International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:9, No:2, 2015

Stereo Motion Tracking

Yudhajit Datta, Jonathan Bandi, Ankit Sethia, Hamsi Iyer

Abstract—Motion Tracking and Stereo Vision are complicated, albeit well-understood problems in computer vision. Existing softwares that combine the two approaches to perform stereo motion tracking typically employ complicated and computationally expensive procedures. The purpose of this study is to create a simple and effective solution capable of combining the two approaches. The study aims to explore a strategy to combine the two techniques of two-dimensional motion tracking using Kalman Filter; and depth detection of object using Stereo Vision. In conventional approaches objects in the scene of interest are observed using a single camera. However for Stereo Motion Tracking; the scene of interest is observed using video feeds from two calibrated cameras. Using two simultaneous measurements from the two cameras a calculation for the depth of the object from the plane containing the cameras is made. The approach attempts to capture the entire three-dimensional spatial information of each object at the scene and represent it through a software estimator object. In discrete intervals, the estimator tracks object motion in the plane parallel to plane containing cameras and updates the perpendicular distance value of the object from the plane containing the cameras as depth. The ability to efficiently track the motion of objects in three-dimensional space using a simplified approach could prove to be an indispensable tool in a variety of surveillance scenarios. The approach may find application from high security surveillance scenes such as premises of bank vaults, prisons or other detention facilities; to low cost applications in supermarkets and car parking lots.

Keywords—Kalman Filter, Stereo Vision, Motion Tracking, Matlab, Object Tracking, Camera Calibration, Computer Vision System Toolbox.

I. Introduction

THE first step in motion tracking involves mapping the moving objects into a software environment that can process live video-feed in real time. The environment *Matlab* is an excellent platform that meets the requirements of the study. Processing is done using image analysis procedures and tracking results are displayed to the user in the form of *Bounding Boxes* that follow the moving objects in the scene. The implementation is split into three sub-tasks. The first task involves capturing the images from the physical space using two cameras. The intermediary task of *Camera Calibration* is performed next [5]. This step ensures the two cameras are tuned correctly for obtaining accurate results by the depth detection procedure[3][7]. The final task of the implementation involves mapping the live video feeds from the two cameras into the virtual space of *Matlab* as a sequence of RGB images. The RGB images are processed using image analysis procedures and the moving objects in the scene of study are separated out using *Foreground Detection* and *Blob Analysis* procedures[4]. In the final step *Kalman Filter* objects are created, which track the objects in the field of view. The moving objects along with the *Kalman Filter Bounding Boxes* corresponding to each camera are displayed to the user using two video-screens.

Y. Datta is a student at the Department of Electrical and Electronics Engineering, National Institute of Technology, Calicut, Kerala, 673601 India (e-mail: yudhajit.datta@gmail.com).
J. Bandi, A. Sethia and H. Iyer are students at NIT Calicut.

II. Theory

A. Kalman Filter

The *Kalman Filter*, also known as a linear quadratic estimation, is an algorithm that uses a series of measurements observed over time, containing noise and other inaccuracies and produces estimates of unknown variables[1]. The *Kalman Filter* addresses the general problem of trying to estimate the state $x_k \epsilon R^n$ of a discrete-time controlled process that is governed by the linear stochastic difference equation:

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \tag{1}$$

With a measurement $z_k \epsilon R^m$ that is:

$$z_k = Hx_k + v_K \tag{2}$$

The random variables w_k and v_k represent the *process noise* and {textitmeasurement noise respectively:

$$p(w) = N(0, Q) \tag{3}$$

$$p(v) = N(0, R) \tag{4}$$

We define $x_k^{\wedge -} \epsilon R^n$ to be our *a priori state estimate* at step k given knowledge of the process prior to step k, and $x_k^{\wedge} \epsilon R^n$ to be our *a posteriori state estimate* at step k given measurement. We can then define *a priori estimate error* and *a posteriori estimate error* as:

$$e_k^- = x_k - x_k^{\wedge -} \tag{5}$$

$$e_k = x_k - x_k^{\wedge} \tag{6}$$

The *a priori estimate error covariance* and *a posteriori estimate error covariance* are then:

$$P_k^- = E[e_k^- {e_k^-}^T] \tag{7}$$

$$P_k = E[e_k {e_k}^T] \tag{8}$$

In deriving the equations for the *Kalman filter*, we begin with the goal of finding an equation that computes an *a posteriori state estimate* as a linear combination of an *a*

TABLE I
OPERATION OF KALMAN FILTER

Symbol	Parameter	Unit
x_k	State vector at time step k	n x 1
u_k	Input vector at time step k	l x 1
z_k	Measurement vector at time step k	m x 1
w_k	Process Noise	n x 1
v_k	Measurement Noise	m x 1
A	Characteristic Matrix: relates the state x_k at the previous time step $k-1$ to the state at the current step k. Here we assume it is constant	n x n
B	Input Matrix : relates the optional control input u_{k-1} to the state x_k	n x l
H	Output Matrix : relates the measurement z_k to the state x_k . Here we assume it is constant	m x n
$x_k^{\wedge -}$	a priori state estimate at time step k	n x 1
$x_k^{\wedge}$	a posteriori state at time step k	n x 1
e_k^-	a priori estimate error at time step k	n x 1
e_k	a posteriori estimate error at time step k	n x 1
P_k^-	a priori estimate error covariance at time step k	n x n
P_k	a posteriori estimate error covariance at time step k	n x n
$Hx_k^{\wedge}$	Measurement prediction at time step k	m x 1
K	Gain Matrix	n x m

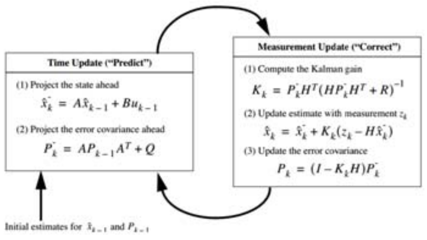

Time Update ("Predict")
(1) Project the state ahead
$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$
(2) Project the error covariance ahead
$P_k^- = AP_{k-1}A^T + Q$
Measurement Update ("Correct")
(1) Compute the Kalman gain
$K_k = P_k^- H^T(HP_k^- H^T + R)^{-1}$
(2) Update estimate with measurement z_k
$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$
(3) Update the error covariance
$P_k = (I - K_k H)P_k^-$
Initial estimates for $\hat{x}_{k-1}$ and P_{k-1}

Fig. 1. Equations of Kalman Filter

priori estimate and a weighted difference between an actual *measurement* and a *measurement prediction* $Hx_k^{\wedge}$.

$$x_k = x_k + K(z_k - Hx_k^{\wedge}) \tag{9}$$

The matrix K is chosen to be the gain that minimizes (7). One form of the resulting K that minimizes it is given by:

$$K_k = P_k^- H^T(HP_k^- H^T + R)^{-1} \tag{10}$$

The *Kalman Filter* estimates a process by using a form of feedback control: the filter estimates the process state at some time and then obtains feedback in the form of measurements which are inherently noisy. As such, the equations for the *Kalman Filter* fall into two groups: time update equations and measurement update equations. The time update equations are responsible for projecting forward in time the current textitstate and *error covariance estimates* to obtain the *a priori estimates* for the next time step [2]. The measurement update equations are responsible for the feedback; i.e. for incorporating a new measurement into the *a priori estimate* to obtain an improved *a posteriori estimate*. The time update equations can also be thought of as predictor equations, while the measurement update equations can be thought of as corrector equations [11].

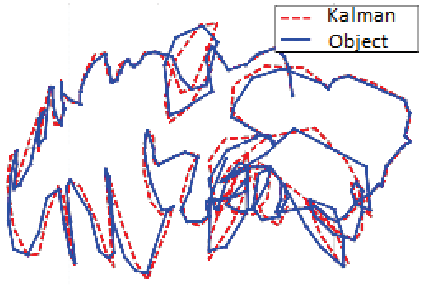


Fig. 2. The result of a Kalman Filter Object tracking a point object in two dimensional space

B. Stereo Vision

Stereo Vision is the recovery of the 3D structure of a scene using two or more images of the 3D scene, each acquired from a different viewpoint in space [4]. An illustration of a visual system that applies the principle of stereo vision with great accuracy and reliability is the human visual system. The perception of depth is a result of stereo calculations performed by the brain on the similar but slightly different simultaneous images of the same object projected on the two eyes [6], as can be observed from Fig. 3. The perceived depth depends on how similar or dissimilar the two stereo images of the object are to the two capturing devices. Although stereo calculations are possible for any mutual orientation of the two capturing devices, they are enormously simplified for the configuration where the optical axes of the two are aligned to be mutually parallel. As a result, the cameras in this study have been arranged in the special mutual configuration wherein there optical axes are aligned parallel to each other[4].Consider the following configuration of two identical cameras shown in Fig. 4. Let the two camera center's be *Camera l* and *Camera r* corresponding to the left and right cameras respectively. Let the *optical center's* of the left and right cameras then be O_l and O_r. T is the distance between the two optical center's O_l and O_r. Z is the distance of the *object point* along the line perpendicular to the line joining the *optical center's* of the two cameras. Let a point of interest on the object be taken as the *object point* P. If the image of the object point on the screen of *Camera l* is represented as P_l and that on screen of *Camera r* is represented as Pr, then the *image distances* X_l and X_r are the distances of the image points from the *optical axes* of the two cameras. *Image offset* d is found as the difference between the image distances X_l and X_r. Each of the Image points can be thought of as forming a triangle with the *optical centre* and *screen* of its respective camera. Let us refer to these as the *image triangles*. In a similar manner the *object point* can be thought of as forming a triangle with each of the cameras. Let these be the *object triangles* [4][10]. From applying the property of similar triangles on the *object triangles* and *image triangles* on either camera, following relation between the *object distance, focus, image offset* between the two cameras, and distance between *optical center's* of the two cameras is obtained as:

$$Z = Tf/d \tag{11}$$

TABLE II
PRINCIPLE OF STEREO VISION

Symbol	Parameter	Unit
P	Object Point	Nil
P_l	Image point l : image of Object Point in Camera l	Nil
P_r	Image Point r :image of Object point in Camera r	Nil
O_l	Optical Centre of Camera l	Nil
O_r	Optical Centre of Camera r	Nil
T	distance between the Optical Centres of the two cameras	cm
Z	distance of Object Point from line joining Optical Centres of the two cameras, i.e. the object distance	cm
f	Foci of the two cameras, i.e. the image distance	Nil
X_l	distance of Image Point l from optical axis ofCamera l	cm
X_r	distance of Image Point r from optical axis of Camera r	cm
d	Image Offset : relative distance between the Image Points in the two cameras	cm

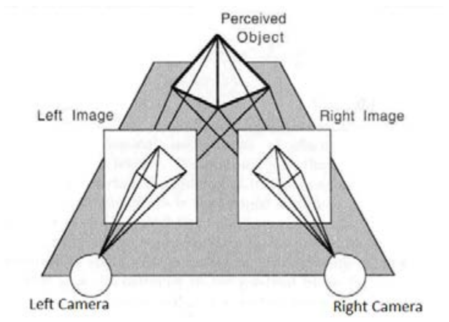


Fig. 3. The result of a Kalman Filter Object tracking a point object in two dimensional space

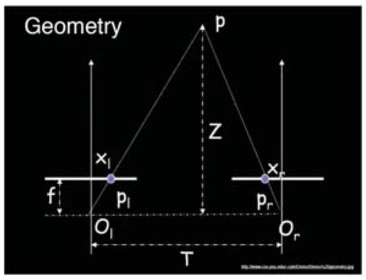


Fig. 4. The result of a Kalman Filter Object tracking a point object in two dimensional space

where $d = Xl - Xr$

III. FLOW OF VARIABLES

Two identical video cameras having their optical axes aligned parallelly are fixed on a common surface and are used to monitor the scene of interest. The two cameras are initially subjected to a one time *Camera Calibration* procedure[5]. First the *Camera Calibration* procedure captures a series of simultaneous stereo images by calling a procedure *Cams.m*. These stereo images are then passed to the procedure *Calibrate.m* and used to stereo-calibrate the two cameras [9][11]. The calibration results are stored in the main tracking function and are used as static values for the rest of the program schedule. In the main tracking function *MultiObjectTrackingMultiCam.m*, the video feeds from the cameras are taken as inputs for two Kalman Filter objects that each produce as an output an estimation of the motion of the objects in the scene at any instant[7][8]. The estimator is visible to the user in the form of a bounding box that tracks the object motion. The centroid of this bounding box is assumed to be a working estimate of the centre of the object being tracked, and the principles of *Stereo Vision* are applied to the centroid attribute of the bounding box [7][11]. Thus, simultaneous measurements from the two estimators tracking the object are used to calculate the depth of the object from the camera screens using *Stereo Vision*. The procedure is repeated iteratively for the desired period so that during the period the *Kalman Filter* objects constantly keep updating their estimations based on the motion of the objects and *Stereo Vision* returns the depth values corresponding to the estimations. In principle, an accurate detection of the centre of an object is a reasonable approximation of the location of the entire object in all scenarios where the dimensions of the object are small compared to the distance of the object centre from the screen [11]. It is interesting to note how the accuracy of the *Kalman Filter* objects in estimating the motion of the objects is crucial for accurate stereo detection.

IV. CONTROL FLOW IN MATLAB

The code performs automatic detection and motion-based tracking of moving objects on videos from stationary cameras. The implementation of motion-based object tracking consists essentially of two parts; detecting moving objects in each frame and associating the detections corresponding to the same object over time. The detection of moving objects is done by performing a background subtraction algorithm based on *Gaussian Mixture Models*. Morphological operations are applied to the resulting Foreground Mask to eliminate noise. Finally, *Blob Analysis* detects groups of connected pixels, which are likely to correspond to moving objects. The association of detections to the same object is based solely on motion [9]. A *track* is an attribute associated with a *Kalman Filter* object that helps the user uniquely identify the reference between the physical object detection and the software object tracking it. The motion of each track is thus estimated by a corresponding *Kalman Filter* which is used to predict the track's location in each frame, which in turn helps determine the likelihood of an object detection being assigned to that track. *Track* maintenance is an important aspect of the schedule. In any given frame, some detections may be assigned to tracks, while other detections and tracks may remain unassigned. The assignment is done on the basis of a *Hungarian Cost Assignment Matrix* that aims to minimize the sum of costs of assigning each detection to a certain track and not assigning it to the rest . The assigned tracks are updated using the corresponding detections. The unassigned tracks are marked invisible. An unassigned detection begins a new track. Each track keeps count of the number of consecutive *frames* where it remained unassigned. If the count exceeds a specified threshold, the schedule assumes that the object left the field of view and it deletes the track [9].

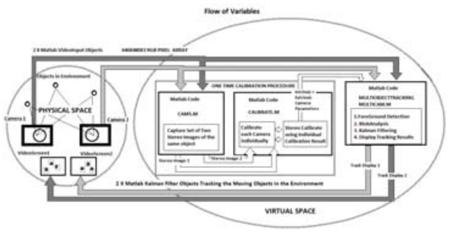


Fig. 5. Flow of variables

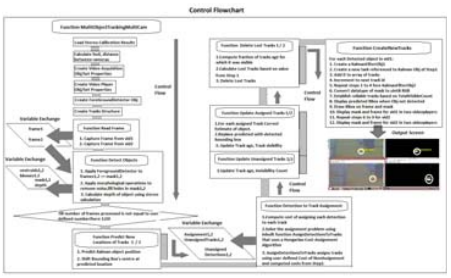


Fig. 6. Variable Exchange in Matlab

V. Results

A. Outputs of the Program

Corresponding to each *videoinput*, two video players display the video stream; one named *Videoplayer* displays the stream as it is and the other named *Maskplayer* displays it in binarized format where well-connected objects in motion are seen to be white while the stationary background is seen to be black. The *Videoplayer* and *Maskplayer* screens from one camera are placed side by side to juxtapose the real scene and the foreground detected binarized scene. This serves to illustrate the differences in human perception and machine vision while inspecting the same scene of interest. The output from a *Kalman Filter* object corresponding to each input is presented as a *Bounding Box* that tracks the motion of the object in the *Videoplayer* as well as *Maskplayer*. The *Bounding Box* contains information regarding the *Track Id* of the detection track. Lastly, the screens corresponding to each input are stacked on top of one another allowing for a comparison of the stereo images of the scene at any instant as captured by the two cameras. There are two variations of the implementation: For *single object detections*, the depth of the object is projected on the output of one set of bounding boxes while the other set contains the track number. For *multiple object detections*, the depths of the individual objects are printed at the workspace of Matlab.

B. Results from Experiments

The code was tested in a diverse test conditions and succeeded in tracking objects in a variety of test environments. The convergence of the 2-dimensional tracking algorithm was found to be quite successful even in challenging environment conditions like poor lighting, background disturbances, and disconnected objects. The depth detection measurement for the single object variation of the code was found to be extremely

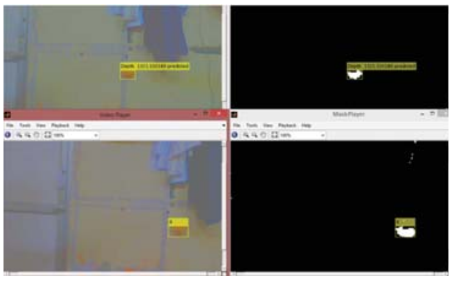


Fig. 7. Matlab Window showing Video Display Screen for pendulum oscillation where object detection is missing and object location is predicted. Here the upper panes display the depth value of the predicted object while lower panes display track id of Kalman object tracking the predicted object

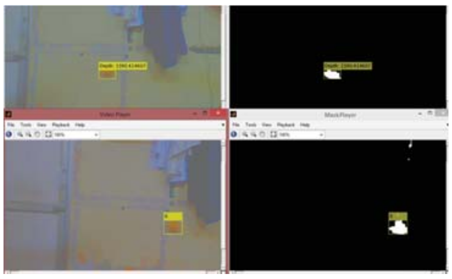


Fig. 8. Matlab Window showing Video Display Screen for pendulum oscillation where object detection is present and thus object location is detected. Here the right panes display the scene of interest in a Foreground Masked binarized format while the left panes display the scene as captured by the two Cameras

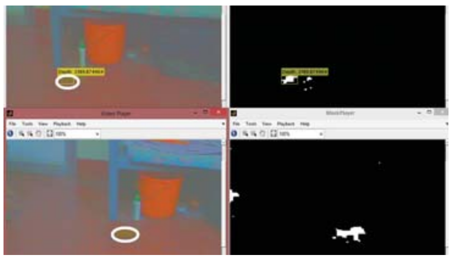


Fig. 9. Matlab Window showing Video Display Screen for a neon colored Ball bouncing past the field of view. Here object detection is present despite quick motion and detected depth converges to accurate value despite constraint of time and visibility. Note: The White Outline in the left panes has been added externally to mark out the location of object in the field of view

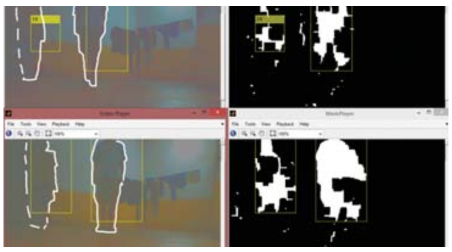


Fig. 10. Matlab Window showing Video Display Screen for multiple moving figures. Object detections are present meaning objects are detected. Extremely accurate 2-D tracking despite the presence of severe constraints such as disconnected objects and poor lighting conditions. Note: The White Outline in the left panes has been added externally to mark out the location of object in the field of view

accurate where predicted results from stereo calculations fell within 90 - 95 % of the actual depths. In the case of a single object scenario in well lit condition, the *Kalman filter* object was found to converge onto the moving object with greater than 90 % accuracy. The depth detection schedule for multiple objects was found to be relatively inaccurate, owing to ambiguity in *Detection to Track Assignment* procedure of the depth detection schedule.

VI. Conclusion

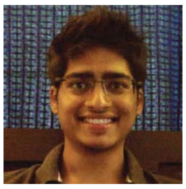
The kernel of the program was found to be solid and dependable. There is scope for making changes to the program to make it more robust and efficient. To achieve accurate depth assignment to objects in case of multi-object tracking, Modifications may be made to the program to make the left and right Kalman filter objects work more synchronously in detecting the same objects in the field of view. This would involve incorporating a knowledge system that would compare the tracked objects in both the filters to ascertain whether or not they are the same physical object in the field of view.

Acknowledgment

The authors would like to thank Dr. Jagadanand G, Assistant Professor, EED, NIT Calicut; for his guidance, encouragement and support through the duration of the study. The authors acknowledge their gratitude towards Matlab Computer Vision System Toolbox Development team. The solutions of this toolbox have been extensively used as the basis for much of the development and implementation of the project.

References

[1] Benjamin C. Kuo, Automatic Control Systems, 3rd edition, New Jersey: Prentice Hall, Inc., 2007, pp. 588-594.
[2] Simon S. Haykin, Neural Networks: A Comprehensive Foundation, 3rd ed., Pearson Education Asia, 1999, pp. 762-765.
[3] R. C. Gonzalez, R. E. Woods, S. L Eddins, Digital image Processing using Matlab: Pearson Education, 2004, (Chap 10)pp. 379 407, (Chap 12) pp. 484 498.
[4] R. Jain, R. Kasturi, B. G. Schunck, Machine Vision, Singapore: McGraw-Hill Book Co., 1995, (Chap -15) pp. 459 481, (Chap -11) pp. 289 305, (Chap -12) pp. 309 357.
[5] www.vision.caltech.edu/bouguetj/calib_doc/htmls/parameters.html (as of 21-07-13).
[6] stemkoski.github.io/Three.js/ (as of 21-07-13).
[7] www.mathworks.in/help/vision/ref/assigndetectionstotracks.html (as of 21-07-13).
[8] www.mathworks.in/help/matlab/ref/avifile.html (as of 21-07-13).
[9] www.mathworks.in/help/imaq/controlling-logging-parameters.html (as of 21-07-13).
[10] www.mathworks.in/help/vision/ref/epipolarline.html (as of 21-07-13).
[11] G. Welch, G. Bishop, An Introduction to the Kalman Filter, University of North Carolina at Chapel Hill, ACM :1995.



Yudhajit Datta Yudhajit Datta was born in Kolkata, India, in 1993. He is currently pursuing the final year of his B.Tech degree in Electrical and Electronics Engineering from National Institute of Technology Calicut, India. He has a keen interest in the applications of computer engineering. His research interests include artificial intelligence, robotics and computer vision.

Jonathan Bandi Jonathan Bandi is currently pursuing the final year of his B.Tech degree in Electrical and Electronics Engineering from National Institute of Technology Calicut, India.

Ankit Sethia Ankit Sethia is currently pursuing the final year of his B.Tech degree in Electrical and Electronics Engineering from National Institute of Technology Calicut, India.

Hamsi Iyer Hamsi Iyer is currently pursuing the final year of her B.Tech degree in Electrical and Electronics Engineering from National Institute of Technology Calicut, India.