Alphabet Recognition using capacitive sensing

CS6650: Smart Sensing for IOT - COURSE PROJECT

CS18B042 - R Prahlada Reddy
CS18B022 - K.V Pradeep Reddy
CS18B037 - P Akshay Thomas
CS18B048 - Y Jashwanth Sai

Problem statement

Using Capacitive sensing to identify the alphabets written on a 2d surface.

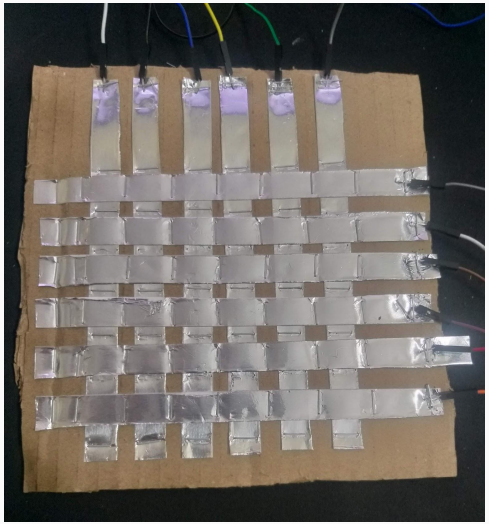
Why this?
We don't need touch pads with high resolution to recognise finite gestures.

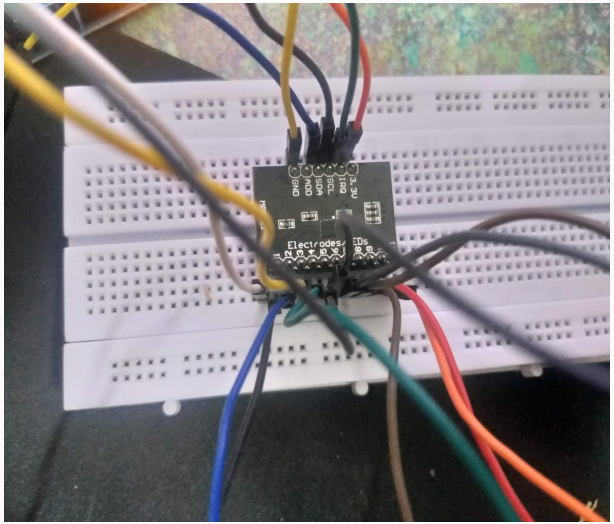
In this project we try to recognise Alphabets using a 6x6 grid, MPR121 touch sensor and an Arduino.

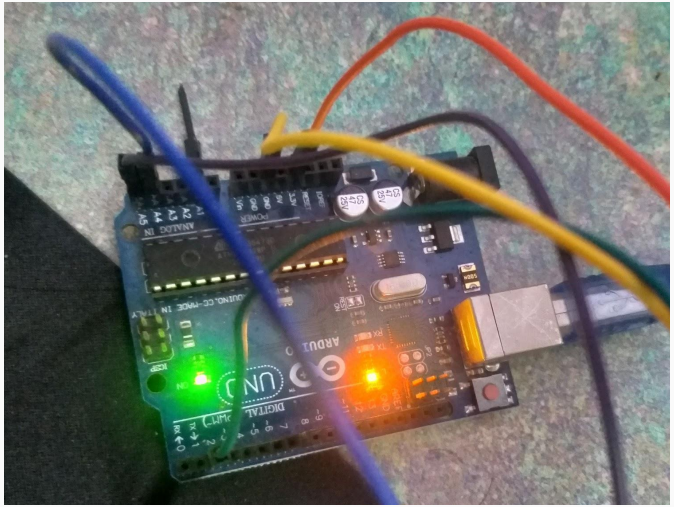
SETUP

GRID	MPR121	ARDUINO	PC
Touch happens	Senses touch and sends electrode touch info to Arduino using I2C Bus	Reads the touch data and groups them in a matrix and sends it to PC via USB	Analyses the matrix and recognises the alphabet

SETUP

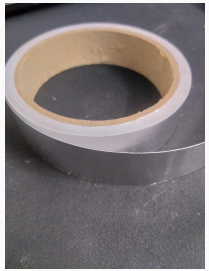
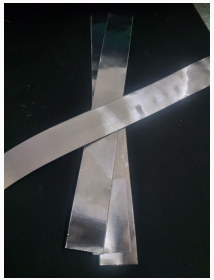


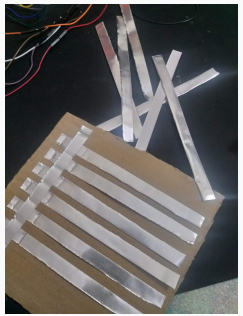
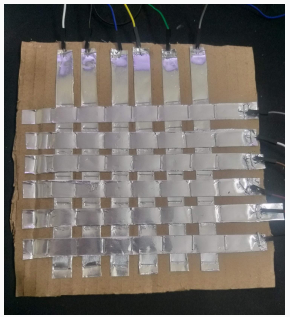




GRID

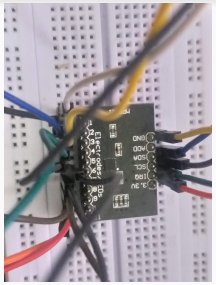
- We used Aluminium tape to form grid.
- We arranged grids such that at least one electrode is touched when finger is placed in grid area
- Ensured that horizontal grid and vertical grid do not touch so that capacitance of system doesn't go high, so that voltage drop become low (Initially we experienced it when we don't insulate them)
- Each row and column is connected to electrode pin of MPR121

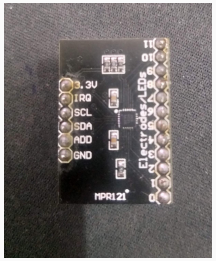



MPR121

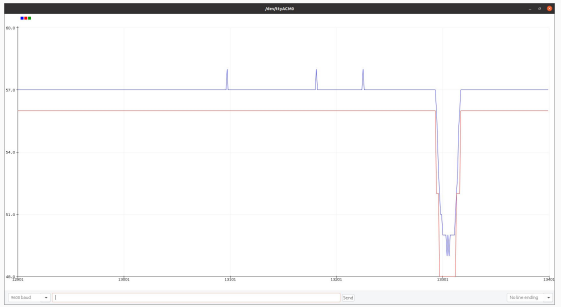
- We connected it to Arduino using I2C bus
- We connected grid as electrodes to it
- How it senses touch?
  It supplies constant current to electrode for some time and measure voltage .
- It maintains a base value updates it with time neglecting high changes.
- High changes are reported as touch but small changes are used to update base.
- Problem? Now since the vertical grid is below only indirect tuch happens which leads to small change in capacitance

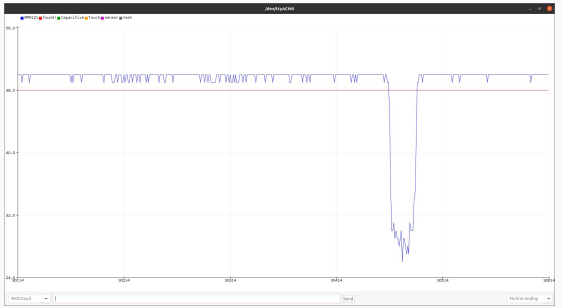




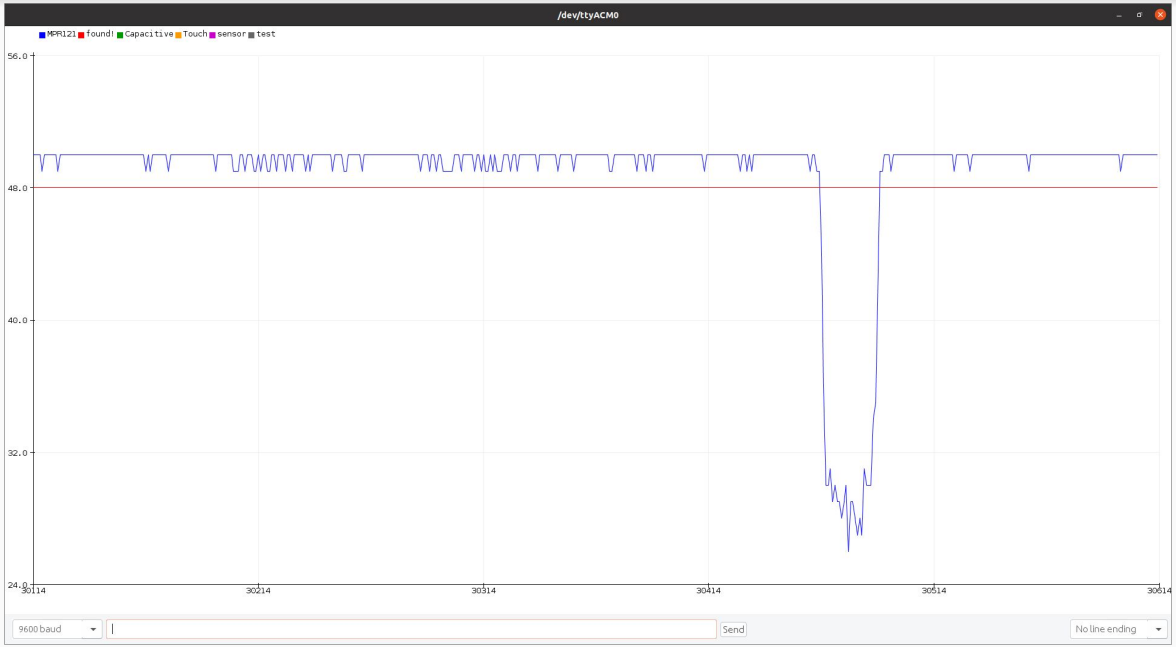
MPR121

- Top graph shows the changes in base and filtered value for column electrode when touched on intersection point Both base and filtered value decreased leading to no touch detection
- Bottom graph shows the changes in base and filtered value for column electrode when touched on intersection point Both filtered value decreased and bace remaining almost constant which lead to touch detection

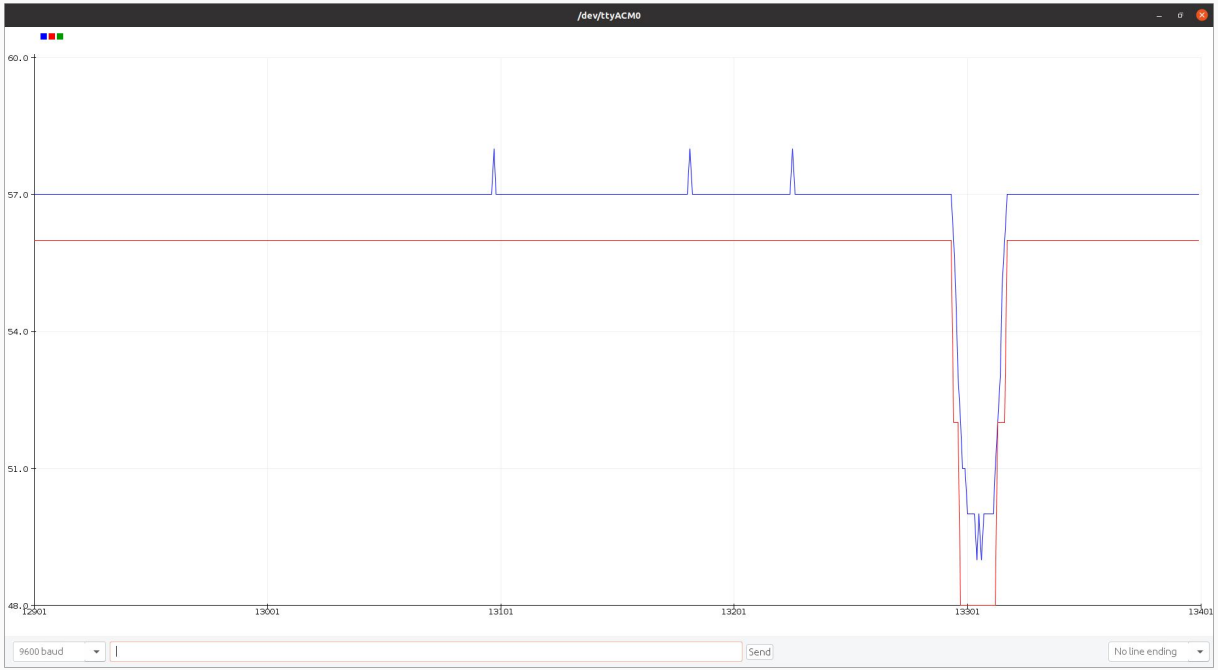




Top Surface (row)



Bottom surface (column)



ARDUINO

- Since sensing column cannot be done directly by MPR121 we implemented it here.
- Grouping all the touched vertices and forming matrix is also done here.
- Detecting when alphabet input is completed is also done here

ARDUINO

Initialization

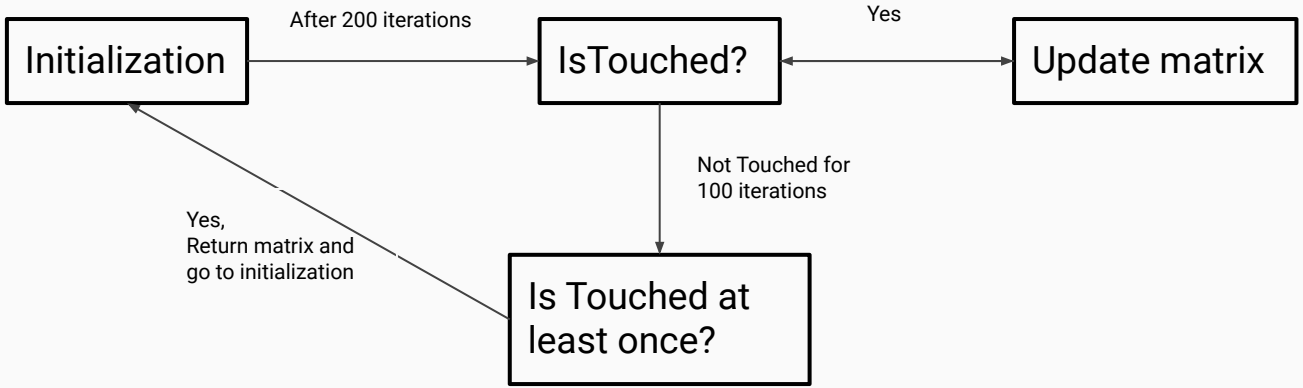
For some 100 iterations we absorb voltage values of each electrode and find mean voltage.we absorbed deviation +-1 for small iterations so variance is not measured

Populating Matrix

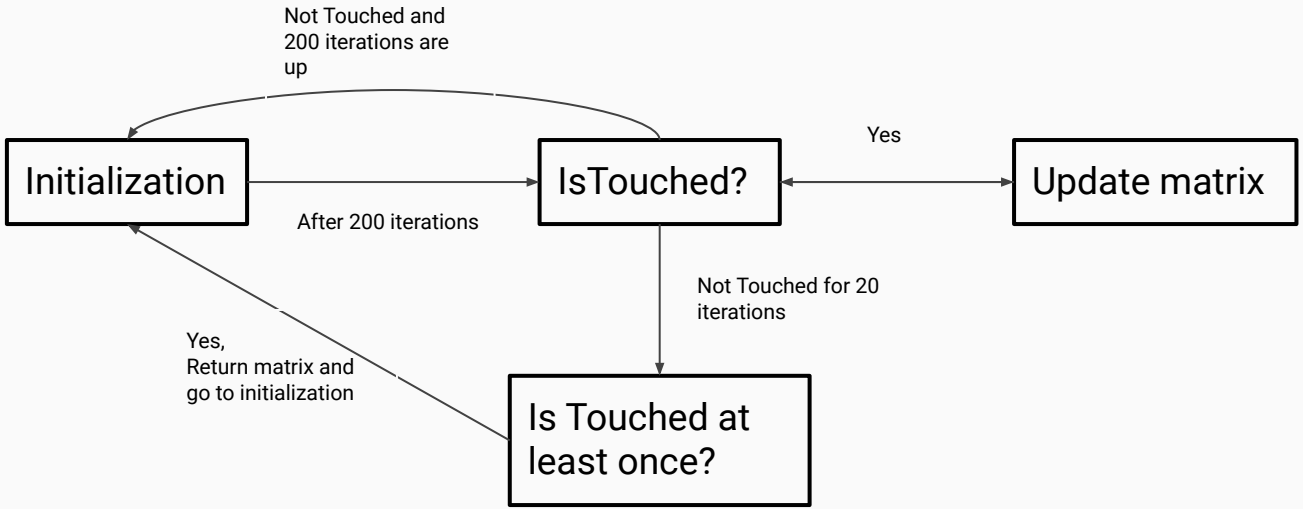
In this phase we try to recognise touch and update matrix. We can notice touch along rows easily as there is large voltage difference ,For columns we see 10 iterations and find which column has max average voltage difference .

We will be switching between these two modes

ARDUINO


Initialization
After 200 iterations
IsTouched?
Yes
Update matrix
Not Touched for
100 iterations
Yes,
Return matrix and
go to initialization
Is Touched at
least once?

ARDUINO


Not Touched and
200 iterations are
up
Initialization
IsTouched?
Yes
Update matrix
After 200 iterations
Not Touched for 20
iterations
Yes,
Return matrix and
go to initialization
Is Touched at
least once?

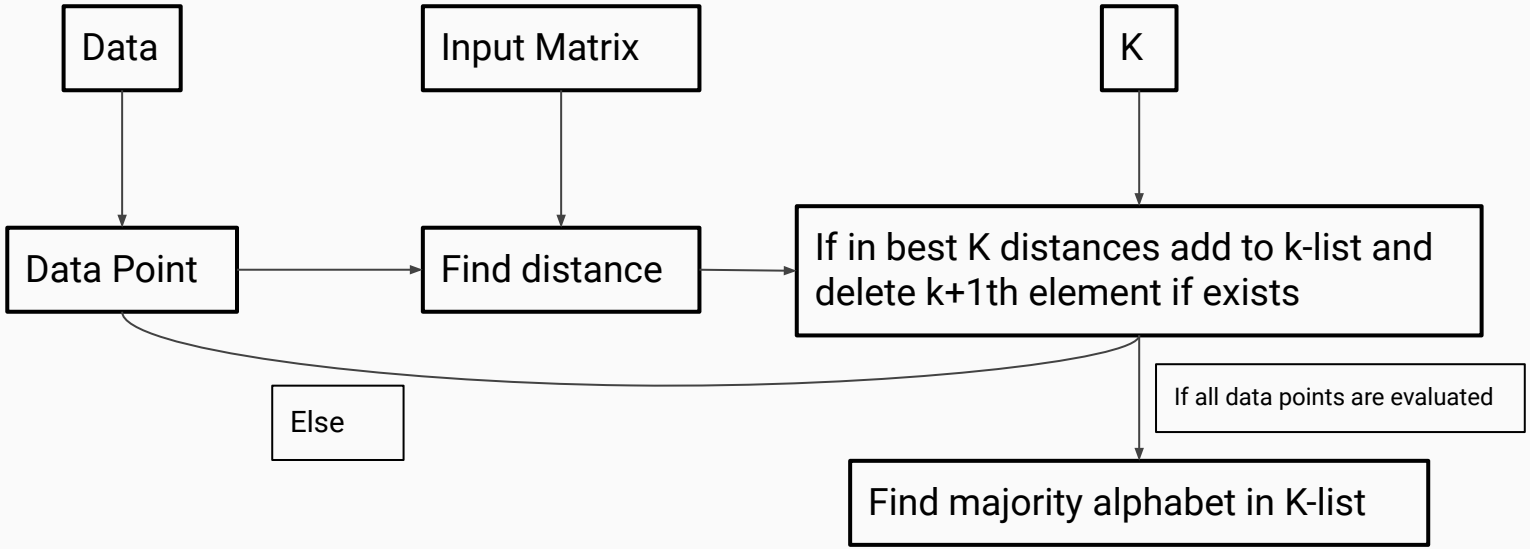
ARDUINO PC connection

- Sends status MPR121 whether it is identified or not
- Send initialization message to PC so that user can know it is in initialization phase and touch will not be considered
- Send a started listening message after initialization so user can start writing letter
- Sends the matrix when user stops writing

PC

- PC needs to identify character using the matrix sent by the Arduino
- We collected 20 data points for each character and using this we use KNN algorithm to identify the character.

KNN Algorithm



Distance Metrics

Metric 1:

We considered distance between two matrices as number of times there is touch in one matrix and not present in another matrix

0 4 0 0 0 0	0 0 4 0 0 0
0 2 0 0 0 0	0 0 2 0 0 0
0 2 0 0 0 0	0 0 1 0 0 0
0 9 1 1 3 0	0 0 1 0 0 0
0 0 0 0 0 0	0 0 4 0 0 0
0 0 0 0 0 0	0 0 5 1 3 1

Distance = 14

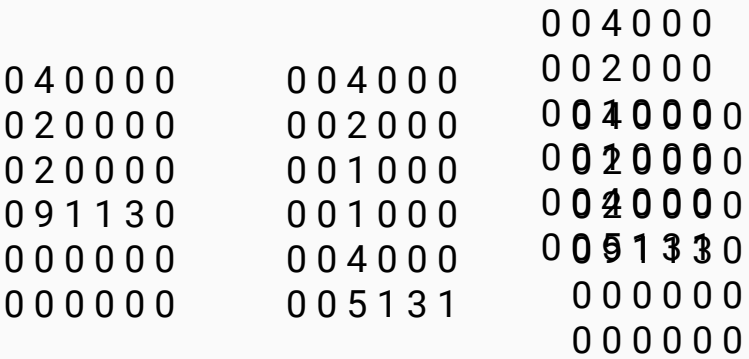
Clearly not a good distance parameter

Distance Metrics

Metric 2:

Mean shift distance metric

We shift 0,0 in one matrix to another and find distance using metric 1 and take least of them



Distance = 2

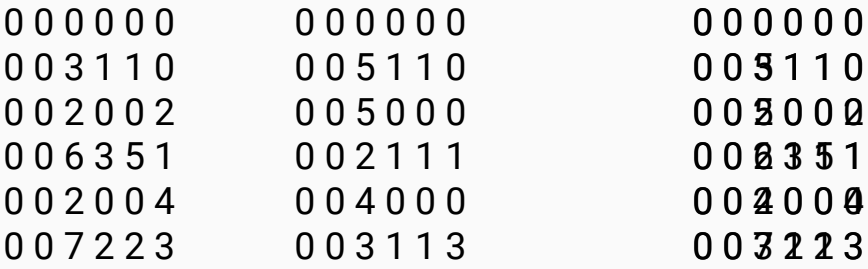
Clearly a better distance parameter

Results

```
[[5. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 3. 0. 0. 2. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 3. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 4. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 4. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 4. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 4. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 5. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 5. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 2. 3. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 4. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 4. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 3. 2. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 5. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 5. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 5. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 4. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 3. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 2. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 2. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 5. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 4. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 4. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 4. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 3. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 3. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 4.]]
```

77.7% accuracy for K=3

Why low



Distance only 2

Alternatives

- Maybe we can use direction of movement while writing also in data that may increase the accuracy
- May have used a probabilistic model
- Somehow using weight information

THANK YOU