

**Islamic University of Technology (IUT)  
Organization of Islamic Cooperation (OIC)**

**COURSE NAME: Digital Signal Processing Lab**

**PROJECT NAME: Speed Detection of a Dynamic Object**

**Prepared by:**

Md. Siraz-uz-Zaman 152450

Syed Tousiful Haque 152458

Mirza Abid Mahmud 152474

Contents

1. **Objective-----------------------------------------------------------03**

**2. Introduction----------------------------------------------------- 03**

**3. Description------------------------------------------------------- 03**

**3.1 Training of the system--------------------------------------03**

**3.1.1 Image Labeler -------------------------------------------05**

**3.1.2 train.cascadeobjectdetector----------------------------05**

**4.2 Object Detection---------------------------------------------07**

**4.2.1 vision.CascadeObjectDetection----------------------07**

**4.2.2 Working procedure of**

**vision.CascadeObjectDetection----------------------08**

**5.Speed Detection-----------------------------------------------10**

**6.Code------------------------------------------------------------10**

**7.Result-----------------------------------------------------------11**

**8.Problems Faced------------------------------------------------ 12**

**1.Objective:**

The system will detect a object in a video & measure it’s speed .  
Example: If a ball is travelling in a certain video ,the system will detect the ball and will measure it’s speed.

**2.Introduction:**

We have to design our system in such a way that when the object starts moving, the object will be detected in it’s first instance and will be differentiated from all other objects . Then the speed of the object is measured . For detecting the object we have to training the system with images of the object with various position of the object & also with the images that doesn’t contain the object.

**3. Description:**

The project is divided into two parts.

* + Training of the system
  + Object detection

**3.1. Training of the system:**

This section is the building block of the system .The system is trained to the object in it’s motion by the positive and negative image of the object.

**Positive Image:** Positive image is the picture of the object . It is a set of image .The positive images contain various position of the of the object . This is because the object can be in various position & in various alignment in a video . So to detect an image in video properly the system has to trained with every possible alignment. A example of positive image is given below:

**Negative Image:** Negative image is the image of the background or the image that doesn’t contain the object . The background is also crucial for the training of the system. The system processes to it’s next step of detection after taking the decision that portion of the image is negative or positive. So to increase the efficiency of the system the negative image is also provided.

To detect the real image part of the positive samples we used the  **Image Labeler** from the App portion of matlab.

**3.1.1 Image Labeler:**

The **Image Labeler** app enables us to label ground truth data in a collection of images. Using the app, we can:

* Define rectangular regions of interest (ROI) labels, pixel ROI labels, and scene labels, and use these labels to interactively label the ground truth data.
* Use built-in detection or tracking algorithms to label your ground truth data.
* Write, import, and use custom automation algorithm to automatically label ground truth .

After labeling the images with the **Image Labeler** ,we then start our training session with **trainCascadeObjectDetector**  function .

**3.1.2: trainCascadeobjectDetector:**

The  **trainCascadeObjectDetector** function syntax is

trainCascadeObjectDetector([outputXMLFilename](https://www.mathworks.com/help/vision/ref/traincascadeobjectdetector.html?s_tid=srchtitle#btrjwkt-1-outputXMLFilename),[positiveInstances](https://www.mathworks.com/help/vision/ref/traincascadeobjectdetector.html?s_tid=srchtitle#btrjwkt-1-positiveInstances),[negativeImages](https://www.mathworks.com/help/vision/ref/traincascadeobjectdetector.html?s_tid=srchtitle#btrjwkt-1-negativeImages))

Here the output XML file is the file that will be generated from the training session . The positive instance & negative image are the character vector which will contain the directory of the file in which the negative & positive images are stored .

The code section of the training is given below:

pos\_ins= pos;

pos\_dir= fullfile('D:\matlab\files\DSP Assignments\tousif\positive');

addpath(pos\_dir);

neg\_dir = fullfile('D:\matlab\files\DSP Assignments\tousif\negetive');

trainCascadeObjectDetector('trained\_model.xml',pos\_ins,neg\_dir,...

'NumCascadeStages',35,'FeatureType','Haar');

As shown in the code section the feature type of the training detector is selected as **HAAR .**  Actually there are three types of feature for the **traincascadeobjectdetector .** They are:

* Haar
* Local Binary Patterns (LBP)
* Histograms of Oriented Gradients (HOG)

Among this three types of feature Haar & Local Binary Patterns (LBP) is used to detect faces because they work well for representing fine-scale textures. On the other hand HOG can detect the overall structure of an object. To save time   LBP or HOG features can be used as they work on a small subset of data. Training a detector using Haar features takes much longer. After that, the Haar features can be run to see if the accuracy improves.

**4.2 Object Detection:**

In this section the system detects the moving object from a video. The video of the moving object is taken as a input & the object detection is done with **vision.CascadeObjectDetector** built in function.

**4.2.1 vision.CascadeObjectDetection:**

The **vision.CascadeObjectDetection** System object comes with several pretrained classifiers for detecting frontal faces, nose ,eyes stop sign & car etc. The Computer Vision Toolbox cascade object detector can detect object categories whose aspect ratio does not vary significantly. Objects whose aspect ratio remains fixed include faces, stop signs, and cars viewed from one side.

The **vision.CascadeObjectDetection**   System object detects objects in images by sliding a window over the image. The detector then uses a cascade classifier to decide whether the window contains the object of interest. The size of the window varies to detect objects at different scales, but its aspect ratio remains fixed. The detector is very sensitive to out-of-plane rotation, because the aspect ratio changes for most 3-D objects. Thus, every orientation of the object has to be trained . Training a single detector to handle all orientations will not work.

**4.2.2 Working procedure of vision.CascadeObjectDetection:**

The vision.cascadeobjectdetector works on the positive & negative samples those were generated after the Image Labelling .After Image Labelling only positive samples are produced. The negative samples are not explicitly given ,the train.cascadeobjectdetector generates negative samples from the  user-supplied negative images that do not contain objects of interest. The function detects the object in user defined stages . Before training each new stage, the function runs the detector consisting of the stages already trained on the negative images. Any objects detected from these image are false positives, which are used as negative samples. In this way, each new stage of the cascade is trained to correct mistakes made by previous stages. So the more the stages the less the false positive rate. As more stages are added, the detector's overall false positive rate decreases, causing generation of negative samples to be more difficult. For this reason, it is helpful to supply as many negative images as possible. There is a trade-off between fewer stages with a lower false positive rate per stage or more stages with a higher false positive rate per stage. Stages with a lower false positive rate are more complex because they contain a greater number of weak learners. Stages with a higher false positive rate contain fewer weak learners. Generally, it is better to have a greater number of simple stages because at each stage the overall false positive rate decreases exponentially. Now if a negative sample is found the classification of that region is complete and the window slides on to the next stage , if the sample is positive the window inform that the object is found . The final window

classifies the region as positive. To work well the each step in the cascade needs to have a low false negative rate because if a sample is falsely classified as negative then the process stops & the mistake can’t be undone . On the other hand if the false positive rate is high it is not a problem at all because if a sample is falsely classified as positive that mistake can be corrected in the next stage.

Here is the flow chart of the works process of thevision.CascadeObjectDetector:











**5.Speed Detection:**

Speed detection is a easy process .

-- First we count how can pixel the object takes in the frame. Like if an object is a 4cm & takes 100 pixels . So the length is 4/100 cm per pixel.  
--Now we will observe in how many frame the movement happens & calculate the pixel in every second by calculating the pixels in 25 frame . As our video was shot in 25 FPS.

---Multiplying this pixels in 25 frame with length per pixel with give us the speed of the object

**6.Code:**

vid=VideoReader('test3.mp4');

detector=vision.CascadeObjectDetector('trained\_model.xml');

count=0;

i=420;

while hasFrame(vid)

vf=readFrame(vid);

count=count+1;

if count<=145

box=step(detector,vf);

if count>=92&&count<=147

box=[i 151 95 75];

i=i-7;

end

if isempty(box)

count;

end

if count==145

pos2=box;

end

if count==95

pos1=box;

end

detectedImg=insertObjectAnnotation(vf,'rectangle',box,'Sharpner');

imshow(detectedImg);

% pause(1);

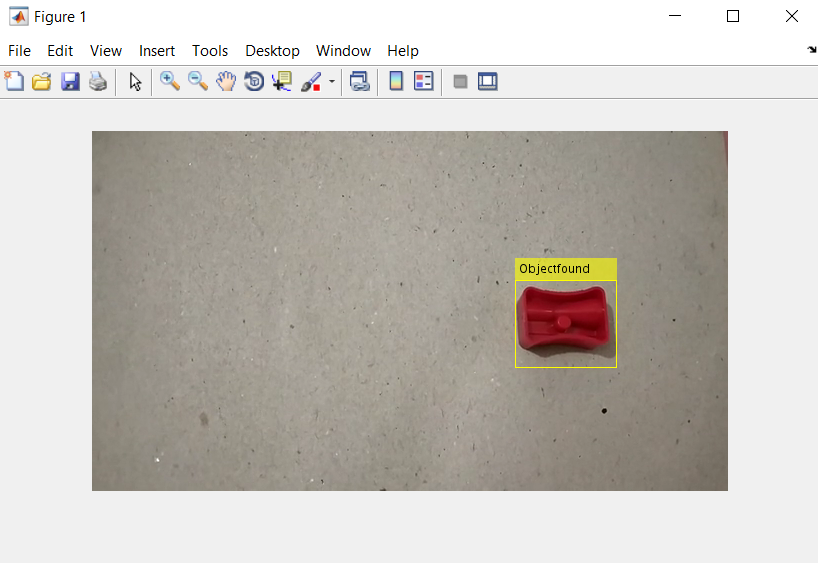
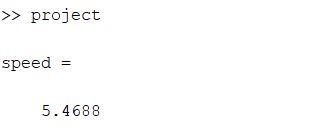
end

end

speed=(pos1(1)-pos2(1))\*25/50\*2.5/95

%%25fps

**7.Result:**

The object is found & the speed of the object will be shown after the end of the video.  
  

**8.Problems Faced:** We faced some problem during doing this project. The object are not always differenciated properly . As we used HAAR feature it took a lot of time to complete the training session & the accuracy is also not up to the mark . Still we tried to bring the results of the project to satisfactory level .