Shubhangam Dutta

CMPEN 454

Honors Project

The Honors Project was on Virtual Object Tracking. Tracking is the task of being given an image identifying and object, and being able to follow it through a video sequence. Something that seems trivial for a human observer can become immensely complicated for a computer when we begin to consider the varying problems such as image matching with motion and rotation of the subject, occasional object occlusion, and other challenges.

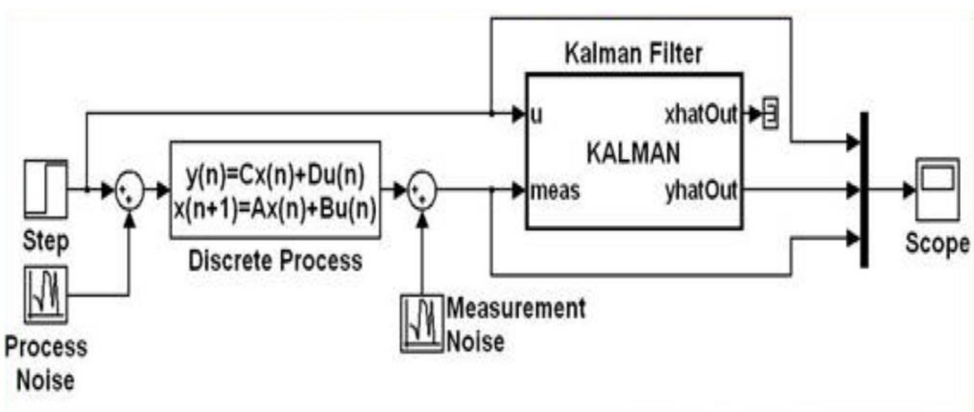
Object tracking is the process of locating moving objects over time in videos. Object tracking is the problem of determining (estimating) the positions and other relevant information of moving objects in image sequences. Object tracking is about locking onto a particular moving object(s) in real-time. It includes: Motion detection: Often from a static camera. Common in surveillance systems. Often performed on the pixel level, Object localization: Focuses attention to a region of interest in the image, Data reduction: Often only interest points found which are used later to solve the correspondence problem, Motion segmentation: Images are segmented into region corresponding to different moving objects and Object tracking: A sparse set of features is often tracked.

Object tracking has a wide range of applications in computer vision, such as surveillance, human-computer interaction, and medical imaging, traffic flow monitoring, human activity recognition, etc.

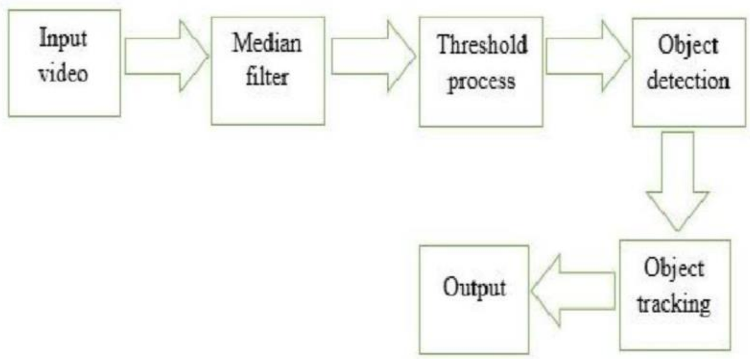
Therefore, for the purpose of Virtual Object tracking, I used Kalman filter to estimate the position of an object moving in a two-dimensional space from a series of noisy inputs based on past positions. The position vector has two components, x and y, indicating its horizontal and vertical coordinates. Apart from that, I also added a bounding box to mark the noise in the frames.

Kalman filters have a wide range of applications, including control, signal and image processing; radar and sonar; and financial modeling. They are recursive filters that estimate the state of a linear dynamic system from a series of incomplete or noisy measurements. The Kalman filter algorithm relies on the state-space representation of filters and uses a set of variables stored in the state vector to characterize completely the behavior of the system. It updates the state vector linearly and recursively using a state transition matrix and a process noise estimate.

The Kalman filter is a set of mathematical equations that provides an efficient computational (recursive) means to estimate the state of a process, in a way that minimizes the mean of the squared error. The filter is very powerful in several aspects: it supports estimations of past, present, and even future states, and it can do so even when the precise nature of the modeled system is unknown.



The filter has two distinct phases: Predict and Update. 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. 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 posterior state estimate.



The function f can be used to compute the predicted state from the previous estimate and similarly the function h can be used to compute the predicted measurement from the predicted state. However, f and h cannot be applied to the covariance directly. Instead a matrix of partial derivatives (the Jacobian) is computed. At each time step the Jacobian is evaluated with current predicted states. These matrices can be used in the Kalman filter equations. This process essentially linearizes the non-linear function around the current estimate.

To generate the input video I used the videoWrite and looped through all the frames to generate the input video.This was then fed into the track.m file from the main file and the bounding box was checked for initialization.

Since it wasn’t initialized it was fed into the initialize1 function which initilazed the bounding box.

Since there was no background frame given. I used averaging to generate a background frame to ease the process of tracking and to detect noise using the Backgrounframe.m file. The video was inputted in this matlab file and we get the averaged background frame by looping over the frames.

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 noisy measurements. The equations for Kalman filters fall in two groups: time update equations and measurement update equations. The time update equations are responsible for projecting forward (in time) the current state and error covariance estimates to obtain the a priori estimate for the next time step. The measurement update equations are responsible for the feedback. That is used 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.

The initial state of the object now was processed by the sampletracker function and a threshold fvalue of 10 was used or generating binary image of the noise.

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 noisy measurements. The equations for Kalman filters fall in two groups: time update equations and measurement update equations. The time update equations are responsible for projecting forward (in time) the current state and error covariance estimates to obtain the a priori estimate for the next time step. The measurement update equations are responsible for the feedback. That is used 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.

I initialized the values and the Kalman filter was created as follows: -

```
A = [1 0 dt 0;
 0 1 0 dt;
 0 0 1 0 ;
 0 0 0 1 ;
 ];
```

```
B = [(dt^2)/2 (dt^2)/2 dt dt]';


% acceleration

u = 10^-2;

H = [1 0 0 0;

 0 1 0 0];
```

H is matrix converting state space into measurement space and R is measurement noise covariance. Determining R for set of measurement is difficult, many Kalman implementations statistically analyze training data to determine fixed R for all future time updates.

```
% Covariance Matrices :

State Uncertainty was found to be 10;


 S = A*S*A' + Q

 K = S*H'*inv(H*S*H'+R);


 if( the input is not empty)

 x = x + K*(input - H*x);

 end
```

S = (eye(size(S,1)) - K*H)*S;

Kalman_Output = H*x;

We are observing the X, and Y position. So our observation is: y = [X Y]

y(t) = H.State(t) + <Measurement Noise>

The dynamic noise characterizes the transition noise from one state to another, therefore the value of the Dyn_Noise_Variance was given as (0.01)^2;

Since I assumed the variables X and Y are independent, I calculated

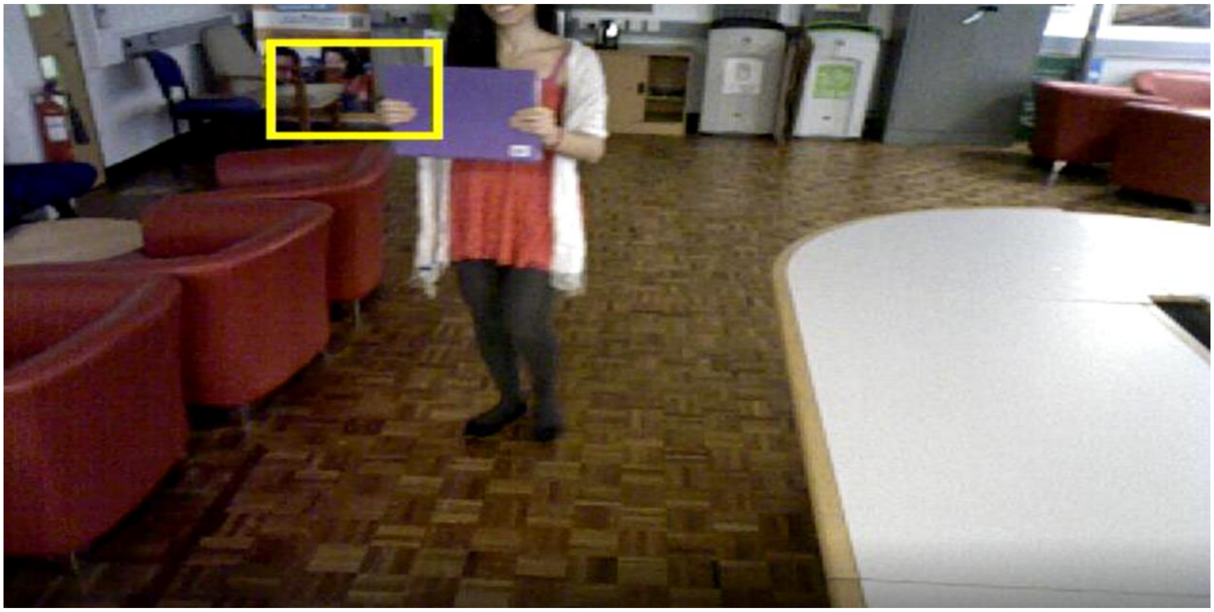
Q as [(dt^2)/4 0 (dt^3)/2 0;

0 (dt^2)/4 0 (dt^3)/2;

(dt^3/2) 0 (dt^2) 0;

0 (dt^3)/2 0 (dt^2);

];



Generated Video

Box and Noise (Uncommenting the imshow on track.m)

The algorithm then then tested for the noise (indicated by black) and the predicted values of the filter was then fed into the bounding box function. On compiling the code, anyone can view the bounding box for each frame separately. The video was also generated (the code has been commented out for storage purposes) and I have compressed and attached the file with the report.

Results

The algorithm that I used produced reasonable results. I generated 2 bounding boxes, one in yellow to track the resulting object and one in black to show the noise (if you uncomment the imshow command in track.m). The tracker failed on certain situations such as when the book opened and on certain times at the beginning. The reason why it behaved in such a way was because of a few reasons such as noise in the frames (as displayed by the black bounding box) and imperfect estimated values for the filter. Since the values of velocity and acceleration were estimated, the bounding box often wasn't able to accurately track the motion of the object.

Advantages of using this filter

- Linear models interact uniquely well with Gaussian noise i.e. make the prior Gaussian, everything else Gaussian and the calculations are easy.
- This is a statistical technique that adequately describes the random structure of experimental measurements. This filter is able to take into account quantities that are partially or completely neglected in other techniques (such as the variance of the initial estimate of the state and the variance of the model error).
- Gaussians are easy to represent i.e. once the mean and covariance is known, you're done.
- The tracker is effective in tracking single objects as the tracker is successful in detecting high moving objects.
- It provides information about the quality of the estimation by providing, in addition to the best estimate, the variance of the estimation error.
- The Kalman filter is well suited to the online digital processing. Its recursive structure allows its real-time execution without storing observations or past estimates.
- This tracker also detects the noise and can be helpful in development of a better tracker.

Group work

The project was done by me individually, No one else contributed towards the project.

References:

https://www.mathworks.com/help/simulink/ug/tutorial-integrating-matlab-code-with-a-simulink-model-for-tracking-a-moving-object.html#bslqaef-1

https://www.hindawi.com/journals/jhe/2018/8079713/

https://www.mathworks.com/help/vision/ref/imagelabeler-app.html