ISSN 2249-6343
International Journal of Computer Technology and Electronics Engineering (IJCTEE)
Volume 3, Special Issue, March-April 2013, An ISO 9001: 2008 Certified Journal.
E-NSPIRE, A National Level Conference held at Pravara Rural Engineering College, Loni, Maharashtra, INDIA.

Autonomous Tracking of Person Using Particle Filter

S. V. Arote, Gayatri J. Barhate, Archana A. Dhanwate, Nikita A. Soni

Abstract—In this project, we discuss the design of Particle filter algorithm to track the given target. It specifically shows the resulting improvement in tracking. The proposed architecture is suitable for indoors and outdoors scenes with static background. It also overcomes the problem of stationary targets fading into the background, which is commonly found in variations of Stauffer's adaptive background algorithm. The project uses Baumberg and Hogg method, which uses a median filter at each pixel to construct a background model. A thresholded absolute image difference is used to identify foreground pixels. The approach to modeling background is to consider the first frame to be the background, and to subtract intensities pixel by pixel in all successors. Non-zero differences then represent movement. The proposed algorithm is more robust in tracking performance is of utmost importance for high-performance real-time applications. It is of great help for applications such as Video Surveillance, Gesture recognition, State estimation of Rovers in outer-space, Traffic monitoring etc. The work may further be useful for Automatic initialization of reference window, Multi part color window, Multi-object tracking, etc.

***Keywords*- Estimation, Gesture, Robust, Utmost.**

I. INTRODUCTION

Target tracking is one of the most important applications of sequential state estimation. It has received significant attention during the past several years due to its crucial value in visual applications. In a tracking scenario, an object can be defined as anything that is of interest for further analysis. For instance, boats on the sea, fish inside an aquarium, vehicles on a road, planes in the air, people walking on a road, or bubbles in the water are a set of objects that may be important to track in a specific domain .Target tracking is an important task within the field of computer vision. The proliferation of high-powered computers, the availability of high quality and inexpensive video cameras, and the increasing need for automated video analysis has generated a great deal of interest in object tracking algorithms. Systems developed for such tasks as people tracking face tracking and vehicle tracking have come in many shapes or size. Real-time object tracking is the critical task in many computer vision applications such as surveillance perceptual user interfaces augmented reality , smart rooms , object-based video compression , and driver assistance. Motion-based recognition (i.e. human identification based on gait, automatic object detection, etc ;).

- Automated surveillance (i.e. monitoring a scene to detect suspicious activities or unlikely events ;).
- Video indexing,(i.e. automatic annotation and retrieval of the videos in multimedia databases;).
- Human-computer interaction, (i.e. gesture recognition, eye gaze tracking for data input to computers, etc. ;).
- Traffic monitoring (i.e. real-time gathering of traffic statistics to direct traffic flow).
- Vehicle navigation (i.e. video-based path planning and obstacle avoidance capabilities.)

II. TARGET TRACKING AND THE ASSOCIATED PROBLEMS

Target tracking can be described as the process of determining the location of a target feature in an image sequence over time. Examples include tracking cars in an intersection via a traffic camera, or tracking the head of a computer user with a web-cam. The target to be tracked might be a complete object (e.g. a person) or a small area on an object (e.g. a corner). In either case, the feature of interest is typically contained within a target region. Ideally, a tracking algorithm would be able to locate the object anywhere within the image at any point in time. However typically only a limited region of the image is searched (usually the same size as the target region). Reasons for this are efficiency (especially necessary for real-time applications) and the fact that there might be many other similar-looking objects in the image. The intuitive approach is to search within a region centered around the last position of the object. But as Figure 1 illustrates, this approach will fail if the object moves outside the target range.

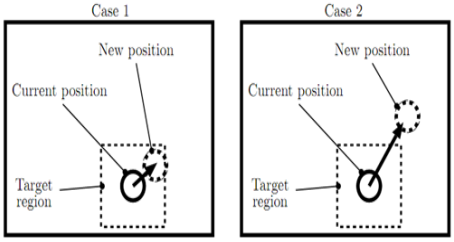


Fig. 1: Case 1: Tracking the Object without Position Prediction Might Be Successful
Case 2: Tracking without position prediction will fail.

ISSN 2249-6343
International Journal of Computer Technology and Electronics Engineering (IJCTEE)
Volume 3, Special Issue, March-April 2013, An ISO 9001: 2008 Certified Journal.
E-NSPIRE, A National Level Conference held at Pravara Rural Engineering College, Loni, Maharashtra, INDIA.

There are many possible reasons why the object might not stay within this region:

- The object is moving too fast.
- The frame rate is too low.

The searched region is these problems are related to each other and could be avoided by ensuring a high enough frame rate for example. But given other constraints, these problems are often unavoidable.

Tracking objects can be complex due to:

- Loss of information caused by projection of the 3D world on a 2D image,
- Noise in images,
- Complex object motion,
- Nonrigid or articulated nature of objects,
- Partial and full object occlusions,
- Complex object shapes,
- Scene illumination changes, and
- Real-time processing requirements.

To solve the first problem, we can attempt predicting the location of the target, and searching in a region centered around that location. But in making the prediction, it is necessary to consider the second problem as well. The previously obtained location measurements are likely not accurate, so the prediction method needs to be robust enough to handle this source of error. This report will discuss how these problems can be addressed with the Kalman filter, and studies how well they are solved.

In summary, two major problems have been identified:

- The object can only be tracked if it does not move beyond the searched region.

Various factors such as lighting and occlusions can affect the appearance of the target, thus making accurate tracking difficult. III. Algorithm for Background Modelling

Background maintenance by median filtering

1. Initialize: Acquire K frames. At each pixel, determine the median intensity of these K. If the image sequence is in colour, this is done for each of the R, G, B streams. The median represents the current background value.
2. Acquire frame K+1. Compute the difference between this frame and the current background at each pixel(a scalar for gray image; a vector for RGB)
3. Threshold this difference to remove/reduce noise. Simple thresholds may be too crude at this point and there is scope for using, e.g., hysteresis
4. Use some combination of blurring and morphological operations to remove very small regions in the difference image, and to fill in 'holes' etc. in larger ones. Surviving regions represent the moving objects in the scene.
5. Update the median measurement, incorporating frame K+1 and discarding the current earliest measurement.
6. Return to (2) for the next frame.

Step 5 here is potentially very expensive, demanding the storage of the last K images and a per-pixel sort at each frame. An efficient short cut is available : only the current background is stored, and intensities are incremented by one if the current frame is brighter at the pixel, and decremented if it is darker. This trick converges the background onto an intensity for which half the updates are brighter and half darker- that is, the median.

III. OUR APPROACH

As we are tracking Target assuming a stationary camera position and constant illumination, we will use Baumberg and Hogg approach to construct a background model.. We chose to use a simple median filter to remove non-hand components of the binary image since once again; our background was not too complicated. The approach to modeling background is to consider the first frame to be the background, and to subtract intensities pixel by pixel in all successors. Non-zero differences then represent movement (classified as part of a foreground object).These may be thresholded (with varying degrees of sophistication) and grouped (perhaps using morphological approaches) to derive the blobs that will be the objects moving in the scene.Such an approach detects every single movement, howsoever small-this would include wind shaking trees, and the tiniest camera movement. More subtly, changes in lightening of any degree would have catastrophic effects. We would only take this very simple approach in conditions under very strict control; an absolutely rigid camera with no environmental variation in lightening or shadow, and minimal or absent moving 'clutter'. These constraints obviously exclude most scenes of any

IV. TARGET TRACKING METHODS

A.Kalman Filter

- **Definition**

Within the significant toolbox of mathematical tools that can be used for stochastic estimation from noisy sensor measurements, one of the most well known and often-used tools is what is known as the Kalman Filter.

Working of Kalman Filter

The time/measurement form of the Kalman Filter is executed in two steps.

ISSN 2249-6343
International Journal of Computer Technology and Electronics Engineering (IJCTEE)
Volume 3, Special Issue, March-April 2013, An ISO 9001: 2008 Certified Journal.
E-NSPIRE, A National Level Conference held at Pravara Rural Engineering College, Loni, Maharashtra, INDIA.

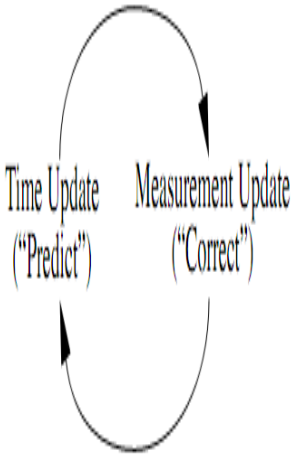


Fig. 2: The Ongoing Discrete Kalman Filter Cycle

1. The time update:

The time update projects the state variable vector estimate ahead in time taking into account the input into the system (that is, it makes a prediction of the new state). The predict phase uses the state estimate from the previous time step to produce an estimate of the state at the current time step. This predicted state estimate is also known as the a priori state estimate because, although it is an estimate of the state at the current time step, it does not include observation information from the current time step.

2. The measurement update:

The measurement update adjusts the time update state estimate to take into account measurements made during the time interval. In the update phase, the current a priori prediction is combined with current observation information to refine the state estimate. This improved estimate is termed the a posteriori state 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. Indeed the final estimation algorithm resembles that of a predictor-corrector algorithm for solving numerical problems as shown below in Figure.

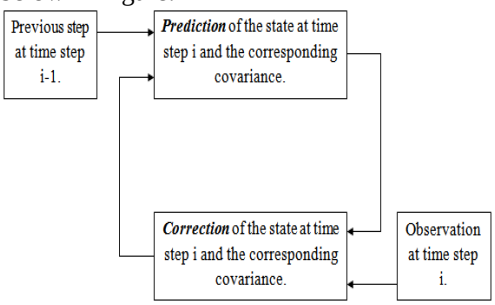
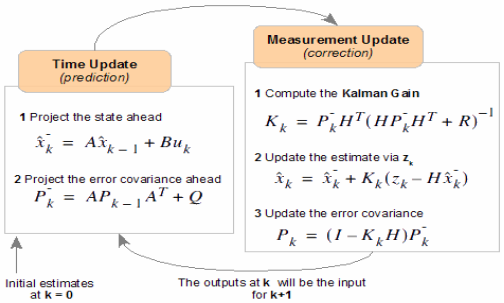


Fig.3: Kalman Filter Cycle

In the prediction step, the time update is taken where the one-step ahead prediction of observation is calculated (i.e. The time update projects the current state estimate ahead in time);

In the correction step, the measurement update is taken where the correction to the estimate of current state is calculated (i.e. The measurement update adjusts the projected estimate by an actual measurement at that time).



Step 1: The Time Update Equations:

The time update equations are equations (1) and (2) Equation 1 calculates what is called the "a priori" estimate of the state variable vector - an estimate of X at time k given measurements up to time k-1. This updates the state estimate based on the knowledge of the previous time state and the input to the system since the last time update. Equation 2 updates the "a priori" error covariance matrix P. This update represents the fact that knowledge of the system gradually decays over time due to errors in the assumed input or in the model which contribute to the uncertainty in the estimate.

Step 2: Measurement Update equations:

The measurement update equations are equations (1), (2) and (3).The time update state estimate in the previous step is corrected based on the feedback from the sensor measurements that have been taken during the time interval. Equation 1 calculates the Kalman gain J to minimize the a posteriori error covariance matrix P. Equation 2 calculates the "a posteriori" estimate of the state variable vector - an estimate of X at time k given measurement data up to time k. This updates the state variable estimate for the new measurement. It compares the actual sensor measurement (Y) with the sensor measurement predicted from the state estimate (CX) and uses the difference between these values (called the "innovation" or "residual") and the Kalman gain to determine the amount by which to adjust the time update state estimate. Equation 3 calculates the "a posteriori" error covariance matrix P. These equations are repeated at every time interval. If the Kalman Filter does not receive any measurement information it performs the time update (Equations (1) and (2)) and no measurement update. In essence a Kalman filter will (vaguely):

1. Estimate the state (from previous)
2. Determine the error
3. Predict the next state
4. Predict the next error
5. Measure the system
6. Update the state calculations
7. Update the error calculations

ISSN 2249-6343
International Journal of Computer Technology and Electronics Engineering (IJCTEE)
Volume 3, Special Issue, March-April 2013, An ISO 9001: 2008 Certified Journal.
E-NSPIRE, A National Level Conference held at Pravara Rural Engineering College, Loni, Maharashtra, INDIA.

Below are the equations that represent this process, followed by the definitions of each symbol:

x_temp_est = A * x_est(last) + B * u_control_input

P_temp = A * P(last) * A^T + Q

K = P_temp * H^T * (H* P_temp * H^T + R)^-1

x_est = x_temp_est + K * (z_measured - H * x_temp_est)

P = (I - K*H) * P_temp

Where,

x - is the state we want to estimate (e.g. position)

u - is the input to the control system (if it exists) (e.g. a force)

A- State (model) matrix, how to get from last state, to next state (e.g. : position+= velocity *dt). (Some people (eg: engineers) use an alternative notation 'F' for this matrix)

B - is the input matrix (Some people use an alternative notation 'G' for this matrix)

H - is the measurement matrix (what are we measuring?) e.g.: position or velocity, etc.

Q - is the process noise covariance matrix (you just 'know' this)

R - is the measurement noise covariance matrix (likewise, you 'know' this, usually from measurements, data sheets, etc, ie: how noisy are your sensors?)

P - The estimate error covariance matrix (calculated)

K - The kalman gain (calculated)

V. FEW IMPORTANT CONCEPTS

- Filtering refers to estimation of the "present" state vector, based upon all "past" measurements.
- Prediction refers to is the estimation of the state vector at a " future" time.
- Smoothing refers to estimation of the state vector at some time in the "past" based on all measurements taken up to the "present" time

Particle Filter

Since their introduction in 1993, particle filters have become a very popular class of numerical methods for the solution of optimal estimation problems in non-linear non-Gaussian scenarios. In comparison with standard approximation methods, such as the popular Extended Kalman Filter, the principal advantage of particle methods is that they do not rely on any local linearization technique or any crude functional approximation. The price that must be paid for this exibility is computational: these methods are computationally expensive. However, thanks to the availability of ever increasing computational power, these methods are already used in real-time applications appearing in fields as diverse as chemical engineering, computer vision, financial econometrics, target tracking and robotics. Moreover, even in scenarios in which there are no real-time constraints, these methods can be a powerful alternative to Markov chain Monte Carlo (MCMC) algorithms | alternatively, they can be used to design very efficient MCMC schemes. Kalman filters, they're a great way to track the state of a dynamic system for which we have a Bayesian model. That means that if we have a model of how the system changes in time, possibly in response to inputs, and a model of what observations we should see in particular states, we can use particle filters to track our belief state. Well, the main reason is that for a lot of large or high-dimensional problems, particle filters are tractable whereas Kalman filters are not. The key idea is that a lot of methods, like Kalman filters, try to make problems more tractable by using a simplified version of your full, complex model. Then they can find an exact solution using that simplified model. But sometimes that exact solution is still computationally expensive to calculate, and sometimes a simplified model just isn't good enough. So then we need something like particle filters, which let you use the full, complex model, but just find an approximate solution instead. First, we can easily adjust the amount of computation that we need to do. By increasing or decreasing the number of particles in the system, we can trade off the accuracy of our estimates for faster results. If we reduce the number of particles, we have less work to do but our approximation will not be as close to the correct answer. In fact, we can even adjust the number of particles on-the fly as conditions permit. For example, a rover may be able to change the frequency of its processor depending on the amount of power that's available. Then, when a lot of light is falling on its solar panels, it can use a lot of particles and get an accurate approximation and when it's cloudy it can scale back to fewer particles and reduce its accuracy.

VI. PARTICLE FILTER OBJECT TRACKING

Visual object tracking is a difficult problem, but in recent years, particle filter-based object trackers have proven to be very effective. Conceptually, a particle filter-based tracker maintains a probability distribution over the state (location, scale, etc.) of the object being tracked. Particle filters represent this distribution as a set of weighted samples, or particles. Each particle represents a possible instantiation of the state of the object. In other words, each particle is a guess representing one possible location of the object being tracked. The set of particles contains more weight at locations where the object being tracked is more likely to be. This weighted distribution is propagated through time using a set of equations known as the Bayesian filtering equations, and we can determine the trajectory of the tracked object by taking the particle with the highest weight or the weighted mean of the particle set at each time step. The particle filter tracker available here is a simple single-object tracker that uses a color histogram-based observation model and a second-order autoregressive dynamical model. It is implemented in C using Open CV and the GSL.

ISSN 2249-6343
International Journal of Computer Technology and Electronics Engineering (IJCTEE)
Volume 3, Special Issue, March-April 2013, An ISO 9001: 2008 Certified Journal.
E-NSPIRE, A National Level Conference held at Pravara Rural Engineering College, Loni, Maharashtra, INDIA.

VII. EXPERIMENT & RESULT

During the explanation of the model we have shown examples of its performance. Abroader experimentation has been done to test tracking of person under different velocity and different distances from the vision system. Person tracking result are shown below.

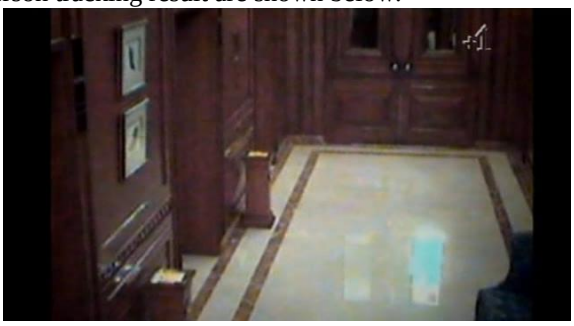


Fig 4: Background Image

Fig 5: Image of Person 1

Fig 6: Tracked image of Person 1

Fig 7: Image of Person 2

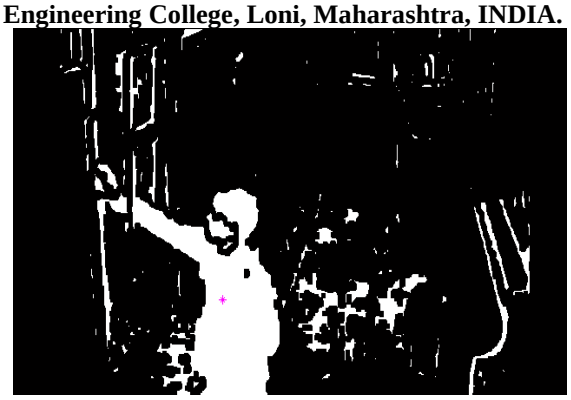


Fig 8: Tracked Image of Person 2

VIII. CONCLUSION

In this paper, we present a particle filtering technique for target tracking using a linear state model and a non-linear measurement model with additive white Gaussian noise. The results suggest that particle filtering technique performs much better than Kalman filtering technique. Future work includes applying particle filtering technique to a non-linear state model and by incorporating non-Gaussian noise. The tracking solution presented in this paper has several desirable properties: it is efficient, modular, has straightforward implementation, and provides superior performance on most image sequences. Acknowledgement "The probability of any event is the ratio between the value at which an expectation depending on the happening of the event ought to be computed, and the value of the thing expected upon its happening."

IX. FUTURE SCOPE

1. Automatic initialization of reference window.
2. Multi part color window.
3. Multi-object tracking.

REFRENCES

[1] Alper Yilmaz, Ohio State University, and Omar Javed's, Object Tracking: A Survey Object Video, Inc. and Mubarak Shah, University of Central Florida.

[2] A. McIvor's, "Background Subtraction Techniques".

[3] M.S. Grewal and A.P. Andrews, "Kalman Filtering: Theory and Practice". Prentice Hall, 1993.

[4] Greg Welch, Gary Bishop's "An Introduction to the Kalman Filter", University of North Carolina at Chapel Hill, Department of Computer Science ,Chapel Hill, NC 27599-3175.

[5] Stephen Rohr, Richard Lind, Robert Myers, William Bauson, Walter Kosiak, and Huan Yen's, "An integrated approach to automotive safety systems". Automotive engineering international, September 2000.

[6] Martin A. Salzmann's, "Some aspects of Kalman Filtering", Department of Geodesy and Geometrics Engineering University of New Brunswick, Fredericton, N.B. Canada E3B 5A3, August 1988,Latest Reprinting February 1996.

[7] C.T. Leondes, Editor, NATO AGARD "Theory and Applications of Kalman Filtering" Chapter 10-Brock, L.D., Schmidt, G.T's,

ISSN 2249-6343
International Journal of Computer Technology and Electronics Engineering (IJCTEE)
Volume 3, Special Issue, March-April 2013, An ISO 9001: 2008 Certified Journal.
E-NSPIRE, A National Level Conference held at Pravara Rural Engineering College, Loni, Maharashtra, INDIA.

"General Questions on Kalman Filtering in Navigation Systems", (1970).

[8] Steffen L. Laurite, Thiele: Pioneer in Statistics, Oxford University Press, 2002. ISBN 0-19-850972-3.

[9] M. Loeve, Probability theory, 3rd edition, Van Nastrand Reinhold, New York, 1963.