eYSIP2016

Automatic Theme Evaluation from Videos

Keshav Bihani
Raj Krishna Srivastava
Khalid Waseem
Duration of Internship: 21/05/2016 – 10/07/2016

2016, e-Yantra Publication

Automatic Theme Evaluation from Videos

Abstract

This project aims at automatically evaluating themes that are provided to teams during eYantra competition with help of image and audio processing, without the need for any manual intervention. Each year, there are many videos submitted before the finals of the competition and automating the evaluation process would help in increasing the number of participants as well as in speeding up the evaluation process. Moreover this project can find its application in surveillance with slight modifications. In the project we are tracking a red colored object (i.e robot) but we can program it to track any other colored objects(like vehicles,humans and so on).

Completion Status

Automatic Evaluation of puzzle solver has been achieved. Here is the brief idea of the theme.The Matlab code generates three log files(.txt files).

- The first one is the trace file generated after applying the mean shift algorithm to track the robot.
- Second file contains the the on and off time of the LEDs as well as the numbers that are picked and deposited.
- The third files contains the time of the buzzer beeps that are used to indicate picking up and deposition of blocks along with indicating the starting and ending of the run.

These three text files are read as input by a C program which then generates the score.The C code is independent of the source that generates the log file and takes as input the data corresponding to 1 run.

1.1 Software Used

- Matlab.
- Detail of software: Version R2012a.
- Installation steps

1.2 Software and Code

This is the Github link for the repository of code and research done during the intership.
TranformVideo.m contains the logic for converting the video into orthographic projection.This is done to bring, all the videos shot from different angles, to a same viewing angle.This help to ease the processing as the obtained image is rectangular and we can easily map to the original image of the arena. Basic logic is explained below- Homography relates any two images of the same planar surface in space (assuming a pinhole camera model).This has many applications, such as image rectification, computation of camera motion,rotation and translation,between two images. Once camera rotation and translation have been extracted from an estimated homography matrix H, this information may be used for navigation, or to insert models of 3D objects into an image or video, so that they are rendered with the correct perspective and appear to have been part of the original scene.

Two parallel lines on the same plane can never intersect in Euclidean space geometry. How ever in projective space geometry two parallel lines can intersect at horizon, eg: picture of a railroad, although the two rails are parallel but in picture you can see them intersecting at horizon.
Euclidean space(or Cartesian space) describes 2D/3D geometry very well but they are not sufficient to handle the projective space.Actually Euclidean geometry is subset of projective geometry.
Homogeneous Coordinates, are a way of representing N dimensional (Cartesian coordinates) with N+1 numbers in projective space. Homogeneous coordinates makes calculation of graphics and geometry possible in projective space. eg: To make 2D Homogeneous coordinates, we simply add an additional variable (ω), into existing coordinates. Therefore a point(X,Y) in Cartesian coordinate becomes(x,y,ω) in homogeneous coordinates.
such that: $X = x/\omega$ and $Y = y/\omega$ where X and Y Cartesian Coordinates.

Let $\begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$ be the Cartesian coordinate of P_1 and $\begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$ be the Cartesian

coordinate of P_2 and $\begin{bmatrix} u_1 \\ v_1 \\ \omega_1 \end{bmatrix}$ and $\begin{bmatrix} u_2 \\ v_2 \\ \omega_2 \end{bmatrix}$ be homogeneous coordinates of P_1 and P_2 respectively.

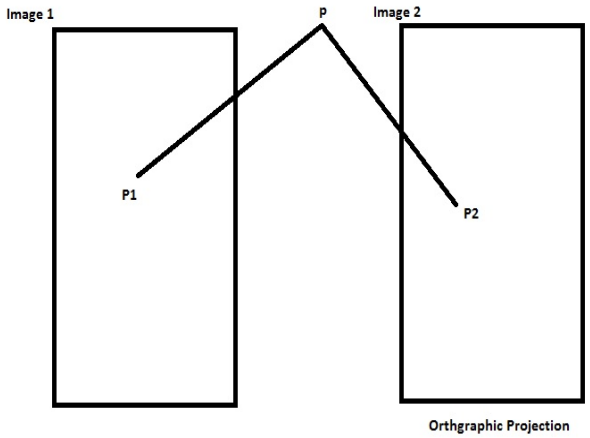


Figure 1.1: Sample

Then we have $x_1 = u_1/\omega_1$ and $y_1 = v_1/\omega_1$
Similarly $x_2 = u_2/\omega_1$ and $y_2 = v_2/\omega_1$.

Given two images of coplanar scene, the two points P_1 and P_2 in the image 1 and 2 are related by a transformation matrix(Homography matrix) H.

i.e. $P_2 = \begin{bmatrix} u_2 \\ v_2 \\ \omega_2 \end{bmatrix} = HP_1 = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \\ \omega_1 \end{bmatrix}$

Hence $x_2 = \dfrac{u_2}{\omega_2} = \dfrac{H_{11}u_1 + H_{12}v_1 + H_{13}\omega_1}{H_{31}u_1 + H_{32}v_1 + H_{33}\omega_1} = \dfrac{H_{11}x_1 + H_{12}y_1 + H_{13}}{H_{31}x_1 + H_{32}y_1 + H_{33}}$ – eqn(i)

Similarly $y_2 = \dfrac{H_{21}x_1 + H_{22}y_1 + H_{23}}{H_{31}x_1 + H_{32}y_1 + H_{33}}$ –eqn(ii)

Rearranging eqn(i) we get

$x_2x_1H_{31} + x_2y_1H_{32} + x_2H_{33} - x_1H_{11} - y_1H_{12} - H_{13} = 0$ –eqn(iii)

Similarly

$y_2x_1H_{31} + y_2y_1H_{32} + y_2H_{33} - x_1H_{21} - y_1H_{22} - H_{23} = 0$ –eqn(iv)

For 1 point P we obtained 2 equations. In homography matrix H there are 8 degrees of freedom($H_{11}, H_{12}, ..., H_{32}$) and H_{33} is always 1. Therefore we need atleast 4 points correspondence to solve the homography matrix. In

above eqn(iii) and (iv) $(H_{11}, H_{12}, ...H_{32})$ are unknown.Arranging the above equations in matrix form and using suitable number pf point correspondence (atleast 4) we get

$$\begin{bmatrix} -x_{1i} & -y_{1i} & -1 & 0 & 0 & 0 & x_{2i}x_{1i} & x_{2i}y_{1i} & x_{2i} \\ 0 & 0 & 0 & -x_{1i} & -y_{1i} & -1 & y_{2i}x_{1i} & y_{2i}y_{1i} & y_{2i} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} H_{11} \\ H_{12} \\ H_{13} \\ H_{21} \\ H_{22} \\ H_{23} \\ H_{31} \\ H_{32} \\ H_{33} \end{bmatrix} = 0$$

where

$$\begin{bmatrix} -x_{1i} & -y_{1i} & -1 & 0 & 0 & 0 & x_{2i}x_{1i} & x_{2i}y_{1i} & x_{2i} \\ 0 & 0 & 0 & -x_{1i} & -y_{1i} & -1 & y_{2i}x_{1i} & y_{2i}y_{1i} & y_{2i} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} = A \text{ and } \begin{bmatrix} H_{11} \\ H_{12} \\ H_{13} \\ H_{21} \\ H_{22} \\ H_{23} \\ H_{31} \\ H_{32} \\ H_{33} \end{bmatrix} = h$$

Therefore we get Ah=0 –eqn(v)
A = 2Nx9 and h=9x1 where N is point of correspondence (atleast 4.)

Eqn (v) can be solved using the SVD of A i.e. $A = USV^T$.Vector h is given by singular vector corresponding to smallest regular value.

Thereafter once video is converted to orthographic projection we will use logic of video tracking specified in the Mean_shift_with_led_and_path.m to generate trace file as well as log file containing on and off time of LEDs.
The project uses the mean shift algorithm to track objects. Mean shift is a technique to find the maxima of a density function. It is also called mode-seeking algorithm. Mean shift finds the target using iteration and find the target that is most similar to the given target. In this algorithm initially a region of interest is selected, then in consecutive iterations it looks in the region of interest and shifts the mean in the direction of increasing density until no more shift is possible, i.e, it has reached the peak of the Probability Distribution Function (densest area).It would be better understood from the following figure.

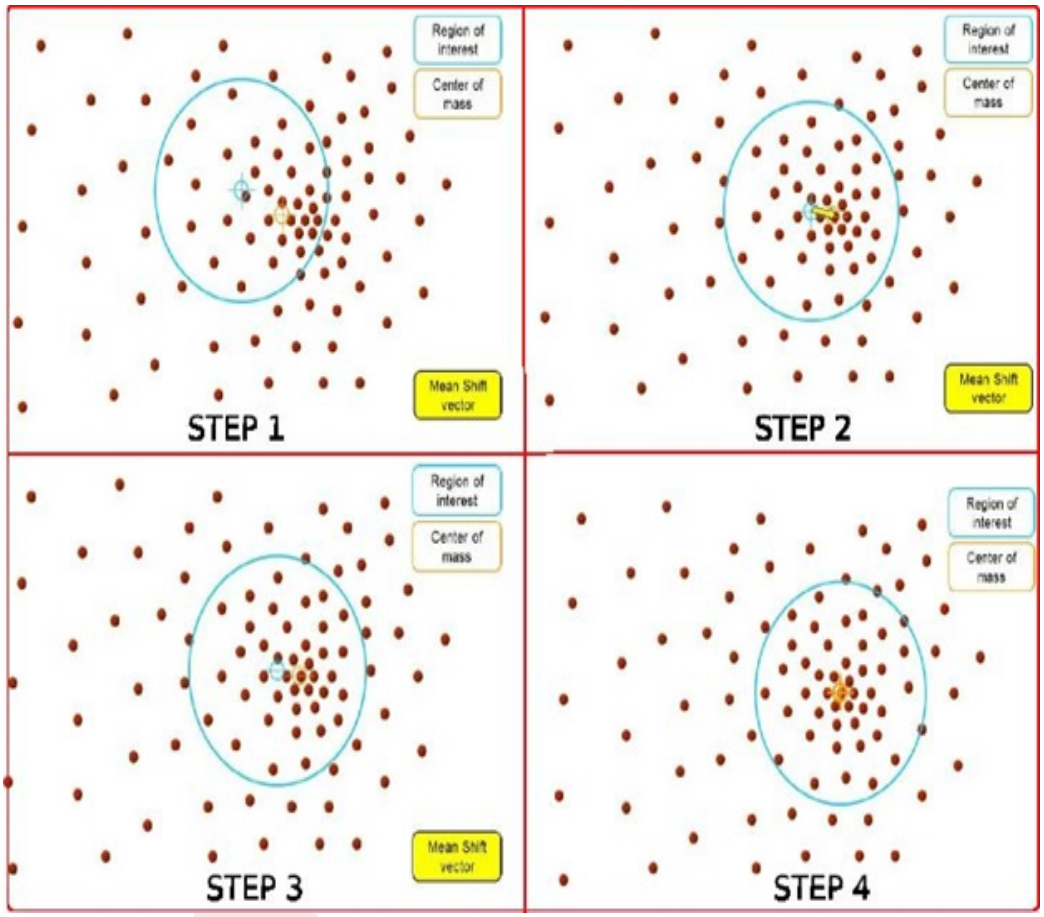


Figure 1.2: Illustration of Mean Shift.

Robot Tracking using Mean Shift For tracking an object using mean shift we specify the region in the frame that is our target patch. Then we use a feature space of the object to track that object. Feature space can be color, intensity, gradient, etc. We use a weighted histogram in our algorithm and represent the target patch. The target patch is converted in to m-dimensional vector by obtaining m-bin histogram of the image. This is the PDF of the target patch.

$$\vec{q} = \{q_u\}_{u=1,2,\ldots m}, \quad \text{such that, } \sum_{u=1}^{m} q_u = 1$$

Now PDF of the candidate patch, which is the initial estimate of the target, in the next frame is given by-

$$\overrightarrow{p(y)} = \{p_u(y)\}_{u=1,2,\ldots,m}, \text{ such that, } \sum_{u=1}^{m} p_u = 1$$

This PDF is centered at y. We have to find y such that the similarity between the two PDFs is maximum. Now to find the similarity between these two PDFs we use the Bhattacharyya Coefficient (ρ).

$$\rho(p(y),q) = \sum_{u=1}^{m} \sqrt{p_u(y) q_u}$$

So to find the new target location we need to maximize the Bhattacharyya coefficient.
Calculating q and p(y):
The probability of occurrence of a particular intensity value u is given by-

$$p(u) = \sum_{} (k||x_i||^2)\, \partial[S(x_i) - u]$$

where, $k\|x_i\|$- is a Gaussian kernel to provide weights based on spatial distance. u varies from 1 to m (In our code it varies from 0 to 255 as we took 256 bins, one for each pixel value).
$\delta[S(x_i) - u]$- Delta function is 1 when $S(x_i) = u$
$S(x_i)$-It is the pixel intensity at location x_i inside patch.

In simple words to calculate the probability density for a particular u , we check how many times intensity value u occurs inside the patch. Let u occur at location x_1, x_2, x_3,, x_4

So its probability distribution is given by $p(u) = w(x_1) + w(x_2) + w(x_3) + w(x_4)$ In the figure weights w(x) are derived from Gaussian Kernel at position x1 and so on. Gaussian kernel assigns the weights such that the pixels closer to the center have large weightage than the one that are far from the center. Lets come back to similarity of target and candidate PDFs.

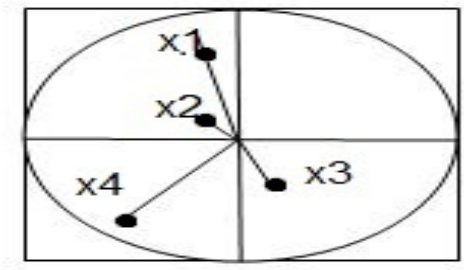


Figure 1.3: Sample Patch with distribution of a certain intensity value- 'hue'.

To maximize the Bhattacharyya Coefficient we firstly expand (y) using Taylor series.

$$\rho\left[\hat{\mathbf{p}}(\mathbf{y}),\hat{\mathbf{q}}\right] \approx \frac{1}{2}\sum_{u=1}^{m}\sqrt{\hat{p}_u(\hat{\mathbf{y}}_0)\hat{q}_u} + \frac{1}{2}\sum_{u=1}^{m}\hat{p}_u(\mathbf{y})\sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{\mathbf{y}}_0)}}$$

To maximize the above term we have to maximize the second term of the Taylor expansion as the first term is independent of y, so it remains constant. It is assumed that the target candidate does not change drastically.
$p_u(y)$ is probability of occurrence of intensity value u, when center of the patch is located at y.
Substituting $p_u(y)$in the above eqn. we obtain-

$$\rho\left[\hat{\mathbf{p}}(\mathbf{y}),\hat{\mathbf{q}}\right] \approx \frac{1}{2}\sum_{u=1}^{m}\sqrt{\hat{p}_u(\hat{\mathbf{y}}_0)\hat{q}_u} + \frac{C_h}{2}\sum_{i=1}^{n_h} w_i k\left(\left\|\frac{\mathbf{y}-\mathbf{x}_i}{h}\right\|^2\right)$$

where, C_h- Constant
$b(x_i)$- Intensity at x_i
y- Kernel center
m- Number of bins (256) and

we have to maximize weights w_i, to maximize weights we will use Mean Shift Algorithm, which also gives us the shift of target.

$$w_i = \sum_{u=1}^{m} \delta\left[b(\mathbf{x}_i) - u\right] \sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{\mathbf{y}}_0)}}.$$

$$\hat{\mathbf{y}}_1 = \frac{\sum_{i=1}^{n_h} \mathbf{x}_i w_i g\left(\left\| \frac{\hat{\mathbf{y}}_0 - \mathbf{x}_i}{h} \right\|^2\right)}{\sum_{i=1}^{n_h} w_i g\left(\left\| \frac{\hat{\mathbf{y}}_0 - \mathbf{x}_i}{h} \right\|^2\right)}.$$

We repeat the process until the similarity between two PDFs is greater than a certain threshold. Once we obtain a similarity greater than out threshold, we have found our new target location.

The Mean Shift Algorithm for object tracking is explained in the flowchart given below.

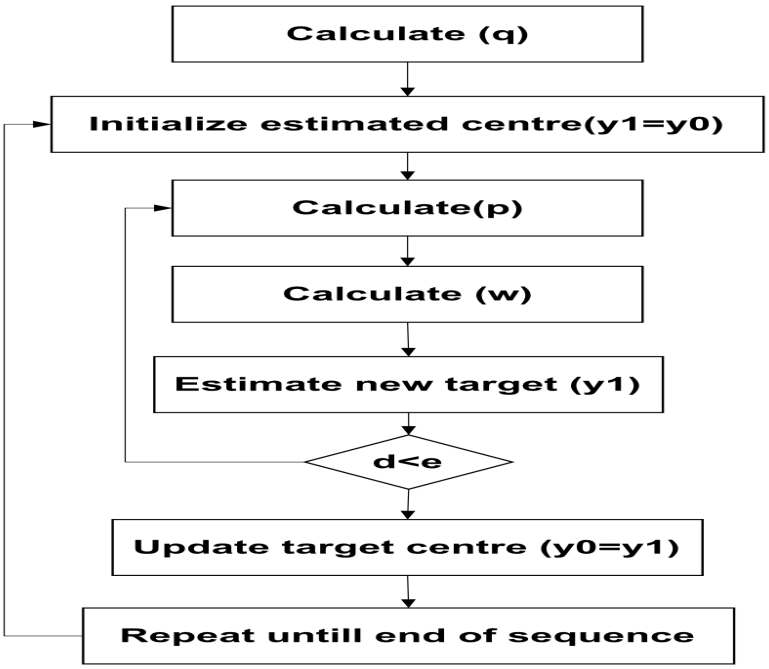


Figure 1.4: Flowchart of mean shift tracking.

Detecting Glowing of Led and Determining Pick and Drop Positions For detecting the glow of the LED we select a rectangular area around the robot and sum up number of blue pixels in each consecutive frame and subtract it with the one obtained in the previous frame. If the difference is above a certain threshold, it corresponds to LED on and a large negative value of difference below a certain threshold corresponds to LED being switched off. The number picked is then identified by mapping the position of LED glow to the original .pdf image of the arena.

The audio processing module ampli.m is independent of the above listed processes and generates text files containing on and off time of buzzer. We know the frequency of buzzer to near about 2.7kHz to 3kHz.We use DFT to detect this frequency and hence the buzzer beeps. Using band pass filter we filter out all the frequencies except those in this range and then output the time in a file. Choice of design of the filter and values of different parameters can be found here.
Using the log files as input,there is C code that generates the score according to the formulae-

$$\textbf{Total Score} = (600 - T) + (CD \times 10) + \sum_{i=0}^{n} \left[100 - error_i \, X \, 10 \right] + B - P$$

Where

- error = abs(Required no- Placed no)
  n can take value between 0 to 4, as there can be maximum of 4 numbers will be present in D2.
- T is the total time in seconds to complete the task.
- CD is the number of correct buzzer beeps.
- P is a penalty where 30 (thirty) points are deducted each time when LED is not turned ON or OFF correctly.
- B is a bonus of 100 points awarded, if no error is committed.

List of things tried during the course of project that failed-

Things tried	Reasons for failure.
Image masking using RGB	Fixed threshold value pose a problem when lightening conditions change
Image masking using HSV	There is no particular valve of Hue or Saturation for white.
Image masking using LAB	It is the measure of degree of variation from red to green and green to yellow, so no help to identify arena, but helped us choose marker color as red.
Corner Detection using Harris Corner	Does not always give us exact four corners.
Corner Detection using vector dot product	The edges that are masked are not smooth enough.
Corner Detection using polar co-ordiantes	Since many videos did not have all corners visible.
Path following by providing equations	All the equations needed to be provided manually
Path following by providing specific path by drawing path	There may be more than one correct path and drawing each is not possible
LED detection by Hue and RGB values	Since arena was white and glowing LED would not change the pixel intensity to the extent that it could be detected.
LED detection by difference in intensity values	Since different parts of arena had different lightening conditions, their intensity values changed without the LED glowing.
Audio Detection By Amplitude	Affected too much by noise.

1.3 Use and Demo

Final Setup Image.
User Instruction for demonstration

- The position of the arena should be as shown with all the corners visible.
- 4cm thick black chart paper must be pasted on the borders of the arena.

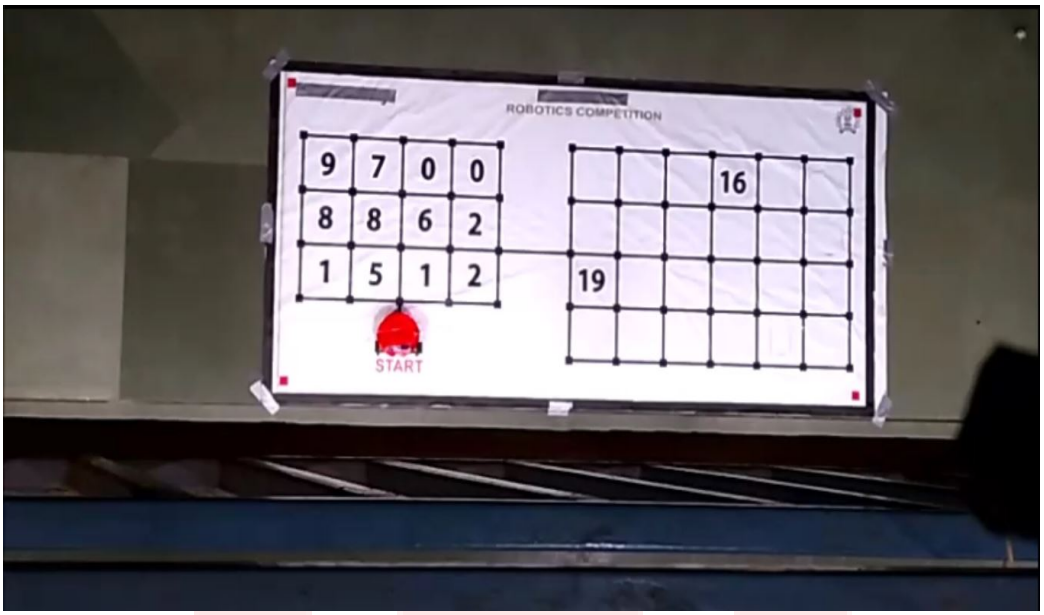


Figure 1.5: Testing Purpose

- Four 2cmx2cm red markers need to be pasted on four corners.
- Camera should be entirely stable without any movement and it should be above directly above the arena.
- Adequate lightening must be present and there should not be much variation in conditions during filming the video.
- The robot should be covered with red colored paper as shown in the image.

Demonstration Videos
Original video with theme implemented.
Part of the original video that is being processed.
Homographed Video of task.
Object tracking implemented on the video.

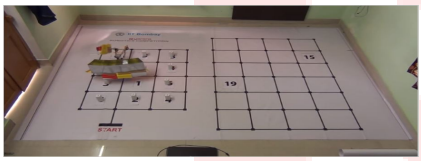
1.4 Future Work

- First of all we need to implement parallel processing in our code since the time for evaluation is humongous to be of any practical use. Parallel processing is useful when previous outputs do not affect the present inputs.

Here we know each frame is represented as matrix and to transform it to orthographic projection we perform operations on each point of the matrix.Now each point is independent to any other point in matrix and operations on them can be done parallely via GPU.

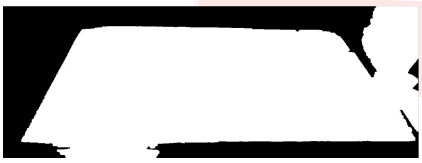
- Secondly relying just on image processing is not going to allow us achieve our goal of making this evaluation generic. We can incorporate machine learning as well.Machine learning helps computers to learn without being explicitly programmed. It focuses on the development of computer programs that can teach themselves to grow and change when exposed to new data.
  We can use concepts of ML to identify the arena, the robot and even to identify a perfect run(i.e by giving it samples of prefect task completion).

1.5 Bug report and Challenges



(a) Earlier

(b) Later



(a) Thresholding earlier image

(b) Thresholding later with same threshold values

- Change in lightening condition during videos pose an large risk of error generation.Errors such as incorrect detection of robot, improper detection of the area of the arena. These errors occur because we have taken RGB colorspace to threshold the image and detect various objects and these thresholding values are very sensitive to lightening conditions.

- The audio processing module isn't accurate. The manually set threshold works in about 70% of the cases.Rest of the time minor changes in thresholding needs to be done to get desired result.

- Another problem with the code is that the homography part isn't optimized. The pre-processing time is too much to be of any practical use,since video of about a minute and half takes nearly 20 minutes to convert.

Bibliography

[1] Dr. Niket Kaisare *Introduction to Matlab*.

[2] Dorin Comaniciu, Visvanathan Ramesh, Peter Meer *Research paper on mean shift*.

[3] Rashi Agrawal *Introduction to DIP using Matlab*.

[4] Dr. Mubarak Shah *UCF Computer Vision Video Lectures*, 2012.

[5] Mathworks Community *Audio Processing.*

[6] Wikipedia *Support Vector Machine.*