INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, ALLAHABAD

(Deemed University)

(A Centre of Excellence in Information Technology Established by Ministry of HRD, Govt. of India)

Project Report

REAL TIME OBJECT SURVEILLANCE IN SPATIAL DOMAIN USING COMPUTER VISION

Project Supervisor:

Mr. Neetish Purohit

Submitted by:

Dushyant Singh (IEC2011019)

Nikhil Pipal (IEC2011020)

Ayush Dobhal (IEC2011081)

Students Declaration

We, hereby declare that the work presented in this project report entitled "**Real Time Object Surveillance in Spatial Domain using Computer Vision**" submitted towards compilation of the 6th semester project of B.Tech (Electronics and Comm. Engg) at the Indian Institute of Information Technology Allahabad, is an authentic record of our original work carried out under the guidance of Dr. Neetish Purohit. The project has been done in full compliance with the requirements and constraints of prescribed curriculum to the best of our knowledge.

Dushyant Singh (IEC2011019)
Nikhil Pipal (IEC2011020)
Ayush Dobhal (IEC2011081)

Certificate

This is to certify that the project work entitled “**Real Time Object Surveillance in Spatial Domain using Computer Vision**” has been carried out for the compilation of 6th semester project by B.Tech (Electronics and Comm. Engg.) students namely Dushyant Singh (IEC2011019), Nikhil Pipal (IEC2011020) and Ayush Dobhal (IEC2011081) at the Indian Institute of Information Technology Allahabad. The above declaration made by the candidates is correct to the best of my knowledge and belief.

Dr.Neetish Purohit
(Associate Professor)

Acknowledgement

Our project titled as "**Real Time Object Surveillance in Spatial Domain using Computer Vision**" would not have been completed if not for the help and encouragement that we got from our mentor Dr. Neetish Purohit. We would like to sincerely thank you for the support and the proper guidelines that we got regarding this project. This genre was completely new for us but with his intense knowledge and constant pushing us to work hard, we could finally complete the project with contention. We thoroughly gained knowledge through the course of this semester. We are thereby submitting this report to you with full honesty and integrity.

Abstract

Real Time Object Surveillance in Spatial Domain using Computer Vision

Object detection and segmentation is the most important and challenging fundamental task of computer vision. It is a critical part in many applications such as image search, image auto-annotation and scene understanding. However it is still an open problem due to the complexity of object classes and images.

The easiest way to detect and segment an object from an image is the color based methods. The colors in the object and the background should have a significant color difference in order to segment objects successfully using color based methods. The objective of this project is to design a hybrid intelligent surveillance system. Due to its programming flexibility, easy availability and user friendly, MATLAB has been used to implement object tracking. A tracking system is uttered to be stable if it is robust and accurate. Robust tracking is e.g. when the system is resistant against noise, or partly occlusion, while the accuracy deals with how well the original object is tracked throughout the sequence.

Literature Survey

Objects are detected in every frame by an object detector. These objects are represented by points, and the association of detected points with the detections in the previous frame is based on the previous object state which can include object position and motion. The problem of association of points is called the point correspondence problem. Deterministic methods for point correspondence define a cost of associating each object in frame $t - 1$ to a single object in frame t using a set of motion constraints. Minimization of the correspondence cost is formulated as a combinatorial optimization problem. Given a cost matrix algorithms like Hungarian algorithm [5] exist to get an optimal correspondence.

Statistical methods model the object properties such as position, velocity etc. as states with some probability. The pdf's of these states are estimated using measurements taken in each frame, which are the object detections obtained by a detection mechanism and by choosing an appropriate motion model to describe the motion of object. Methods like Kalman filtering and particle filtering [1] are used to estimate the state of the object at each time instant, using these measurements.

More recently a different approach for tracking of multiple objects through extended occlusions has emerged. It involves using short-term trackers to track objects until they get occluded or the tracker accuracy drops below a threshold [2]. This short term track is called a tracklet. In the end all these tracklets are linked with each other minimizing a cost function associated with the linking of different tracklets with each other. This is a normal correspondence problem, which can be solved by defining a cost matrix and using Hungarian Algorithm. [3,4] extend the approach used by [2] by allowing a single tracklet to contain multiple objects. They do this by producing a hypothesis of two tracklets which are close in time and space, merging into one tracklet. Also, the cost of linking two tracklets, depends upon the similarity of the appearances of the object at the end of the first tracklet and the start of the second tracklet. If dissimilarity is high then the two tracklets would be perceived as two different objects. In real time videos with unrestricted illumination and pose changes of objects, this possibility is very high.

Kernel is refers to an object region of primitive shape. Tracking is done by computing the motion of this region, which can be translational, affine or projective, from one frame to the next. These algorithms differ by the appearance representation used and the method used to estimate the object motion. Templates and density-based appearance models (histograms) have been widely used because of their simplicity and low computational cost. Comaniciu and Meer [6] use a weighted histogram computed from a circular region to represent the object. Instead of performing a brute force search for locating the object, they use the mean-shift procedure. Extending optical flow methods to compute the translation of a rectangular region inn 1994, Shi and Tomasi [7] proposed the KLT tracker which iteratively computes the translation of a region centered on an interest point. Tao et al. [8] propose an object tracking method for multiple objects based on modelling the whole image as a set of layers. This includes a single background layer and one layer for each object. Each layer consists of shape parameters, motion parameters and appearance parameters, which are continuously, estimated using each other, in an EM algorithm. Mittal and Larry [9] propose a multiple person tracker by matching segmentations of the persons, across multiple views.

Problem Statement and Motivation

Object tracking in real time is one of the most important topics in the field of computer Vision. Detection and tracking of moving objects in the video scenes is the first relevant step in the information extraction in many computer vision applications. This idea can be used for the surveillance purpose, video annotation, traffic monitoring, human-computer interaction, intelligent transportation, and robotics and also in the field of medical. Object tracking can be defined as the process of segmenting an object of interest from a video scene and keeping track of its motion, orientation, occlusion etc. in order to extract useful information. Real-time object detection and tracking is a critical tasks in many computer vision applications such as surveillance, driver assistance, gesture recognition and man machine interface. Here, object tracking is done using MATLAB TOOL interfaced with Arduino to focus the camera on object using stepper motor.

Introduction

The goal of the object tracking is to track the position of objects in real-time. It is a critical part in many applications such as image search, image auto-annotation and scene understanding. However it is still an open problem due to the complexity of object classes and images. Unlike a multimedia system, the output of a surveillance system is not always an image
sequence, but could also be the positions of objects image content features and so on. Object tracking can be defined as the process of segmenting an object of interest from a video scene and keeping track of its motion, orientation, occlusion etc. in order to extract useful information. Visual tracking is as the name implies tracking of objects using a camera. Using a camera for object tracking changes the perception from world coordinates to camera coordinates. This means that in the projection from camera, the dimensions is reduced from 3D to 2D. The tracker is made to run in the image, on the 2D projected data of the 3D, and it is thus important have knowledge about the motion models of the object moving in the 3D environment.
The objective of video tracking is to associate target objects in consecutive video frames.
Typically this process involves four stages, namely,

• Object detection: Detecting and localizing a moving object in a video. Typically this is done through background subtraction to identify the moving objects.

• Object representation: Description of the object in a video frame. For example, an object can be represented by simply its centroid or by simple geometrical shapes like rectangle and ellipse etc. marked around it, or by either its contour or silhouette.

• Feature Selection: After choosing a suitable object representation, a set of features suitable for the application like edges, color histograms, optical flow etc. are extracted over the region of the object. Choice of features has always been governed by the underlying application domain.

• Tracking the features: The chosen features are finally tracked through different methods which can be broadly divided into point tracking, kernel tracking and silhouette tracking. The method of tracking depends on the features chosen.

Proposed Project Framework

The colors in the object and the background should have a significant color difference in order to segment objects successfully using color based methods. MATLAB usually captures images and videos in 8-bit, unsigned integer, BGR format. In other words, captured images can be considered as 3 matrices, BLUE, RED and GREEN with integer values ranges from 0 to 255.

Blue matrix

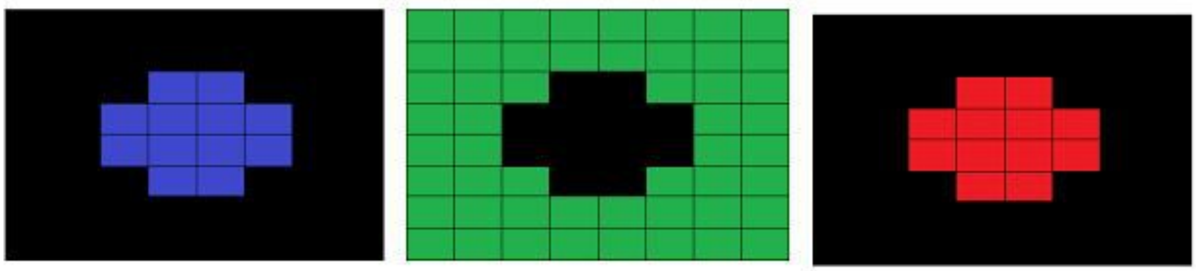
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	255	255	0	0	0
0	0	255	255	255	255	0	0
0	0	255	255	255	255	0	0
0	0	0	255	255	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Green matrix

255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255
255	255	255	0	0	255	255	255
255	255	0	0	0	0	255	255
255	255	0	0	0	0	255	255
255	255	255	0	0	255	255	255
255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255

Red matrix

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	255	255	0	0	0
0	0	255	255	255	255	0	0
0	0	255	255	255	255	0	0
0	0	0	255	255	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0



Resultant Image

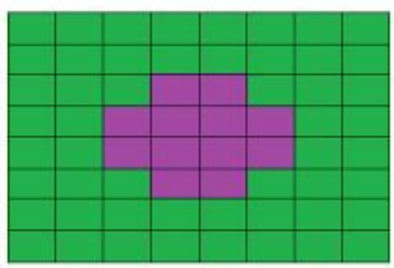


Figure 0 : BGR planes

Image segmentation is the process of dividing an image into multiple parts. This is typically used to identify objects or other relevant information in digital images

First, we connect to a Windows video device using the videoinput function

Now we start acquiring images.
By default, acquisition begins immediately. Since acquisition occurs in the background, the MATLAB command line is free and we can start processing images in the background. The processing does not need to start before the acquisition is done, so we'll use the wait function to wait for the acquisition to stop. The getdata function transfers the acquired images into the MATLAB workspace.

Now that we have the image sequence in MATLAB, we'll explore simple technique for tracking.

<u>Steps:</u>

- First acquire an RGB Frame from the Video.
- Extract the Red Layer Matrix from the RGB frame.
- Get the grey image of the RGB frame.
- Subtract the grayFrame from the redFrame.
- Filter out unwanted noises using Median Filter.
- Now convert the diffFrame into corresponding Binary Image using proper threshold value. Change its value for different light conditions. We have used its value as 0.15.

- Now you are all done. Your Red color has been detected. Now you can put any blob statistics analysis on this image. You can calculate the centroid, area or bounding box of those blobs.

- We get the centroid coordinates of the object.

- These coordinates are sent to Arduino board using serial communication.

- These coordinates are used to control the movement of stepper motor using Arduino board.

Block Diagram

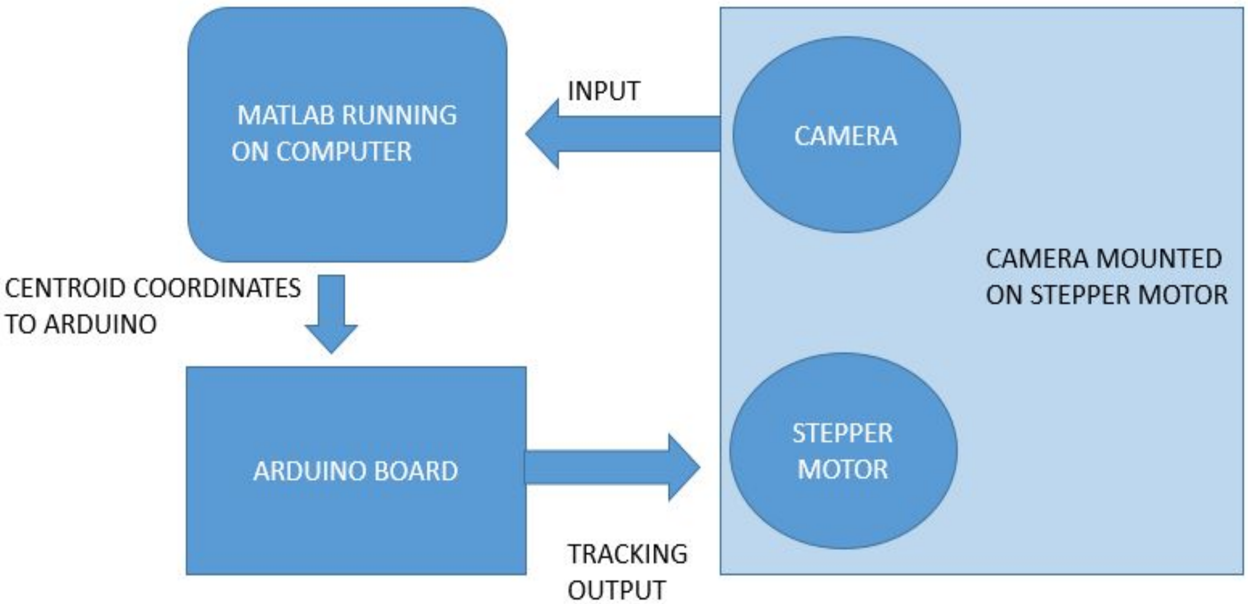
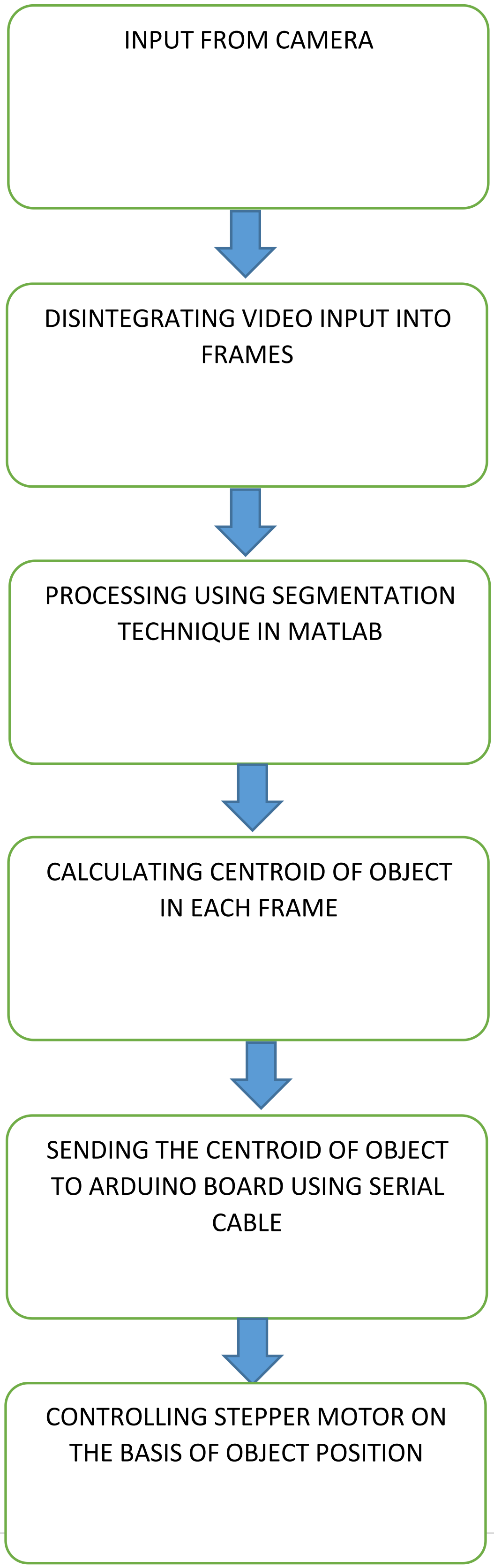


Figure 1: Block diagram

Process Flow Chart



Methods Used

- videoinput('winvideo', 2, 'YUY2_640x480') : Video input from camera.
- arduino=serial('COM9','BaudRate',9600) : Create serial communication object on port COM
- data = getsnapshot(vid) : Get a frame from camera feed.
- diff_im = imsubtract(data(:,:,1), rgb2gray(data)) : Detect red part in the image.
- diff_im = im2bw(diff_im,0.18) : Gray scale to binary conversion
- stats = regionprops(bw, 'BoundingBox', 'Centroid') : Draw a box around the centroid of the object

Software Specifications

MATLAB:

MATLAB (matrix laboratory) is a multi-paradigm numerical computing environment and fourth-generation programming language. Developed by MathWorks, MATLAB allows matrix manipulations, plotting of functionsand data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, and Fortran.

The MATLAB application is built around the MATLAB language, and most use of MATLAB involves typing MATLAB code into the Command Window (as an interactive mathematical shell), or executing text files containing MATLAB code, including scripts or functions. Most MATLAB functions can accept matrices and will apply themselves to each element. For example, mod(2*J,n) will multiply every element in "J" by 2, and then reduce each element modulo "n". MATLAB does include standard "for" and "while" loops, but (as in other similar applications such as R), using the vectorized notation often produces code that is faster to execute.

Hardware Specifications

Webcam:

The LifeCam is a webcam from Microsoft for home PC users available in various models. Its special features include an embedded microphone and integration with Windows Live Messenger, thus enabling direct Internet support.

GENERAL	
Compatible Software	Windows Live Messenger, Yahoo Messenger, Skype, and Microsoft Office Communicator
Focus Type	Fixed
Connectivity	USB
Has Tilt	No
Model Name	LifeCam VX-700
Category	Webcam
VIDEO AND IMAGE	
Is HD	No
Sensor Type	CMOS
Video Capture Resolution	640 x 480
Image Capture Resolution	640 x 480
SYSTEM REQUIREMENTS	
Memory	256 MB RAM
Processor	Intel Pentium III 550 MHz(Pentium 4 1.4 GHz Recommended)
Operating System	Windows: Vista, XP(SP2)

Figure 2: Web camera

Arduino Uno:
Arduino is a single-board microcontroller, intended to make the application of interactive objects or environments more accessible. The hardware consists of an open-source hardware board designed around an 8-bit Atmel AVRmicrocontroller, or a 32-bit Atmel ARM. Current models feature a USB interface, 6 analog input pins, as well as 14 digital I/O pins which allows the user to attach various extension boards.

Specifications

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

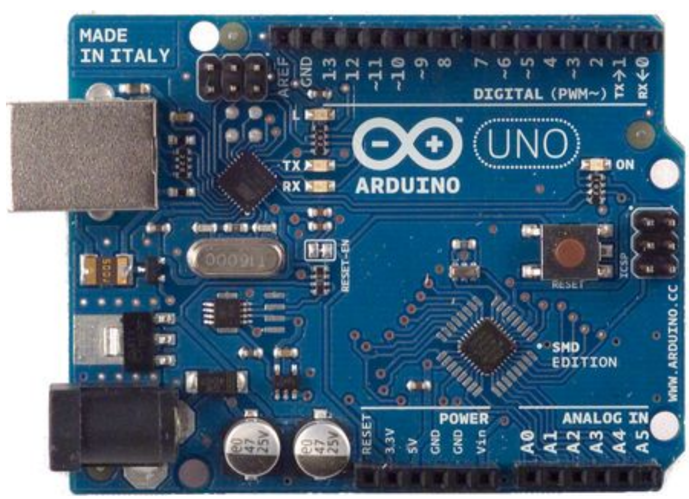


Figure 3: Arduino Board

L293D

A quadruple half H-bridge bidirectional motor driver IC that can drive current of up to 600mA with voltage range of 4.5 to 36 volts. They have separate bridge enable option and are suitable to drive small DC-Geared motors, bipolar stepper motor etc.

Specifications:

- Supply Voltage: 4.5V to 36V
- Output current capability per driver: 600mA
- Pulsed Current: 1.2A Per Driver
- Package: 16-pin PDIP

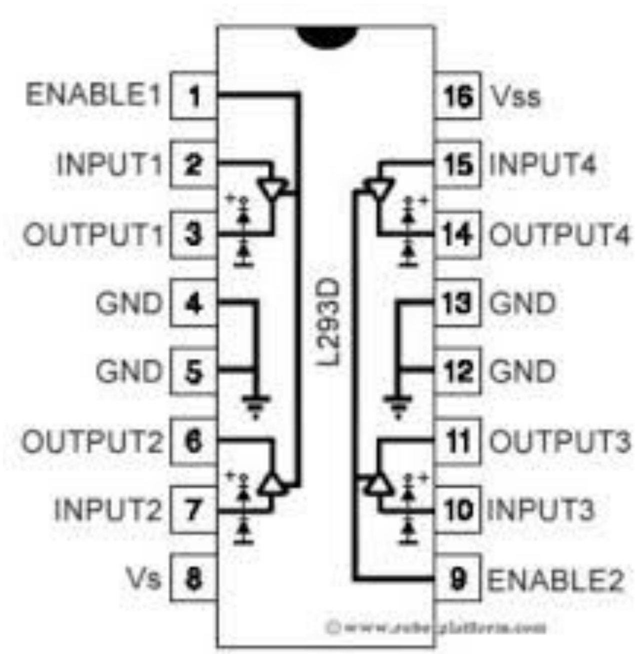


Figure 4: L293D Pin diagram

Stepper Motor

A stepper motor (step motor) is a brushless DC electric motor that divides a full rotation into a number of equal steps. The motor's position can then be commanded to move and hold at one of these steps without any feedback sensor (an open-loop controller), as long as the motor is carefully sized to the application.

Specifications

- Step Angle : 3.75 Degree
- Configuration: 4 wire bipolar stepper motor
- Holding Torque: 0.65 Kgcm at 0.6A per winding
- Phase current : 0.6A
- Resistance/phase: 5.5 ohm
- Inductance/Phase : 7.4 mH
- Length :27 mm
- Shaft Length : 9.1 mm
- Shaft Dia : 5mm
- Weight: 180 grams

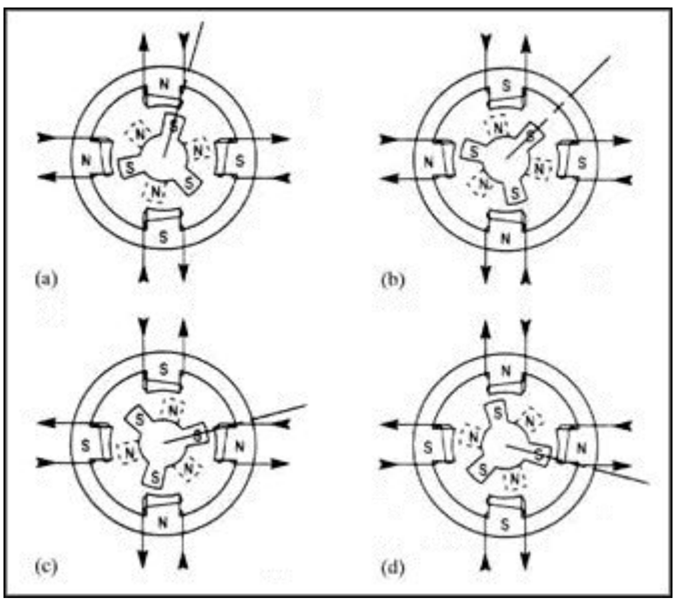


Figure 5: Working of stepper motor

Activity Time Chart

Weeks	Task	Status
Week 1(Jan 20-27)	Configuring,Installing and getting started with MATLAB	Completed
Week 2(Feb 2-9)	Interfacing Camera and learning MATLAB.	Completed
Week 3(Feb 11-18)	Taking Video input from interfaced camera and processing the video frame by frame.	Completed
Week 4(Mar 2-9)	Segmentating the video feed based on color property.	Completed
Week 5(Mar 23-30)	Working on Arduino Uno.	Completed
Week 6(April 3-10)	Interfacing stepper motor with Arduino Uno using L293D.	Completed
Week 7(April 11-18)	Connecting Arduino with MATLAB using serial cable.	Completed
Week 8(April 19-26)	Moving camera using stepper motor.	Completed

Output

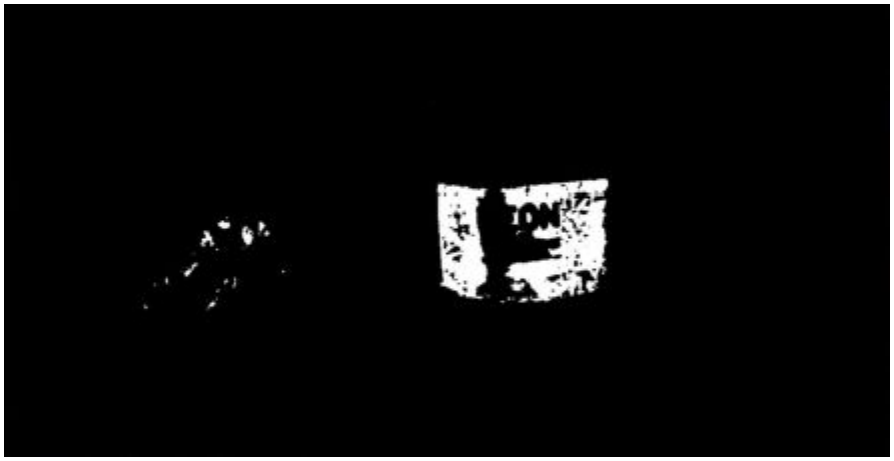


Figure 6: Thresholding red colored object

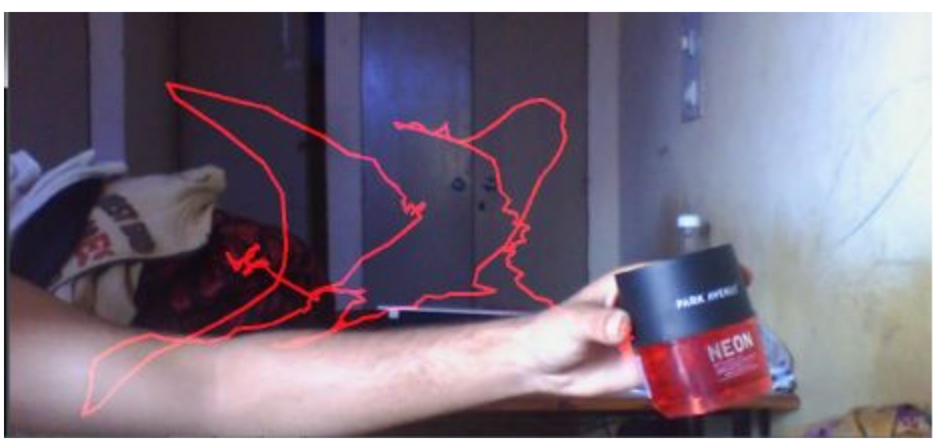


Figure 7: Tracking Red colored object

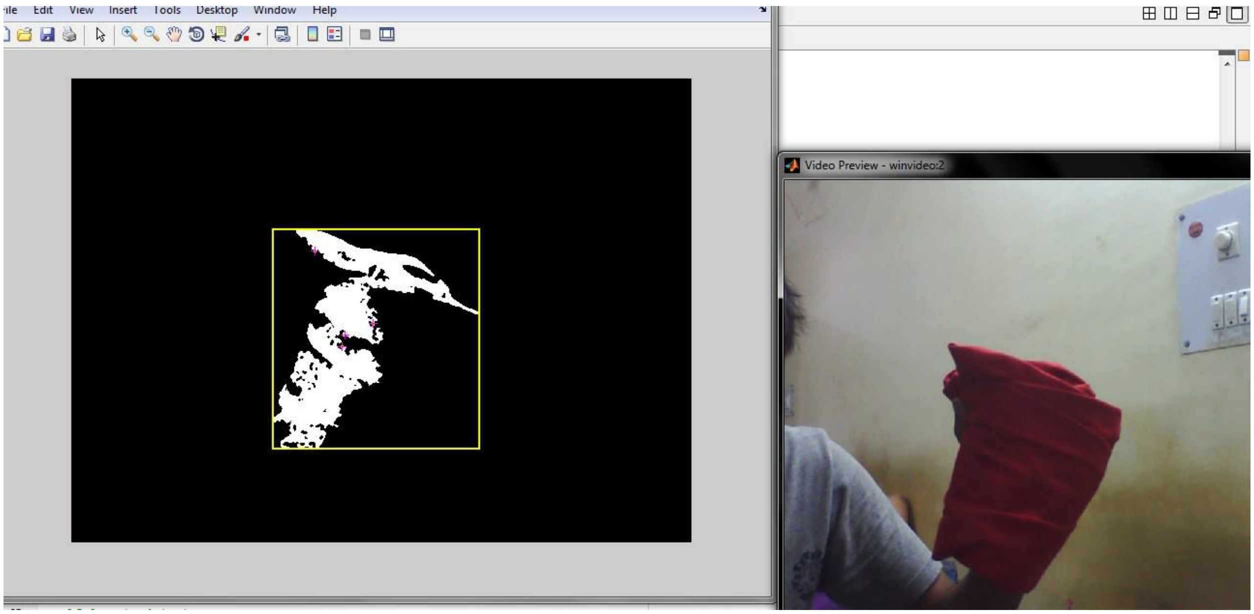


Figure 8: Segmenting red color

Future Aspects

More robust techniques can be used which are invariant to angular displacement and scalability and shape invariant. Detection and tracking can be used in various applications like

• Motion-based recognition, i.e. identifying humans based on gait, automatic object detection, etc.

• Automated surveillance, i.e. monitoring a scene to detect suspicious activities or unlikely events automatically, tracking a suspicious person over a long period of time automatically, reducing the amount of manual labor required in surveillance.

• Video indexing, which involves annotating the videos based on the type of the motion of the objects in it and retrieving the videos using a similar analysis on the videos.

• Human-computer interaction, that is, gesture recognition, eye gaze tracking for data input to computers, etc. This is used in systems like SixthSense [22] to provide a more intuitive interface to computing.

• Traffic monitoring, that is, real-time gathering of traffic statistics to direct traffic flow.

• Vehicle navigation, that is, video-based path planning and obstacle avoidance capabilities

References

1. Streit, R. L. and luginbuhl, T. E. Maximum likelihood method for probabilistic multi-hypothesis tracking. In Proceedings of the International Society for Optical Engineering vol. 2235. 394–405 1994.
2. R. Kaucic, A. G. A. Perera, G. Brooksby, J. Kaufhold, and A. Hoogs. A unified framework for tracking through occlusions and across sensor gaps. IEEE Conference on Computer Vision and Pattern Recognition 2005, pp. 990-997 .
3. A. Perera, C. Srinivas, A. Hoogs, G. Brooksby, and W. Hu. Multi-Object tracking through simultaneous long occlusions and Split-Merge conditions. IEEE Conference on Computer Vision and Pattern Recognition 2006, pp. 666-673.
4. João F. Henriques, Rui Caseiro and Jorge Batista. Globally Optimal Solution to Multi-Object Tracking with Merged Measurements. IEEE International Conference on Computer Vision 2011, pp. 2470-2477
5. Hungarian Algorithm, http://en.wikipedia.org/wiki/Hungarian_algorithm
6. Comaniciu, D. and Meer, P. Mean shift: A robust approach toward feature space analysis. IEEE Trans. Patt. Analy. Mach. Intell. 24, 5, pp. 603–619, 2002.
7. Shi, J. and Tomasi, C. Good features to track. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 593–600, 1994.
8. Tao, H., Sawhney, H., and Kumar, R. 2002. Object tracking with bayesian estimation of dynamic layer representations. IEEE Trans. Patt. Analy. Mach. Intell. 24, 1, pp. 75–89.
9. Mittal, A., Davis, L.S. : “M2 tracker: A multi-view approach to segmenting and tracking people in a cluttered scene”. International Journal of Computer Vision 51 (3) , pp. 189-203

Matlab Code

```
vid = videoinput('winvideo', 2, 'YUY2_640x480');
answer=1;
arduino=serial('COM9','BaudRate',9600);

set(vid, 'FramesPerTrigger', Inf);
set(vid, 'ReturnedColorspace', 'rgb')
vid.FrameGrabInterval = 1;

start(vid)

while(vid.FramesAcquired<=200)

  data = getsnapshot(vid);

  % Now to track red objects in real time
  % we have to subtract the red component
  % from the grayscale image to extract the red components in the image.
  diff_im = imsubtract(data(:,:,1), rgb2gray(data));
  %Use a median filter to filter out noise
  diff_im = medfilt2(diff_im, [3 3]);
  % Convert the resulting grayscale image into a binary image.
  diff_im = im2bw(diff_im,0.18);


  % Remove all those pixels less than 300px
  diff_im = bwareaopen(diff_im,300);
  imshow(diff_im)
  % Label all the connected components in the image.
  bw = bwlabel(diff_im, 8);
  % We get a set of properties for each labeled region.
  stats = regionprops(bw, 'BoundingBox', 'Centroid');

  % Display the image
  preview(vid)

  hold on
```

```
    %This is a loop to bound the red objects in a rectangular box.
    for object = 1:length(stats)
        bb = stats(object).BoundingBox;
        bc = stats(object).Centroid;
        rectangle('Position',bb,'EdgeColor','y','LineWidth',2)
        plot(bc(1),bc(2), '-m+')
        a=text(bc(1)+15,bc(2), strcat('X: ', num2str(round(bc(1))), ' Y: ',
num2str(round(bc(2)))));
        set(a, 'FontName', 'Arial', 'FontWeight', 'bold', 'FontSize', 12, 'Color', 'yellow');
    end

    num = round(bc(1));
    fprintf(arduino,'%s',char(answer));
    if num >= 300
        answer = 1;
    else if num <= 300
            answer = 2;
        end
    end
   hold off
end
% Both the loops end here.

stop(vid);
fclose(arduino);
flushdata(vid);

clear all
```

Arduino code

```
int ledPin=13;

int matlabData;


void setup()

{

             pinMode(ledPin,OUTPUT);

            Serial.begin(9600);

}

 void loop()

{


            matlabData=Serial.read(); // read data

            if(matlabData==1) {

      digitalWrite(ledPin,HIGH); // turn light on

    }

    else if(matlabData==2) {

      digitalWrite(ledPin,LOW); // turn light on

      delay(500);

      digitalWrite(ledPin,HIGH);

      delay(500);

    }

}
```

Suggestion Of Board Members