## Group 15

Alan Cheng 22240142

Sigar Sigar 23191175

Matthew Golightly 20771474

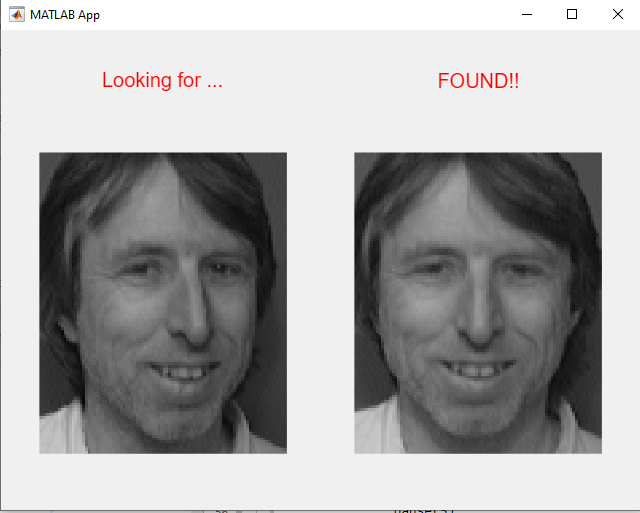
## Approach

Our approach follows Section 2.1 of the ‘Linear Regression for Face Recognition’ paper provided for the project. Downsampling was applied by scaling the image by half across the width and height dimensions. We decomposed the algorithm into separate functions with a main driver script to ensure our code is easily testable and organized.

To partition the image set into a training and test set, the images are first loaded into Matlab as a datastore. Next, the datastore is split into two separate datastores via the function ‘splitEachLabel’ representing the training and test sets. The function takes the parameter ‘randomized’ to ensure each train-test split is random.

From the training data, the class models can then be computed, then test images can be compared against the class models as outlined in the reference paper. The distance metric is computed between the test image and each class model, with the prediction class selected via minimum distance.

## GUI example



# Submission Folders and Files

### Folders

* FaceDataset contains the sample images provided on LMS
* ExtendedDataset contains our personal images in addition to the sample images

### Files

* main.m is the main file to test the image recognition algorithm.
* getClassModel.m contains a function which returns the class model for a set of training images.
* getTrainTest.m contains a function that splits a set of images into a training and test set.
* getRecognitionAccuracy.m is a modified version of main.m, and is used to run a specified number of trials to assess face detection accuracy in a cross-fold validation approach.

# Running the code

## Test phase

* Open main.m file
* Paste location of main folder containing sub folders of photos into the fpath variable.
* Run the code

## GUI

* Run the facedetection.mlapp file.
* When prompted for a directory, select the main folder containing sub folders of photos.

# Recognition Accuracy - Original Data Set

Train-test splits were randomised for each trial.

## 50-50 Train/Test Split

* Trial 1: 96.00%
* Trial 2: 92.50%
* Trial 3: 93.50%
* Trial 4: 96.50%
* Trial 5: 95.00%
* Trial 6: 94.00%
* Average: 94.58%

## 60-40 Train/Test Split

* Trial 1: 96.88%
* Trial 2: 94.38%
* Trial 3: 98.75%
* Trial 4: 98.12%
* Trial 5: 95.00%
* Trial 6: 97.50%
* Average: 96.77%

## 70-30 Train/Test Split

* Trial 1: 97.50%
* Trial 2: 96.67%
* Trial 3: 96.67%
* Trial 4: 99.17%
* Trial 5: 97.50%
* Trial 6: 96.67%
* Average: 97.22%

## 80-20 Train/Test Split

* Trial 1: 96.25%
* Trial 2: 100.00%
* Trial 3: 100.00%
* Trial 4: 98.75%
* Trial 5: 100.00%
* Trial 6: 97.50%
* Average: 98.75%

# Recognition Accuracy - Extended Dataset

We extended the original dataset with our own images. Each group member contributed 10 grayscale images of the same dimensions as the original images. A 50-50 train-test split was used here.

* Trial 1: 93.49%
* Trial 2: 95.81%
* Trial 3: 94.42%
* Trial 4: 97.21%
* Trial 5: 95.81%
* Trial 6: 93.95%
* Average: 95.12%

# Conclusion

Our results showed that a larger training set improved the recognition accuracy of the classification model. This is likely due to having more training data to formulate the class models from, increasing the variation of the training data and making the model more robust by decreasing the amount of unseen features for each person.

Interestingly, the extended dataset including our own images achieved a greater recognition accuracy than the original dataset. However, there was only a 0.54% difference which is quite insignificant. Since our group members looked quite different to the sample images, and too one another, it was to be expected the model’s performance would not change drastically and may slightly improve.