Tracking with Kalman Filter

Scott T. Acton
Virginia Image and Video Analysis (VIVA), Charles L. Brown Department of Electrical and Computer Engineering
Department of Biomedical Engineering
University of Virginia, Charlottesville, VA 22904

Virginia Image and Video Analysis

Tracking

- The tracking problem can usually be broken down into two subproblems
  - (1) **Acquisition/Detection**: finding the object of interest (the target) for the first time
  - (2) **Tracking/Prediction**: guessing where it's going to be in the next frame

Acquisition/Detection

- *Centroid Trackers* -- find the centroid in the region of interest
- *Edge Trackers* -- track the leading edge of a target
- *Outline Trackers* – track the outline of the target (using edge detection)
- *Template Trackers* -- track a target template
- *Deformable Template Trackers* -- change the template as you go
- *Adaptive Template Trackers* -- let image sequence re-define the template
- *Snake Trackers* – track the boundary of an object by matching the corresponding snake between two frames

Prediction/Estimation

- The goal of the prediction portion is to estimate the next position of the target, given the previously acquired image sequence
- Once we have an estimate of the next target position, we can look at a small subimage for the target -- and avoid searching the whole image again
- This subimage is called the **track gate** -- the size the of track gate is dictated by the accuracy of the tracker
- Solution to the prediction problem borrows heavily from estimation theory -- we'll discuss the standard solution

Kalman Filter

- The Kalman filter is the optimal filter (in the least mean squared error sense) for track prediction
- The Kalman filter is used heavily by control theorists -- it's used everywhere from the Space Shuttle to the Patriot missile system to the NY stock exchange
- If we assume a constant velocity model for our target, the Kalman filter reduces to the *alpha-beta* filter -- we'll see alpha and beta soon

Kalman Filter

- The Kalman filter is a combination of a predictor and a filter:

- The predictor estimates the location of the target at time *k* given *k*-1 observations

- When observation *k* arrives, the estimate is improved using an optimal filter to estimate the target position at time *k* +1: the filtered estimate is the best estimate of the true location of the target given *k* observations at time *k*

- Both the predictor and the filter are linear systems
- The Kalman filter will **not** be **derived** here; the equations will be set up and explained for a tracking application

Glossary of Terms

X_k -- an image sequence
k -- time, k=0 denotes the first acquisition
i,j -- row and column positions in the image
$\hat{i}_{k|k-1}$ -- row of the predicted target at time k given the first k - 1 observations
$\hat{j}_{k|k-1}$ -- column of the predicted target
G -- the track gate -- a subimage of X_k
i_k -- row at time k
j_k -- column at time k
$\hat{v}^i_{k+1|k}$ -- velocity estimate for time k+1 in the *i direction*
$\hat{v}^j_{k+1|k}$ -- same for j direction
v^i_k -- velocity at time k in the i direction
v^j_k -- same for j direction
u^i_k -- velocity drift noise process in the i direction
δt -- change in time between frames
i^0_k -- observed i value -- sometimes called z_k in other places (same goes for j) -- we get this observation from the template matcher, centroid finder, matched filter, etc.

Motion Model – Constant Velocity

- We will use a constant velocity model $i_{k+1} = i_k + \delta t\ v^i_k$
- This is a linear system. We're saying that the first derivative (of position) is fairly constant, and the second derivative is almost zero. We will model a small acceleration (called the "drift") as a white noise process.
- So, as long as our temporal sampling rate is sufficient, this should be a good model for motion. Here, acceleration is viewed as a noise process, and we just have to choose a reasonable variance.
- Aside: we can have a constant acceleration model with the Kalman filter -- this is the *alpha-beta-gamma* filter. This model has the acceleration terms in addition to position and velocity (for each direction, *i* and *j*).

Prediction of Position

- Predicted position (*)

$$\hat{i}_{k+1|k} = \hat{i}_{k|k} + \delta t \hat{v}^{i}_{k+1|k}$$

- Filtered estimate of position is (**)

$$\hat{i}_{k|k} = \hat{i}_{k|k-1} + \alpha_k (i^0_k - \hat{i}_{k|k-1})$$
$$= (1-\alpha_k)\hat{i}_{k|k-1} + \alpha_k i^0_k$$

- The gain α_k determines the balance between the previous track history and the new observation
- If α_k is large (near 1), we believe the observations are very reliable (this is essentially ignoring the track history)
- If it's small (near 0), we believe that there is a lot of measurement noise (this is essentially ignoring the observation)

Prediction of Velocity

- Predicted velocity (***)

$$\hat{v}^{i}_{k+1|k} = \hat{v}^{i}_{k|k-1} + \beta_k (i^0_k - \hat{i}_{k|k-1}) / \delta t$$

- Kalman gain β_k controls how we let the new observation affect the predicted velocity
  - near 0, it means that we think the observations are unreliable and that the actual velocity is REALLY a constant -- in this case, the observation does not affect the predicted velocity
  - near 1, then the observations are reliable (we think!). Here, we allow the velocity to drift (acceleration).

- All of the above equations repeat for the *j* direction

Computing Kalman Gains

- For the alpha-beta filter (the constant velocity model), we can pre-compute the gains -- yes, before we implement the tracker (this assumes **stationarity**)
- Also, the gains converge quickly to constants -- so, we don't have compute an infinite number of the gain values
- Then, we can just plug the observations into the three prediction and filtering equations and "Kalmanize"!
- The gains depend upon the **noise variances** and the **state vector error covariance matrix**
- (*See Appendix for details on how to compute gains*)

Executing the Kalman Filter

- (1) Use the starting state conditions to get alpha and beta for several *k*'s (until convergence) -- pre-store these values (the first *k* = 1) (*See Appendix* ****)
- (2) Acquire the target using the whole image to get initial coordinates
- (3) Use (**) to get the filtered position , then (***) to get the predicted velocity , then (*) to get the predicted position (same goes for the corresponding *j* terms)
- (4) Acquire target within track gate centered at predicted position
- (5) Go to (3)
- Processes for *i* and *j* run independently and concurrently

More

- If there are no observations at time *k*, the track is **coasted** -- we use *observed position = predicted position*
  - the next predicted position is then simply the last predicted position plus the velocity multiplied by the frame time

Problems with Kalman

- Some constants used in computing the gains are difficult to obtain
  - actual errors may not comply statistically w/ Kalman model
    - -- the "divergence phenomenon”
  - the real world may not obey the Kalman assumptions:
    - (1) observations are signal plus white noise
    - (2) the signal can be modeled by a linear system driven by white noise
    - (3) all parameters of the two noise processes and the linear system are known precisely
- How do these terms relate to our tracker?
  - the signal: the position of the target
  - noise: we assume white noise drift in the target velocity and measurement noise in the target location
  - the linear system: the constant velocity model

Appendix: Kalman Gains

- Let the state vector X be defined for our system as:

$X_k = \begin{bmatrix} i_k \\ v_k^i \end{bmatrix}$. The state vector of the predictor is $\hat{X}_{k|k-1} = \begin{bmatrix} \hat{i}_{k|k-1} \\ \hat{v}_{k|k-1}^i \end{bmatrix}$.

The state vector for the filter $\hat{X}_{k|k}$ is constructed similarly.
(these are the state matrices for the i direction; the same goes for the j dir.)

- We'd like to measure the error in the predictions and in the filtered results, so that we can minimize error.

- The error in the predicted state vector is:

$$X_k - \hat{X}_{k|k-1}$$

and the error for the filter state vector is:

$$X_k - \hat{X}_{k|k}$$

- The above errors are stochastic vectors -- hence, they have covariance matrices

- The predicted state vector covariance matrix is:

$$P_{k|k-1} = E\left[\left(X_k - \hat{X}_{k|k-1}\right)\left(X_k - \hat{X}_{k|k-1}\right)^T\right]$$

- The filtered state vector error covariance matrix is:

$$P_{k|k} = E\left[\left(X_k - \hat{X}_{k|k}\right)\left(X_k - \hat{X}_{k|k}\right)^T\right]$$

- **Function of the Kalman filter**: choose α_k and β_k to minimize $P_{k|k}$

Why? You want the minimum mean squared error estimate.

- For the noise processes, we have

  σ_n^2 -- the measurement noise variance
  σ_u^2 -- the velocity drift noise variance

- The solution that minimizes $P_{k|k}$ (We won't prove it here) is:

$$\alpha_k = \frac{P^{11}_{k|k-1}}{P^{11}_{k|k-1} + \sigma_n^2} \quad \text{and} \quad \beta_k = \frac{\delta t P^{21}_{k|k-1}}{P^{11}_{k|k-1} + \sigma_n^2} \qquad (****)$$

- So, we have equations for α_k and β_k, but $P_{k|k-1}$ is changing with each iteration

- For our constant velocity alpha-beta model, $P_{k|k-1}$ can be computed recursively as follows:

$$P^{11}_{k+1|k} = P^{11}_{k|k-1} + 2P^{12}_{k|k-1} + P^{22}_{k|k-1} - \frac{\left(P^{11}_{k|k-1} + P^{12}_{k|k-1}\right)^2}{P^{11}_{k|k-1} + \sigma_n^2}$$

$$P^{12}_{k+1|k} = P^{12}_{k|k-1} + P^{22}_{k|k-1} - P^{12}_{k|k-1}\left(\frac{P^{11}_{k|k-1} + P^{12}_{k|k-1}}{P^{11}_{k|k-1} + \sigma_n^2}\right)$$

$$P^{21}_{k+1|k} = P^{12}_{k+1|k}$$

$$P^{22}_{k+1|k} = P^{22}_{k|k-1} + \sigma_u^2 - \frac{\left(P^{12}_{k|k-1}\right)^2}{P^{11}_{k|k-1} + \sigma_n^2}$$

- We just need **initial conditions** for $P_{k|k-1}$ (= $P_{1|0}$)

- To do this correctly, we need to define two additional variances:

σ_i^2 -- variance in the initial row position (there's a corresponding term for column position *j*)
$\sigma_{v_i}^2$ -- variance in the initial velocity in the *i* direction

- If we assume that the initial position is a uniformly distributed random variable over the *N* possible rows, then computing σ_i^2 is easy -- just consult the probability textbooks

- The computation of $\sigma_{v_i}^2$ can be derived in the same way -- determine the minimum and maximum possible velocities -- then assume that velocity is uniformly distributed over the possible velocities

- Now we can compute the filtered state vector error covariance at time 0:

$$P_{0|0} = E\left[(X_k)(X_k)^T\right] = \begin{bmatrix} \sigma_i^2 & 0 \\ 0 & \sigma_{v_i}^2 \end{bmatrix}$$

- Then $P_{1|0}$ can be computed from:

$$P_{1|0} = A_0 P_{0|0} A_0^T + Q_0$$

- Note that the above indicates matrix multiplication. For our tracker:

$$A_0 = \begin{bmatrix} 1 & \delta t \\ 0 & 1 \end{bmatrix} \quad \text{... the state transition matrix}$$

and

$$Q_0 = \begin{bmatrix} \sigma_n^2 & 0 \\ 0 & \sigma_u^2 \end{bmatrix}$$

(covariance of the noise processes)

- Other **Track Initiation** information:
  (1) Set $\hat{i}_{0|0}$ to the first acquired position
  (2) The first velocity estimate is indeterminate (or can be set to a constant -- preferably)