py` file. It can outputs results using Harris and Shi-Thomasi corner detection algorithms. It can also downscale images for faster processing times. To see an output using the Shi-Thomasi algorithm you can use this line:

- `corner_detector(shapes, 13, 1000, 0.05, 1, "shi")`

If you think the algorithm is running a bit slow, try out this line:

- `corner_detector(shapes, 3, 150000, 0.05, 3, "harris")`

It downscales the cropped image by a factor of three as a trick to process faster.

You can run the script using the `'python part3.py'` command.

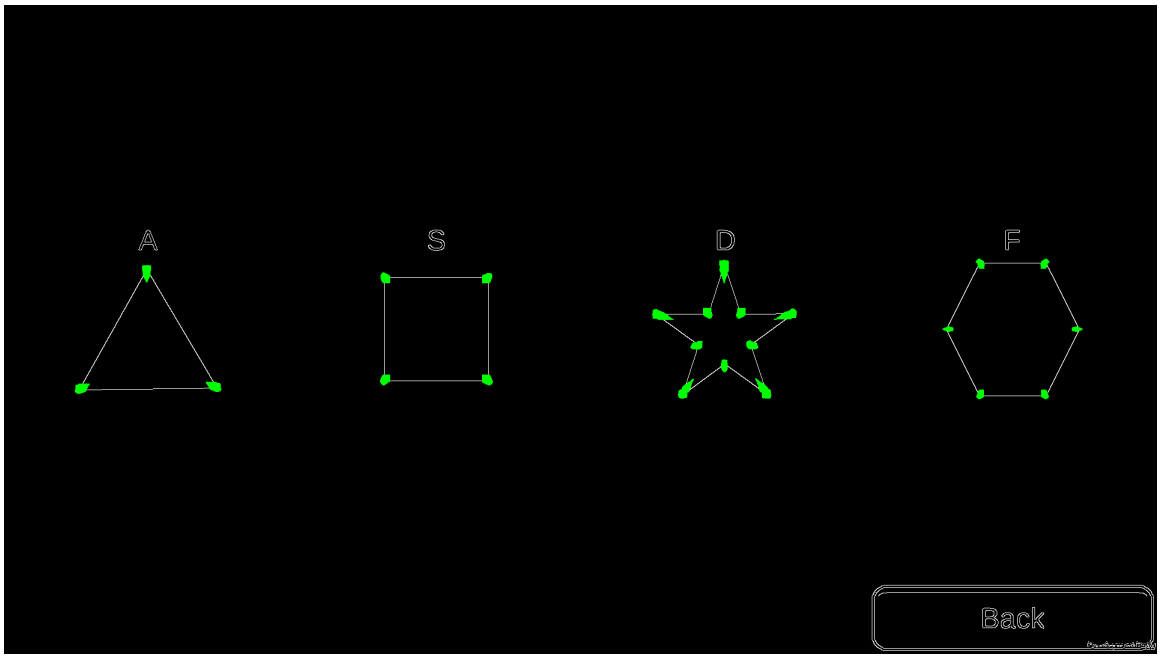

Figure 4: Minimum Eigenvalue Corner Detector Applied All Shapes Page

Notice how only the corners of the shapes are detected, unlike others. This is because shapes are cropped from the screenshot, edge detection is applied then placed back on to the screenshot and saved.

Part 4: Dance Game

In this part, both songs of the game can be played by the script. A prompt to the user is shown and the song to be played is selected. The script plays the game and ends it as

expected. It requires no user input and achieves 100% accuracy. Please note that the script involves some delays within the algorithm and the outcome of the note accuracy maybe different on different devices.

The algorithm works as follows:

1. Start the song
2. While less than 18 (number of notes in both songs) keys are pressed, take a screenshot of the region just in front of the pink region
3. If the note check region (a 3x3 area in the middle of the screenshot) detects a note (all black pixels) apply Harris corner detection to the screenshot
4. Mask the corners and count the number of corners
5. Press the right key depending on the number of corners
6. Once 18 keys are pressed wait five more seconds so the user can see the accuracy and listen to the song

You can run the script using the `'python part4.py'` command.