Hand Gesture based Game using MATLAB & Computer Vision Library

Harsh K. Malde.
Student, Computer Department,
K. J. Somaiya College of Engineering,
Mumbai, India
harsh.malde@somaiya.edu

Darshan D. Gajara.
Student, Computer Department,
K. J. Somaiya College of Engineering,
Mumbai, India
darshan.g@somaiya.edu

Swapnil R. Sutar
Student, Computer Department,
K. J. Somaiya College of Engineering,
Mumbai, India
swapnil.sutar@somaiya.edu

Bhakti N. Palkar.
Faculty, Computer Department,
K. J. Somaiya College of Engineering,
Mumbai, India
bhaktiraul@somaiya.edu

Abstract—Gone are the days when humans could interact with computers only via limited hardware devices like mouse & keyboards. With the recent developments in Virtual Reality, the interaction medium between humans & computers has changed drastically. This paper intends to put some light on the role played by hand gestures to provide inputs to the computer. Detection of gestures do not require any external device other than the webcam. The gameplay is about word formation using hand gestures specified by the user at the start of the game or by the predefined gestures.

Keywords*-hand gesture, virtual reality, human computer interaction, computer vision*

*****

I. Introduction

Pattern Recognition and Gesture Recognition are the growing fields of research. Being a significant part in non-verbal communication, Hand Gestures are playing a vital role in our daily life. The gesture based interaction interface proposed here can be substantially applied towards many applications, though the present paper considers Games as the application domain. Proper Hand Gesture segmentation from background and other body parts of the consecutive images is the primary requirement for the design of this application. The image frames can be captured from the low cost web cam. Gesture based interface proposed in this paper consists of detection, tracking and recognition modules.

A. About the Game (Gameplay)

Alphabets or letters are made to freely fall on the screen; A word is displayed to the user before he/she sees the free fall of the letters. The user is then supposed to form the earlier displayed word by collecting the letters in the same order. Letters can be collected by specified hand gestures.

B. Structure of the paper

Primarily, the work flow is explained in detail, which is followed by the involvement of technologies and the issues faced during the implementation of the same.

II. Work Flow

The work flow consists of two phases viz., Generating a database of gestures and Use of that database to map the gestures from user input to track it.

Gesture detection in real time requires a database of positive & negative images. A positive image contains the specified gesture which is missing in the negative image. Multiple number of these positive & negative images together form a dataset required for generating a XML database.

Steps to generate a XML database:

A. Labelling the Images

trainingImageLabeler invokes an app for labeling ground truth data in images. This app allows you to interactively specify rectangular Regions of Interest (ROIs). The ROIs define locations of objects, which are used to train a classifier. It outputs training data in a format supported by the trainCascadeObjectDetector function. The function trains a model to use with the vision.CascadeObjectDetector detector.

B. Implementation of Training

The example below shows how to set up and train a five-stage, stop-sign detector, using 86 positive samples. The default value for TruePositiveRate is 0.995.

Step 1: Load the positive data samples from a MATLAB file. File names and bounding boxes are contained in the array of structures labeled 'data'.
load('stopSigns.mat');

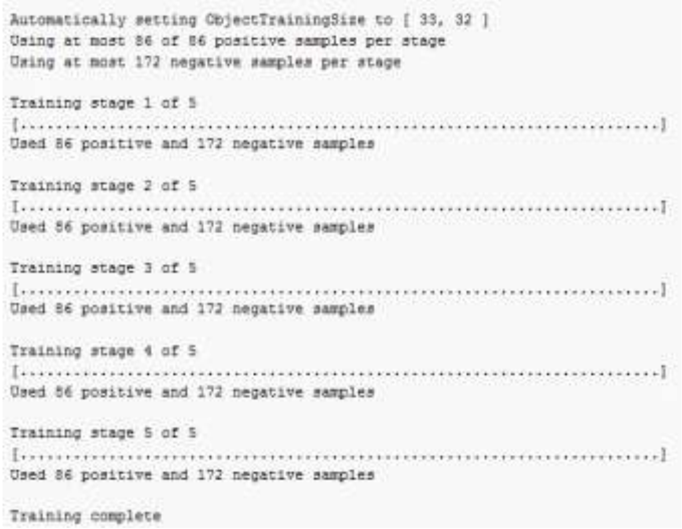
Step 2: Add the image directory to the MATLAB path.
imDir = fullfile(matlabroot,'toolbox','vision','visiondemos','stopSignImages');
addpath(imDir);

Step 3: Specify folder with negative images.
negativeFolder = fullfile(matlabroot,'toolbox','vision','visiondemos','non_stop_signs');

Step 4: Train the detector.
trainCascadeObjectDetector('stopSignDetector.xml',data,negativeFolder,'FalseAlarmRate',0.2,'NumCascadeStages',5);

trainCascadeObjectDetector(outputXMLFilename,positiveInstances,negativeImages) writes a trained cascade detector XML file with the name outputXMLFilename. The name specified by the outputXMLFilename input must have a XML extension.[5] For a detailed explanation on how this function works, refer to Train a Cascade Object Detector in MathWorks forum.

Upon execution of the above commands, Computer Vision software returns the following message:



```
Automatically setting ObjectTrainingSize to [ 33, 32 ]
Using at most 86 of 86 positive samples per stage
Using at most 172 negative samples per stage

Training stage 1 of 5
[......................................................................]
Used 86 positive and 172 negative samples

Training stage 2 of 5
[......................................................................]
Used 86 positive and 172 negative samples

Training stage 3 of 5
[......................................................................]
Used 86 positive and 172 negative samples

Training stage 4 of 5
[......................................................................]
Used 86 positive and 172 negative samples

Training stage 5 of 5
[......................................................................]
Used 86 positive and 172 negative samples

Training complete
```

Figure 1: Training Completion Message Box

Notice that all 86 positive samples were used to train each stage. This is because the true-positive rate is very high relative to the number of positive samples.

Any number of gestures can be added by following the above mentioned technique.

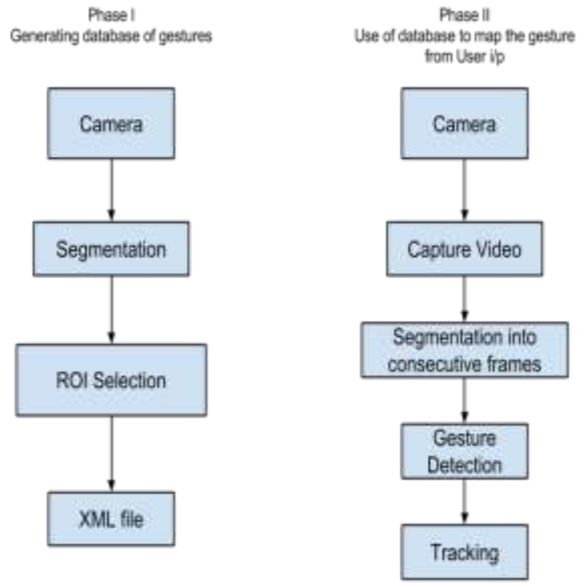

Phase I
Generating database of gestures
Camera
Segmentation
ROI Selection
XML file
Phase II
Use of database to map the gesture from User i/p
Camera
Capture Video
Segmentation into consecutive frames
Gesture Detection
Tracking

Figure 2: Work Flow of the System

In order to generate a database of large number of positive and negative images for a particular gesture, webcam captures the video of the user and breaks it into desired number of images as specified by the user. This process is automated by writing a specific set of code in MATLAB. User is then supposed to select the region of interest (ROI) from the image to generate a positive image. All this image information is stored in a XML file which is later used to map the gestures provided by the user in real time. Once the gestures are matched, user's hand activity can be tracked.

III. TECHNOLOGICAL ASPECTS

This section of the paper discusses the various underlying technologies and their usage with respect to the project at hand.

List of Technologies Used:

- MATLAB R2014a: To develop the application including GUI.

- Computer Vision System Toolbox: For object detection and tracking, video processing.

- Image Acquisition Toolbox: To acquire images and video from cameras and frame grabbers directly into MATLAB and Simulink.

There are no hardware requirements as such. Laptops are by default manufactured with webcams and in case of computers, portable webcams can be used.

The implementation of this game can be looked upon as a module for development of multiple games rather than this only game. The phase where the hand gestures are detected and tracked can be exported to development of other projects, which are then completed with little or more modification.

IV. Implementation Issues

A. Time Lag

MATLAB requires heavy amount of processing power and is unable to detect or track the gesture with the normal speed. User experiences considerable amount of time lag in the system, which makes it tedious to work with.

B. Licensing

MATLAB is a paid software and to purchase a licensed version of MATLAB, one has to invest a huge amount of money, which isn't affordable by all sorts of users.

C. Robustness

Most of the gesture based algorithms for MATLAB aren't robust. This means that the false detection of the system is large. Such systems or applications cannot be used in time and accuracy critical situations.

V. Acknowledgement

We take this opportunity to express our gratitude to everyone who supported us throughout the course of this project. We're thankful for their aspiring guidance, invaluably constructive criticism and friendly advice during the project work. We're sincerely grateful to them for sharing their truthful and illuminating views on a number of issues related to the project.

We would also like to thank our guide Prof. Bhakti N. Palkar & Prof. Poonam M. Bhogle for their continued help throughout the initiation & development of this project.

VI. References

[1] Dharani Mazumdar, Anjan Kumar Talukdar, Kandarpa Kumar Sarma, "Gloved and Free Hand Tracking based Hand Gesture Recognition", Dept. of Electronics & Communication Engineering, Gauhati University Guwahati-14, Assam, India, 2013.

[2] Mohamed Alsheakhali, Ahmed Skaik, Mohammed Aldahdouh, Mahmoud Alhelou, "Hand Gesture Recognition System", Computer Engineering Department, The Islamic University of Gaza, Gaza Strip , Palestine, 2011.

[3] Syed Akhlaq Hussain Shah, Ali Ahmed, Iftekhar Mahmood, Khurram Khurshid, "Hand gesture based user interface for computer using a camera and projector", Communication Systems Engineering, Institute of Space Technology, Islamabad Highway, near Rawat, 44000, Islamabad, Pakistan, 2011.

[4] Meenakshi Panwar, "Hand Gesture Recognition based on Shape Parameters", Centre for Development of Advanced Computing, Noida, Uttar Pradesh, India.

[5] http://in.mathworks.com/help/vision/ug/train-a-cascade-object-detector.html?refresh=true

[6] Isaac Gerg, Adam Ickes, Jamie McCulloch (2003, Dec. 7), Computer Vision I [Online], Available: http://www.gergltd.com/cse486/project5/

[7] http://blogs.mathworks.com/