Pointing Based Object Localization

Group 6

Kaushik Subramanian

Robotics and Computer Vision, ECE 472

April 16th, 2009

Table of Contents

Goal

The goal of the project is to design a Finger Pointing-based interface that -

1. Estimates the 3D position of a target object based on the Hand-Eye Line of Sight vector,
2. Charts a course for the LEGO Robot to reach the target and
3. If computationally feasible, recognize the object.

The project involves concepts of Stereo Camera Calibration, Least Squares Estimation of the Fundamental Matrix, Depth Estimation, Face, Eye and Hand Feature Tracking and associated Computer Vision Algorithms.

Outline of the Approach -

1) Use Left Camera Image to Identify and locate the face of the person, position of the eye and pointing hand.

2) Obtain Line of Sight vector by drawing straight lines across the matching eye and finger tip.

3) If multiple positive points are detected, use Least Squares to resolve the projections.

4) Process the gradient along the LOS and locate target object. Choose the first object if line passes through many.

5) Use Fundamental Matrix to obtain the Corresponding point in the Right camera and Estimate the Depth to the point.

6) Chart a course for the robot to the destination.

1. Setup and Robot Design

We require two cameras for the purpose of Stereo Vision, a fully functional NXT robot, a random array of target objects and a powerful algorithm.

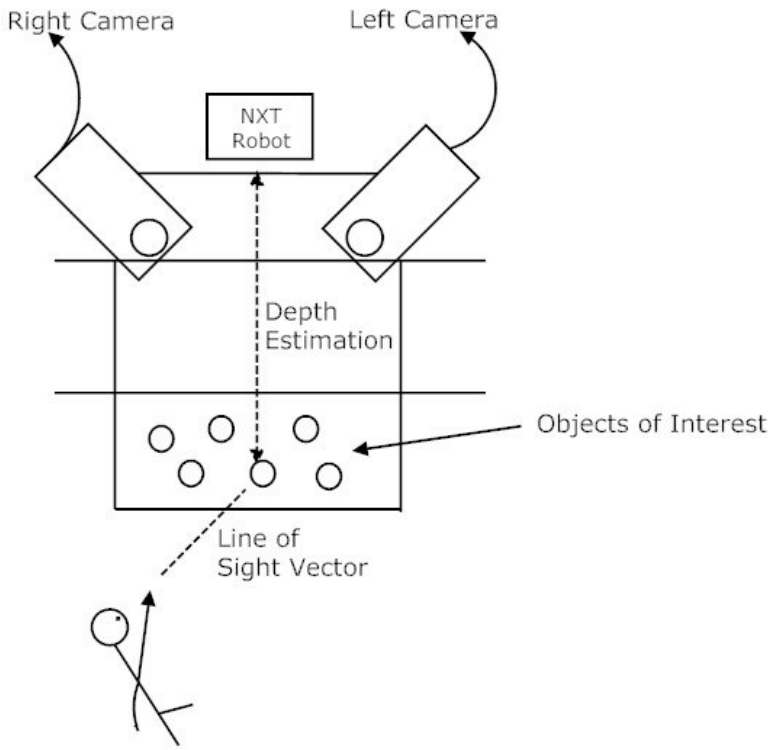


Figure 1: Experimental Setup of the Project with all the necessary components

As shown in Figure 1, the two Logitech Fusion Web Cameras (1.3 MP) are placed on an elevated surface, a distance *T* apart from one another and an angle θ to the horizontal. A certain distance away from the Camera mounts, the target objects are placed in a random order (within the Cameras FOV). The position of the cameras must be such that a complete view of the volunteer as well as the targets is ensured.

We require a portable computer that is compatible with OpenCV, a recent version of Matlab and has USB/Bluetooth Capability. The communication protocol will be taken care of using the RWTH NXT Matlab Toolbox. Using that we can connect to the robot using Bluetooth.

The Robot Construction is a simple one, taking the form of car. The motors are controlled carefully to facilitate the forward motion so that it can move to different locations as computed by the Depth Estimation Function. A picture of the robot is given below Figure 2.

The Agent

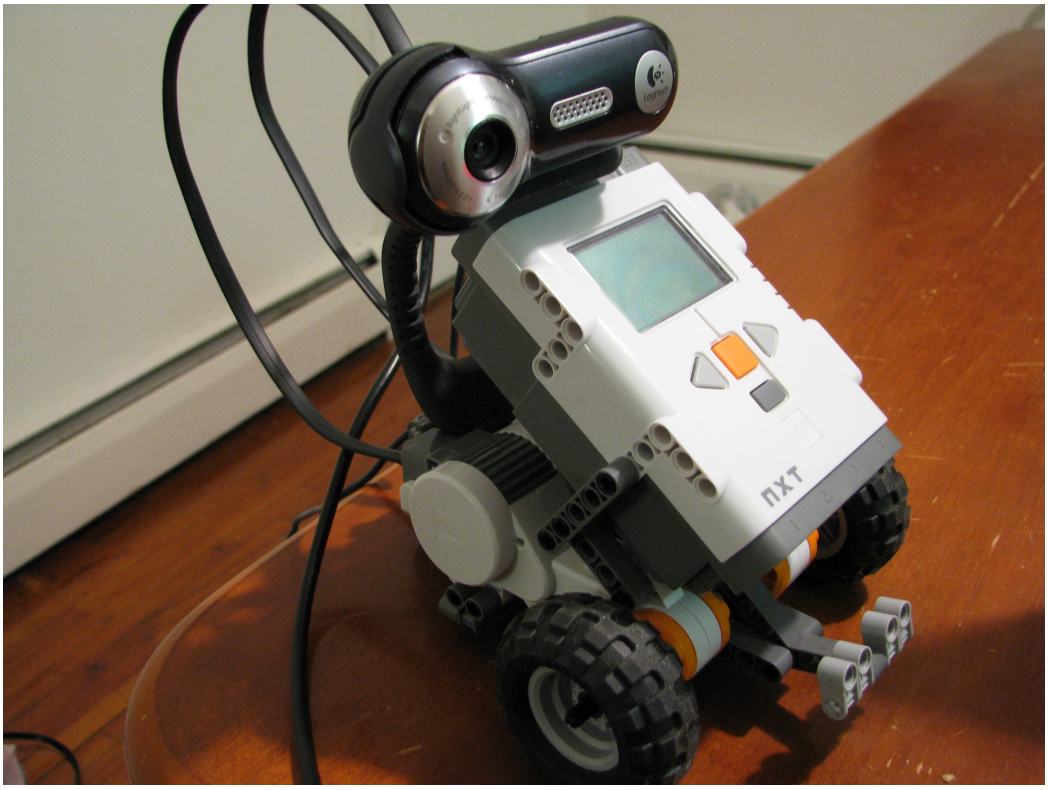


Figure 2: A picture of the Robot - The Agent

1.1. Stereo Camera Calibration

A Rubik's cube is used for calibrating the left and right cameras. After the initial setup as shown before, the Rubik's cube is placed a certain distance *d* away from the two cameras such that each camera can see two vertical faces. Figure 3 gives an example of this.

Figure 3: a) Image of the Rubik's cube with two visible faces as taken from the Left Camera, b) Image of the Rubik's cube with the same two visible faces as taken from the Right Camera,

The images taken are used as input to a Matlab program and the Intrinsic and Extrinsic Parameters of the Left and Right Camera are calculated. Points are selected from the images in a particular sequence, the same as the hard–coded world coordinates. As the algorithm progresses, these parameters can be used to check the accuracy of the computed Fundamental Matrix and hence the Depth Estimator. Also the calculated focal length (Intrinsic Parameter) is used in the computation of Depth.

2. Feature Detection

The first step of the Algorithm is to acquire the position of the Face and then Eyes. In the year 2001 Paul Viola of MERL and Michael Jones of Compaq, Cambridge published an Object Detection Algorithm based on a Boosted Cascade of Haar Classifiers. Their work was published in CVPR 2001. Their idea is based on machine learning algorithms, namely AdaBoost, which selects a small number of critical features from a larger set and yields efficient classifiers. The main advantage in the algorithm is possibility of cascading these classifiers. From their Face Detection Experiments under real-time applications, they have reported the detector to run at 15 frames per second without resorting to image differencing or skin color detection.

2.1. Face and Eye Detection using Cascaded Classifiers

There are several methods posted on the internet detailing how to construct the required .xml classifier files. One such file is present in the OpenCV sample c code and is referenced by the facedetection code that is provided.

A classifier consists of 3 main components – Positive Samples, Negative Samples and Training. For a dataset of about 4000 samples, it takes about a week to build an efficient classifier and generate the necessary .xml file. For the purpose of my project, I made use of the faceclassifier.xml file available in OpenCV. I have used an open source eyeclassifier xml file for the purpose of eye detection. These two files are easily compatible with OpenCV. Therefore, the Left Camera Image is input to the Classifier which detects the position of the head and hence the position of the eyes. We find that the classifier is invariant to background noise and skin tones. This step is shown in Figure 4 below.

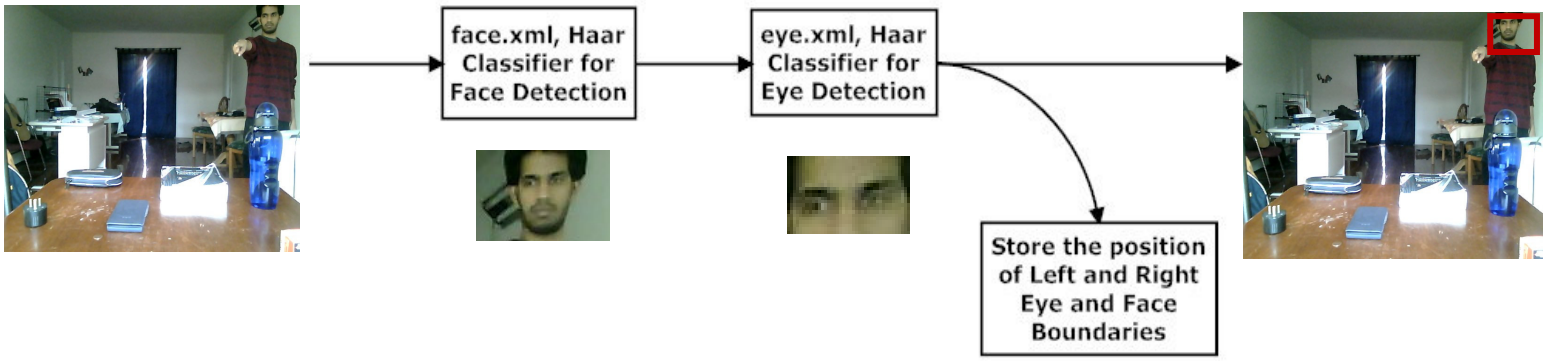


Figure 4: A step by step working of the Face and Eye Classifier using OpenCV

2.2. Hand Detection using Skin Color Segmentation

Once the positions of the Face and the Eye have been stored, these values are used in Matlab to perform skin filter segmentation. Given the input the image and position of the detected features, we convert the image to the YCbCr and HSV Color Space. The most commonly used skin filters have these thresholds –

140 =< Cr <= 165

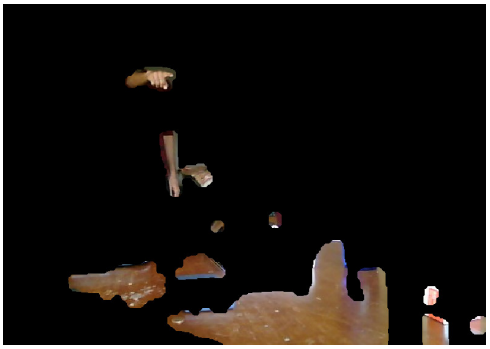
140 =< Cr <= 195

0.01 =< hue <= 0.1

These values have been thoroughly researched and after numerous tests, they were found to give acceptable results. In most cases, the filter tends to extract regions in the environment that are associated with a skin color tone. We can avoid their detection using a simple boundary check.

Hand Detection –

Since we know the lower boundary of the face, the skin filter is applied from that point to the rest of the image. We know apriori that the volunteer in the image is pointing at an object and his/her hand is therefore within a particular radius from the face. We calculate the top and bottom boundaries of the detected pointing hand and set the remaining pixels to zero. This eliminates the chances of the environment skin tones affecting the algorithm.



The Table, left hand and other portions of the image are also segmented out. This is unwanted

Segmentation of the required Pointing Hand only

Figure 5: The effect of the environment skin tones when using the skin filtering segmentation method.

An important note to make is that in the present state the algorithm works only for right-handed pointing. After the hand is segmented out, the boundary pixels locations are evaluated to understand which direction, the hand is pointing (straight, left or right). Based on that information, the fingertip can be located as a simple boundary pixel shift operation.

Once the fingertip has been segmented out, we estimate an equation that joins the x,y coordinates of the right eye and the position of the finger tip. This can be done using Least Squares. The projected line forms the required Line of Sight Vector. This is shown in Figure 6.

2.3. Object Detection from Pixel Matching

After the LOS vector has been computed, a pixel matching operation is performed along the direction of the vector. A suitable window is chosen and a SSD of the pixels within the window will indicate if any uniformity in color. There will be no uniformity when the line is passing through the environment until it passes through an object. As the window reaches that point, the SSD filter produces uniformity in colors and the program declares that an object has been detected.

Note - This program is not functioning optimally yet. It requires further analysis.

Figure 6: The LOS vector joining the calculated eye points, fingertip points and detected object.

3. Depth Estimation

So far we have been working with the Left Image to detect the required features. The only feature we will need at this point is the computed x,y coordinate of the object. In order to determine the position of this object with respect to the Camera, we must estimate the Fundamental Matrix. The Stereo Vision Setup allows us to estimate to acceptable level the depth of the object (direction on the z axis) from the camera plane.

3.1. Determine the Fundamental Matrix

The Fundamental Matrix estimation has been well described by Trucco and Verri in their 3D Computer Vision textbook. Commonly known as the 8-point Algorithm, we need to acquire a set of at least 8 point correspondences between the left and right camera image. In our case we use 12 to 21 points (larger the number of points, better the estimate of *F*). The equations are converted to a Af=0 form and solved using SVD. The Fundamental matrix acquired can be used to determine the corresponding points across the left and right image. It is related to the Pixel coordinates in this manner –

$$\mu_r = F\, p_l$$

where μ_r is the Projective Epipolar line in the right camera image corresponding to p_l

F is the Fundamental Matrix

p_l is the pixel coordinates of a point in the left camera image

μ_r can be represented in the form of a line equation –

$$ax_i + by_i + c = 0$$

Using this equation and the one above, we can determine the epipolar line on which the corresponding point is supposed to lie

Example of a Fundamental Matrix Obtained –

FMat =

0.0000	-0.0001	0.0481
0.0000	0.0000	-0.0094
-0.0459	0.0228	0.0553

Consider the Image given below, the blue highlighted region represents the pixel coordinates in the left camera image. We want to calculate the corresponding point in the right camera image. Using the above given Fundamental Matrix, we compute the corresponding epipolar line (shown below). We can see that it passes through the ipod.

Figure 7: Top – Left Camera Image with Highlighted Blue region (ipod), Bottom – Corresponding Epipolar Line in the Right Camera Image which contains the required point.

3.2. Correlation Matching based on SSD

As shown in Figure 7 we have the corresponding Epipolar line which contains the corresponding point. We perform a Sum of Squared Differences around the pixel coordinate region in the Left Camera Image. This value is then matched with the SSD values computed along the Epipolar line in the right camera image. At the ipod, the correlation match between the two SSD's will be high and the program declares it has found the point.

Note - This program is not functioning optimally yet. It requires further analysis.

Computation of SSD -

$$\Sigma(x_i\text{-mean}(x))^2$$

3.3. Computation of Pixel Disparity

At this point we have the required pixel coordinates in both the camera images. We can calculate depth using the formula

$$Z=f\,T/d$$

Where f is the focal length

T is the distance between the camera centers

d is the disparity between the pixels, $x_r - x_l$

For example, consider the above example in tracking the ipod,

f=700 as calculated initially

T = 27cm

D = 555 - 281 = 274

Z = 650*27/274

Z = 64.05cm

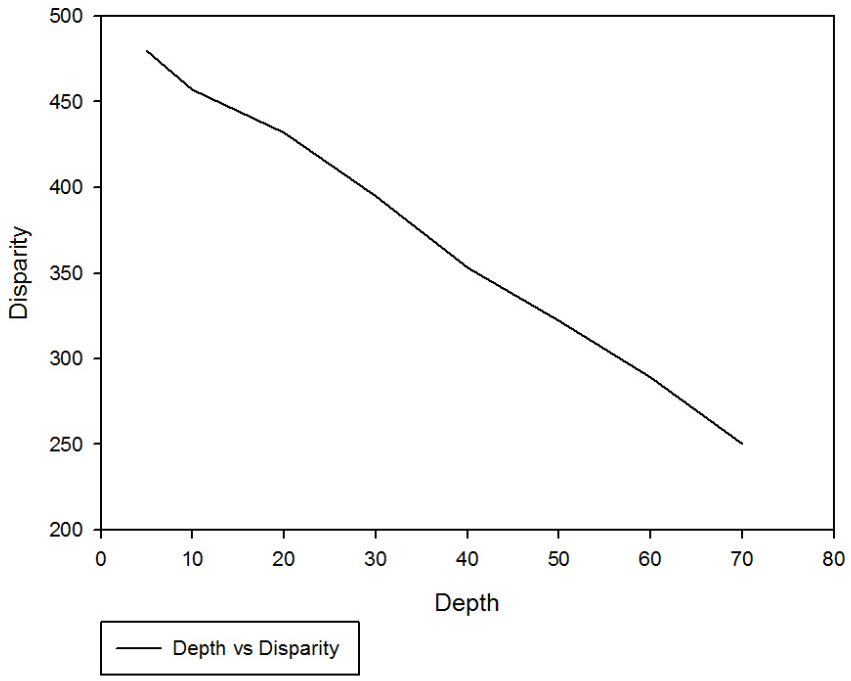
The ipod was actually present at **72cm** from the camera center. Although not exact, the values were acceptable.

4. Robot Movement towards the Target

The 3D depth position of the Object has been computed. The robot simply proceeds forward in that direction until that distance has been covered. Future work would be to take into the account the angle variation of the objects position from the center line. The robot would have to align itself along one camera direction, calculate the angle to the object and then chart a course.

5. Experimental Results

- Disparity vs Depth



We can see that they are inversely proportional to one another.

- Feature Tracking

Total of 30 trials	Positive Results	Negative Results
Face Detection	26	4
Eye Detection	24	6
Finger Detection	15	15

- Effects of Occlusion

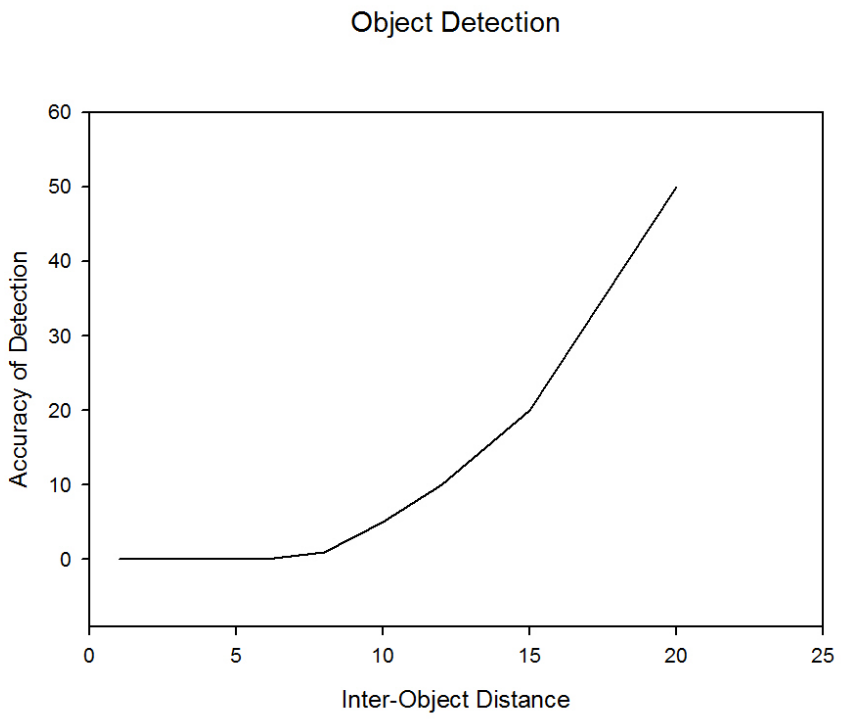
When running test programs, there are trials when the hand would cover a part of the face. This happens more so due to an unconscious tilt by the volunteer. During such inputs, the program returns bad results. It is unable to detect the boundary of the hand. Therefore it is necessary for the pointing hand not to cover any part of the face and for the face to be straight.

- Projection Errors in calculation of corresponding points

When correlation matching using SSD, they values don't always match and points don't always turn out to be the same. For these cases, I have tried using SAD and the results for them are also not concrete. The results seem positive and negative more based on chance rather that robustness of the algorithm.

- Inter-Object Distance

In order to be easily able to separate objects, we find that there should a minimum distance of at least 8 cm between any 2 objects.



6. Recent Trends in Robotics

In 1961, an article titled, “A peep into the automated future” by Paul Mickle detailed about the first functioning robot that joined General Motors. It was an automated die-casting mold that dropped red-hot door handle into pools of cooling liquid on a line that moved them along to workers for trimming and buffing. What is important to note is that it wasn’t until the next millennium that the GM robot was realized as the ‘Eden of Robots’.

Back in those days, Robotics was primarily composed of Control Theory Research. As the years progressed, with the progress of other areas of research namely Computer Vision and Machine Learning, the area of Robotics widened its scope to accommodate the mentioned fields. Present day robotics has been advanced to a great degree, diversified and found applications in almost everywhere. They can be broadly classified into 5 categories – Academic, Defense, Medicine, Industry, Home.

If we take into account all the ongoing research, we see that the aim is to make a robot as human as possible, can we make it learn, can we make it feel, see, hear. Honda’s Asimo and the Nao are only a few examples of this point. Should we think of them as replacements to humans?

In the present state, they are more of an aid to our tasks. For example - Advanced robots are being modeled to aid in surgeries and complex medical techniques. They provide accurate, reproducible and exact results. They are not prone to error. There are even machines that are controlled by humans. This breakthrough has helped the lives of millions if not more.

Research Scientists at Universities and the Industry are looking at making Robot a common things in homes. There is a need to make robots available for home use and make the lives of people easier. Home Automation is no longer a dream. And to reach this goal, they are different events and fields which challenge all areas of research.

Example - Robot Soccer competions are held to better understand different planning algorithms. Robots are designed to be controlled from a distance of light years away – Communication breakthroughs.

One of the most advanced robots I’ve come across is Boston Dynamics – BigDog. It has the ability to walk on any terrain and maintain its balance. If ever there was way to set a trend, it would be with the BigDog. It was released in March 27, 2008.

The Future -

In our current generation, Multi-Core Architectures are being heavily used to facilitate the computationally intensive tasks. Several researches are now making avenues to use these advanced architectures to design tomorrow’s robot. They will take into account the large vision and planning datasets to make the robot robust to environment. An example of the power of super computers is a project in EPFL called the BlueBrain project. It is the first comprehensive attempt to reverse-engineer the mammalian brain, in order to understand brain function and dysfunction through detailed simulations. The computations for this endeavour are being carried out by IBM’s BlueGene.

7. Bibliography

1) Rapid Object Detection using a Boosted Cascade of Simple Features by Paul Viola Michael Jones (CVPR 2001)

2) Evaluation of Hand Pointing System Based on 3-D Computer Vision by P. Serafinavičius, S. Sajauskas

G. Daunys, 2008 ELEKTRONIKA IR ELEKTROTECHNIKA, AUTOMATION, ROBOTICS AUTOMATIZAVIMAS, ROBOTECHNIKA

3) Introductory Techniques for 3-D Computer Vision by Trucco and Verri

4) Stereo Vision for Unrestricted Human-Computer Interaction by Ross Eldridge and Heiko Rudolph

RMIT University, Melbourne, Australia, itechonline Database

5) Real-time Recognition of 3D-Pointing Gestures for Human-Machine-Interaction by Kai Nickel and Rainer Stiefelhagen, DAGM 2003.