Color Vision Tester

ARTIFICIAL INTELLIGENCE, PROJECT 5

Arjun Bastola
RAMAPO COLLLEGE OF NEW JERSEY |
COMPUTER SCIENCE DEPARTMENT

Classes/Java Files:

EV3 Classes:

1. **Class: HomePage**
   Description:
   - Provides user Menu to select between “Scan Board” and “Start K Means Algorithm.”
   - If the user selects K Means, this class displays an option to select value for k.

2. **Class: Board**
   Description:
   - Scans the 10*10 board automatically (without support)
   - Store’s each pixel’s details: (x-axis, y-axis and color value) as Pixel Object

JAVA Classes:

3. **Class: Statement**
   Description:
   - Stores each rule of Text File
   - Used for Backward and Forward Mode

4. **Class: Rules**
   Description:
   - Gets Text File
   - Extract all the rules from the text file using array of Statement Object

5. **Class: Pixel**
   Description:
   - Stores detail for each Pixel
   - Store x-coordinate, y-coordinate and value of color

6. **Class: KMeansAlgorithm**
   Description:
   - Implements KMeansAlgorithm in terms of cluster
   - Divides the Pixels to k clusters/groups and these groups are passed along the KMeansAlgorithmDistance to get Final Characters
   - Also prints bounding box’s coordinates for each color group

7. **Class: KMeansAlgorithmDistance**
   Description:
   - Implements KMeansAlgorithm in terms of distance
   - Divides each group passed from KMeansAlgorithm Class to two groups.

- Coordinates of both groups are then passed along NormalizePixels to get Absolute value of coordinates of each Pixel in the groups
- Also prints bounding box's coordinates for each group

8. **Class: NormalizePixels**
   Description:
   - Each group of pixels passed from KMeansAlgorithmDistance is normalized in this class.
   - This class finds the lowest value of x and y and subtracts co-ordinates of each Pixel by minimum value of x and y.
   - The normalized groups are then passed to DetectCharacter to get Final Characters.

9. **Class: DetectCharacter**
   Description:
   - Each group of pixels passed from NormalizePixels are checked against the provided rule base to detect characters.
   - Uses Forward Reasoning AI algorithm to detect Characters.
   - Prints out each detected Character.

AI Algorithms used:

The following algorithms being used:

1. Forward Reasoning

Description:

The algorithm uses two lists: one to store characters that have passed the test until now and another to store characters that have failed the test. This algorithm will abandon the failed list and will continue on working on the passed list. After each pixel is scanned, it checks all the elements of passed lists and checks if they pass co-ordinate that was just scanned. In this way the passed list is checked until all the pixels have been scanned. At the end any characters left in the passed list in printed as result of the search.

Classes Used In: DetectChracter.java (JAVA Class)
Methods Used In: DetectCharacter(ArrayList<Pixel> arrayList) and LookUpNestedRule(String a, String passed)

2. KMeansAlgorithm

Description:

This algorithm divides the Pixels scanned to k number of Clusters in terms of Color. Value of k is defined by the user. The user is provided with Menu to select the value of k. After the Pixels are divided to k number of clusters using Color, each cluster is then divided to 2 more Clusters in terms of Distance. Therefore, at the end, user will have 2k group of clusters and each cluster will detect a character.

How K Means Clustering Algorithm works?

First, decide the number of clusters k. Then:

1. Initialize the center of the clusters
2. Attribute the closest cluster to each data point
3. Set the position of each cluster to the mean of all data points belonging to that cluster
4. Repeat steps 2-3 until convergence

Classes and Methods Used In:

Class	Methods
KMeansAlgorithm.java (JAVA Class)	PerformClustering(), average(ArrayList<Pixel> arrayList)
KMeansAlgorithmDistanc.java (JAVA Class)	KMeansAlgorithmDistance(ArrayList<Pixel> arrayList), centroidOfPoints(ArrayList<Pixel> arrayList)

Bug Report:

1. Didn't print actual bounding box dimensions (This was fixed after the demonstration)
2. Didn't print all the characters present in the board (See Observation Log)

Missing Features:

1. None

Additional Features:

1. No significant additional feature

Observation Log:

Observation Type	Observation	Possible Solutions
Bounding Box of Each Clusters	• Displayed incorrect Bottom Right co-ordinates • Was displaying coordinates of bottom right Pixel instead of bottom right corner's coordinates of the bounding box • The error has been fixed	• This error has been fixed
Character Detected	• Only 4 characters was detected • There was an error detecting difference between Tan, Yellow and White • Detected Characters of Red and Green but not of Yellow, White and Tan because the difference was minimum.	• Scan each Pixel 3 times and add each value of color. • This will lead to greater difference between the color values and more precise results.

Project Log:

Date:	Work Completed	Time Spent
April 19	• Worked with Color Sensor • Tried different color samples to get estimate of how color values changes according to the colors • Got highest difference between colors using RGBMode	2.5 Hours
April 22	• Worked on the HomePage • Created HomePage class	1.0 Hour
April 23	• Created Sample 5*5 board data (25 Pixel Objects) • Programmed KMeansAlgorithm using Color • Programmed KMeansAlgorithm using Distance (k2) • Combined two of these algorithm to work together. • Used sample date to get output • Got correct output	4 .0Hours
April 24	• Combined EV3 classes and Java Classes to work together • Made Board class which scanned 10*10 board • Checked algorithm using actual data • Output was correct	2.0 Hours
April 25	• Worked to get Bounding Box dimensions for each clusters • Tested sample board but this time the Pixels were smaller • Didn't get expected output • Few Characters were missing	3.0 Hours
April 26	• Worked on Documentation	1.5 Hours
	Total Hours	14.0 Hours