Assignment 7

Gaurav Baney, Wei-Han Chang gbaney@usc.edu, weihanc@usc.edu
05/05/2020

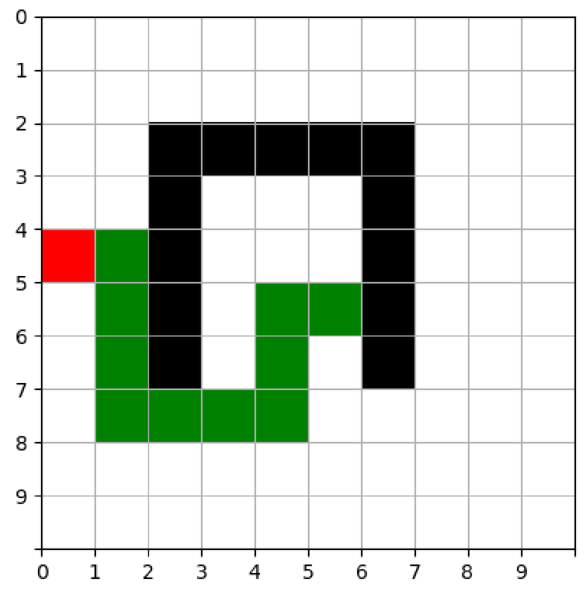
1. Implementation

Our implementation of HMM is in the file 'hmm.py'
The input file "hmm-data.txt" has been hardcoded to run upon executing the file.

The resulting output (trajectory) is as follows:

```
[['1', '1', '1', '1', '1', '1', '1', '1', '1', '1'],
 ['1', '1', '1', '1', '1', '1', '1', '1', '1', '1'],
 ['1', '1', '0', '0', '0', '0', '0', '1', '1', '1'],
 ['1', '1', '0', '1', '1', '1', '0', '1', '1', '1'],
 ['G', '^', '0', '1', '1', '1', '0', '1', '1', '1'],
 ['1', '^', '0', '1', '<', 'S', '0', '1', '1', '1'],
 ['1', '^', '0', '1', 'V', '1', '0', '1', '1', '1'],
 ['1', '<', '<', '<', 'V', '1', '1', '1', '1', '1'],
 ['1', '1', '1', '1', '1', '1', '1', '1', '1', '1'],
 ['1', '1', '1', '1', '1', '1', '1', '1', '1', '1']]
```

Which, when visualized in a graph, looks like:



1.1 Data Structures

For the HMM, we created a custom "Point" data structure to represent grid and tower locations. Within the Point class contains the coordinates as well as several methods used for comparison and finding neighbors. Storing locations in a custom data structure allows us to simplify our code, and reduce the amount of times we have to write the same code

We also have another custom data structure, "PathNode", that stores information on each step in the predicted trajectory. The class contains information such as timestep, probability, and the coordinates of that step

All other data, such as noisy distance, are stored as a python list.

1.2 Code Level Optimzations

We decided to create our own data structures so we can reduce the amount of lines of code we have to write. Operations such as location comparison and finding neighbors are built into these data structures so we can simply call a method instead of writing these operations over again.

1.3 Experience

The biggest challenge we faced in this assignment was understanding the Viterbi algorithm. We spent by far the largest amount of time with this section. We also spent a good amount of time deciding how to store our data for optimization purposes.

2. Applications

HMM is used extensively in computer vision, particularly facial recognition and feature extraction. HMM can also be used in speech recognition to extract features from speech

3. Contribution

Gaurav Baney: HMM (2/3)
Wei-Han Chang: HMM (1/3), Report