International Journal of Application or Innovation in Engineering & Management (IJAIEM)
Web Site: www.ijaiem.org Email: editor@ijaiem.org
Volume 4, Issue 10, October 2015 **ISSN 2319 - 4847**

Shuttlecock tracking and trajectory estimation using Microsoft Kinect sensor

Aseem Raina[1], Nilay Mokashi[2], Pushkar Nimkar[3],Sanket Gujar[4]

Pune Institute of Computer Technology, Pune

ABSTRACT

The main objective of this paper is devising a method to detect the shuttlecock during a badminton game and to estimate its trajectory. The Microsoft Kinect™ sensor, for shuttlecock detection, and a Linux+Python environment, for data processing and filtering, was used. The trajectory estimation information, in the form of an equation of a straight line, was transmitted via Bluetooth™ to the robot, which extracted the given coordinates and with the help of mapping techniques, aligned itself to at a specific position in the arena to intercept the shuttlecock. Innovative algorithms were used to distinguish the shuttlecock from background objects. This project was undertaken as a task for ABU-Robocon 2015, the biggest international robotics competition in Asia- Pacific, held annually.

Keywords:- Shuttlecock detection, trajectory estimation, Kinect™, Python.

1. INTRODUCTION

The KinectTM sensor was introduced in November 2010 by Microsoft. The sensor unit is connected to a base with a motorized pivot. It was designed to be positioned above or below a video display and to track player body and hand movements in 3D space, which allows users to interact with the Xbox 360. The Kinect contains a RGB camera, depth sensor, IR light source, three-axis accelerometer and multi-array microphone, as well as supporting hardware that allows the unit to output sensor information to an external device. Along with the RGB values, Kinect also gives the depth values associated with every pixel. It uses structured infrared light to determine the depth values. Kinect builds on software technology developed internally by Rare, a subsidiary of Microsoft Game Studios owned by Microsoft, and on range camera technology by Israeli developer PrimeSense, which developed a system that can interpret specific gestures, making completely hands-free control of electronic devices possible by using an infrared projector and camera and a special microchip to track the movement of objects and individuals in three dimensions. This 3D scanner system called Light Coding employs a variant of image-based 3D reconstruction. Kinect is capable of simultaneously tracking up to six people, including two active players for motion analysis with a feature extraction of 20 joints per player.

Reverse engineering has determined that the Kinect's various sensors output video at a frame rate of ~9 Hz to 30 Hz depending on resolution. The default RGB video stream uses 8-bit VGA resolution (640 × 480 pixels) with a Bayer colour filter, but the hardware is capable of resolutions upto1280x1024 (at a lower frame rate) and other colour formats such as UYVY. The monochrome depth sensing video stream is in VGA resolution (640 × 480 pixels) with 11-bit depth, which provides 2,048 levels of sensitivity. The Kinect can also stream the view from its IR camera directly (i.e. before it has been converted into a depth map) as 640x480 video, or 1280x1024 at a lower frame rate. The Kinect sensor has a practical ranging limit of 1.2–3.5m (3.9–11.5 ft) distance when used with the Xbox gaming station software. The area required to play Kinect is roughly 6 m2, although the sensor can maintain tracking through an extended range of approximately 0.7–6 m (2.3–19.7 ft). The sensor has an angular field of view of 57° horizontally and 43° vertically, while the motorized pivot is capable of tilting the sensor upto 27° either up or down. The horizontal field of the Kinect sensor at the minimum viewing distance of~0.8m(2.6ft) is therefore ~87cm (34 in), and the vertical field is ~63 cm (25 in), resulting in a resolution of just over 1.3 mm (0.051 in) per pixel. The microphone array features four microphone capsules and operates with each channel processing 16-bit audio at a sampling rate of 16kHz.

Because the KinectTM sensor's motorized tilt mechanism requires more power than the Xbox 360's USB ports can supply, the device makes use of a proprietary connector combining USB communication with additional power. Redesigned Xbox 360 S models include a special AUX port for accommodating the connector, while older models require a special power supply cable (included with the sensor) that splits the connection into separate USB and power connections; power is supplied from the mains by way of an AC adapter.

Figure 1 Output of Kinect after horizontalstack

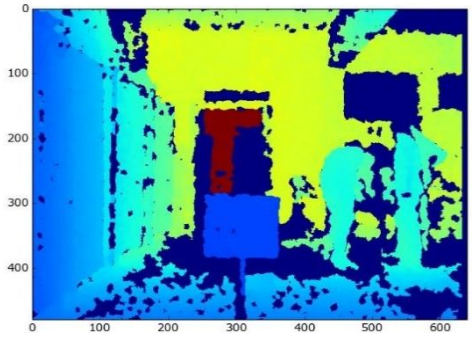

Figure 2 Depth Image ofKinect

In the field or Robotics, KinectTM has proved to be of a significant application in design of autonomous robots. Libraries for Kinect are available in a many languages. In this project, Python was used for programming on a Linux based open source platform (Ubuntu).Since KinectTM provides output at a frame rate of 30 frames per second, our algorithm and code had to be optimized to detect the shuttlecock and relay the coordinates in minimum amount of time. The theme for the ABU Robocon2015 was a badminton doubles game which demanded designing of 2 robots, manual or autonomous, which were pitted against the robots of the opponent team. The standard rules of doubles badminton were applicable. We decided to use Image Processing to detect the shuttlecock after the opponent's serve, so that our robot could easily align itself to intercept the serve and return the shuttlecock. One of the major challenges was time constraint, badminton being a fast paced game. We had a window of about 1s till the shuttlecock landed in our court and we had to make sure our robot was properly positioned by then.

2. RELATED WORK

Since going open source Kinect has attracted many researchers all over the globe. In [1], Zhang has presented a comprehensive review of recent Kinect-based computer vision algorithms and applications. The reviewed approaches are classified according to the type of vision problems that can be addressed or enhanced by means of the Kinect sensor. The covered topics include pre-processing, object tracking and recognition, human activity analysis, hand gesture analysis, and indoor 3-D mapping. The paper serves as a helpful tutorial and source of references for Kinect-based computer vision research. Kinect has also been used for indoor navigation of robots. [2] shows the results from testing the Kinect sensor on an autonomous ground vehicle.

Tracking any object in real time is always a tough task. As badminton playing robots are very uncommon, application of computer vision around the scope of this paper has been very less studied. Shuttlecock tracking has been worked upon by a few researchers, but that is passive. The video is recorded during match and the analysis done later. Work has been done upon plotting the trajectories for post-match analysis to help coaches find out where the player went wrong, what are the strengths of opponents etc.

When any moving object is captured on a video camera, motion blur is observed due to the intrinsic limit of frames per second. [3] introduces a trajectory estimation method using blur characteristics in 3D space. The images were captured from two different viewpoints. This method can extract both the position and speed of moving object in frame. [4] uses 2-D seriate images to obtain statistical data of badminton games. The shuttlecock trajectory is extracted from all detected trajectories using various characteristic parameters. In [5], Yoshikawa et al. have showed a very novel approach for tackling the situation. They describe an automatic serve scene detection method employing cubic higher-order local auto-correlation (CHLAC) and multiple regression analysis

(MRA). CHLAC can extract features of postures and motions of multiple persons without segmenting and tracking each person by virtue of shift-invariance and additivity, and necessitate no prior knowledge. Then, the specific scenes, such as serve, are detected by linear regression (MRA) from the CHLAC features. In [6], Li studied the two-dimensional detection in the badminton Hawkeye system with background subtraction. The mass centre coordinates of the badminton were obtained after eliminating the influence of the isolated noise by morphological opening operation, which have found the base of badminton 3D information obtaining and reconstruction for the following research.

Linear regression is a widely used method to interpolate or extrapolate data from known random measurements. Linear regression models could be fitted using various techniques, least squares approach being the common one. In [7] Iu and Wohn argue that the 3-D velocity of a single point up to a scalar factor can be recovered from its 2-D trajectory under the perspective projection. Then they extend the idea to the recovery of 3-D motion of rigid objects. In both cases measurements are collected through temporal axis first, while keeping the amount of measurements in each frame minimal. We may use multiple features to get a more accurate estimate if they are available. This approach called temporally oriented approach requires us to introduce the explicit model for the evolution of 3-D motion. The analysis is based on the assumption that the 3-D motion is smooth so that its 3-D velocity can be approximated as a truncated Taylor series. Regression relations between unknown motion parameters and measurements for a single point and rigid

body are derived. The method of Maximum Likelihood is used to estimate the motion. Also, the uniqueness of determining the 3-D motion of a single point is discussed.

3.IMPLEMENTATION

3.1Approach to the problem

Before deciding and designing suitable algorithms for our project, after some research on the competition environment, we came up with some calculated assumptions on which we later on based our algorithms-

i. There won't be much of a variation in the shuttlecock trajectory as the effect of wind in an indoor environment can be neglected.
ii. The drop point of the shuttlecock won't vary much as according to the rules of the game, the shuttlecock was required to fall in the service zone only.
iii. Other than the shuttlecock, no object, close to the area of the shuttlecock, would come in the frame of Kinect for the duration of the game.

Other than these, a part of the net and some surrounding props were removed from our detection frame using horizontal and vertical slicing. We were successful in removing some of the background noise using depth slicing techniques.

3.2Algorithms involved

3.2.1Contouring

Contours can be described simply as a curve joining all continuous points (along the boundary), having same colour or intensity. The function used for finding contours was cv2.findContours().There are three arguments in the function- image, contour retrieval mode, approximation mode and output consists of image, contours and hierarchy. The result "contours" is a Python list containing all objects boundary points as separate lists.

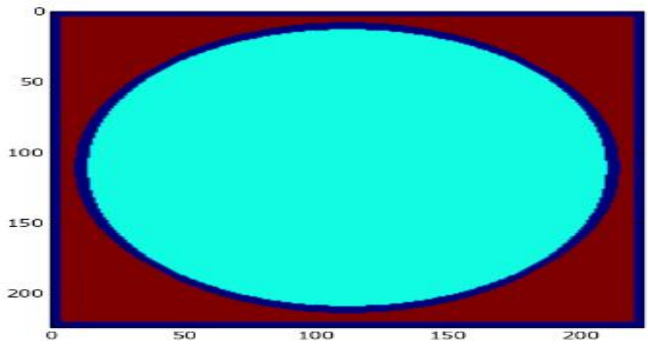

Figure 3 Contouring on an image

3.2.2Motion based detection

In this algorithm, two frames were captured, almost immediately apart using Kinect. Then the absolute difference function was used to subtract the two frames from each other. Any moving object would be highlighted after that because of its difference in position in the two frames.

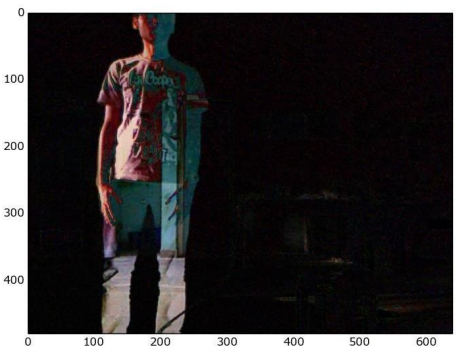

Figure 4 Subtracting two images to detect motion

The problem with this method was that the exact coordinates of the moving object could not be extracted, as the contrast of the detected object was not suitable for contouring. Also, since the difference had to be stored in a third image, it added to the computing time.
A more advanced detection algorithm was needed if motion detection was to be implemented. This was a relatively difficult task considering the time crunch we were facing.

3.2.3Color based detection

This method detects objects based on their colour and shade. The colour values are stored in a RGB matrix. Any object can be detected by hardcoding the values in the matrix. The extreme RGB values are white (255,255,255) and black (0,0,0).

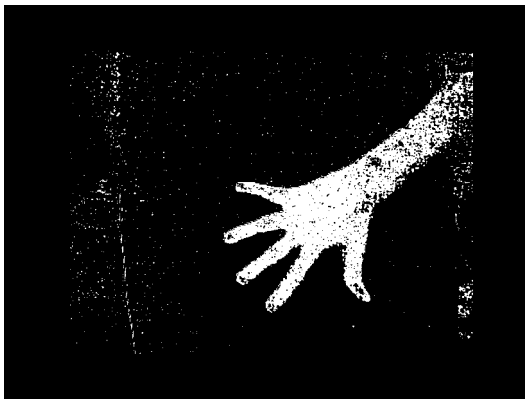

Figure 5.1 Detection of skin color

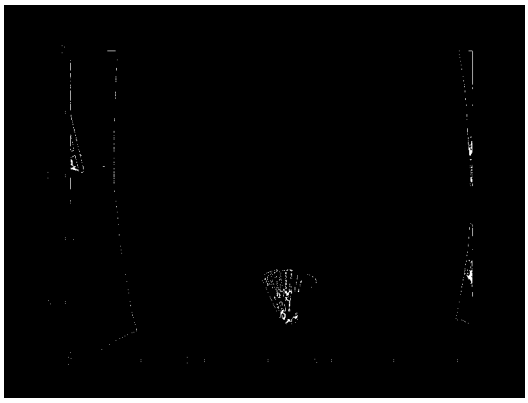

Figure 5.2 Shuttlecock detected using color

The major problem with this method is that colour based detection is not very reliable. The colour of the object might vary due to its reflective characteristics. Also sunlight, floodlights and other factors can cause a hindrance to colour based detection.

3.2.4Area based detection

OpenCV has a built-in function for contouring an image. Apart from the main contouring there were a few smaller aspects involved. They are listed below.

A. Canny edge detection

Canny Edge detection is used to detect the edges in an image. This method was used to filter Kinect's output to show only the edges of the objects detected in the frame. This is generally the precursor step carried out before contouring the image.

Figure 6 Canny edge detection output

B. Writing frames to video

Debugging becomes easier when it is possible to visualize the result of the algorithm used although presenting video real time may overload the system hence affecting overall throughput of the system. Thus, it was considered to store the video and watch it later to see how the program worked. The xvid (fourcc) video codec was used for storing consecutive frames in the form of video which was simplified due to support provided by OpenCV library functions. All videos were recorded at 15fps rate and as stated earlier obtained frames have 640 × 480 resolution.

C. Area/Depth Ratio

During the trials analyzing the shuttlecock trajectory, it was concluded from observations that some kind of relation must exist between the projected area of the shuttlecock and its actual distance from a specific point on the line of reference. On normal observation it could be easily deduced that as the shuttlecock came closer to the point of reference, its projected area also increased. Similarly, as the shuttlecock moved away, its projected area decreased. The line of reference should be perpendicular to the path of the shuttlecock. We can thus state that-

$$Projected\ Area(A) \propto \frac{1}{Distance\ from\ Reference\ point\ (D)}$$

Some readings for the distance and its corresponding projected area were taken. The readings were more or less consistent, but there were a few variations at certain points which couldn't be ignored. So it was concluded that we couldn't entirely rely on this approach, but there would be no harm in adding it as another filter, in tandem with our area filter.

D. Trimming the detection range

The frame needed to be depth sliced to avoid further noise interference. The limits of depth range were decided after a series of experiments. The depth image was initially obtained in mm format. The frame was masked according to the limits set.

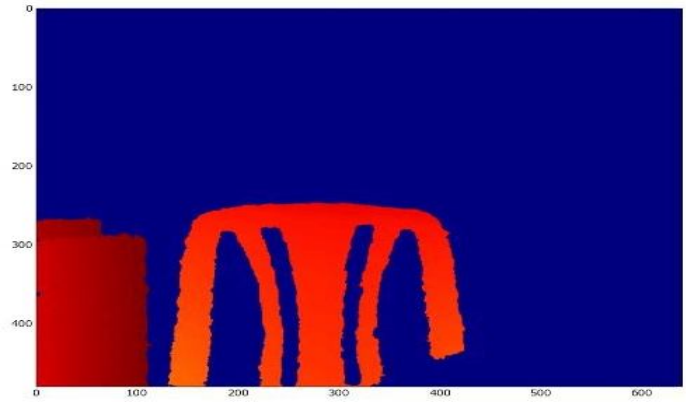

Figure 7 Applying depthslicing

So our final algorithm worked as follows-

i. A frame was captured. All the edges in the frame were detected and contoured. The area enclosed by the contours was calculated by:

area=cv2.contourArea(cnt)

ii. If the area lied within a certain pre-defined range (corresponding to the shuttlecock), then that object would be the shuttlecock ideally. So all the contours satisfying this criterion were displayed using:

iii. cv2.drawContours()

iv. The coordinates and centroid of the contour area, labelled as the shuttlecock, was calculated. For this we used the 'moments' function of OpenCV.

moments =cv2.moments(cnt)

i. Modification was done in the code so that the Z- coordinate given by the 'depth' function in OpenCV would be in mm.

ii. For the X and Y coordinates, we derived certain equations to translate the pixel coordinates into real world coordinates. The accuracy and correctness of the equations was verified using Matplotlib of Python language and by actually measuring the distance and comparing it with the calculated coordinates.

iii. Readings were taken continuously. As soon as 10 readings were obtained, a regression line equation was calculated using the obtained coordinates (regression line negates any variation or deviation in the readings).

iv. The equation of the obtained line was transmitted via Bluetooth

3.3Positioning of Kinect

On analyzing the field conditions and the probable positions of the opponent robots and controllers, it was concluded that the most ideal position for Kinect™ would be at the center line, perpendicular to net, because

i. It would ensure least interference from the opponent team and would not hinder their movements in any way.

ii. The service robot of the opposing team would not come in the frame of detection.

iii. The shuttle would be detected in the first half of the trajectory itself, thus facilitating a faster response from our side.

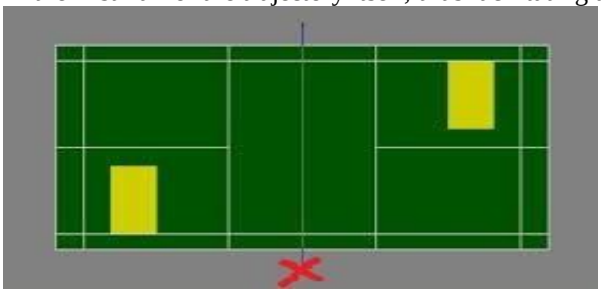

Figure 8 Positioning of Kinect in the arena

3.4Regression Line

In statistics, linear regression is an approach for modelling the relationship between a scalar dependent variable y and one or more explanatory variables (or independent variable) denoted by x. Linear regression consists of finding the best-fitting straight line through the points. The best-fitting line is called a *regression line*.

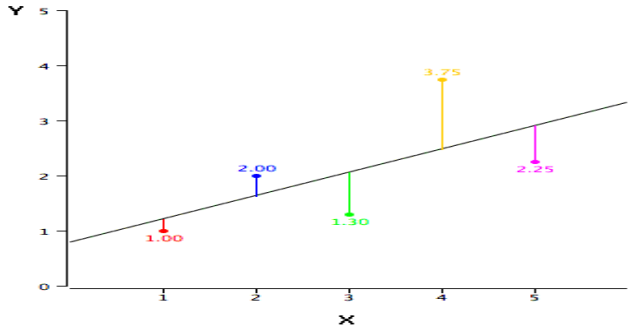

Figure 9 Example of regression line for sample data

The formula for regression line is

$$Y' = bX + A$$

Where Y' is the predicted value, b is the slope and A is the Y intercept.
Slope b can be calculated as

$$b = r\frac{S_y}{S_x}$$

And intercept A can be calculated as

$$A = M_y - bM_x$$

Where,
S_y = Standard deviation ofY
S_x = Standard deviation ofX
M_y = Mean ofY
M_x = Mean ofX
r = correlation between X andY

4.CHALLENGES FACED

4.1Regression Line

The output from Kinect sensor contained a lot of noise. To actually use the relevant data, we had to write a few basic noise filtering algorithms. Depth slicing was used to solve the problem by eliminating certain part of the arena/output which was irrelevant. Apart from that, since the shuttlecock's shape and projected area kept on changing throughout the trajectory, we had to come up with an 'area filter' based algorithm to filter out unnecessary objects from our frame. Though not very accurate, the algorithm worked for us primarily because we wouldn't normally have anything in the frame which would come close to the projected area of the shuttlecock in any orientation, that being our underlying assumption.

4.2 External factors

The infrared sensor of Kinect™ didn't give clear readings in the presence of sunlight. So most of the testing took place at night or indoor. Though the arena was indoor, we had to make sure that the halogen lighting or any other shining surface (reflecting light) wouldn't affect ourreadings.

4.3 Coordinate mapping

ThexandycoordinateswhichwereextractedfromKinect™ were pixel coordinates. We managed to tweak our program so that Kinect™ would give us the real world reading for depthorzcoordinateinmillimeters.Buttranslatingtheother two coordinates to real world coordinates was still a big challenge.Weworkedonitforsometimeandlaterwereable to come up with a normalized equation for the required conversion. On testing we found out that the conversionwas fairly accurate with negligibleerror.

4.4 Hardware limitations

Kinect™ has a 30 fps output rate. Keeping in mind the swift game of badminton, we had to detect the shuttlecock, perform the necessary calculations, transmit the data to the microcontroller on our robot and finally our robot had to

alignitselfaccordingtotherelayedcoordinatesonthearena, all this in a meagre time of 1.5s at the most. This certainly posedabigchallenge,keepinginmindthecomplexityofthe problem statement. Even though our detection was fairly fast, there were certain restrictions on the locomotion of our robot which introduced some time lag in theresponse.

2. RESULTS

We were successfully able to detect the shuttlecock in midair, in the first half of its trajectory and transmit thecoordinates to the robots.

Figure 10.1

Figure 10.2

In figure 10.1 shuttlecock is detected (highlighted) and human shape is ignored. In figure 10.2, the shuttlecock suspended from a thread is detected (while the environment objects are ignored).

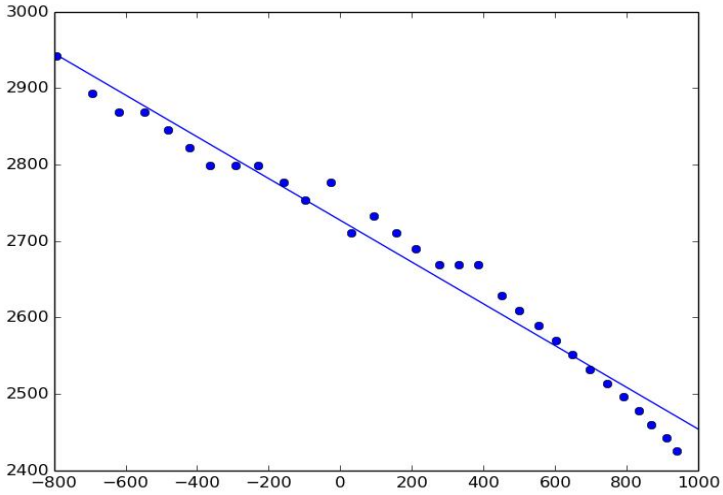

Figure 11 Plotting the regression line (x axis are the realworld Cartesian values and y axis is the depthvalue)

6. CONCLUSION AND FURTHER SCOPE

Even though our objective of shuttlecock detection was achieved, there were some drawbacks in our approach. Also, there is a lot of scope for optimization and enhancement of the algorithm:

i. Use of predictive Kalman filter
ii. Even though Python had a lot of library support and was easier to implement, using a language such as C/C++ would have reduced our computation and processing time.
iii. Area based detection has certain limitations. It could be used only after a few calculated assumptions. This algorithm will fail if some other objects are in the frame with an area comparable to that of a shuttlecock.
iv. Due to the resolution of Kinect and its Shutter speed (30 fps), the detected shuttlecock has a distorted shape.

7. ACKNOWLEDGEMENTS

We would like to extend our heartfelt gratitude to Castalia Labs, Pune, for their guidance, support and valuable inputs which helped us achieve our objective. We also would like to thank our team, without their encouragement and support, this certainly wouldn't have been possible.

REFERENCES

[1]. Z. Zhang, "Microsoft Kinect sensor and its effect", Multimedia, IEEE, vol. 19, no. 2, pp. 4-10, 2012.
[2]. R.A. El-laithy, Jidong HuangYeh, M."Study on the use of Microsoft Kinect for robotics applications", Position Location and Navigation Symposium (PLANS), 2012 IEEE/ION pp. 1280-1288
[3]. Hidehiko Shishido, ItaruKitahara, Yoshinari Kameda, Yuichi Ohta, "A Trajectory Estimation Method for Badminton Shuttlecock Utilizing Motion Blur", Image and Video Technology Lecture Notes in Computer Science Volume 8333, 2014, pp 325-336
[4]. Chen Bingqi, Wang Zhiqiang, "A statistical method for analysis of technical data of a badminton match based on 2-D seriate images", Tsinghua Science and Technology(Volume:12, Issue: 5), pp. 594-601
[5]. Fumito Yoshikawa, Takumi Kobayashi, Kenji Watanabe, and Nobuyuki Otsu, " Automated Service Scene Detection for Badminton Game Analysis Using CHLAC and MRA ", World Academy of Science, Engineering and Technology International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol:4, No:2, 2010
[6]. Li Li, "Application of two-dimensional detection based on the background subtraction in badminton Hawkeye system"
[7]. Siu-Leong Iu, KwangyoenWohn, "Estimation of 3-D Motion and Structure Based on a Temporally-Oriented Approach With the Method of Regression", Technical Reports (CIS), Department of Computer & Information Science, University of Pennsylvania, November 1998
[8]. Rafael Gonzalez and Richard Woods, "Digital Image Processing", 3rd Edition, Pearson Publications
[9]. www.xbox.com/en-IN/Kinect
[10]. www.python.org
[11]. www.ubuntuforums.org
[12]. www.samontab.com

Authors

Aseem Raina is Information Technology engineering student at the Pune Institute of Computer Technology, Pune. His areas of interest include object oriented programming and robotics.

Nilay Mokashi is Electronics and Telecommunication engineering student at the Pune Institute of Computer Technology, Pune. His areas of interest include computer vision, astronomy and robotics.

Pushkar Nimkar is Computer engineering student at the Pune Institute of Computer Technology, Pune. His areas of interest include programming and robotics.

Sanket Gujar is Electronics and Telecommunication engineering student at the Pune Institute of Computer Technology, Pune. His areas of interest include electronic circuits and robotics.