Design and Development of Ball Catching Robotic Arm

Kartik Sharma, Gianetan Singh Sekhon

Abstract— This paper presents a ball catching robotic arm system with 3DOF assembled from the commercially available parts. Other than the previous research work a mechatronically complex system were designed. We have designed a very simple arm which is able to move for catching the ball when it has to move. The given robotic arm system is low cost implementation compare to the previous one. In this system, a single camera system is use to perceive the trajectory of the ball, the system detects the ball in each frame with the help of a fast mean shift algorithm. It calculates the shift of the mean of the identified color intensity and according to that it sends the control commands over the serial port to the robotic arm via ZigBee. The basic objective to catch the flying object at the expected location. This catcher robotic arm can catch the ball thrown to it from 5-6 meter with an average success rate of 70-75%.

Index Terms—Ball Catching Robotic Arm; Robotics; Robotic Vision

I. INTRODUCTION

Catching a moving object with a hand is one of the most difficult tasks for humans as well as for robot systems. In order for a robot to catch a moving object, the pose of the object must be determined. When the trajectory of the object is known, the object motion can be calculated. Common approaches to utilizing vision information for controlling robotic manipulators are the (i) position (velocity) control and the (ii) image feature control (color).[1]

A large number of ball catching robot arms have been designed for the couple of decades, and several of these have become standard platforms for R&D efforts. The most widely used is without a doubt the Ultimate PUMA 5xx series, DLR Institutes robotics Arms. The general availability of design components and computational software matter's a lot here, also the reasonable price is important to allow design of systems that can be replicated and further developed by others.

The goal of our work is to visually track a flying object throwing towards the robotic arm and send the data to the robotic arm to move for catch the ball at expected location. As we use a new low cost self developed robotic arm (figure 1) to catch a ball that is thrown by someone towards the robot. The same problem has been addressed by several researchers and also they proposed several successful results but they differ from ours in terms of cost of the entire system and catching accuracy of ball.

Manuscript received on August, 2012

Kartik Sharma, M.Tech Student, Yadavindra College of Engineering(YCOE), Punjabi University Patiala

Asst. Prof. Gianetan Singh Sekhon, Incharge C.E. Section, Yadavindra College of Engineering(YCOE), Punjabi University Patiala

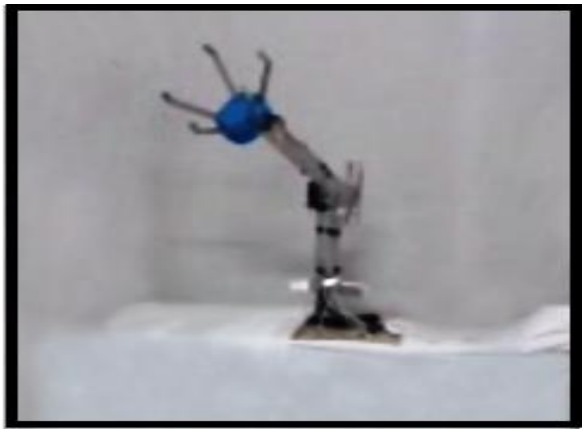

Figure 1

However, for research work in robotics it is often a challenge to get access to a high performance and expensive robots which is also available to other researchers. In many respects robotics has lacked standard systems based upon which comparative research could be performed. Too much research is performed on a basis that cannot be replicated, reproduced or reused [2]. So to continue in the similar field every researcher has to produce its own robots to do comparative analysis and to get results.

The most important problem that had to be addressed is that of selecting the feature to use for segmenting the ball. Using approach where to detect the ball by its shape is not an efficient approach, in this approach we have require to set our background static or stable because here tracking of moving object is obtained by computing the difference between the actual image (new frame) and some reference image i.e. background image which is static. This is very difficult when we used our system in real time constraints. Instead we using color of the object in which we just take care about the lightning conditions we never bother about setting the threshold value of background with the original object.

In our general setting, the object is in the air for about 0.5s to 1s, covering a distance of about 5 to 6m. Therefore it is essential to provide an early prediction of the trajectory to quickly command the catch position to the robot. As the ball is tracked for a longer time, the prediction becomes increasingly precise and catching accuracy of the arm is improved.

II. RELATED WORK

In the famous and pioneering work [4], [5] the 7 DOF DLR-LWR-III arm with a gripper to grasp the ball and an active vision system is used.

The task of tracking and predicting balls was studied as part of robotic ball catching systems ([4], [3], [6], [7], [8]). All of these have a static setup using stereo cameras with rather wide baselines. Detecting the ball is done by pixel-wise segmentation, using color ([6], [7], [8]) or using the difference to a reference image [3]. (Real time trajectory perception paper). A system with a 5 DOF arm on a humanoid robot (only the arm is moving) with a "cooking basket" at the end effector for catching the ball and an active vision system is presented in [9].

In work [3] they used the 7 DOF DLR-LWR-II arm with a small basket at the end effector and a stationary stereo camera and image processing system built from cheap "off-the-shelf" components.

III. CONTRIBUTION

In this paper we present a ball catching robotic arm setup, which significantly differs in two aspects from previous research. First, we develop a robotic arm with commercially available parts with a low cost. This is challenging for us to build an arm with such a low cost because the hardware and motors of the arm has to be able to reach to the ball fast enough and in addition it has to be light weight, so that the impact of the weight doesn't effects the motors. Moreover, this is also challenging for us to implement the algorithms for the arm movements, because all the work here is to be carried out for the real time results so we need such algorithms which are more efficient than the previous work and also which are able to work on our robotic arm setup.

The second new aspect is, that, instead of using dual or stereo vision cameras (300 fps) we use a 60 fps camera to determine the ball and to calculate its future trajectory. Our camera is here responsible for tracking ball, determines velocity of ball and also determines angle at which the ball is getting turned. So using these simple steps we are here able to calculate the trajectory of ball which then allows the arm to move up to the ball. As early mentioned we left the hand grasping work on the future work because of low funds availability.

IV. GENERAL SETUP AND ARCHITECTURE

Predicting the ball with stereo vision camera is extremely expensive as we need Gige interface cards and more powerful programming language to interface. We therefore use a single camera system mounted on a vertical pole on the beside on the thrower. The viewing angle of camera is 5m horizontally and about 4 m vertically. This is enough for us for tracking of ball from around 5 m. Simple studies of human performance indicates that our system is able to catch the ball if the ball is throwing in virtual 150 cm by 150 cm window. From these basic requirements it is possible to compute flight time and velocities for the trajectory perception, as summarized below.

- Throwing distance will be approximately 5-6m.
- Ball should be in air or flight time is about 0.9 s.
- The ball will travel with an approximate velocity of 5.5 m/s.
- The ball should be caught if it arrives within a virtual 150cm × 150cm window.

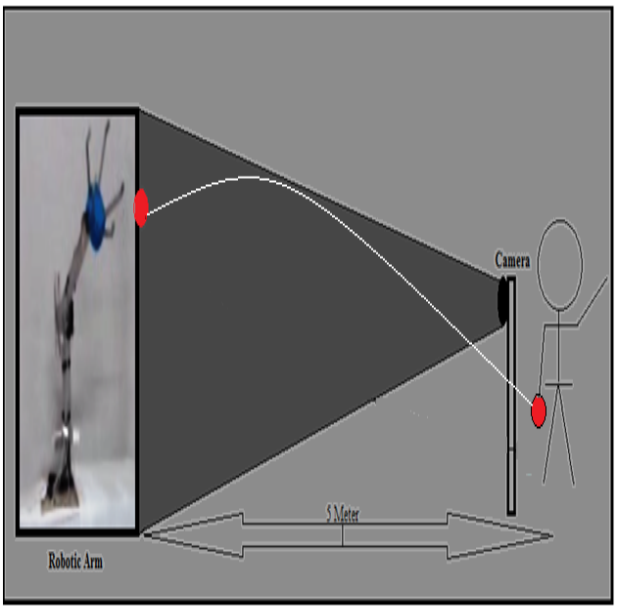

Figure2 Schematic of ball catching experiment

The ball is thrown by a human from a distance of about 5 m onto the robot with speed of typically 5.5 m/s, resulting in a flight time of about .9s. The robotic arm is placed on a table about 2.5 m above the ground level. The single camera system (60fps) is placed beside the throwing person at 1m height from ground level. The ball has a diameter of 8.5 cm and a weight of 70 g.

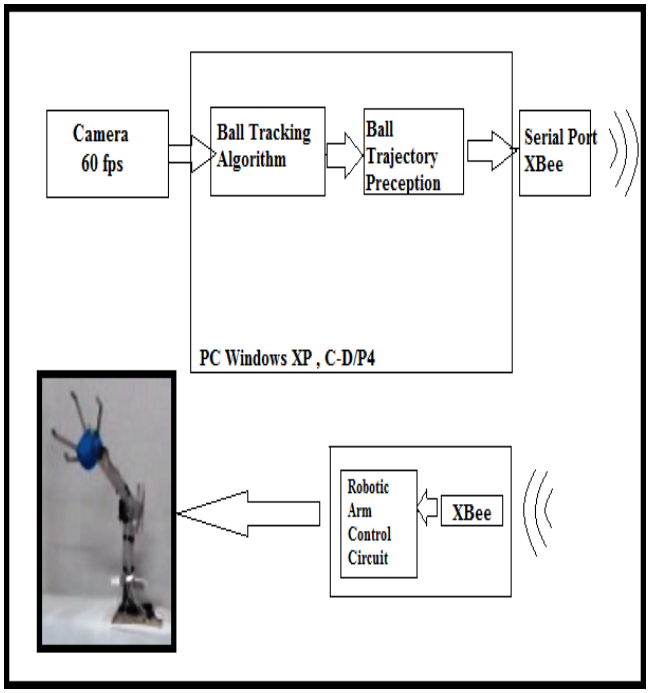

Figure 3 System architecture showing the processing modules

V. OBJECT TRACKING WITH MEAN SHIFT ALGORITHM

Mean Shift is a powerful and versatile non parametric iterative algorithm that can be used for lot of purposes like finding modes, clustering etc.

Mean Shift was introduced in Fukunaga and Hostetler [12] and has been extended to be applicable in other fields like Computer Vision. This part of our paper will provide a discussion of Mean Shift.

Intuitive Idea of Mean Shift

This section provides an intuitive idea of Mean shift and the later sections will expand the idea. Mean shift considers feature space as an empirical probability density function. In probability theory, a probability density function (pdf), or density of a continuous random variable, is a function that describes the relative likelihood for this random variable to take on a given value. If dense regions (or clusters) are present in the feature space, then they correspond to the mode (or local maxima) of the probability density function. We can also identify clusters associated with the given mode using Mean Shift.

For each data point, Mean shift associates it with the nearby peak of the dataset's probability density function. For each data point, mean shift defines a window around it and computes the mean of the data point. Then it shifts the center of the window to the mean and repeats the algorithm till it converges.

At the high level, we can specify Mean Shift as follows:

1. Fix a window around each data point.
2. Compute the mean of data within the window.
3. Shift the window to the mean and repeat till convergence.

The most popular way to detect a ball in an image is the circle Hough-transform ([13], [14], [15]). Conceptually, it counts the number of pixels along every hypothetical circle. This technique involves much calculation as it need two threshold values to be calculates every time [16]. Another popular technique for the circle detection is Contrast Normalized Sobel Filter (CNS). [16]

Here we present object tracking with mean shift algorithm the main idea of the mean-shift algorithm for detecting and tracking the segmented face in HCI, is a procedure for locating the maxima of a density function given discrete data sampled from that function. The algorithm for solving this problem has been addressed in Bradski [10] and Cheng [11]. The main steps in the mean-shift algorithm are as follows here first select the first pixel (color to be tracked) of the object to track.

- Define the area of the search window and calculate its centre
- Define the area of the object and calculate its centre
- Compute the distance between the centre of the search window and centre of the object which is moves in whole image and translate the search windows equal to the centre of the object.
- Repeat the step 2 until the value of i.

Object color intensity which we exactly need to track it may result poor object localization. If the window size is chosen too large, it will contain too many background pixels. So always there is need to select appropriate search window size to achieve better results.

Here in this section fig(4-8) we presents results of ball tracking by mean shift in 2d,mean shift algorithm here provide efficient results for ball tracking even under the poor environmental conditions.

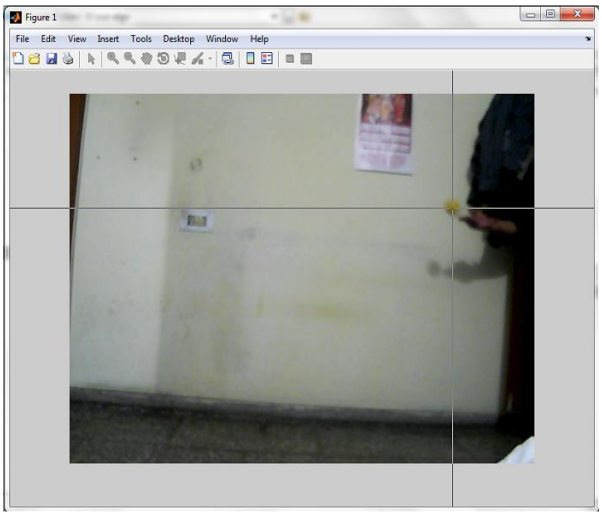

Figure 4 feature selection (color)

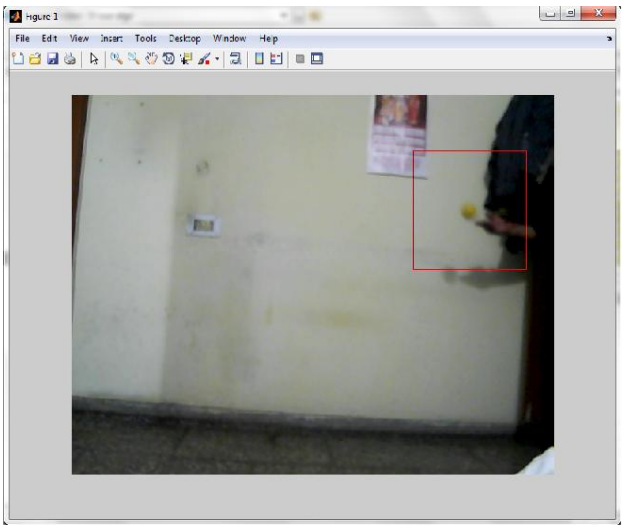

Figure 5 search Window Initialization

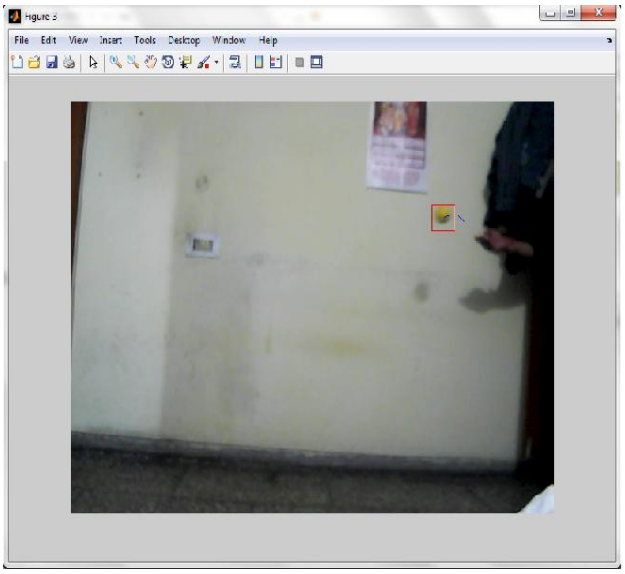

Figure 6 defining area of object (ball)

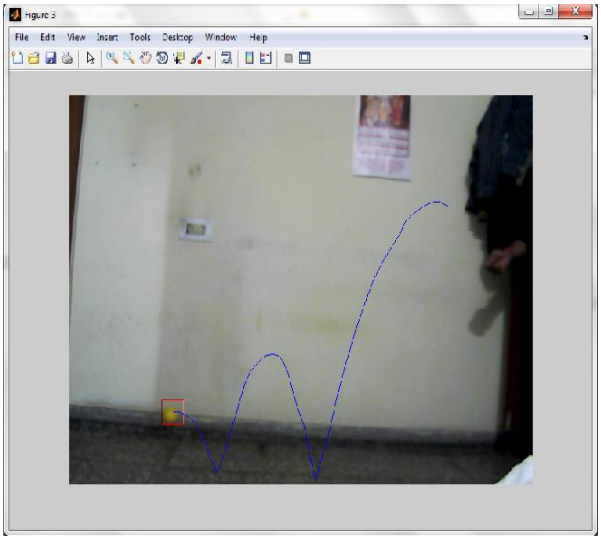

Figure 7 Mean shift algorithm tracking

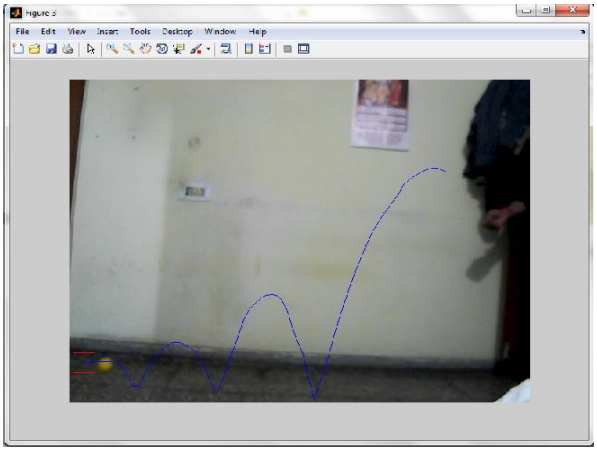

Figure 8 complete tracking of object (ball)

As we are here concerned not only with object tracking but also with the trajectory interception of the ball which is throwing towards robotic arm. So along with object tracking with mean shift we also calculates the velocity of the throwing object and angle, both of these velocity and angle we calculates ball trajectory.

VI. PREDICTION AND INTERSCEPTION

Modeling the trajectory of a ball throw can be done threw the simple Newton mechanics. The only difficulty is the need of a perfect mechanical stable model because if the model is not stable in any sense then the catch point has to be predicted from the modeling is enough for exact catching. Here we need the velocity of the ball with the angle to calculate the trajectory of the given throw. After the half way of the ball throw the values of velocity and angle of projectile is calculated by matlab with trajectory equation with the gravitational force 1g.after the half ways the precepted values of velocity and angle of projectile are sent to the controller for the future calculations and on the behalf of these calculations the catch point of the robotic hand is getting decided.

Here in the figures (9-10) the 3d and 2d plots are shown of the throwing ball , in figure 11 the pictorial representation about the vision system tracking and predicts thrown ball in real-time.

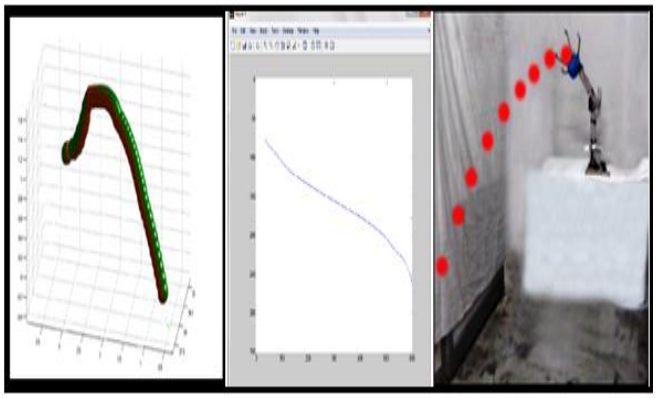

Figure 9 (from left to right) shows (1)3d Modelling of ball (2)2d modelling of ball(3)The vision system tracking and predicts thrown ball in Real-Time

VII. IMPLEMENTATION

This section describes the technical details of the actual implementation of the robot arm.

A. Hardware implementation

The arm proposed and specified in the earlier sections was constructed with commercially available parts and with lots of reasearch and development as our main goal was to develop a low cost arm system than the previous arm systems e.g (Puma 5x,DLR arms).we have satarted our experiments with simple stepper motors with high torque capacities but that fails in our early experiments. After that we have used the dc motors with the iron gear boxes that are highly powerful. These motors are used at two joints of our robotic arm

1. Base i.e for left and right rotaion
2. For up and down rotaion.

The specifications of our motors are

Sr. No	Part	Product name	Mass	Comments	Torque
1	1 joint	DC Geared motor	1.8 Kg	75:1 gear system	3 Nm
2	2 joint	Dc Geared Motor	1.8 kg	75:1 gear system	3 Nm
3	3 joint	Servo Motor	80 Grams	50:1 gear System	1.3 Nm

Table 1 Specifications for Motor Used In the Robotic Hand

These motors are operated by the matlab with the help of controller circuit(robot control circuit). The matlab gives direct command to start or stop the motors over the serial port. All the communication is done with the help of ZigBee.

ZigBee modules are a family of really nice little radio devices that use the ZigBee or 802.15.4 protocol. They send and receive data via the 2.4GHz or 900 MHz band at a relatively low power and can be used to set up simple point-to-point links. Interfacing with a ZigBee module is easy. They all have the same pin outs, and you talk to them with simple serial commands. They can be configured to be very low power.

In our early experiments we have faced a lot of problems due to collisions of robotic arm with its own wires and due to hanging of the Controller or Matlab due to long wires for data communication over serial port. In order to avoid these collisions in the robotic arm we provide the mechanical limit switches in the robotic arms for avoidance of collision form wires of and other parts. These switches enforce a limit to the rotation of arm so that it cannot rotate beyond its limit that prevents it from mechanical damage. Table 2 shows the limits of the robotic arm from its home position (i.e. 90o) and Figures (10-15) shows the mechanical switches that prevent it from damage.

Sr. No.	Joint No.	Limits from its Homeposition(i.e. 90^o)
1	Base Motor (Joint1)	-90^o to +90^o
2	Upper Motor(Joint 2)	-60^o to +60^o

Table 2 Limits on Joint Angles.

Figure 10 Limit Switch right side **Figure 11** Limit Switch left side **Figure 12** Top View of Limit Switches

Figure 13 Limit Switch down side **Figure 14** Limit Switch up side **Figure 15** Side View of Limit Switches

B. Software implementation

Matlab 7.12.0 version R2011a high-level technical computing language is used here for the algorithms implementation with Intel Celeron D Processor. As mentioned earlier, for tracking we have used the mean shift algorithm that is also responsible for calculating velocity and angle of the ball. All this tracking perception and sending of control signals is done *with Intel^{TM} Celeron D @ 3.33 MHz* equipped desktop system.

There is still some fine tuning remaining to be done for the robot arm, but even so, it already fulfills all the specified requirements that can be measured, and has a performance that is similar to previous research which are very high cost implementation(about $81000) than our work which is about $1200 implementation. The remaining section shows the experimental results of our system.

VIII. EXPERIMENTS AND RESULTS

In Fig. 16 we show experimental results for the catch modes recorded for similar throws. The catch success rate (number balls caught versus the number of balls thrown into the feasible catch spaces) is > 70% and the initial distance of the ball from Arm is about 5-6 meter. The main cause of missed catches was errors in the early predictions of the ball path, causing the robot to start motion in the wrong direction.

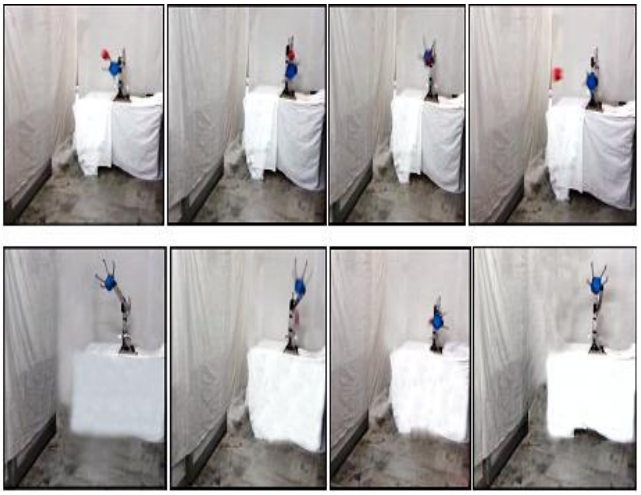

Figure 16 Image sequence (from upper left to lower right) shows the catching experiments done at our labs, in upper left to upper right the catching of ball is shown from different angles and in lower left to right shows about the different positions of robotic arm when it reaches at its catch point.

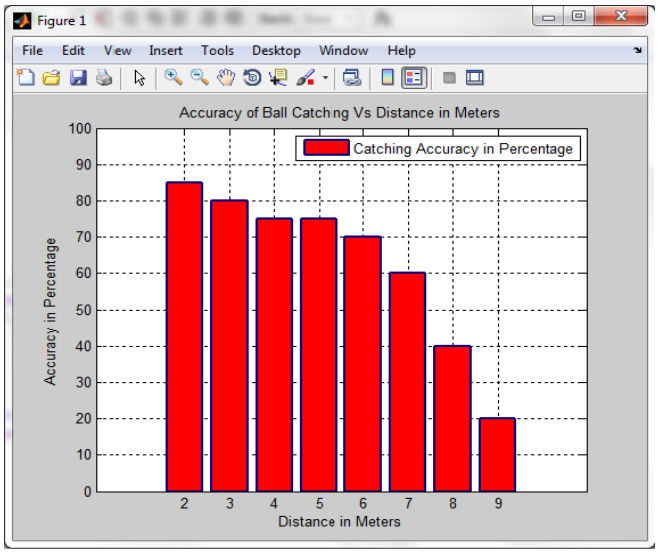

Figure 17 Catch Success Rate Graph

The figure 17 here shows the catch success rate graph. Here we have plot a bar graph between the distances in meter v/s percentage accuracy of catching ball.

As results shown in graph the accuracy of the catching ball is better when we throw the ball towards the robot from 2 meter distance and the results are worse when we increase the distance from 6 meter onwards so here we conclude that the given system provides good catch success rate for the distance 5-6 meter the average catch success rate here is about 70-75 %.

Sr. no.	Author	Title & Year	Method used for ball tracking	Initial distance of ball from robotic arm	Accuracy in %
1	Barbara Hove et al	Experiments in Robotic Catching 1991	Initial motion algorithm	Up to 5m	70-80%
2	U. Frese et al	Off-the-shelf Vision for a Robotic Ball Catcher, 2001	Threshold based object segmentation	Up to 1 m	67%
3	Akio Namiki et al	Development of a High-speed Multifingered Hand System and Its Application to Catching, 2003	object segmentation and extraction algorithm with high speed vision system	1 m upward	90%
4	Chyi-Yeu Lin et al	The DSP Based Catcher Robot System with Stereo Vision2008	Background image subtraction with feature extraction	Fixed 4 m	65%
5	Tasuku Yamawaki et al	Robot Catching with High Manipulability Grasp Configuration using Vision, 2009	Segmentation and Extraction algorithm with high speed vision system	Up to 4	30%
6	Berthold Bauml et al	Kinematic ally Optimal Catching a Flying Ball with a Hand-Arm-System, 2010	Threshold based object segmentation	Up to 5m	80%
6	Oliver Birbach et al	Realtime Perception for Catching a Flying Ball with a Mobile Humanoid, 2011	Contrast normalized sobel method for segmentation	Up to 5m	80%
7	Thomas Wimbock et al	Experiments on Grasp Acquisition, 2007	Threshold based object segmentation	Up to 4	80%
8	Magnus Linderoth et al	Object Tracking with Measurements from Single or Multiple Cameras, 2010	Threshold based object segmentation	Up to 4	50%
9.	**Kartik Sharma et al**	**Proposed System**	**Mean Shift Algorithm**	**5-6m**	**70-80 %**

Table 3 Comparison Table

The Above table compares our work with the previous research work. Here the proposed system has the catch success rate is 70-80 % with a initial distance of 5-6 m. The main cause of missed catches was errors in the early predictions of the ball path, causing the robot to start motion in the wrong direction.

IX. CONCLUTION AND FUTURE WORK

We learned several lessons during this research work. We have shown that it is possible to catch a thrown ball with a robot arm using computer vision. Here we have used simple commercially available parts to complete our arm which is much lesser in cost as compare to the previous work.

The presented method clearly has the limitation of not grasping the ball when it is strike at the hand palm. This hand grasp acquisition work is pending on the future work due to low availability of funds. Furthermore, for computer vision system we advice the researchers to developed own software on Linux or others. Future work will concentrate on these difficult issues.

REFERENCES

1. T. Koivo and A. J. Koivo," Catchability of a Moving Object by a Robot", IEEE/ICMC Proceedings of International Conference on Man and Cybernetics Systems, pp 2911- 2916 Vol. 3 2005.
2. Christian Smith and Henrik I Christensen," Constructing a High Performance Robot from Commercially Available Parts" IEEE/RAS Robotics and Automation Magazine, Vol 16:4, pp. 75-83, Dec 2009.
3. U. Frese et al.," Off-the-Shelf Vision for a Robotic Ball Catcher", IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 1623- 1629 vol.3, 2001
4. B. Hove and J. Slotine, "Experiments in robotic catching," in Proceedings of the 1991 American Control Conference, 1991, pp. 380–385.
5. W. Hong and J. Slotine, "Experiments in hand-eye coordination using active vision," in Proceedings of the Fourth International Symposium on Experimental Robotics, ISER95, 1995.
6. M. Riley and C. G. Atkeson, "Robot catching: Towards engaging human-humanoid interaction," Autonomous Robots, vol. 12, pp. 119–128, 2002.
7. C. Smith and H. I. Christensen, "Using COTS to construct a high performance robot arm," in Proc. of the IEEE Intern. Conf. on Robotics and Automation, 2007.
8. G. Batz, et al., "Dynamic manipulation: Nonprehensile ball catching" in Proc. of the IEEE Mediterranean Conf. on Control and Automation, 2010.
9. K. Nishiwaki at al. "The humanoid saika that catches a thrown ball," in Proceedings of the IEEE International Workshop on Robot and Human Communication, 1997, pp. 94–99.
10. Bradski, G.R.et al. "Computer Vision Face Tracking For Use in a Perceptual User Interface", Microcomputer Research Lab, Santa Clara, CA, Intel Corporation, 1998.
11. Cheng. Y, "Mean shift, mode seeking, and clustering". IEEE Transactions on Pattern Analysis and Machine Intelligence, *pp* 790-799 vol 7, 1995.
12. Fukunaga, K. & Hostetler, L. (1975). The estimation of the gradient of a density function, with applications in pattern recognition. IEEE Transactions on Information Theory, pp 32–40 vol 21, 1975
13. R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," Comm. of the ACM, vol. 15, , pp. 11–15, 1972.
14. C. Kimme et al, "Finding circles by an array of accumulators," Comm. of the ACM,vol. 18, pp. 120–122, 1975.
15. H. Yuen et al. "A comparative study of Hough transform methods for circle finding," in Proc. of the Alvey Vision Conf., 1989.
16. Oliver Birbach et al.," Real-time Perception for Catching a Flying Ball with a Mobile Humanoid" (ICRA), 2011 IEEE International Conference on Robotics and Automation, pp 5955- 5962, May 2011