An Image Processing Technique for the translation of Indian Sign Language Finger-Spelling to Audio And Text

Project Guide

Sandeep Kumar E

Assistant Professor

Department Of TCE

JNNCE, Shimoga

Team Members :

1. Dhenuka M
   4JN11TE009
2. Shreekantha A N
   4JN11TE038
3. Spoorthi M N
   4JN11TE047
4. Sujay Sharma
   4JN11TE051

Overview

- Introduction
- Aim of the project
- Literature Survey
- Structure of a Sign Language
- Block Diagram
- HOG Feature Extraction
- Classification (SVM)
- Working Principle
- System Flowchart
- Results
- Hardware Implementation
- Challenges
- Project Progress

Introduction-facts

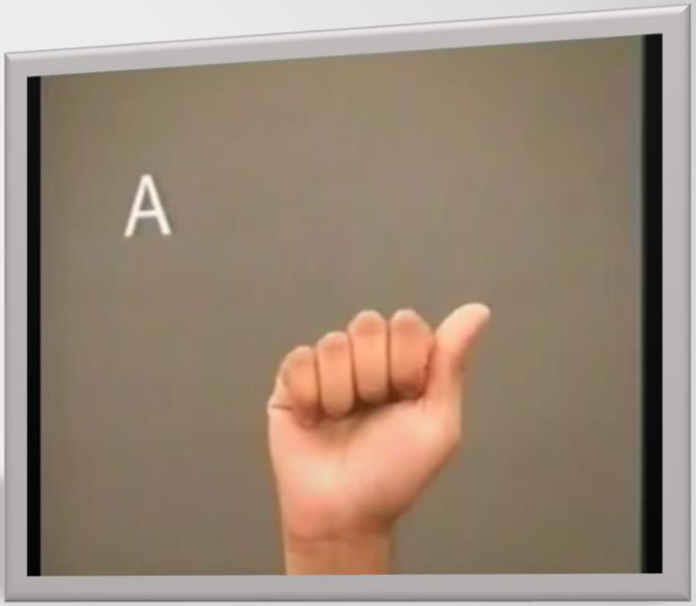
- Over 5% of the world's population – 360 million people have disabling hearing loss

  (328 million adults and 32 million children)
- The Indian deaf population of 1.1 million is 98% illiterate
- At least 1 in 12 Indians suffer from hearing loss
- Sign Language is taught in less than 2% of the schools in India
- The trained teachers for this population is only 3,006 in 388 schools whereas untrained teacher strength is 889 in 243 schools

Facts courtesy : WHO and National Association of the Deaf

Introduction

- **Sign Language:** A language which uses manual communication and body language to convey meaning, as opposed to acoustically conveyed sound patterns
- Used primarily by hearing-impaired people
- **NOT** derived from spoken language



Aim of the project

- Development of an efficient Image Processing algorithm
- Successful implementation of Image Processing algorithm
- Detection of Sign Language Finger-Spelling
- Conversion to Audio and Text
- Build a portable model which is cost effective and helpful
- Bridge the gap between the normal and hearing impaired people

Literature survey

- Sign Language has static and dynamic letters.

C

J

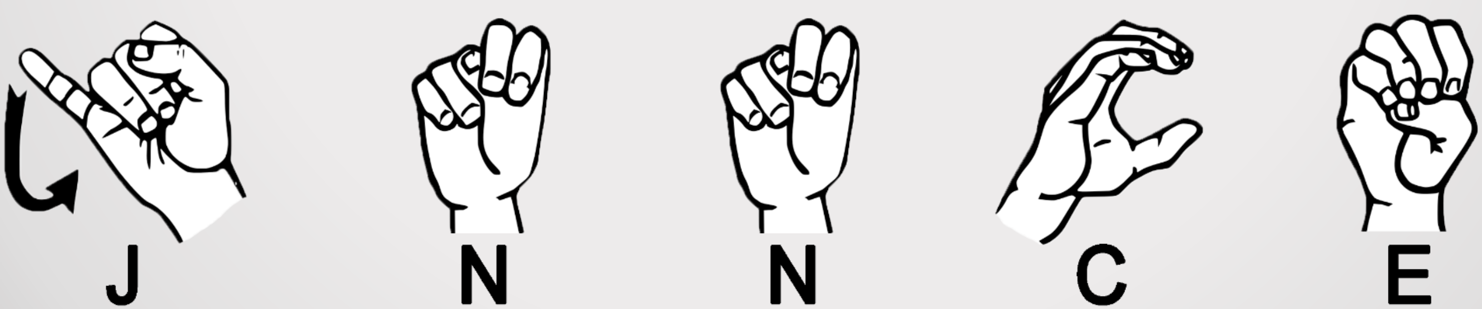
- Ghotkar [1] in his paper tracked hand features and recognition was achieved using Genetic Algorithm. This method has success rate of 86%.
- Starner in [2] used Hidden Markov Model for recognition of American Sign Language. His work was limited to **coloured gloves**. He achieved a success rate of 99%.
- Singha in [3] used Indian sign language was recognized using Eigen value weighted Euclidean distance based classifier on **still images**. He got accuracy rate of 97%.

Structure of a sign language

- Five basic parameters
  - Shape of the hand
  - Place of articulation (location)
  - Movement
  - Palm Orientation
    - ✓ Region of the hand contacting body
    - ✓ Orientation of the hand to body
    - ✓ Orientation of hands to each other
  - Facial Expressions

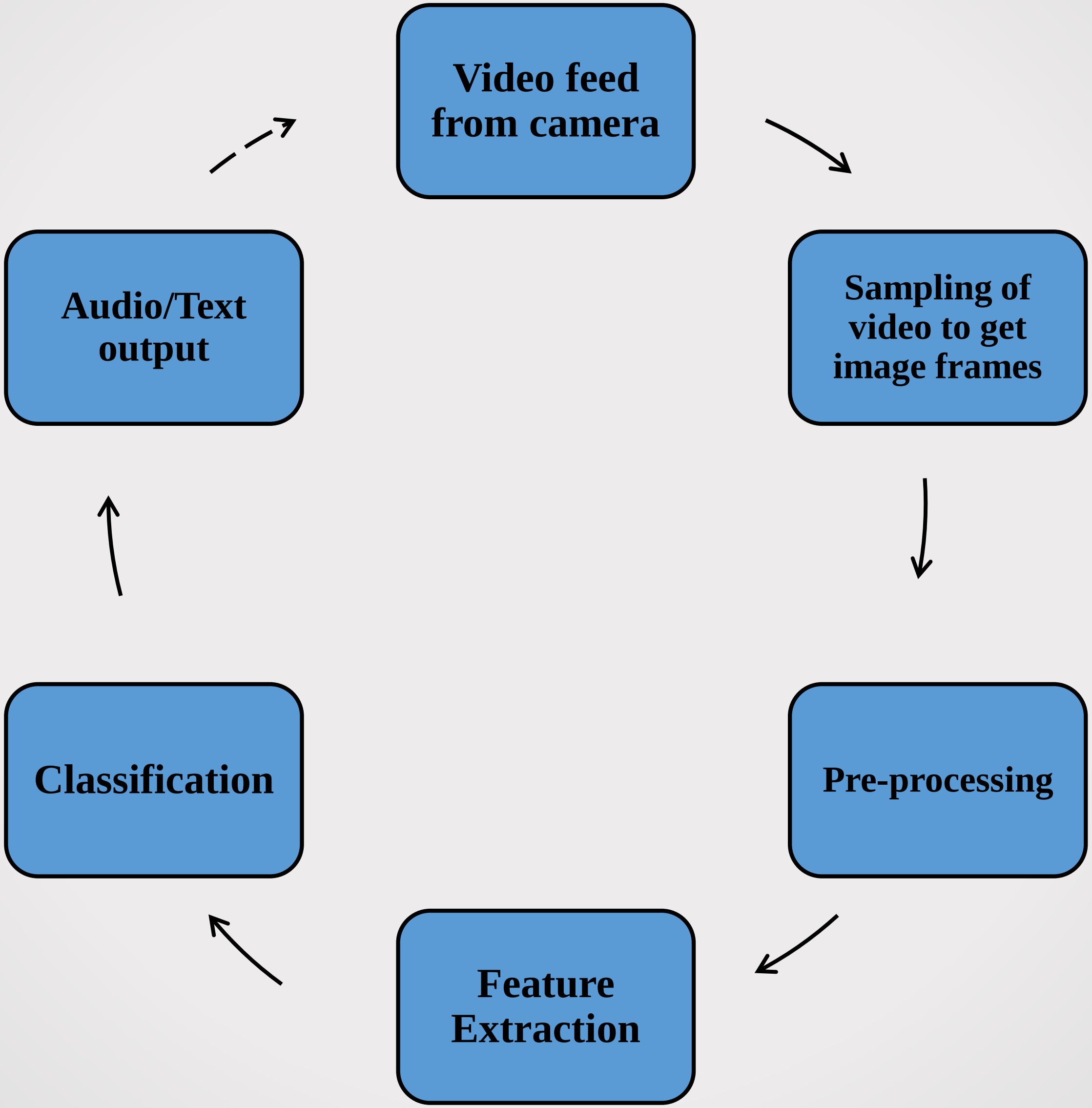
Structure of a sign language

- Fingerspelling/Manual Alphabet
  - Words without assigned signs may be spelled
  - Some commonly spelled words become lexicalized (e.g. *NO*)
  - Not the same hand shape is used in every sign language



Alphabets

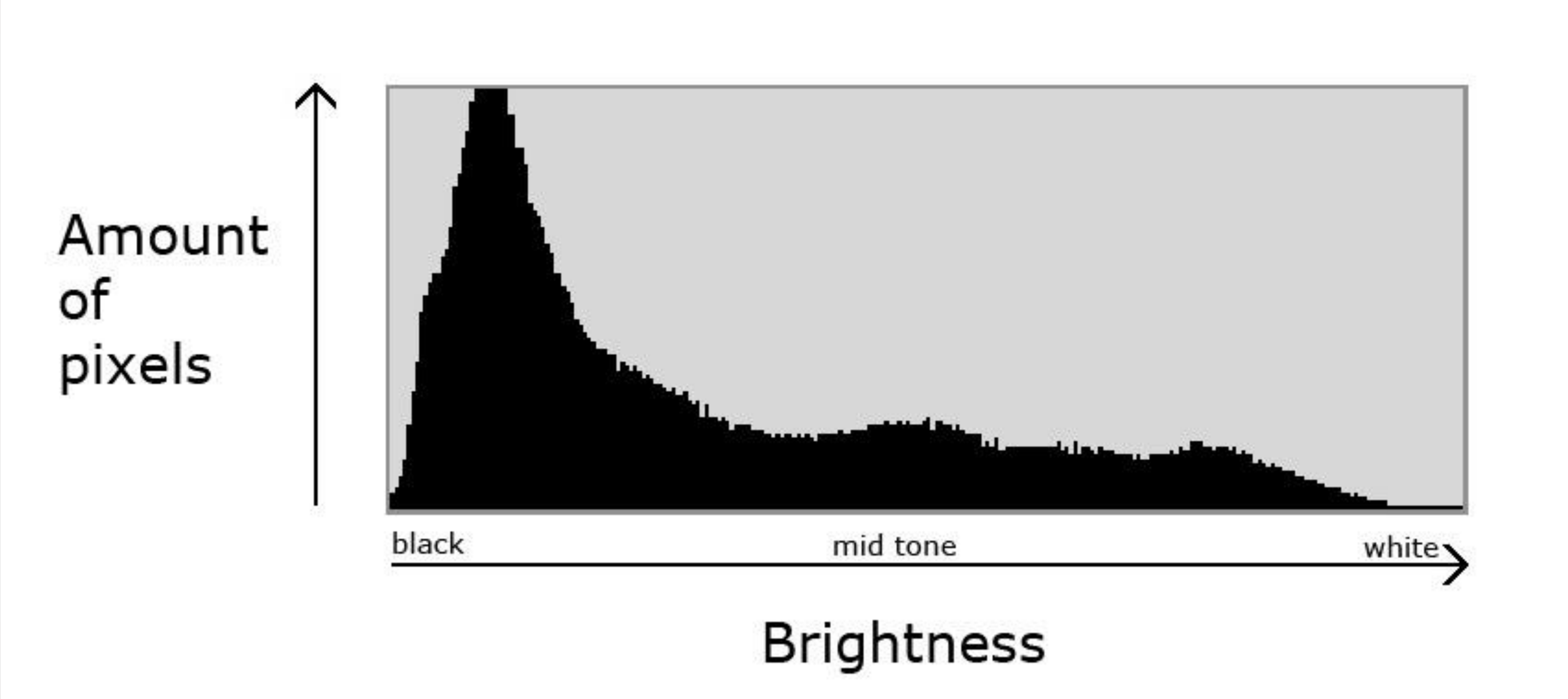
Block Diagram



HOG Feature Extraction

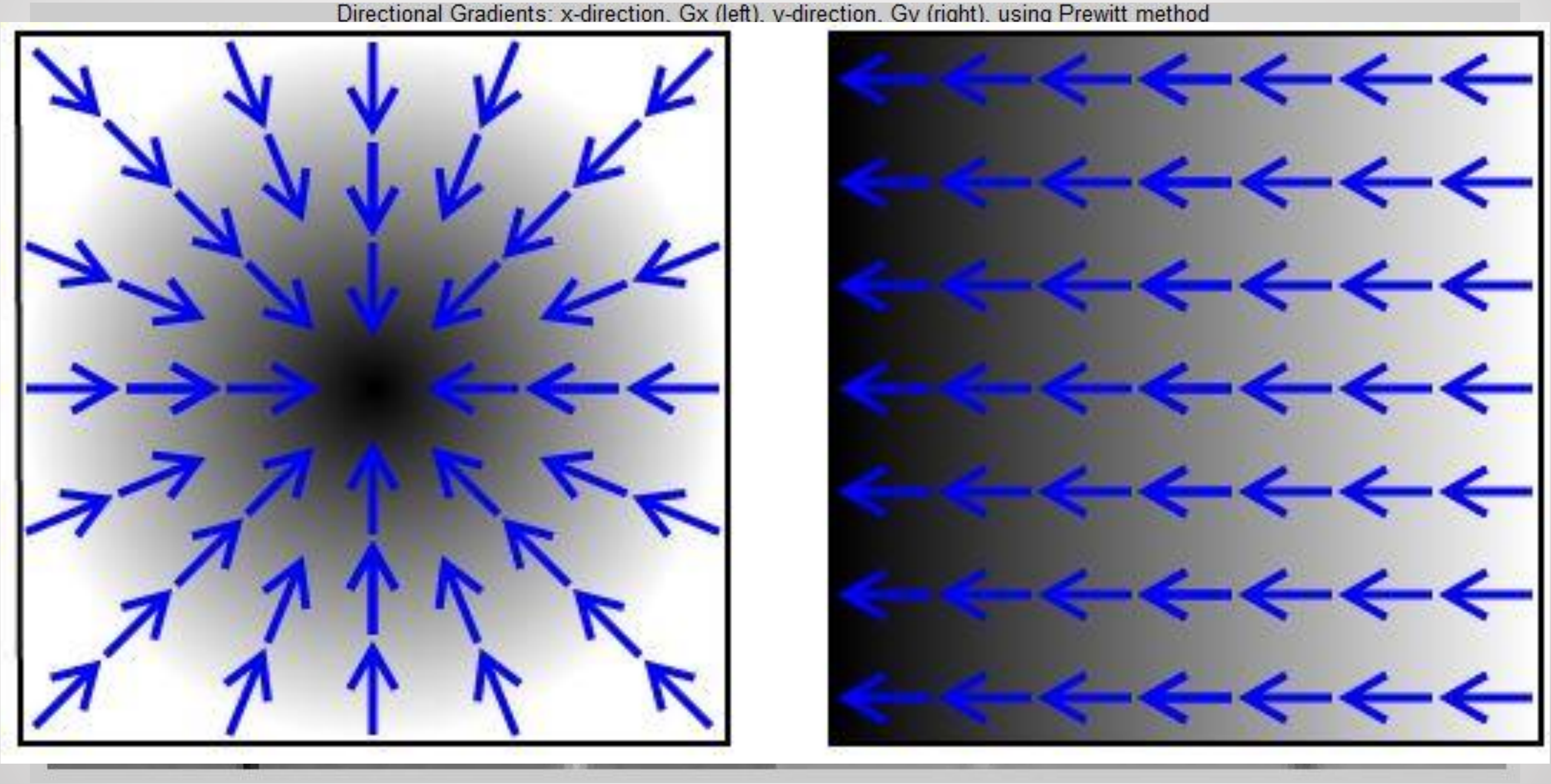
- Histogram of Oriented Gradients
- Used for the purpose of object detection.
- Counts occurrences of gradient orientation in localized portions of an image
- Navneet Dalal and Bill Triggs, researchers for the French National Institute for Research in Computer Science and Control (INRIA), first described HOG descriptors.
- The image is divided into small connected regions called cells
- For the pixels within each cell, a histogram of gradient directions is compiled
- The descriptor is then the concatenation of these histograms

Histogram

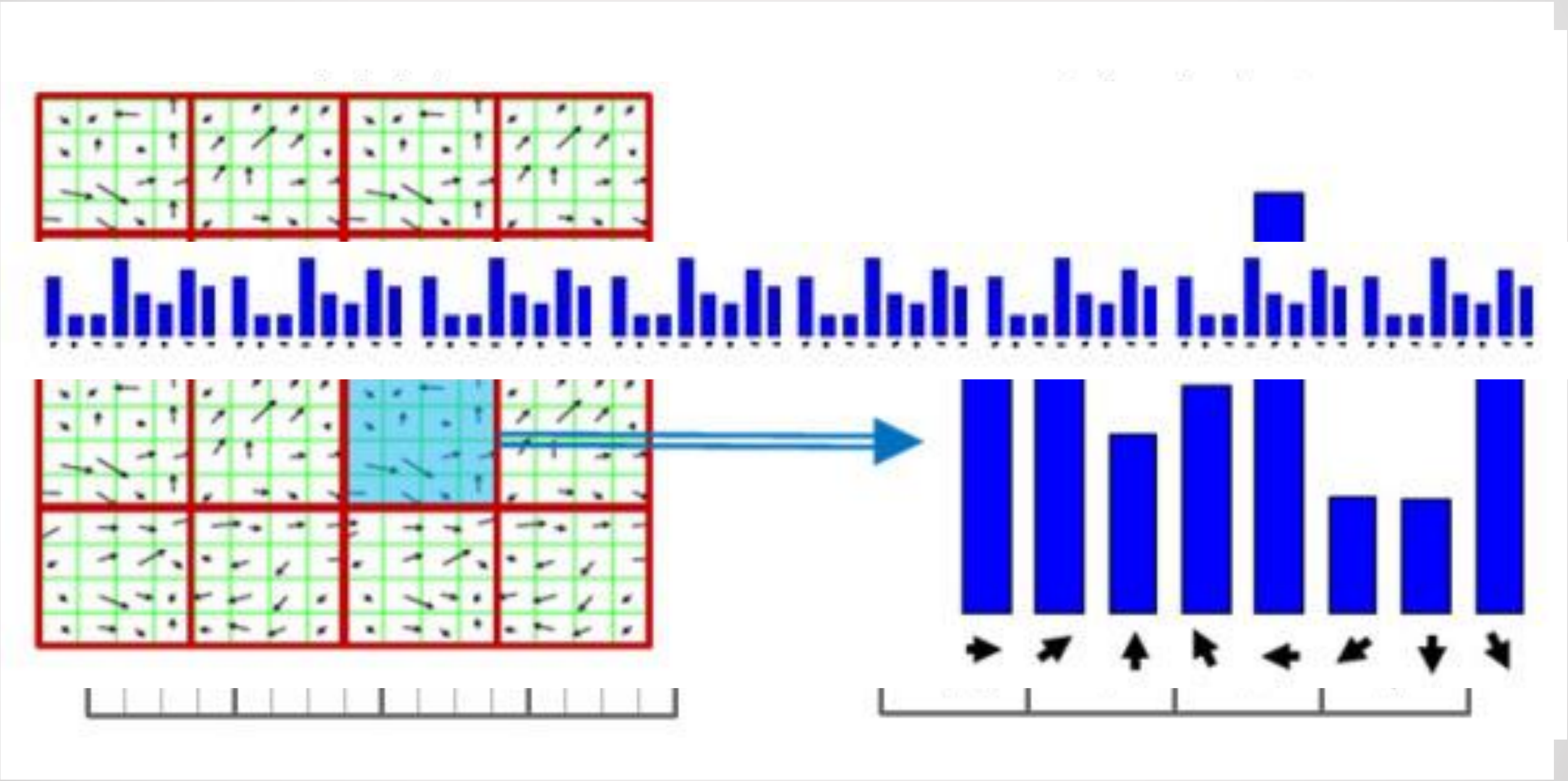


Oriented Gradients

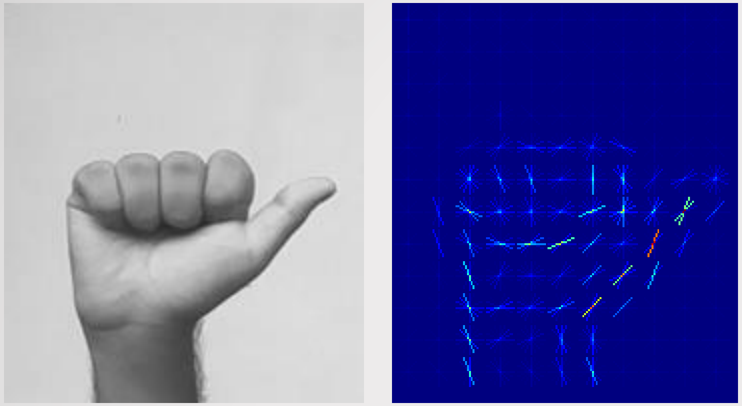
Image Gradient



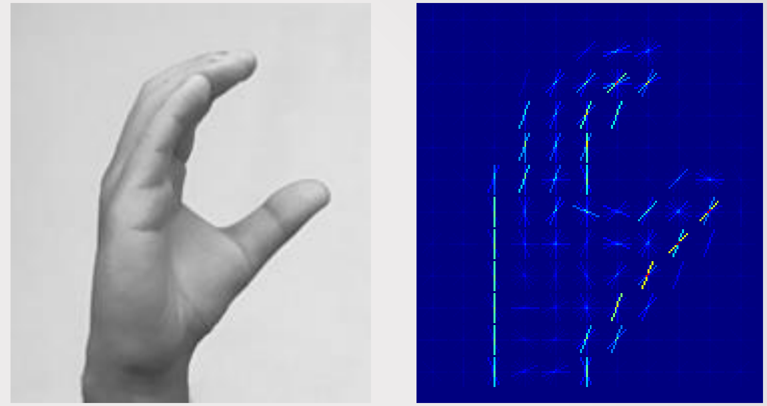
HOG Descriptor



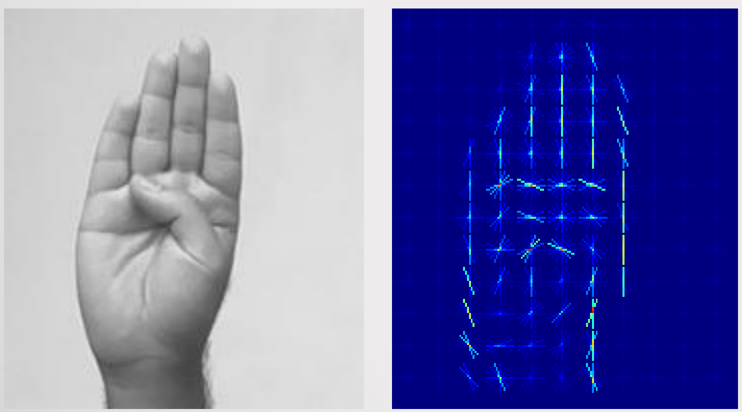
HOG



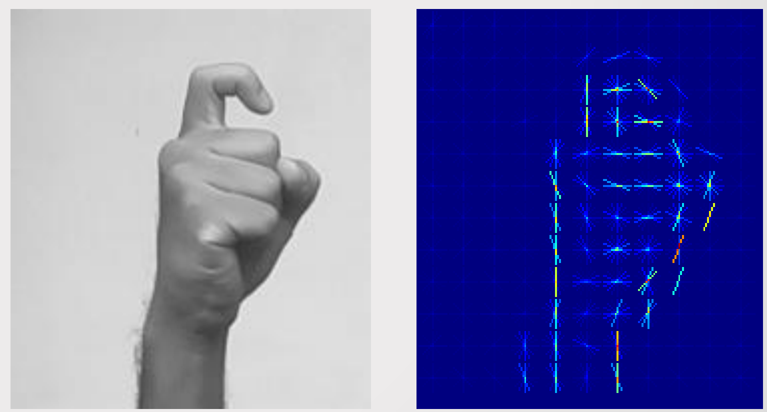
A



C



B



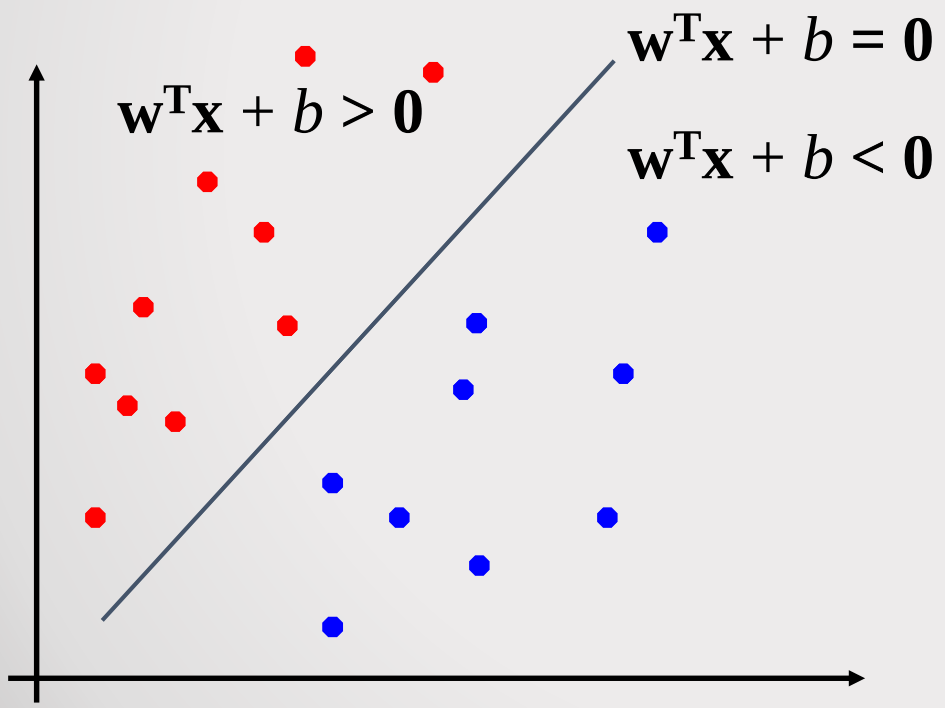
X

Support Vector Machine

- Supervised learning models with associated learning algorithms that analyse data and recognize patterns
- Given a set of training examples, each marked as belonging to one of two categories, an SVM builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear classifier
- An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible

Linear Separators

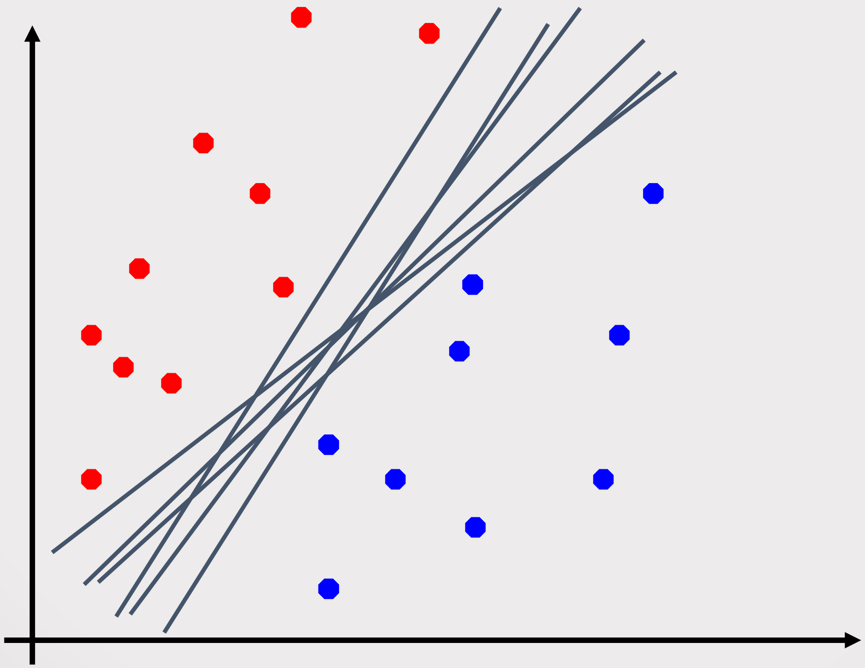
- Binary classification can be viewed as the task of separating classes in feature space:



$$f(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} + b)$$

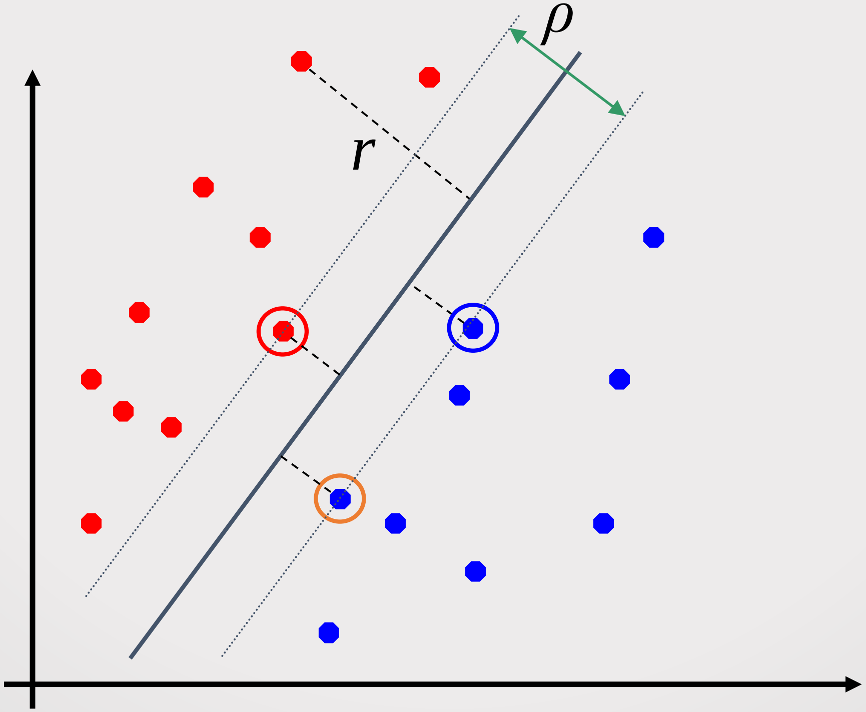
Linear Separators

- Which of the linear separators is optimal?



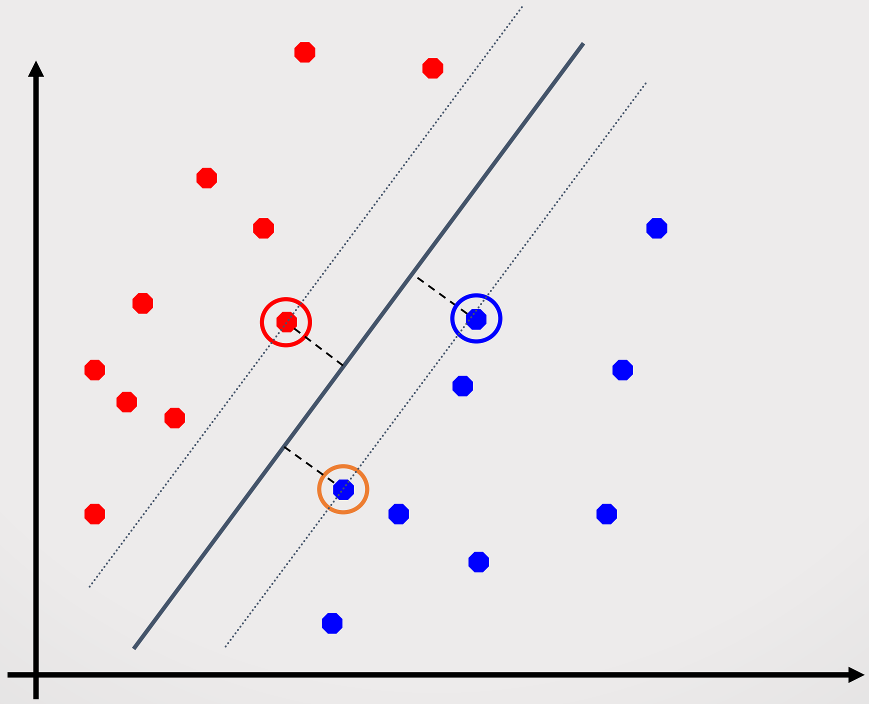
Classification Margin

- Distance from example $\mathbf{x}_i$ to the separator is $r = \dfrac{\mathbf{w}^T\mathbf{x}_i + b}{\|\mathbf{w}\|}$
- Examples closest to the hyperplane are ***support vectors***.
- ***Margin*** ρ of the separator is the distance between support vectors.



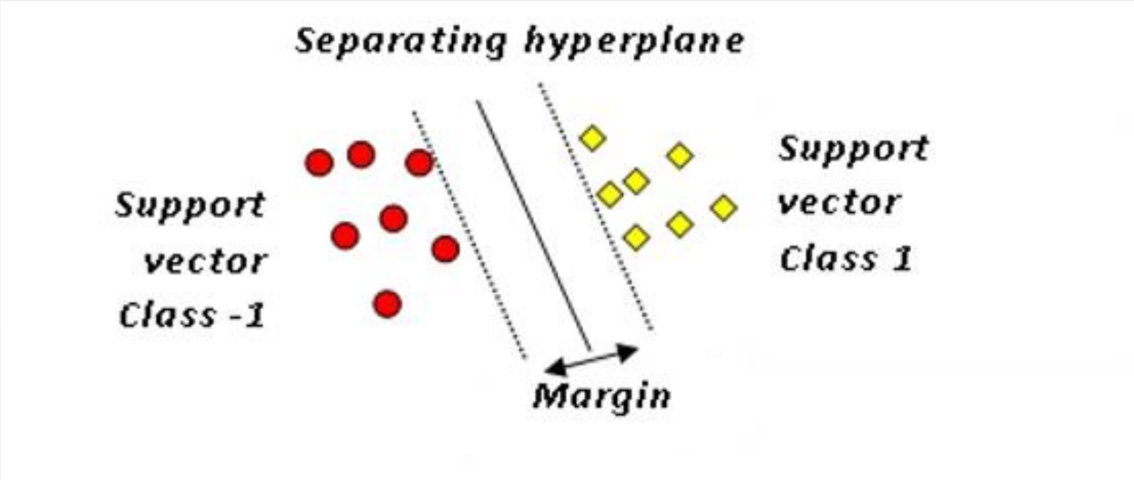
Maximum Margin Classification

- Maximizing the margin is good according to intuition and PAC theory.
- Implies that only support vectors matter; other training examples are ignorable.



Overview of SVM

- A Support Vector Machine (SVM) performs classification by finding the hyperplane that maximises the margin between two classes
- It draws the widest channel between two classes
- The two class labels are +1 (positive samples) -1 (negative samples)



Linear SVM Mathematically

- Let training set $\{(\mathbf{x}_i, y_i)\}_{i=1..n}$, $\mathbf{x}_i \in \mathbf{R}^d$, $y_i \in \{-1, 1\}$ be separated by a hyperplane with margin ρ. Then for each training example $(\mathbf{x}_i, y_i)$:

$$\begin{array}{l} \mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b \leq -\rho/2 \quad \text{if } y_i = -1 \\ \mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b \geq \rho/2 \quad \text{if } y_i = 1 \end{array} \quad \Leftrightarrow \quad y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq \rho/2$$

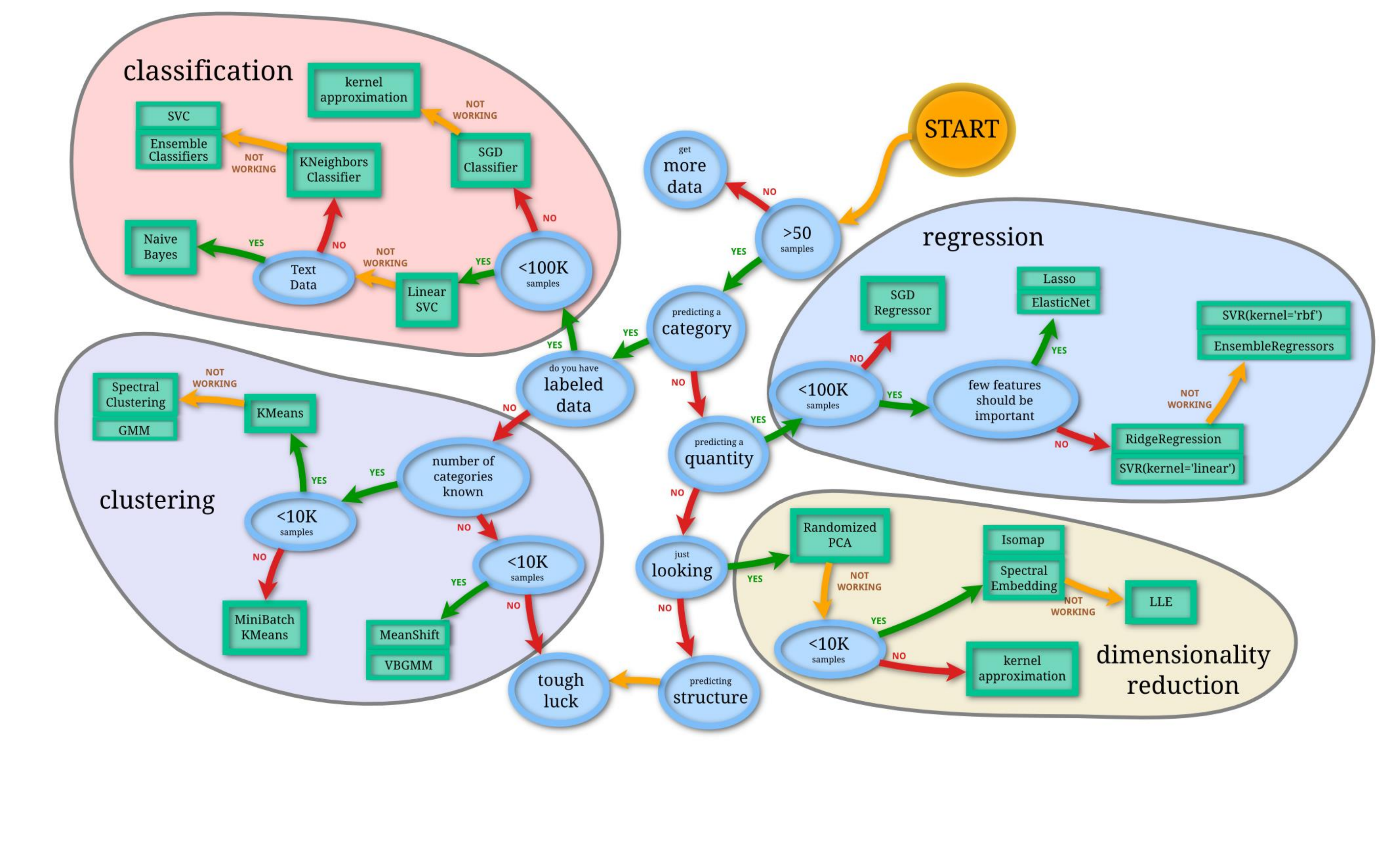
- For every support vector $\mathbf{x}_s$ the above inequality is an equality. After rescaling **w** and *b* by $\rho/2$ in the equality, we obtain that distance between each $\mathbf{x}_s$ and the hyperplane is

$$r = \frac{\mathrm{y}_s(\mathbf{w}^T\mathbf{x}_s + b)}{\|\mathbf{w}\|} = \frac{1}{\|\mathbf{w}\|}$$

- Then the margin can be expressed through (rescaled) **w** and b as:

$$\rho = 2r = \frac{2}{\|\mathbf{w}\|}$$

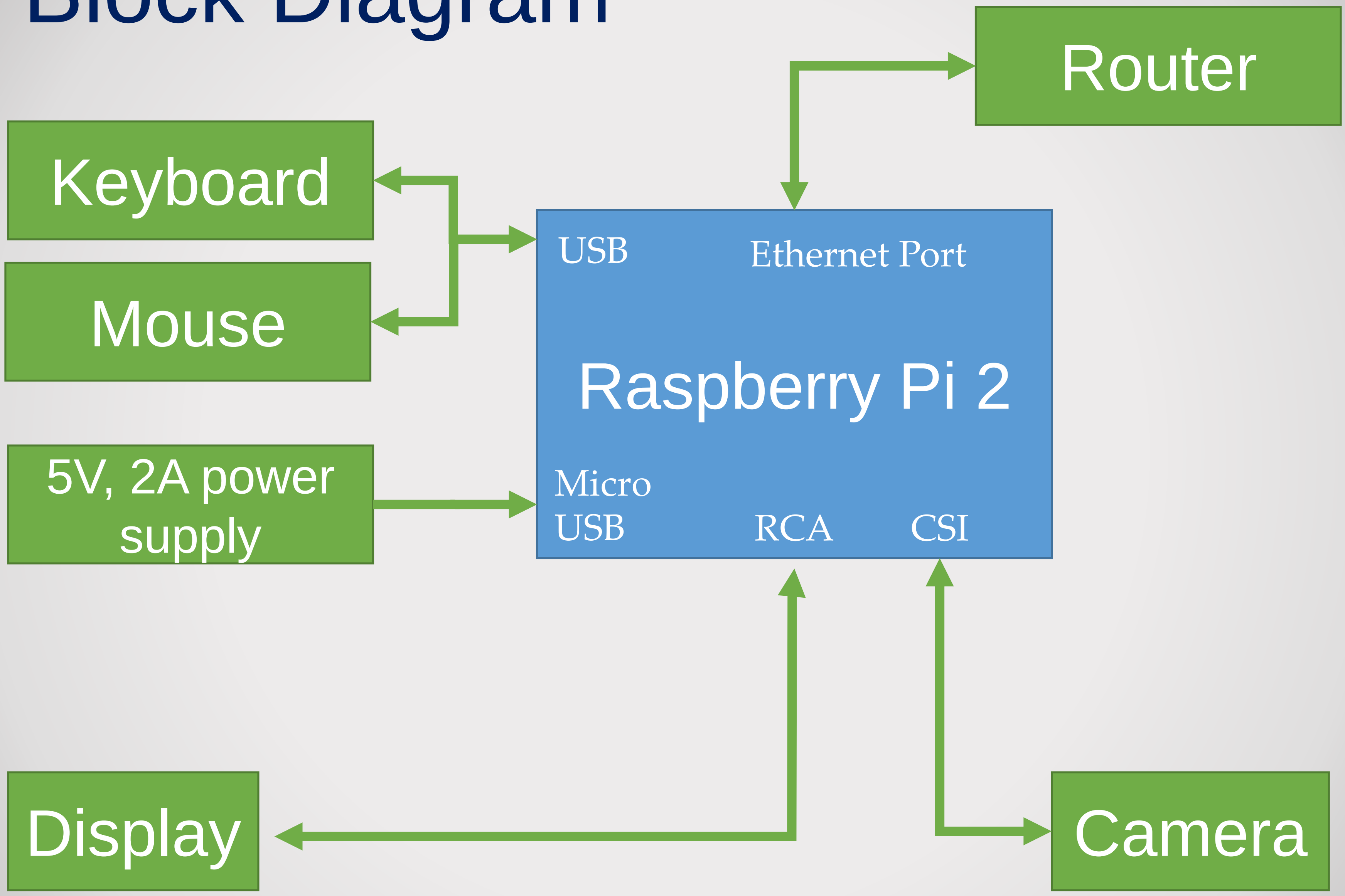
Flowchart to approach problems to estimate data



Software dependencies in Raspberry Pi 2

- Raspbian Wheezy
- Python
- OpenCV
- Scikit-learn
- Pip
- Numpy
- Matplotlib
- Skimage (Image Processing SciKit)

Block Diagram

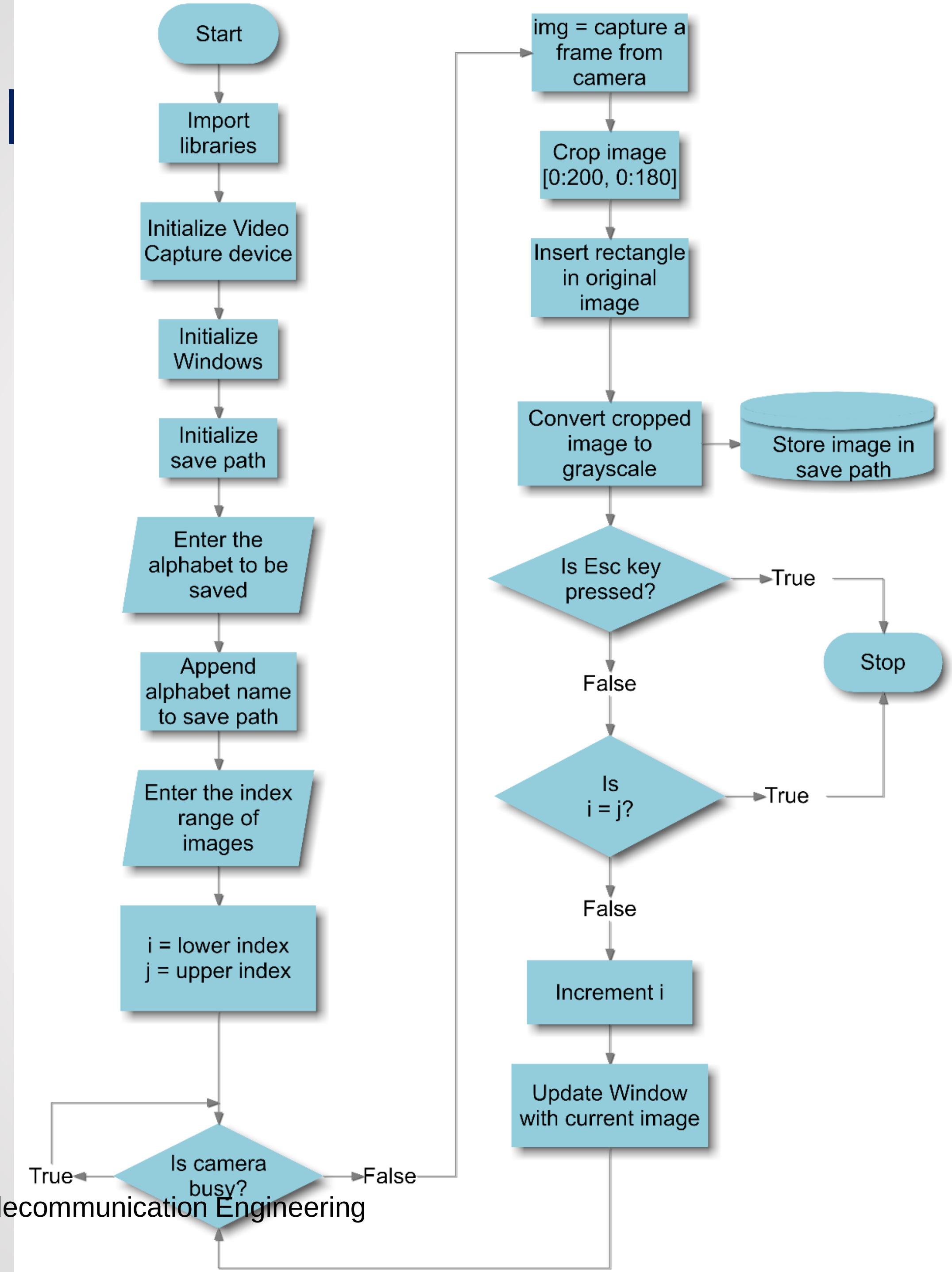


System overview

Our project works by:

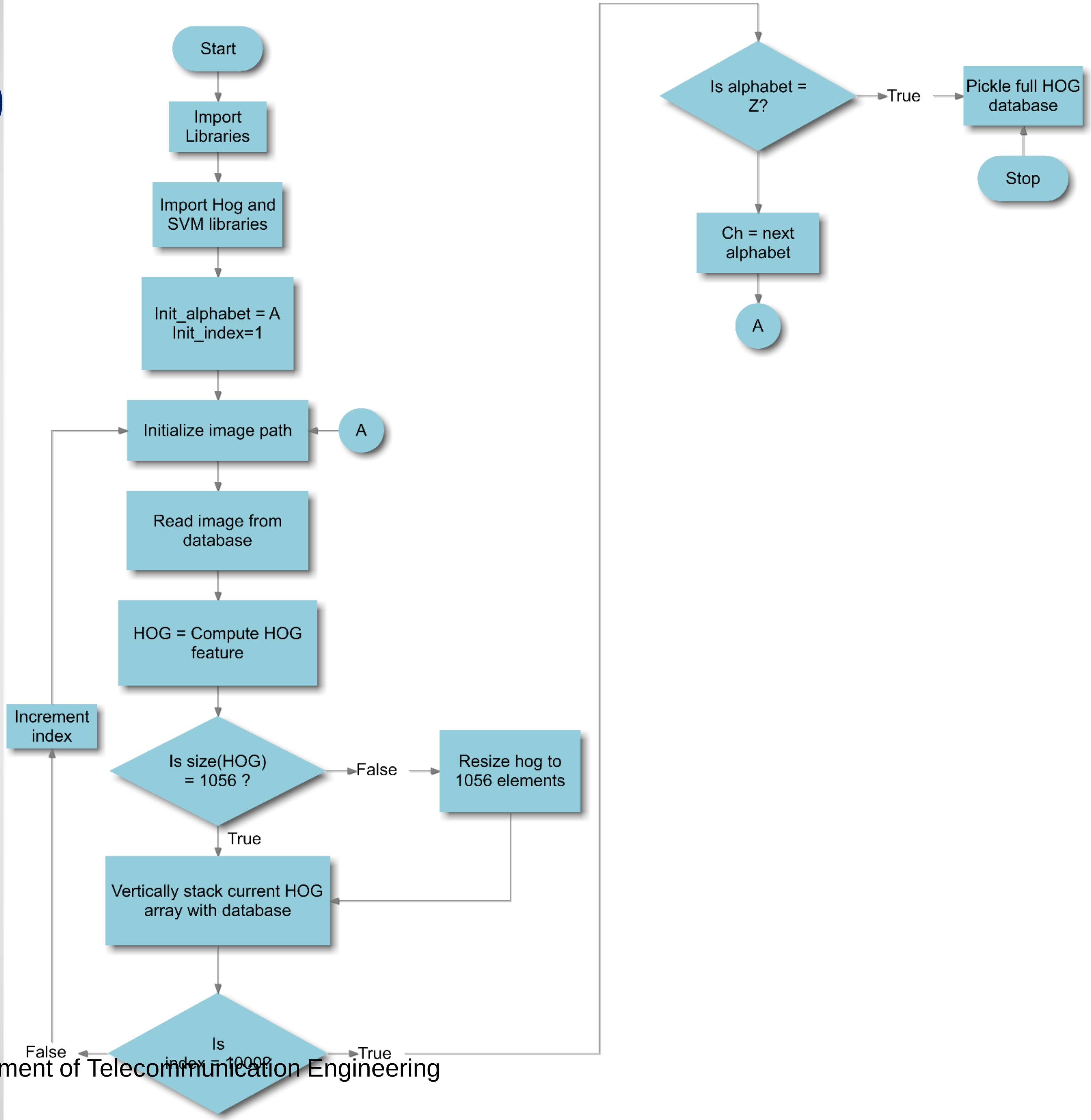
- Taking 1000 samples of each alphabet
- Storing it in memory
- Computing HOG feature for each sample
- Generating HOG database of all samples
- Training a multi class SVM classifier
- Storing SVM classifier in memory
- Using trained SVM classifier in real-time video for classification.

Acqui

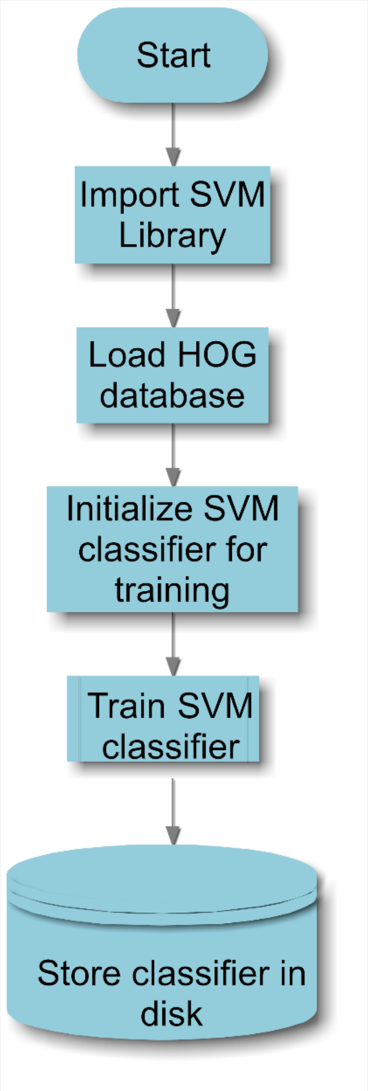


Start
Import libraries
Initialize Video Capture device
Initialize Windows
Initialize save path
Enter the alphabet to be saved
Append alphabet name to save path
Enter the index range of images
i = lower index
j = upper index
Is camera busy?
True
False
img = capture a frame from camera
Crop image [0:200, 0:180]
Insert rectangle in original image
Convert cropped image to grayscale
Store image in save path
Is Esc key pressed?
True
False
Stop
Is i = j?
True
False
Increment i
Update Window with current image

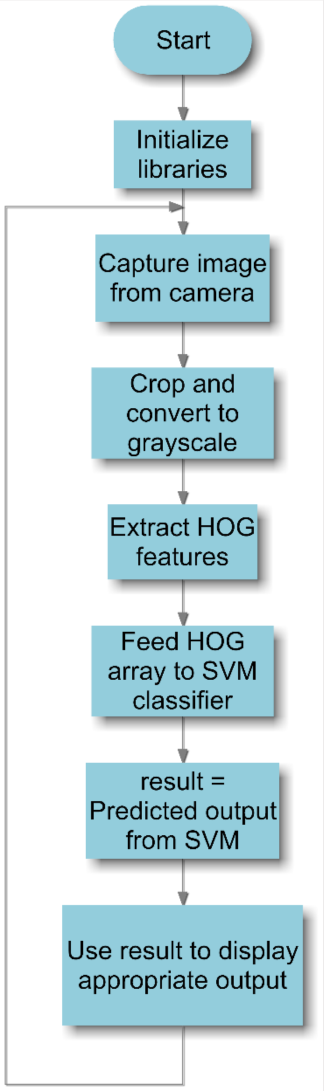
Co

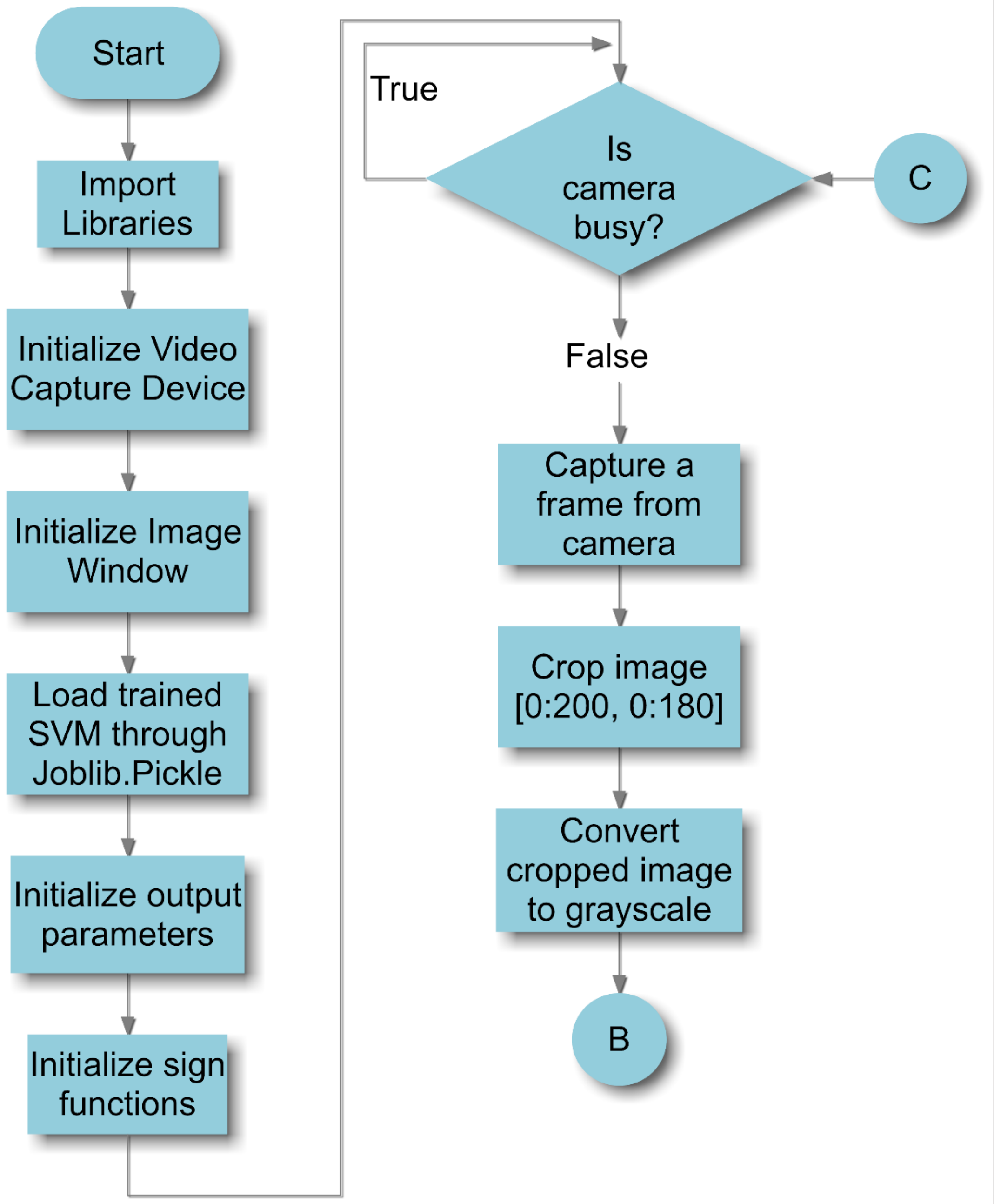

Start
Import Libraries
Import Hog and SVM libraries
Init_alphabet = A
Init_index=1
Initialize image path
A
Read image from database
HOG = Compute HOG feature
Is size(HOG) = 1056 ?
False
Resize hog to 1056 elements
True
Vertically stack current HOG array with database
Increment index
Is
index = 1000?
False
True
Is alphabet = Z?
True
Pickle full HOG database
Stop
Ch = next alphabet
A

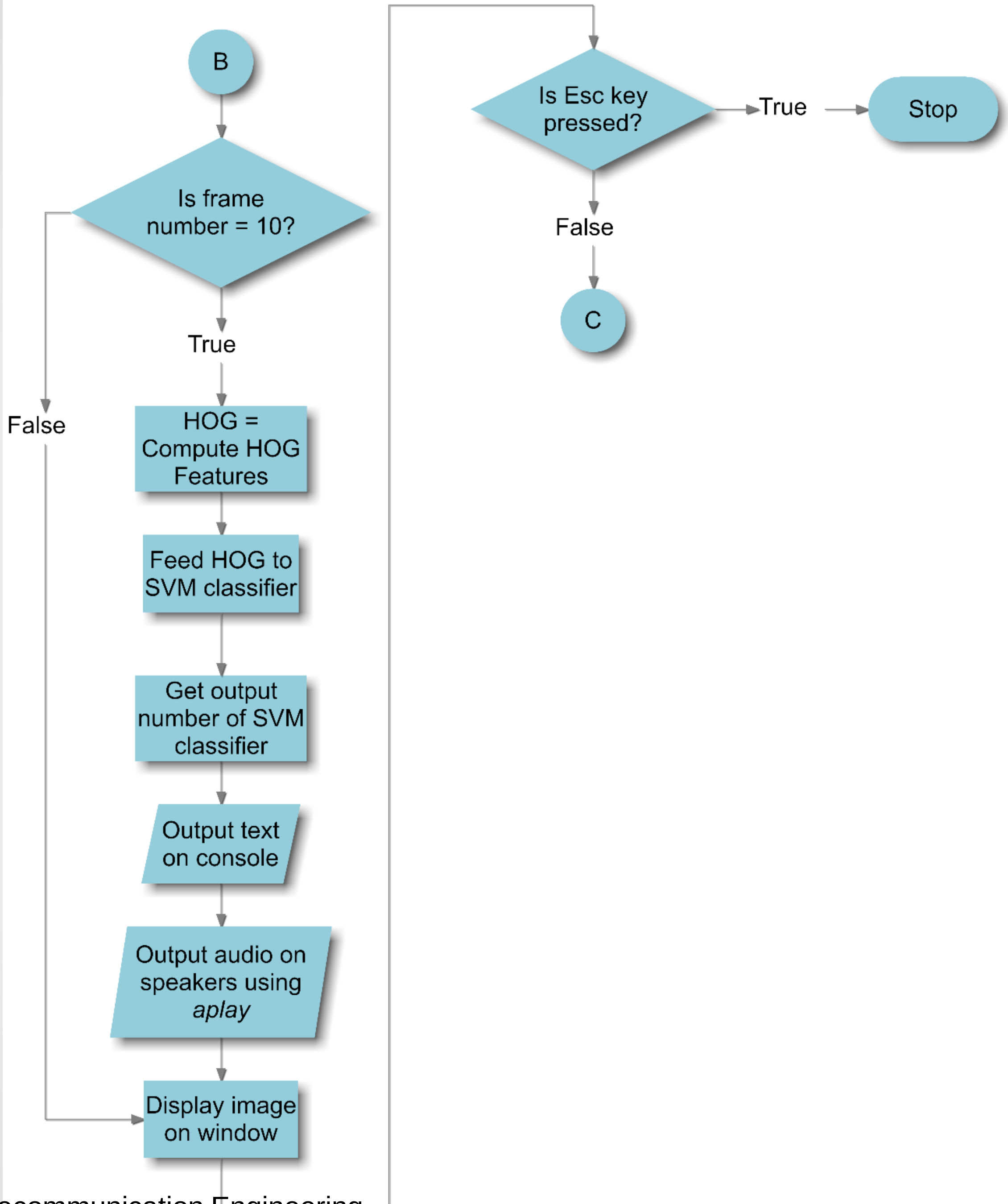
Training SVM classifier



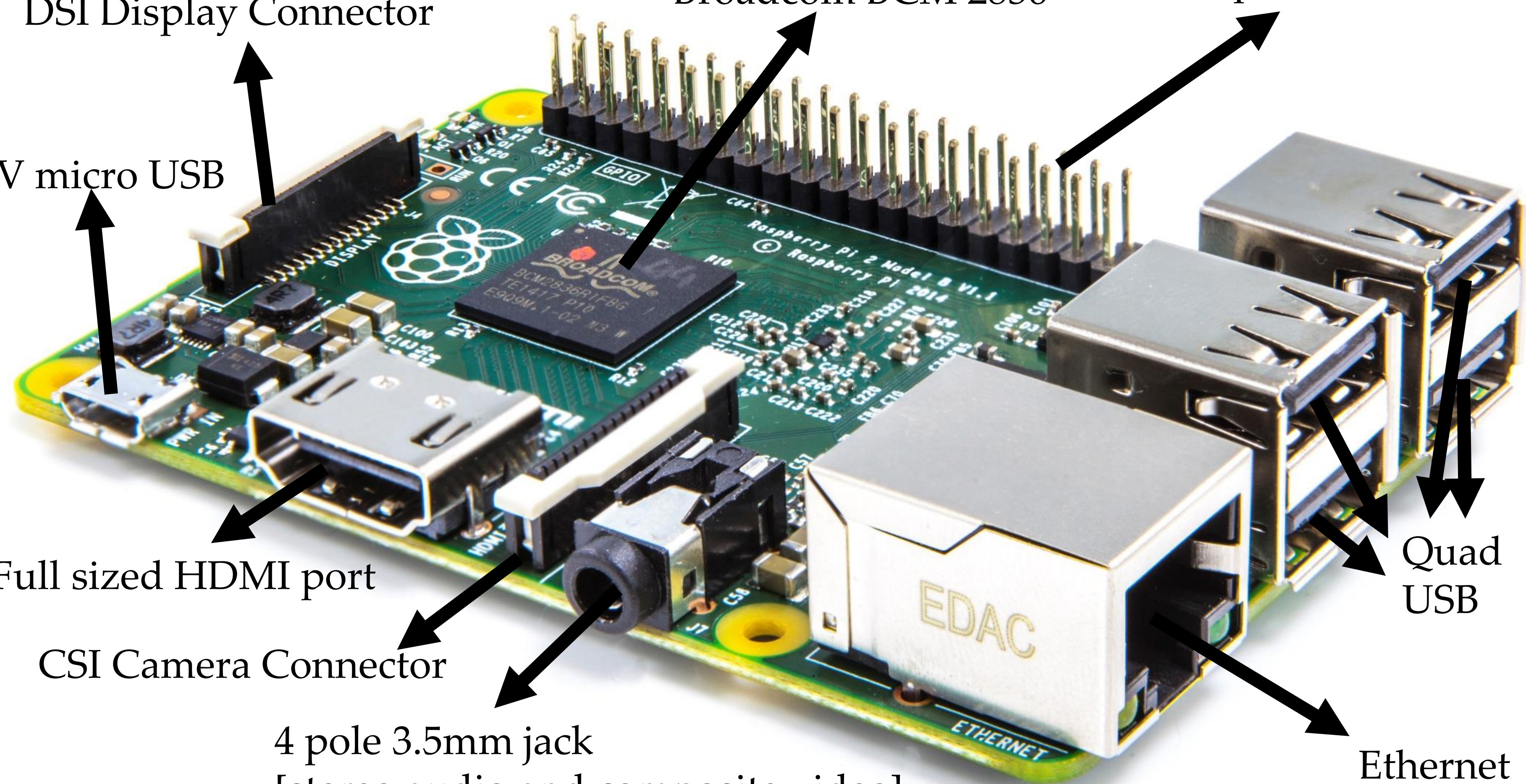
System Flowchart




Start
Import Libraries
Initialize Video Capture Device
Initialize Image Window
Load trained SVM through Joblib.Pickle
Initialize output parameters
Initialize sign functions
True
Is camera busy?
C
False
Capture a frame from camera
Crop image [0:200, 0:180]
Convert cropped image to grayscale
B


B
Is frame number = 10?
True
False
HOG = Compute HOG Features
Feed HOG to SVM classifier
Get output number of SVM classifier
Output text on console
Output audio on speakers using *aplay*
Display image on window
Is Esc key pressed?
True
Stop
False
C

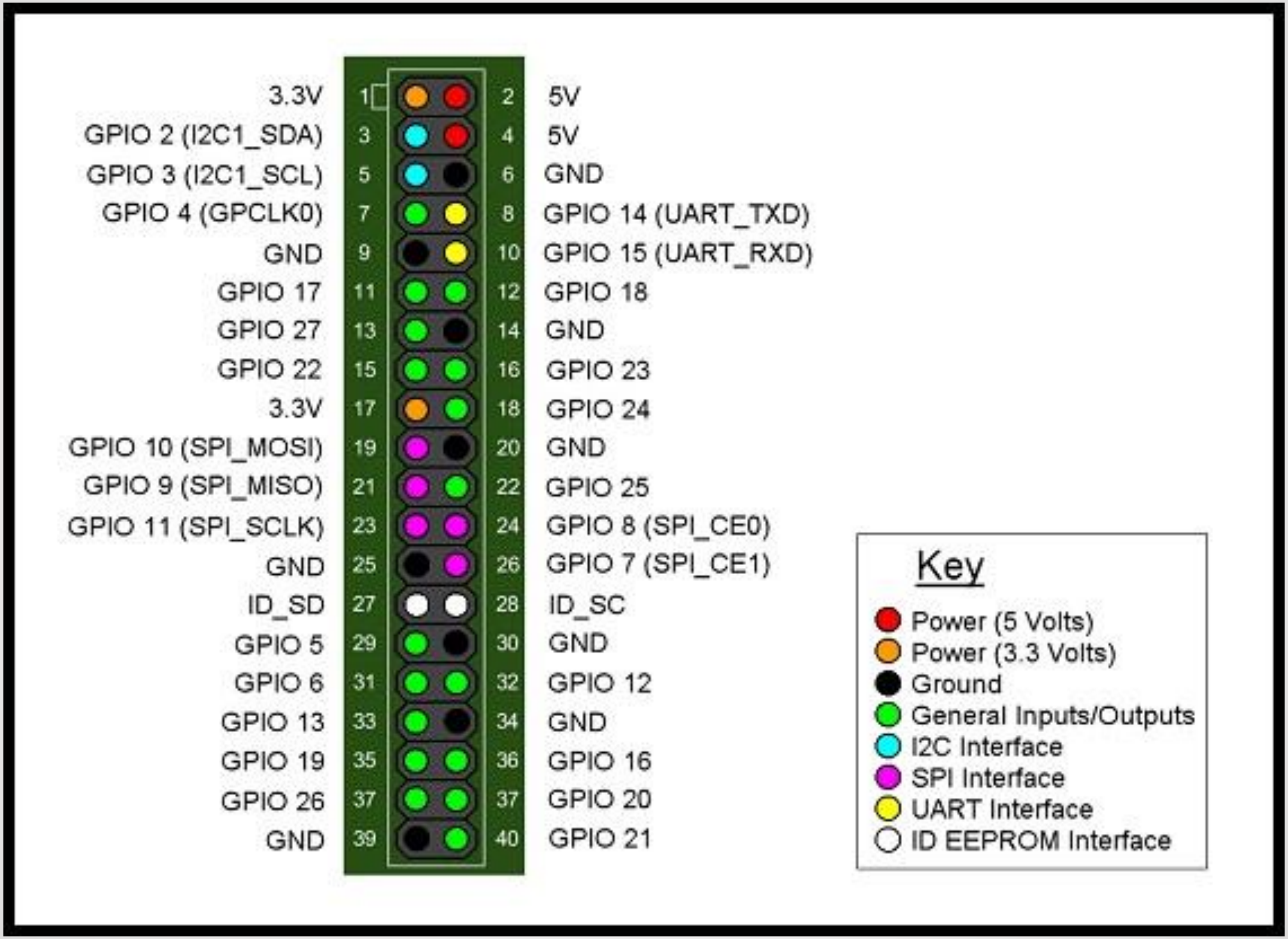
Block Diagram



Raspberry Pi 2

- 900MHz quad-core Broadcom BCM2836 CPU with 1GB DDR2 RAM
- VideoCore IV 3D graphics core
- 40 pin extended pins - with 27 GPIO pins
- Micro SD slot
- Multiple ports: Four USB ports, full sized HDMI, four pole stereo output and composite video port. CSI camera port and DSI display port
- 10/100 BaseT Ethernet
- Micro-USB power source 5V, 2A
- Dimensions: 85 x 56 x 17mm

Pin Diagram



3.3V 1 2 5V
GPIO 2 (I2C1_SDA) 3 4 5V
GPIO 3 (I2C1_SCL) 5 6 GND
GPIO 4 (GPCLK0) 7 8 GPIO 14 (UART_TXD)
GND 9 10 GPIO 15 (UART_RXD)
GPIO 17 11 12 GPIO 18
GPIO 27 13 14 GND
GPIO 22 15 16 GPIO 23
3.3V 17 18 GPIO 24
GPIO 10 (SPI_MOSI) 19 20 GND
GPIO 9 (SPI_MISO) 21 22 GPIO 25
GPIO 11 (SPI_SCLK) 23 24 GPIO 8 (SPI_CE0)
GND 25 26 GPIO 7 (SPI_CE1)
ID_SD 27 28 ID_SC
GPIO 5 29 30 GND
GPIO 6 31 32 GPIO 12
GPIO 13 33 34 GND
GPIO 19 35 36 GPIO 16
GPIO 26 37 37 GPIO 20
GND 39 40 GPIO 21
Key
Power (5 Volts)
Power (3.3 Volts)
Ground
General Inputs/Outputs
I2C Interface
SPI Interface
UART Interface
ID EEPROM Interface

Pi Camera Specification

Size	25 x 20 x 9 mm
Weight	3 g
Still resolution	5 Megapixels
Video modes	1080p30, 720p60 and 640x480p60/90
Linux integration	V4L2 driver available
C programming API	OpenMAX IL and others available
Sensor	OmniVision OV5647
Sensor resolution	2592 x 1944 pixels
Sensor image area	3.76 x 2.74 mm
Pixel size	1.4 µm x 1.4 µm
Optical size	1/4"
Full-frame SLR lens equivalent	35 mm
S/N ratio	36 dB
Dynamic range	67 dB @ 8x gain
Densitivity	680 mV/lux-sec
Dark current	16 mV/sec @ 60 C
Well capacity	4.3 Ke-

Pi Camera Specification

Fixed Focus	1 m to infinity
Focal length	3.60 mm +/- 0.01
Horizontal field of view	53.50 +/- 0.13 degrees
Vertical field of view	41.41 +/- 0.11 degress
Focal ratio (F-Stop)	2.9

Pi Camera Hardware Features

Available	Implemented
Chief Ray Angle Correction	Yes
Global and rolling shutter	Rolling shutter
Automatic exposure control (AEC)	No - done by ISP instead
Automatic white balance (AWB)	No - done by ISP instead
Automatic black level calibration (ABLC)	No - done by ISP instead
Automatic 50/60 Hz luminance detection	No - done by ISP instead
Frame rate up to 120 fps	max 90fps. Limitations on frame size for the higher frame rates (VGA only for above 47fps)
AEC/AGC 16-zone size/position/weight control	No - done by ISP instead
Mirror and flip	Yes
Cropping	No - done by ISP instead (except 1080p mode)
Lens correction	No - done by ISP instead
Defective pixel canceling	No - done by ISP instead
10-bit RAW RGB data	Yes , format conversions available via GPU
Support for LED and flash strobe mode	LED flash
Support for internal and external frame synchronization for frame exposure mode	No

Pi Camera Software Features

Picture formats	JPEG (accelerated) , JPEG + RAW , GIF , BMP , PNG , YUV420 , RGB888
Video formats	raw h.264 (accelerated)
Effects	negative , solarise , posterize , whiteboard , blackboard , sketch , denoise , emboss , oilpaint , hatch , gpen , pastel , watercolour, film , blur , saturation
Exposure modes	auto , night , nightpreview , backlight , spotlight , sports , snow , beach , verylong , fixedfps , antishake , fireworks
Metering modes	average, spot, backlit, matrix
Automatic White Balance modes	off, auto , sun , cloud, shade, tungsten, fluorescent , incandescent , flash, horizon
Triggers	Keypress , UNIX signal , timeout
Extra modes	demo , burst/timelapse , circular buffer , video with motion vectors , segmented video , live preview on 3D models

Display unit

- 7 inch **NTSC TFT-LCD Color Monitor**
- Technical specifications of the display used is as follows:
  - ✓ Power with 9-12VDC only
  - ✓ 320mA power draw at 12V, 450mA at 9VDC
  - ✓ Visible display dimensions: 154.5mm x 86.8mm
  - ✓ Selectable 16:9 or 4:3 ratio via menu
  - ✓ Resolution: 480 x RGB x 234
  - ✓ Brightness: 250cd/m2
  - ✓ Contrast: 400:1
  - ✓ Display plastic case dimensions: 172mm x 114.28mm
  - ✓ Weight (excluding power cable): 383.2g

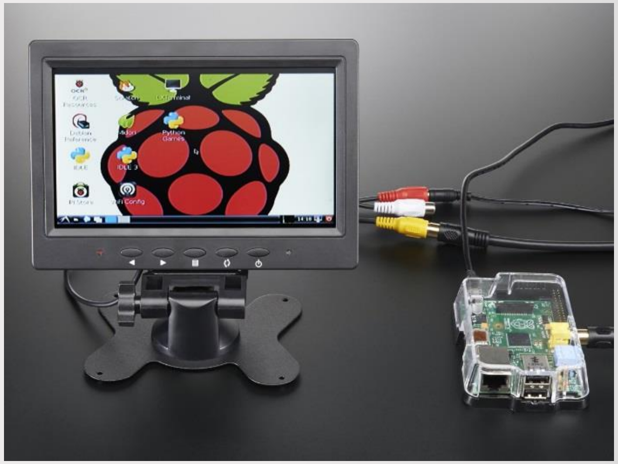


Figure: NTSC TFT-LCD 7´´ display connected to a Raspberry Pi.

Result analysis

Confusion matrix

	A	B	C	D	E	F	G	H	I	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
A	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	73	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E	0	25	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
F	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
K	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0
M	0	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0
O	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0	0	0	0	32	0	0	0	0	0	0	0	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	98	2	0	0	0	0	0
T	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	100	0	0	0	0	0
U	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	82	0	0	0	0
V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0
X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	80	0
Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	25

Advantages and Limitations

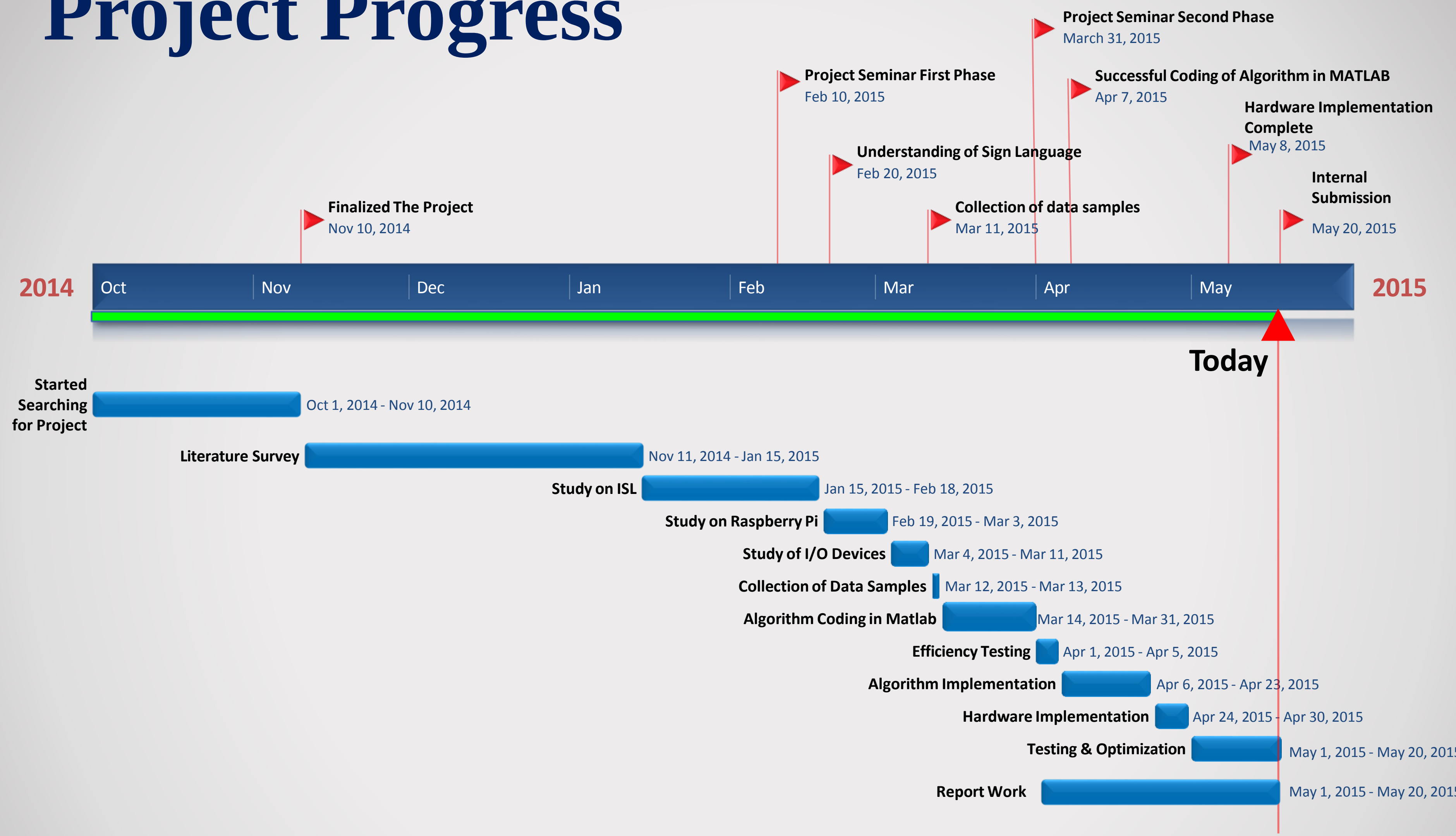
- Advantages:
  - Effective translation of Sign languages.
  - This system helps to mediate between people using other sign languages

- Disadvantages:
  - Cost is high.
  - Time delay.
  - To drive TFT-LCD display interfaced with Raspberry Pi 2 needs 5V and 3A.

Future enhancement

- More features can be extracted for signs ‘M’, ‘N’ and “S’ for better efficiency.
- We can implement translation for other sign languages.
- Mobile application can be developed by using this approach.
- **Server-Client Model** can be used instead of processing on Raspberry Pi 2.

Project Progress



References

1) Ghotkar, A. S., Khatal, R., Khupase, S., Asati, S. and Hadap, M. “Hand Gesture Recognition for Indian Sign Language”, International Conference on Computer Communication and Informatics (ICCCI), Jan. 10-12, 2012, Coimbatore, India.
2) Starner, T. and Pentland, A. “Real-Time American Sign Language Recognition from VideoUsing Hidden Markov Models”, IEEE, 1995, pp. 265-270.Singha, J. and Das, K. “Hand Gesture Recognition Based on Karhunen-Loeve Transform”, Mobile and Embedded Technology International Conference (MECON), January 17-18, 2013, India, pp. 365-371.
3) Singha, J. and Das, K. “Indian Sign Language Recognition Using Eigen Value Weighted Euclidean Distance Based Classification Technique”, International Journal of Advanced Computer Science and Applications, Vol. 4, No. 2, 2013, pp. 188-195.
4) Recognition of Indian Sign Language in Live Video by Joyeeta Singha and Karen Das in International Journal of Computer Applications (0975 – 8887),Volume 70– No.19, May 2013
5) Intelligent Sign Language Recognition Using Image Processing Sawant Pramada1, Deshpande Saylee , Nale Pranita, Nerkar Samiksha, Mrs.Archana S. Vaidya in IOSR Journal of Engineering (IOSRJEN) e-ISSN: 2250-3021, p-ISSN: 2278-8719 Vol. 3, Issue 2 (Feb. 2013), ||V2|| PP 45-51.

THANK YOU