Color Vision Tester: A PIP scanning program for leJOS EV3 Robot

Bishal Regmi

April 2016

1 Description of Classes

1.1 EV3 Classes

1.1.1 Start.java

This class is the entry point of the program. It contains the 'main' function that initializes the menu. This class initializes the Robot class which handles the robot, its LCD screen to display the menu. This is also the calss that scans all the pixels from the board to create a table and then use the model classes to cluster the scanned values.

1.1.2 Robot.java

This class contains the attributes related to controlling the robot. It is where all the sensors and motors are initialized. When this class is initialized from the Start class, the DifferentialPilot that controls the robot is initialized.

1.2 Other Java Classes

1.2.1 Dictionary.java

This is the class where the file to get the rule base from fetched. It then send the InputStream for the file to syntaxReader class to decode the rules. This class also called the k-means clustering algorithm to cluster the values based on distance and color and then use forward search to detect the letters.

1.2.2 ForwardSearch.java

This is the class which handles the algorith for Forward Search. It has two list passed and failed divides and creates the decision tree.

1.2.3 Letter.java

This is the class to hold all the letters. Basically, this class holds a two dimensional array of the board size as boolean value and has getters and setters to set these values. It also has copyFrom and compareWith function which allow to copy the true values from teh given letter and compare the letter with the given letter respectively.

1.2.4 syntaxReader.java

This is the class that decodes the serialized files and creates a list of rules.

1.2.5 Menu.java

This class is an interface for all the menus. It has an LCD object and a function to display the cursor and a function to invoke the Menu.

1.2.6 Pixel.java

This class is used to record the row, column and color value of each pixel scanned from the board.

1.2.7 Cluster.java

This class is used to store the information about a cluster for k-means clustering. It records the centroid and the list of pixels in that cluster.

1.2.8 ClusterLimits.java

This class is used to hold the bounding box values for each cluster.

1.2.9 kMeans.java

This is the class where the actual algorithm of k-means clustering takes place.

1.2.10 kMeansReading.java

This is an enum to define what the clustering is to be done on - Color or Distance.

1.2.11 FullPattern.java

This class is used to hold the color value and the string value for each recognized letter from k-means clustering.

2 AI Algorithms

2.1 1. Forward Search

This search is implemented in the Class **ForwardSearcher.java** and the method **public void divide-Tree(int x, int y, boolean pixelSet)** is the AI part of it. This method basically divides the list into two new lists based on the setting of the given pixel. If there are no deductions left at any point, it rolls back one node in the tree and continues scanning for possible candidates skipping the last pixel.

2.1.1 2. k-Means Clustering

This search is implemented in the class **kMeans.java** and the method **private void update()** Inside the kMeans class, there functions are respectively called in every iteration of clustering. In each iteration, this update method calculates a new centroid using **private void recalculateCentroids()** method and checks to see if the previous centroid an the new centroid are the same to decide the ending of the algorithm. In case it doesn't end, it reassigns all the pixels to these clusters with new centroids using method **private void assignPointToClusters()**.

3 Bug Report

- I haven't encountered any bugs yet.

4 Features

4.1 Additional

1. None

4.2 Missing

1. None

5 Observation Log

The letters in Red, Green and White were recognized. However, the letter in Yellow and Tan were not because they had really small difference in their color value due to which they were scanned as the same color. Thus, when clustering all the Tan and yellow were clustered into the same cluster due to which the deduced clusters didn't match with any rules from the rule base. Thus, I only had 6 letters deduced: ell, d, L, 4, H and uber

6 Project Log

- April 16th (4 hrs)
  Wrote the pseudo-code for k-means clustering
- April 17th(3 hrs)
  Implemeneted k-means clustering using console and reading pixels from a text file. The algorithm works
- April 23rd(5 hrs)
  Tried to implement the clustering first on color and then distance. The entire day was spent on debugging and printing statements
- April 24th (3 hr)
  Figured out that using int values for row and column messed up clustering so changed them to float for precision. k-means clustering works perfectly. Spent the entire day optimiing code.
- April 25th(3 hrs)
  Working on the robot part. trying to scan the board as quickly as possible. Did it by delaying the robot while moving forward for the time it takes to scan 1 pixel. Scanning quickly works.

Total: 18 hours