LABORATORY ORIENTED PROJECT

INSTR F367

GESTURE RECOGNITION-BASED CONTROL OF MOBILE ROBOT

FINAL PRESENTATION

By:

ANIKET SAMANT	2012A8PS354P
TANMAY AGARWAL	2012A8PS500P

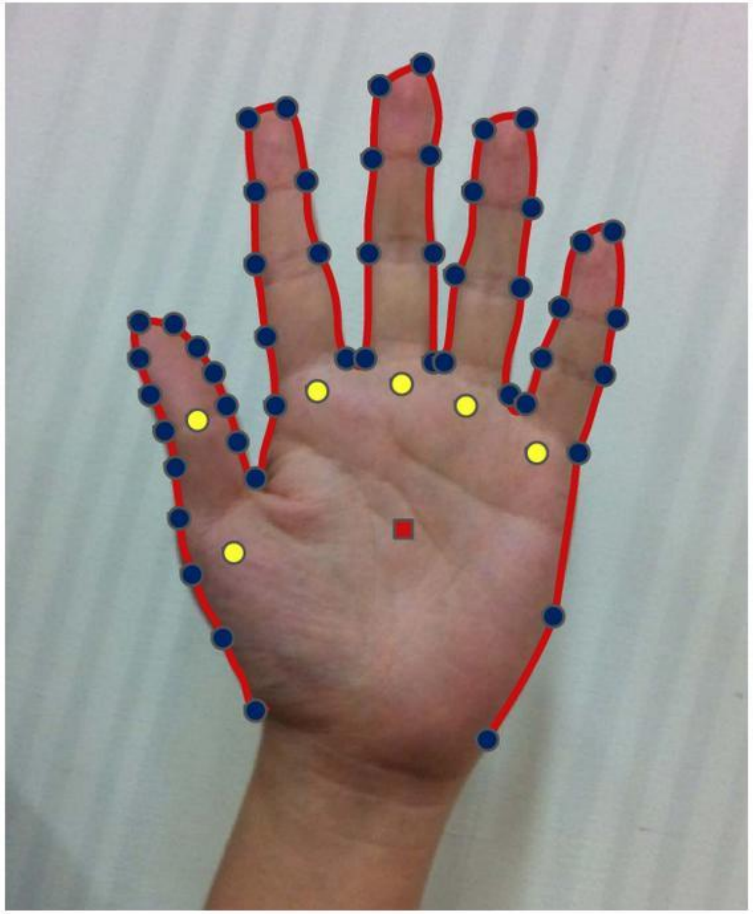
Gesture recognition

Mathematical interpretation of a human motion

Using a computing device

Inputs could be a static gesture or a motion pattern

A camera reads the movements of the human body and communicates the data to a computer that uses the gestures as input to control devices

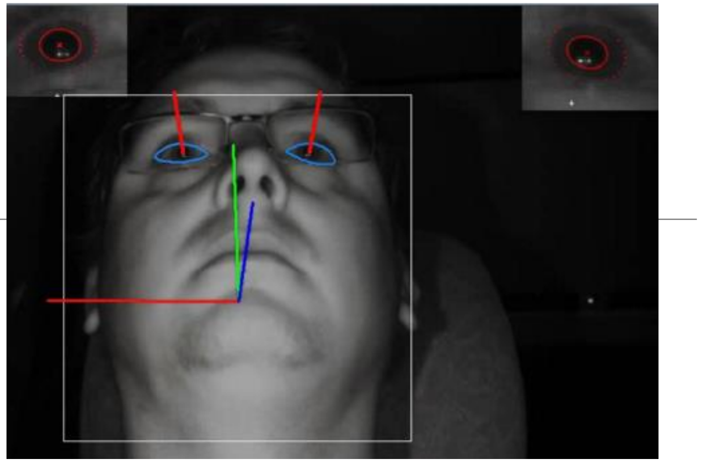


Applications of Gesture Recognition

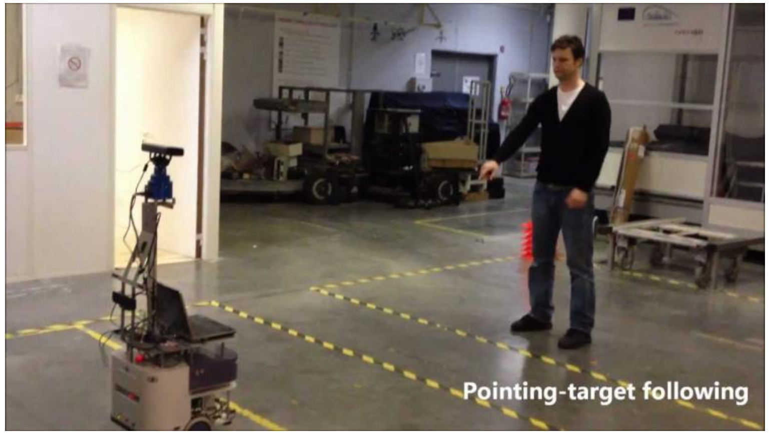
Home automation – Gesture-based control of:

- Fans
- Lights
- Television
- Air-conditioning system, etc

Driving assistance – for driving assistance (control of infotainment system, air-conditioning, etc) and for safety of the passengers (eye-tracking using a camera)



Control of mobile robots – hand gestures can be used to control mobile robots to assist people with disabilities



Control of Mobile Robot

The mobile robot’s direction is controlled using differential drive

An Arduino board (UNO) is used to send control signals to the motor driver ICs (L293D)

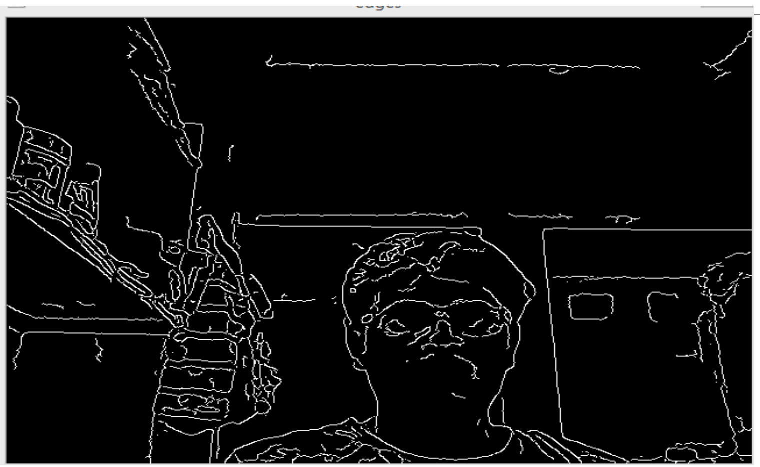
Gesture recognition is carried out using a web camera and the corresponding control signals are sent to the Arduino board

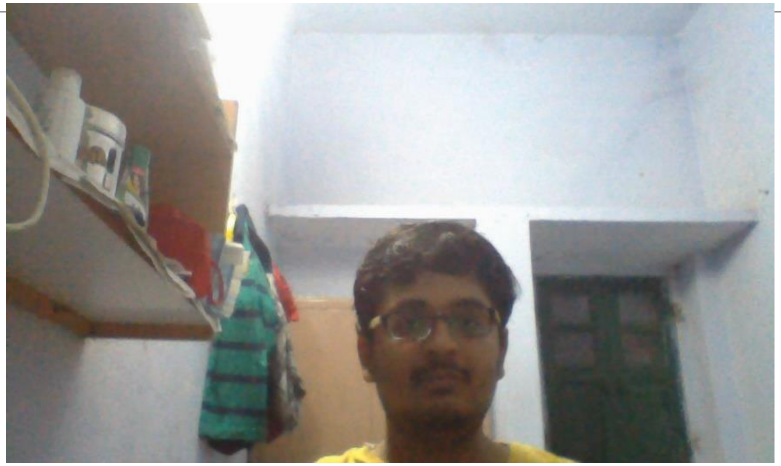
Software

Microsoft Visual Studio – This platform has been used to implement gesture recognition algorithms

- Written in C++
- OpenCV libraries used for gesture recognition

Visual Micro – a plugin for Visual Studio that used to test the Arduino board





Arduino IDE – Used for preliminary testing of the motors and testing the response of the motors to different types of control inputs

Hardware

An Arduino board is used to control four DC motors (12 V, 60 RPM, 800 mA NL / 4 A FL current).

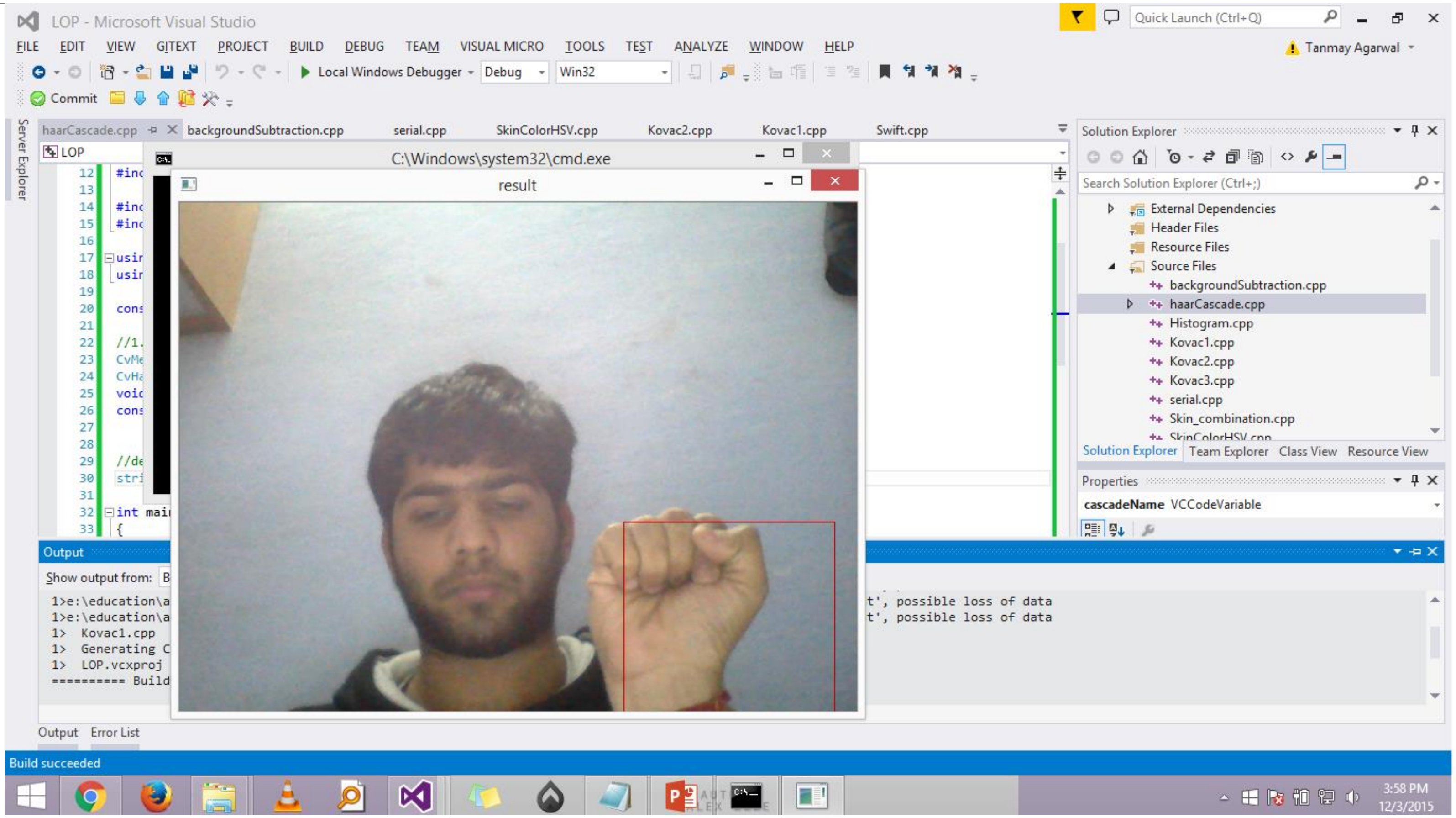
Differential drive concept is used to control the direction of the robot. (Directions being forward, reverse, rotate left, rotate right and controlled using inputs to L293D motor driver ICs)

Two TI L293D motor driver ICs are used and one set of input control signals is common to each of them.

Laptop’s in-built webcam for testing the OpenCV libraries.

Gesture Recognition algorithms are performed on a laptop and the corresponding control signals are sent to the mobile robot using the serial communication channel (COM port).

Haar Cascade



```
#inc

#inc
#inc

usir
usir

cons

//1.
CvMe
CvHa
voic
cons


//de
stri

int mai
{
```

cascadeName VCCodeVariable

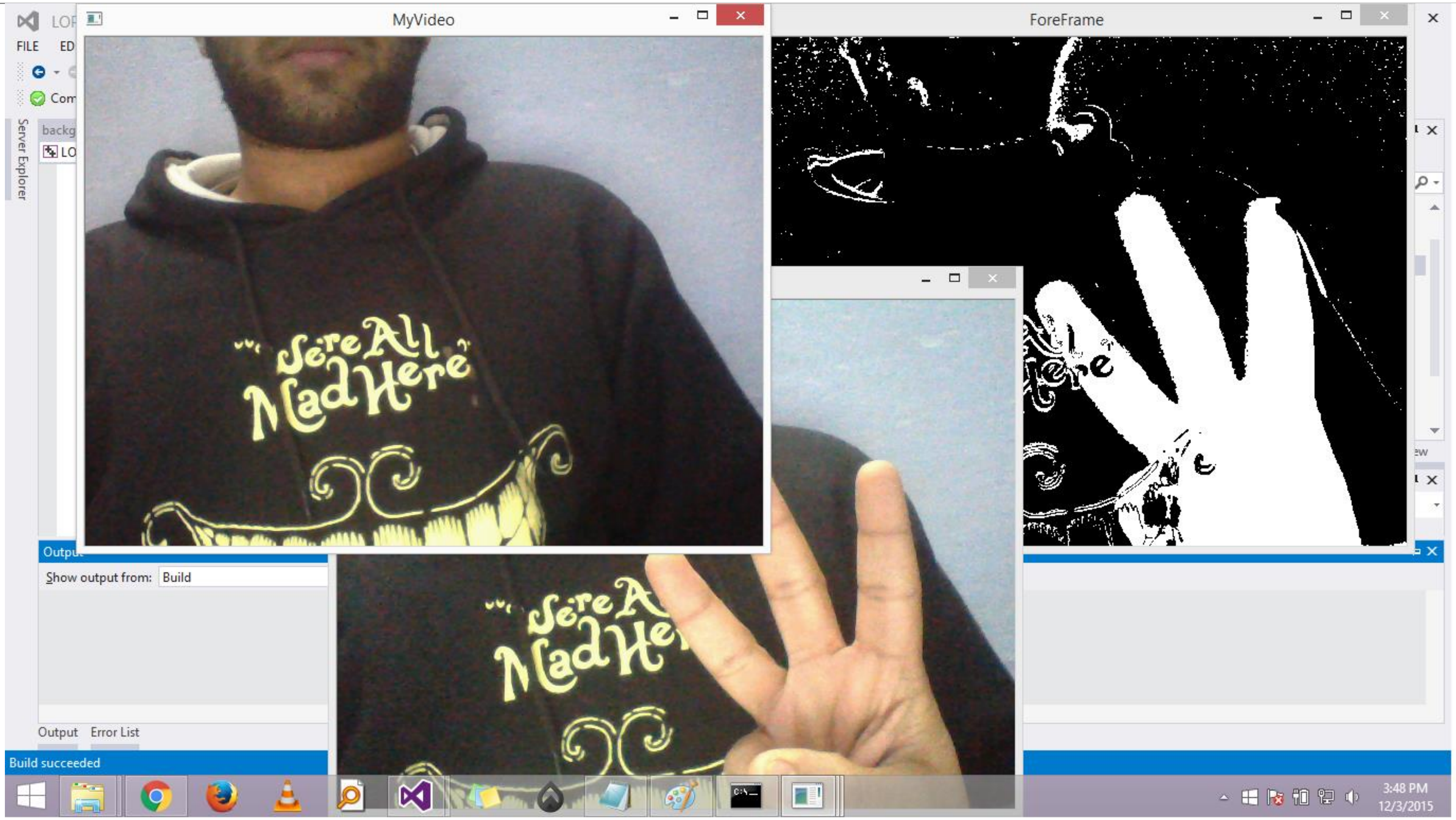
```
1>e:\education\a                                   t', possible loss of data
1>e:\education\a                                   t', possible loss of data
1>  Kovac1.cpp
1>  Generating C
1>  LOP.vcxproj
========== Build
```

Gesture Recognition Algorithms

Hand Recognition based on Haar Cascades

- Algorithm was proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001.
- It is a machine learning approach where a cascade function is trained based on positive and negative sample images.
- Model’s false detection rate decreases as more and more number of samples are given for training.
- Training data consists of positive samples of the object, in our case, our hand, given in different orientations and scale.
- The negative samples images include everything in the environment except the object to be recognized

Background subtraction



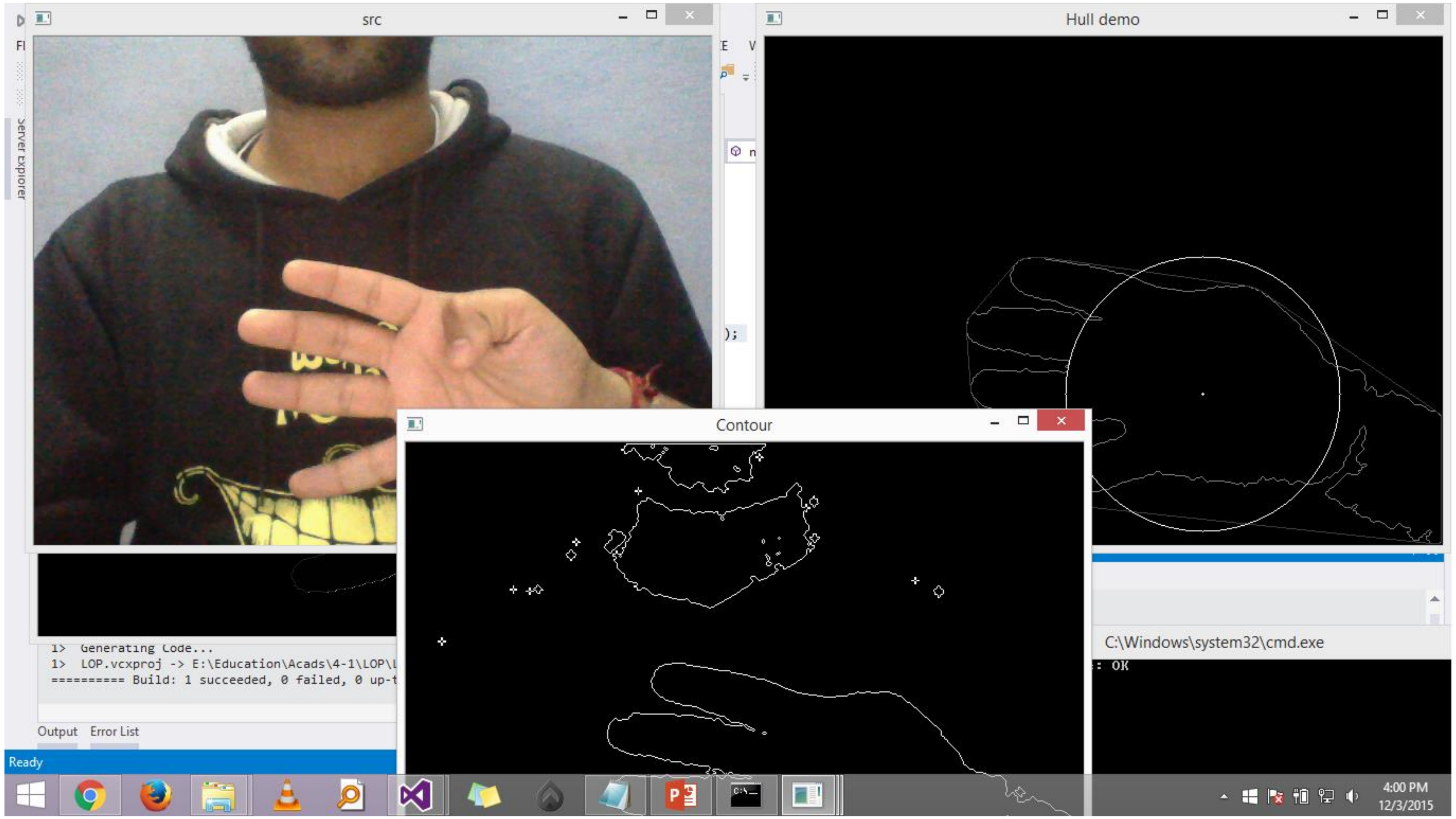
Show output from: Build

Build succeeded

Background subtraction

- Involves capture of static background frame which remains unchanged throughout the duration of the process
- This frame is subtracted from the live video feed and the resulting frame is subjected to a threshold so as to convert the image to binary
- An anomaly such as a hand, or a moving object is easily detected since the difference in pixel values of the anomaly and the background exceeds the value of the threshold
- Gesture recognition algorithms can then be performed on the resulting image

Skin colour – HSV-based

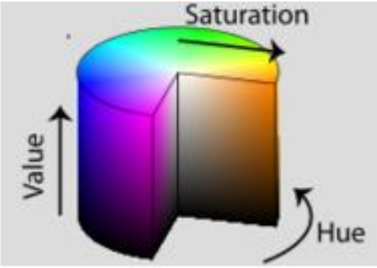


```
1>  Generating Code...
1>  LOP.vcxproj -> E:\Education\Acads\4-1\LOP\l
========== Build: 1 succeeded, 0 failed, 0 up-t
```

```
: OK
```

HSV – based model

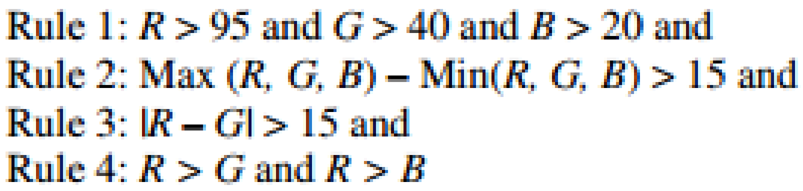
- HSV – Hue, saturation, value



- Skin colour lies within a certain range of this colour space
- This method involves sampling the pixel HSV values of the input video stream and thresholding them considering the HSV range of human skin colour
- Considers only the absolute HSV values of the pixels

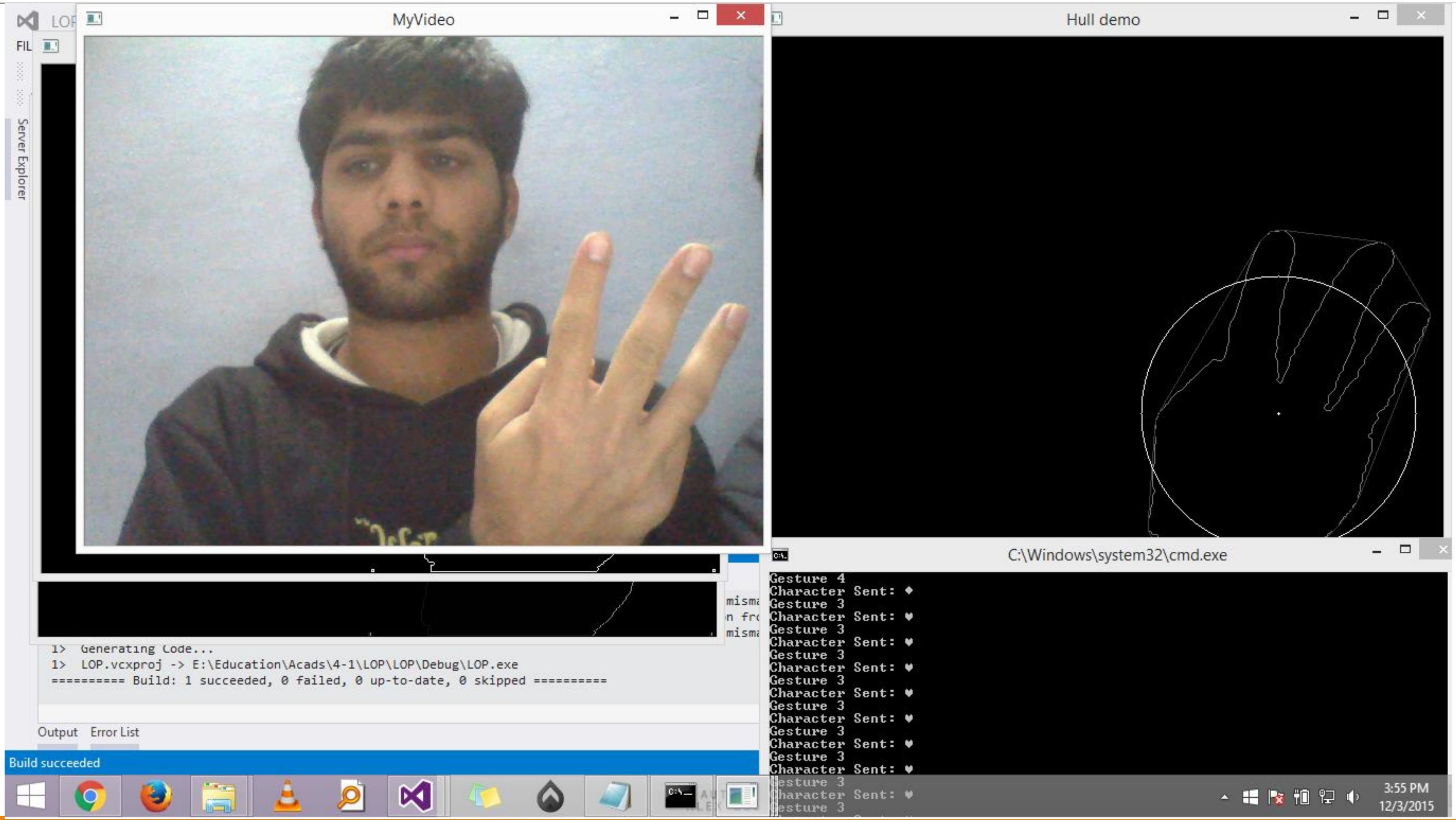
Kovac model

- Similar to HSV model, but relies on certain ratios of the pixel RGB values to identify pixels that are of human skin colour
- Works well for a variety of skin colour ranges, and does not rely solely on the absolute values of the pixel colours
- Works well even in a dynamic background subject to the condition that there is no skin coloured-object present



Rule 1: $R > 95$ and $G > 40$ and $B > 20$ and
Rule 2: $\mathrm{Max}(R, G, B) - \mathrm{Min}(R, G, B) > 15$ and
Rule 3: $|R - G| > 15$ and
Rule 4: $R > G$ and $R > B$

Skin Colour – Kovac model (finger count)



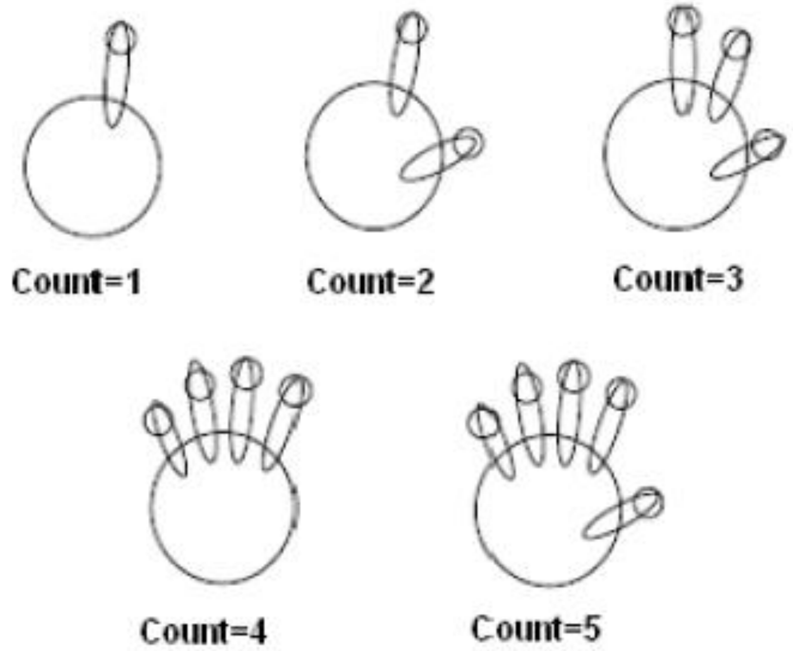
```
misma
n fr
misma
1>  Generating Code...
1>  LOP.vcxproj -> E:\Education\Acads\4-1\LOP\LOP\Debug\LOP.exe
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

```
Gesture 4
Character Sent: ♦
Gesture 3
Character Sent: ♥
Gesture 3
Character Sent: ♥
Gesture 3
Character Sent: ♥
Gesture 3
Character Sent: ♥
Gesture 3
Character Sent: ♥
Gesture 3
Character Sent: ♥
Gesture 3
Character Sent: ♥
Gesture 3
Character Sent: ♥
Gesture 3
```

Kovac model – finding number of fingers

Identification of number of fingers being shown to the camera by the user

- Aims to find the connected components including the hand and the fingers.
- The image is converted to grayscale equivalent.
- Finds the centre of this single component.
- A circle of a radius less than the distance between the centre and the tip of the middle finger is drawn around this centre.
- Finally, the number of transitions between black to white and white to black are counted which gives us the number of fingers being shown by the user.



Count=1

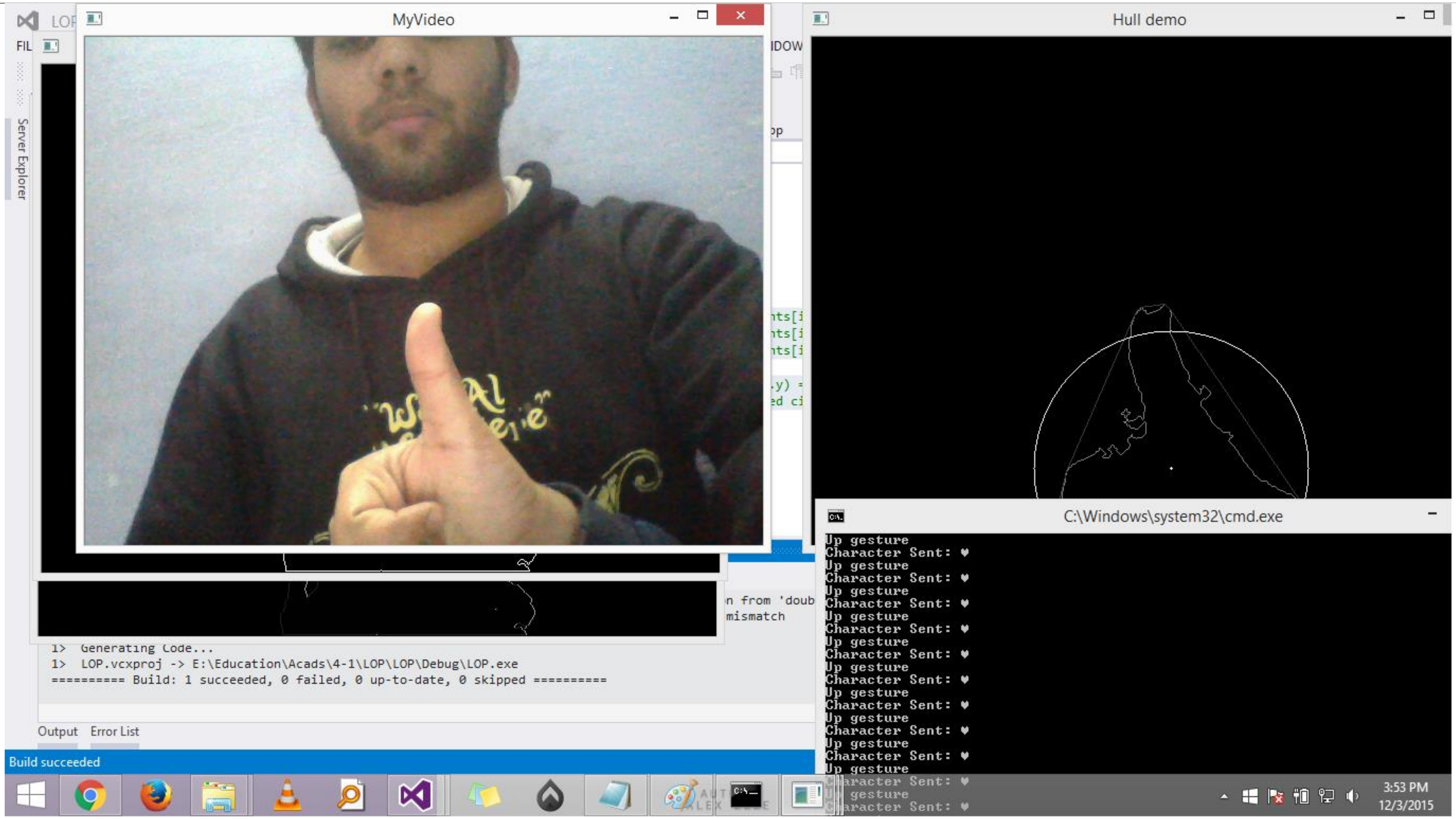
Count=2

Count=3

Count=4

Count=5

Kovac model (direction determination)



```
n from 'doub
mismatch
1>  Generating Code...
1>  LOP.vcxproj -> E:\Education\Acads\4-1\LOP\LOP\Debug\LOP.exe
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

```
Up gesture
Character Sent: ♥
Up gesture
Character Sent: ♥
Up gesture
Character Sent: ♥
Up gesture
Character Sent: ♥
Up gesture
Character Sent: ♥
Up gesture
Character Sent: ♥
Up gesture
Character Sent: ♥
Up gesture
Character Sent: ♥
Up gesture
Character Sent: ♥
Up gesture
Character Sent: ♥
Up gesture
Character Sent: ♥
```

Kovac model – direction determination

- Similar to finger count, we begin with finding the centre of the binarized image
- The contour is iterated over so as to find the maximum distance of the contour pixels from the centre
- The direction is interpreted based on the relative position of the farthest pixel of the contour from the centre
- The robot is controlled using this direction

THANK YOU!