What Is Video Tracking?

- **Video tracking** is
  - The process of locating a moving object (or several ones) in time using a camera.
  - An algorithm analyses the video frames and outputs the location of moving targets within the video frame
Basic Problems of Tracking Systems

- **Motion problem**
  - Predict the location of an image element being tracked in the next frame, that is, identify a limited search region in which the element is expected to be found with high probability

- **Matching problem**: (also known as detection or location)
  - Identify the image element in the next frame within the designated search region.

Solutions for *Motion Problem*

- To define the search area in the next frame as a fixed-size region surrounding the target position in the previous frame.
- The size is chosen according to the characteristic of the problem, crucially the expected frame-to-frame displacement.
  - Obviously, this knowledge is not often available, reliable, or time-independent, so performance is limited.
Solutions for *Matching Problem*

- In essence, a similarity metric to compare candidate pairs of image elements in the previous and current frame.
- This is closely related to the *correspondence problem of stereo vision*
  - The same scene element must be detected in two (or more) images acquired simultaneously from different viewpoints.

Traditional Solutions for Video Tracking

- A well-known solution from control theory is the *Kalman filter* (KF)
  - A well-known optimal, recursive estimator of the state of a dynamic system
- *Particle filtering* is another solution to the multiple-target problem within statistical estimation.
Design requirements for a video tracker

- **Robustness to clutter**: the tracker should not be distracted by image elements resembling the target being tracked.
- **Robustness to occlusion**: tracking should not be definitely lost because of temporary target occlusion (*drop-out*), but resumed correctly when the target reappears (*drop-in*).
- **False positives/negatives**: only valid targets should be classified as such, and any other image element ignored (in practice, the number of false alarms should be as small as possible).
- **Agility**: the tracker should follow targets moving with significant speed and acceleration (“agile” motion).
- **Stability**: the lock and accuracy should be maintained indefinitely over time.

What is Kalman Filter?

- A recursive **state estimator** for processes that are
  - partially observed
  - non-stationary
  - Stochastic
- It gives an **optimal estimate**
  - in the least squares sense of the actual value of a state vector
  - from noisy observations
Application of Kalman Filter

- Object tracking for a video stream
- How to track the object in the scene?
  - Find some points of interest for the object in a scene
    - Interesting point detector
  - Predict where they will be in the next frame from a motion model
    - Linear motion model
  - Look for them in the next frame and update our model
    - Method to update the model as we see a series of frames
    - This is the idea behind the Kalman filter

Kalman Filter

- The Kalman filter
  - Gives an estimate of an unknown state, $s_t$.
    - The value of $s$ at time $t$ is $s_t$
  - This estimate is based on measurements, $m_t$.
    - The measurement made at time $t$ is $m_t$
  - Also estimates the uncertainty in the estimate, $P_t$
One-Dimensional Kalman Filter

- Goal: To estimate some value, $s$, which varies over time
- We have a model of how $s$ changes with time
  - $s_{t+1} = a*s_t + v$
  - where $v$ is a random value with mean 0 and variance $q$
- We will estimate $s$ from measurements
  - At each time, a measurement, $m_t$, is made
  - The measurement is related to $s_t$ by
    - $m_t = h*s_t + w$
    - $w$ is a random value with mean 0 and variance $r$

Example of One-Dimensional Kalman Filter: Population Growth

- A model of how $s$ changes with time: $s_{t+1} = a*s_t + v$
  - where $v$ is a random value with mean 0 and variance $q$
- Estimate $s$ from measurements
  - At each time, a measurement, $m_t$, is made
  - The measurement is related to $s_t$ by
    - $m_t = h*s_t + w$
    - $w$ is a random value with mean 0 and variance $r$
- We want to estimate the population (in millions) of some country
  - We know that the population grows at some rate, say 10% per year
  - However, this figure (population) has a variance of 5 million
    - So we have $a = 1.1$, and $q = 5$
- We have a series of measurements from census forms
  - We know that not everyone fills in the forms. We expect that about 85% will, with a variance of 10 million
    - So we have $h=0.85$ and $r = 10$
An Initial Estimate

• An initial estimate is needed to get things started
  • We have not made any measurements so we have no idea what the population is
• Put some bounds on it
  • It has to be greater than zero
  • It is probably less than a billion
• Pick a value for the initial estimate
  • \( s_0 = 500 \) (million)
• This value is very uncertain, so we give it a large variance
  • \( p_0 = 500^2 = 250,000 \)
• The estimate is probably wrong, but it doesn’t matter since it has a high variance

A First Measurement

• A model of how \( s \) changes with time:
  \( s_{t+1} = a*s_t + v \)
  • where \( v \) is a random value with mean 0 and variance \( q \)
• Estimate \( s \) from measurements
  • At each time, a measurement, \( m_t \), is made
  • The measurement is related to \( s_t \) by
    • \( m_t = h*s_t + w \)
  • \( w \) is a random value with mean 0 and variance \( r \)
• It is the time to start a Kalman filter
  • At each time we make a prediction from the last estimate
  • We then make a measurement
  • We combine the prediction and the measurement to give the final estimate
• Predicting \( s_1 \)
  • We just use our model equation, so
    \( s_1^- = a*s_0 = 1.1 \times 500 = 550 \)
  • The noise term, \( v \), doesn’t affect \( s_1^- \), since it is (on average) zero

Note the superscript ‘-’ of “\( s_1^- \)”, this marks \( s_1^- \) as an initial estimate
Predicting the Variance

• A model of how $s$ changes with time: $s_{t+1} = as_t + v$
  • where $v$ is a random value with mean 0 and variance $q$

• Results from statistics
  • If $a$ and $b$ are independent random variables with variances $v_a$ and $v_b$, and $k$ is a constant
    • $\text{var}(a+b) = v_a + v_b$
    • $\text{var}(ka) = k^2v_a$

• So how to calculate the value of variance $p_1$ of $s_1$ (where $s_1 = as_0 + v$)?
  
  \[ p_1 = a^2 \times \text{var}(s_0) + \text{var}(v) \]
  
  \[ = a^2p_0 + q \]
  
  \[ = 1.12 \times 250,000 + 5 \]
  
  \[ = 302,505 \]

• Since $s_0$ is uncertain, $s_1$ is uncertain also

Note the superscript ‘‘$-$’’ of “$s_1$”, this marks $s_1$ as an initial estimate

Making a Measurement

• A model of how $s$ changes with time: $s_{t+1} = as_t + v$
  • where $v$ is a random value with mean 0 and variance $q$

• Estimate $s$ from measurements
  • At each time, a measurement, $m_t$, is made
  • The measurement is related to $s_t$ by
    • $m_t = h*s_t + w$
    • $w$ is a random value with mean 0 and variance $r$

• Make a measurement of the population $m_1 = 91$ (million)
  • The variance in this measurement is 10 million
  • This is about 85% of the true population

• How do we combine the prediction and the measurement?
  • The one with lower variance should have greater weight
  • We also need to take into account the factor $h$
The Kalman Gain

- A model of how s changes with time: \( s_{t+1} = a \cdot s_t + v \)
  - where \( v \) is a random value with mean 0 and variance \( q \)
- Estimate \( s \) from measurements
  - At each time, a measurement, \( m_t \), is made
  - The measurement is related to \( s \) by
    - \( m_t = h \cdot s_t + w \)
    - \( w \) is a random value with mean 0 and variance \( r \)

- The Kalman filter uses a value called the **Kalman gain**
  - It is computed from the variances of \( s_t \) and \( m_t \)
  - It is chosen so that the variance in the final estimate, \( s_t \), is as small as possible
- Our final estimate will be
  \[
  s_t = s_t^- + k_t (m_t - h \cdot s_t^-)
  \]
  - \( m_t - h \cdot s_t^- \) is the difference between the measurement and the one we would expect if our prediction was right
  - \( k_t \) tells us how much attention to give this difference

\[\text{Note the superscript } ^- \text{ of } "s_t^-", \text{ this marks } s_t^- \text{ as an initial estimate}\]

The Variance in our Estimate

- A model of how \( s \) changes with time: \( s_{t+1} = a \cdot s_t + v \)
  - where \( v \) is a random value with mean 0 and variance \( q \)
- Estimate \( s \) from measurements
  - At each time, a measurement, \( m_t \), is made
  - The measurement is related to \( s \) by
    - \( m_t = h \cdot s_t + w \)
    - \( w \) is a random value with mean 0 and variance \( r \)

\[
\begin{align*}
s_t &= s_t^- + k_t (m_t - h s_t^-) \\
      &= (1 - k_t h) s_t^- + k_t m_t \\
\text{var}(s_t) &= (1 - k_t h)^2 \text{var}(s_t^-) + k_t^2 \text{var}(m_t) \\
p_t &= (k_t h^2 - 2k_t h + 1) p_t^- + k_t^2 r \\
      &= (h^2 p_t^- + r) k_t^2 - 2h p_t^- k_t + p_t^- \\
\end{align*}
\]

\[\text{Note the superscript } ^- \text{ of } "s_t^-", \text{ this marks } s_t^- \text{ as an initial estimate}\]

where "-" of \( p_t^- \) marks \( p_t^- \) as the variance of the initial estimate
Finding the Kalman Gain

\[ s_t^- = s_t^- + k_t^* (m_t^- - hs_t^-) \]
\[ = (1 - k_t^* h) s_t^- + k_t^* m_t \]
\[ \text{var}(s_t^-) = (1 - k_t^* h)^2 \text{var}(s_t^-) + k_t^2 \text{var}(m_t) \]
\[ p_t^- = (k_t^* h^2 - 2k_t^* h + 1)p_t^- + k_t^2 r \]
\[ = (h^2 p_t^- + r)k_t^2 - 2h p_t^- k_t^- + p_t^- \]

\[
\frac{dp_t^-}{dk_t^-} = \frac{d}{dk_t^-} (h^2 p_t^- + r)k_t^2 - 2h p_t^- k_t^- + p_t^- \\
0 = 2k_t^- (h^2 p_t^- + r) - 2h p_t^- \\
k_t^- = \frac{h p_t^-}{h^2 p_t^- + r}
\]

The Kalman Filter

• This expression for \( k_t^- \) allows us to simplify the formula for \( p_t^- \) to \( p_t^- = p_t^- - k_t^- h p_t^- \)
  
• We now have the 1D Kalman filter

• It is based on the model equations
  
• \( s_{t+1} = a s_t + v \)
  
• \( m_t = h s_t + w \)

• The 1D Kalman filter equations are

\[
\begin{align*}
  s_t^- & = a s_{t-1}^- \\
p_t^- & = a^2 p_{t-1}^- + q \\
k_t^- & = (h p_t^-)/(h^2 p_t^- + r) \\
s_t^- & = s_t^- + k_t^- (m_t^- - hs_t^-) \\
p_t^- & = p_t^- - k_t^- h p_t^- \\
\end{align*}
\]

Initial Estimate

Variance of Initial Estimate

Update the Initial Estimate

Update the variance of the Initial Estimate

Note the superscript ‘−’ of “\( s_t^- \),” this marks \( s_t^- \) as an initial estimate

where ‘−’ of “\( p_t^- \)” marks \( p_t^- \) as the variance of the initial estimate
Kalman Filter Calculation: An Example

- We had an initial estimate at $t=1$
  - $s_{1^*} = 550$
  - $p_{1^*} = 302,505$
- We then made a measurement
  - $m_t = 91$
  - $r = 10$
- We can now compute $k_t$

\[
\begin{align*}
s_t &= s_{t-1} + k_t(m_t - hs_{t-1}) \\
p_t &= p_{t-1} - k_thp_{t-1}
\end{align*}
\]

\[
k_t = \frac{hp_{t-1}}{h^2p_{t-1} + r}
\]

\[
k_t = \frac{0.85 \times 302,505}{0.85^2 \times 302,505 + 10} = 0.85 \times 302,505 + 10
\]

\[
k_t = \frac{257,129.25}{218,569.8625} = 1.176
\]

Kalman Filter Calculation: An Example

- We now use $k_t$ to combine $s_{t-1}$ and $m_t$ and find $s_t$ and $p_t$

\[
\begin{align*}
s_t &= s_{t-1} + k_t(m_t - hs_{t-1}) \\
p_t &= p_{t-1} - k_thp_{t-1}
\end{align*}
\]

\[
s_t = s_{t-1} + k_t(m_t - hs_{t-1}) \\
\approx 550 + 1.176(91 - 0.85 \times 550) \\
\approx 107
\]

\[
p_t = p_{t-1} - k_thp_{t-1} \\
\approx 302,505 - 1.176 \times 0.85 \times 302,505 \\
\approx 13.9
\]
Kalman Filter Calculation: An Example

- The computation then iterates with each measurement

\[ s_t^* = a s_{t-1} \]
\[ p_t^* = a^2 p_{t-1} + q \]
\[ k_t = (h p_t^*)/(h^2 p_t^* + r) \]
\[ s_t = s_t^* + k_t (m_t - h s_t^*) \]
\[ p_t = p_t^* - k_t h p_t^* \]

<table>
<thead>
<tr>
<th>t</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s_t )</td>
<td>550</td>
<td>118</td>
<td>132</td>
<td>147</td>
<td>164</td>
<td>180</td>
</tr>
<tr>
<td>( p_t )</td>
<td>302K</td>
<td>21.7</td>
<td>15.2</td>
<td>13.8</td>
<td>13.2</td>
<td>13.2</td>
</tr>
<tr>
<td>( m_t )</td>
<td>91</td>
<td>103</td>
<td>115</td>
<td>129</td>
<td>140</td>
<td>153</td>
</tr>
<tr>
<td>( k_t )</td>
<td>1.176</td>
<td>0.719</td>
<td>0.616</td>
<td>0.587</td>
<td>0.578</td>
<td>0.575</td>
</tr>
<tr>
<td>( s_t )</td>
<td>107</td>
<td>120</td>
<td>134</td>
<td>149</td>
<td>165</td>
<td>180</td>
</tr>
<tr>
<td>( p_t )</td>
<td>8.46</td>
<td>7.25</td>
<td>6.90</td>
<td>6.79</td>
<td>6.76</td>
<td>6.75</td>
</tr>
</tbody>
</table>

Filter Convergence

\[ s_t^* = a s_{t-1} \]
\[ p_t^* = a^2 p_{t-1} + q \]
\[ k_t = (h p_t^*)/(h^2 p_t^* + r) \]
\[ s_t = s_t^* + k_t (m_t - h s_t^*) \]
\[ p_t = p_t^* - k_t h p_t^* \]

<table>
<thead>
<tr>
<th>t</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s_t )</td>
<td>550</td>
<td>118</td>
<td>132</td>
<td>147</td>
<td>164</td>
<td>180</td>
</tr>
<tr>
<td>( p_t )</td>
<td>302K</td>
<td>21.7</td>
<td>15.2</td>
<td>13.8</td>
<td>13.2</td>
<td>13.2</td>
</tr>
<tr>
<td>( m_t )</td>
<td>91</td>
<td>103</td>
<td>115</td>
<td>129</td>
<td>140</td>
<td>153</td>
</tr>
<tr>
<td>( k_t )</td>
<td>1.176</td>
<td>0.719</td>
<td>0.616</td>
<td>0.587</td>
<td>0.578</td>
<td>0.575</td>
</tr>
<tr>
<td>( s_t )</td>
<td>107</td>
<td>120</td>
<td>134</td>
<td>149</td>
<td>165</td>
<td>180</td>
</tr>
<tr>
<td>( p_t )</td>
<td>8.46</td>
<td>7.25</td>
<td>6.90</td>
<td>6.79</td>
<td>6.76</td>
<td>6.75</td>
</tr>
</tbody>
</table>
Generalization of Kalman Filter

• There is no reason for \( q, r, a, \) and \( h \) to be fixed
  • They can all vary with time if needed
  • Often they are fixed, but not always

• For example, we could have \( q \) and \( r \) being a percentage of the population in our example

\[
\begin{align*}
 s_t &= a_{t-1} s_{t-1} \\
p_t &= a^2 p_{t-1} + q \\
k_t &= (h p_t) / (h^2 p_t + r) \\
s_t &= s_t + k_t (m_t - h s_t) \\
p_t &= p_t - k_t h p_t
\end{align*}
\]

\[
\begin{align*}
 s_t &= a_{t-1} s_{t-1} + v_{t-1} \\
m_t &= h s_t + w_t \\
s_t &= a_{t-1} s_{t-1} \\
p_t &= a^2 p_{t-1} + q_{t-1} \\
k_t &= (h p_t) / (h^2 p_t + r) \\
s_t &= s_t + k_t (m_t - h s_t) \\
p_t &= p_t - k_t h p_t
\end{align*}
\]

Generalization of Kalman Filter

• Usually our state and measurements are sets of values
  • We can represent these as vectors for \( s \) and \( m \) at each time
  • \( a, h, q, r, k, \) and \( p \) become matrices, which we write as \( A, H, Q, R, K, \) and \( P \)

• This makes the filter equations more complicated
  • We can’t divide by a matrix to find \( K \), but the matrix inverse does much the same thing
  • The terms like \( a2p \) become terms like \( APA^T \)
Multi-Dimensional Kalman Filter

\[
\begin{align*}
 s_t^- &= a s_{t-1} \\
p_t^- &= a^2 p_{t-1} + q \\
k_t &= (h p_t^-)/(h^2 p_t^- + r) \\
s_t &= s_t^- + k_t (m_t - h s_t^-) \\
p_t &= p_t^- - k_t h p_t^-
\end{align*}
\]

\[
\begin{align*}
 s_t &= a_{t^2} s_{t-1} + v_{t-1} \\
m_t &= H s_t + w_t \\
s_t^- &= a_{t^2} s_{t-1} \\
p_t^- &= a_{t^2} p_{t-1} + q_{t-1} \\
k_t &= (h p_t^-)/(h^2 p_t^- + r_t) \\
s_t &= s_t^- + k_t (m_t - h s_t^-) \\
p_t &= p_t^- - k_t h p_t^-
\end{align*}
\]

Kalman Filter Assumptions

- The Kalman filter is based on a number of assumptions:
  - It assumes that the relationships, \( A \) and \( H \), between \( s_t \) and \( s_{t-1} \) and \( m_t \) are linear
  - It assumes that these linear relationships are known beforehand
  - We’ll look at a way to relax this constraint
- It also relies on a Gaussian error model:
  - The noise terms \( v \) and \( w \) are assumed to be Gaussian
  - It assumes that their (co)variances are known beforehand
  - The formulation of \( K \) to minimise \( P \) relies on this assumption
Application of Kalman Filter: Traffic Tracking

• We want to track vehicles on a road
  • Eg: The truck in the images to the left
  • They are moving with a (fairly) constant velocity
  • In each frame we can measure the position of a feature on the vehicle we want to track

![t=0](image1) ![t=10](image2)

State Update Equation

• We assume the truck is moving with constant velocity
  • Our state is the truck position \((x,y)\) and velocity \((u,v)\)
  \[
  s = [x, y, u, v]^T
  \]
  • At each time the velocity adds on to the position

\[
\begin{align*}
x_t &= x_{t-1} + u_{t-1} \\
y_t &= y_{t-1} + v_{t-1} \\
u_t &= u_{t-1} \\
v_t &= v_{t-1}
\end{align*}
\]

\[
s_t = A s_{t-1}
\]
Measurement Equation

• At each time we can detect features in the image
  • These make our measurements, $m_t$
  • We can directly measure the position of the truck, but not its velocity
  • $m_t = [x, y]^T$

\[
\begin{bmatrix}
    x_t \\
    y_t \\
    u_t \\
    v_t
\end{bmatrix} =
\begin{bmatrix}
    1 & 0 & 0 & 0 \\
    0 & 1 & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
    x_{t-1} \\
    y_{t-1} \\
    u_{t-1} \\
    v_{t-1}
\end{bmatrix}
\]

$m_t = Hs_t$

An Initial Estimate

• The initial estimate of the state
  • We give a rough value of $x$ and $y$ to say which feature we are tracking
  • We probably won’t have any idea about $u$ and $v$
  • So we will use $s_0 = [100, 170, 0, 0]^T$

• We also need to give the (un)certainty
  • Our estimate of the position is good to within a few pixels
  • Our motion estimate is not good, but we expect the motion to be small
  • We represent this as a covariance matrix
Recap: Covariance Matrices

- So what is a covariance matrix?
  - It gives the relationships between sets of variables
  - The variance of a variable, \( x \), is \( \text{var}(x) = E((x-x)^2) \)
  - The covariance of two variables, \( x \) and \( y \), is \( \text{cov}(x,y) = E((x-x)(y-y)) \)
- Given a vector of variables \( x=[x_1,x_2,\ldots,x_k] \)
  - The covariance, \( C \), is a \( k \times k \) matrix
  - The \( i,j^{th} \) entry of \( C \) is: \( C_{i,j} = \text{cov}(x,y) \)
  - A diagonal entry, \( C_{i,i} \), gives the variance in the variable \( x_i \)
  - \( C \) is symmetric

Covariance in Noise

- The noise terms \( v \) and \( w \) need to be estimated
  - They have zero mean, and covariance \( Q \) and \( R \) respectively
  - We need an estimate of these matrices
  - \( Q \) and \( R \) say how certain we are about our model equations
- To estimate \( Q \)
  - Our initial estimate will be within a few pixels, say \( \sigma=3 \)
  - The velocity is a bit less certain, but won’t be large, say \( \sigma=5 \)
  - There is no reason to think that the errors are related, so the covariance terms will be zero
Initial Covariance

\[
P_0 = \begin{bmatrix} 9 & 0 & 0 & 0 \\ 0 & 9 & 0 & 0 \\ 0 & 0 & 25 & 0 \\ 0 & 0 & 0 & 25 \end{bmatrix}
\]

- The variances of \( x \) and \( y \) are \( 3^2 = 9 \)
- The variances of \( u \) and \( v \) are \( 5^2 = 25 \)
- Since we assume independence, the off-diagonal entries are all 0

Uncertainty in the Model

- Our model equations have noise terms
  - \( v \) represents the fact that our state update model may not be accurate
  - \( w \) represents the fact that measurements will always be noisy
  - We need to estimate their covariances
- In general
  - Often the terms will be independent. If this is the case the off-diagonal entries will be zero
  - Choosing the diagonal entries (variances) is often more difficult
State Update Covariance

- The state update equation is not perfect
  - It assumes that the motion is constant but \( u \) and \( v \) might change over time
  - It assumes that all the motion is represented by \( u \) and \( v \) but other factors might affect \( x \) and \( y \)
- These errors will probably be small
  - The motion is slow and quite smooth
  - So the variance in these terms is probably a pixel or less, say \( \sigma = \frac{1}{2} \)

\[
Q = \begin{bmatrix}
0.25 & 0 & 0 & 0 \\
0 & 0.25 & 0 & 0 \\
0 & 0 & 0.25 & 0 \\
0 & 0 & 0 & 0.25
\end{bmatrix}
\]

State Update Covariance

- The measurements we make will be noisy
  - The features are located only to the nearest pixel
  - Because of image noise, aliasing, etc, they might be off by a pixel or so
- These errors are a bit easier to estimate
  - The feature is probably in the right place, or a pixel off
  - So the variance in these terms is probably \( \sigma^2 = 1 \)

\[
R = \begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix}
\]
Predict the State

• We can now run the filter
  • First we make a prediction of the state at \( t=1 \) based on our initial estimate at \( t=0 \)

\[
\begin{align*}
S_1^- &= A S_0 \\
\begin{bmatrix}
1 & 0 & 1 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
100 \\
170 \\
0
\end{bmatrix}
\begin{bmatrix}
100 \\
170 \\
0
\end{bmatrix}
\end{align*}
\]

Prediction Covariance

\[
\begin{align*}
P_1^- &= A P_c A^T + Q \\
= \begin{bmatrix}
1 & 0 & 1 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
100 \\
100 \\
0.25
\end{bmatrix}
\begin{bmatrix}
100 \\
100 \\
0.25
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 1 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
100 \\
0.25 \\
0.25 \\
0.25
\end{bmatrix}
\begin{bmatrix}
100 \\
0.25 \\
0.25 \\
0.25
\end{bmatrix}
\end{align*}
\]
Making a Measurement

• The state prediction gives us
  • a guide to where the feature will be
  • We expect it to be near (100, 170)
  • The variance in the \( x \) position is 34.25
  • The variance in the \( y \) position is 34.25

• We can use this to restrict our search for a feature
  • We are 95% certain that the feature lies in a circle of radius 2\( \sigma \) of the prediction
    \[ \sigma = \sqrt{34.25} \approx 5.85 \]
    
  • We look for a feature in this region

Making a Measurement

• Within the search region
  • We compute a value that tells us how likely each point is to be a feature (interesting point detector)
  • We find the point with the largest value within this region
  • This is \( m_1 = [103, 163]^T \)
The Kalman Gain

- We now combine the prediction and measurement
  - We compute the Kalman gain matrix
  - This takes into account the relative certainty of the two pieces of information

\[
\begin{align*}
    s_t &= A_{t-1} s_{t-1} + v_{t-1} \\
    m_t &= H_t s_t + w_t \\
    s_{t|t} &= A_{t-1} s_{t-1} \\
    P_t &= A_{t-1} P_{t-1} A_{t-1}^\top + Q_{t-1} \\
    K_t &= H_t P_t (H_t P_t H_t^\top + R_t)^{-1} \\
    s_t &= s_{t|t} + K_t (m_t - H_t s_{t|t}) \\
    P_t &= P_{t|t} - K_t H_t P_{t|t}
\end{align*}
\]

\[
K_t = P_{t|t} H_t^\top \left( H_t P_{t|t} H_t^\top + R_t \right)^{-1}
\]

* The first components are close to 1, which will give more weight to the measurement

The Final Estimate

- We can now make a final state estimate
  - We combine the prediction and the measurement
  - We also compute the covariance in this estimate
    - This can be used to tell us how far we can trust the estimate
    - It is also used to make a prediction for the next frame

\[
\begin{align*}
    s_t &= A_{t-1} s_{t-1} + v_{t-1} \\
    m_t &= H_t s_t + w_t \\
    s_{t|t} &= A_{t-1} s_{t-1} \\
    P_t &= A_{t-1} P_{t-1} A_{t-1}^\top + Q_{t-1} \\
    K_t &= H_t P_t (H_t P_t H_t^\top + R_t)^{-1} \\
    s_t &= s_{t|t} + K_t (m_t - H_t s_{t|t}) \\
    P_t &= P_{t|t} - K_t H_t P_{t|t}
\end{align*}
\]
The State Estimate

\[ s_t = A_t s_{t-1} + v_{t-1} \]
\[ m_t = H_t s_t + w_t \]
\[ s^*_t = \hat{A}_t s_{t-1} \]
\[ P^*_t = A_t P_{t-1} A_t^T + Q_{t-1} \]
\[ K_t = H_t P_t^* (H_t^T H_t + R_t)^{-1} \]
\[ s_t = s^*_t + K_t (m_t - H_t s_t) \]
\[ P_t = P_t^* - K_t H_t P_t \]

The State Covariance

\[ P_t = P_t^* + K_t H_t P_t \]
\[ m_t = H_t s_t + w_t \]
\[ s^*_t = \hat{A}_t s_{t-1} \]
\[ P^*_t = A_t P_{t-1} A_t^T + Q_{t-1} \]
\[ K_t = H_t P_t^* (H_t^T H_t + R_t)^{-1} \]
\[ s_t = s^*_t + K_t (m_t - H_t s_t) \]
\[ P_t = P_t^* - K_t H_t P_t \]
Iteration

- We repeat this computation for each frame
  - Over time the state predictions become more accurate
  - The Kalman gain takes this into account and places more weight on the predictions
- To implement the Kalman filter
  - We need a lot of matrix routines
  - These are tiresome to code by hand, but there are several libraries (Matlab, Mathematic) available
  - Only need basic operations: +, −, ×, transpose, inverse

Questions?