Exercise 4 | **Line-based Extended Kalman Filter for Robot Localization**

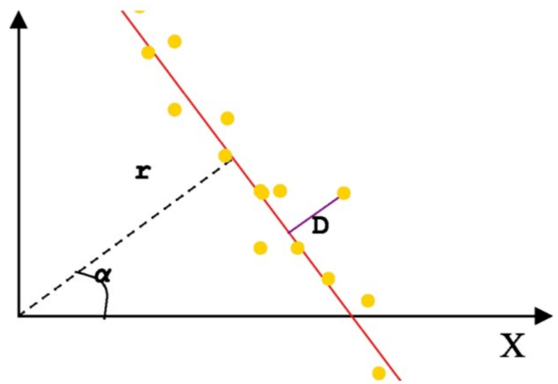
Lukas Bernreiter and Hermann Blum

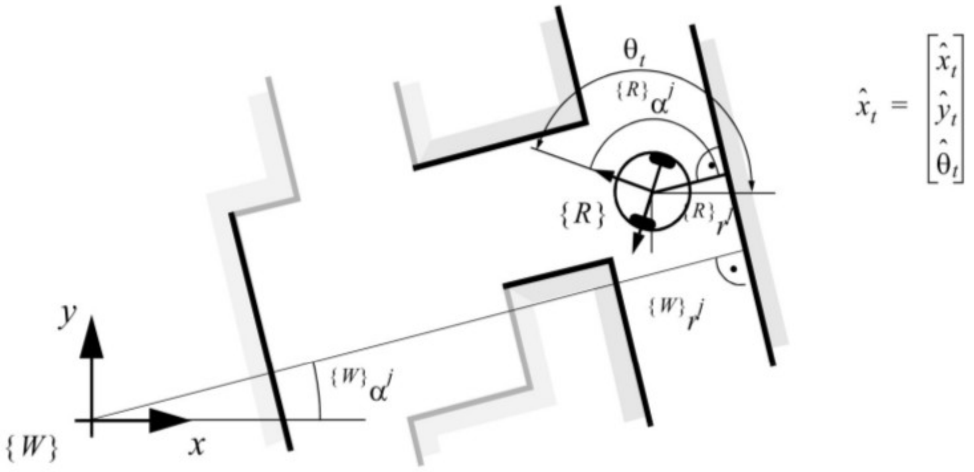
Line extraction, **EKF**, SLAM

Exercise 3
- Line extraction
- Line fitting

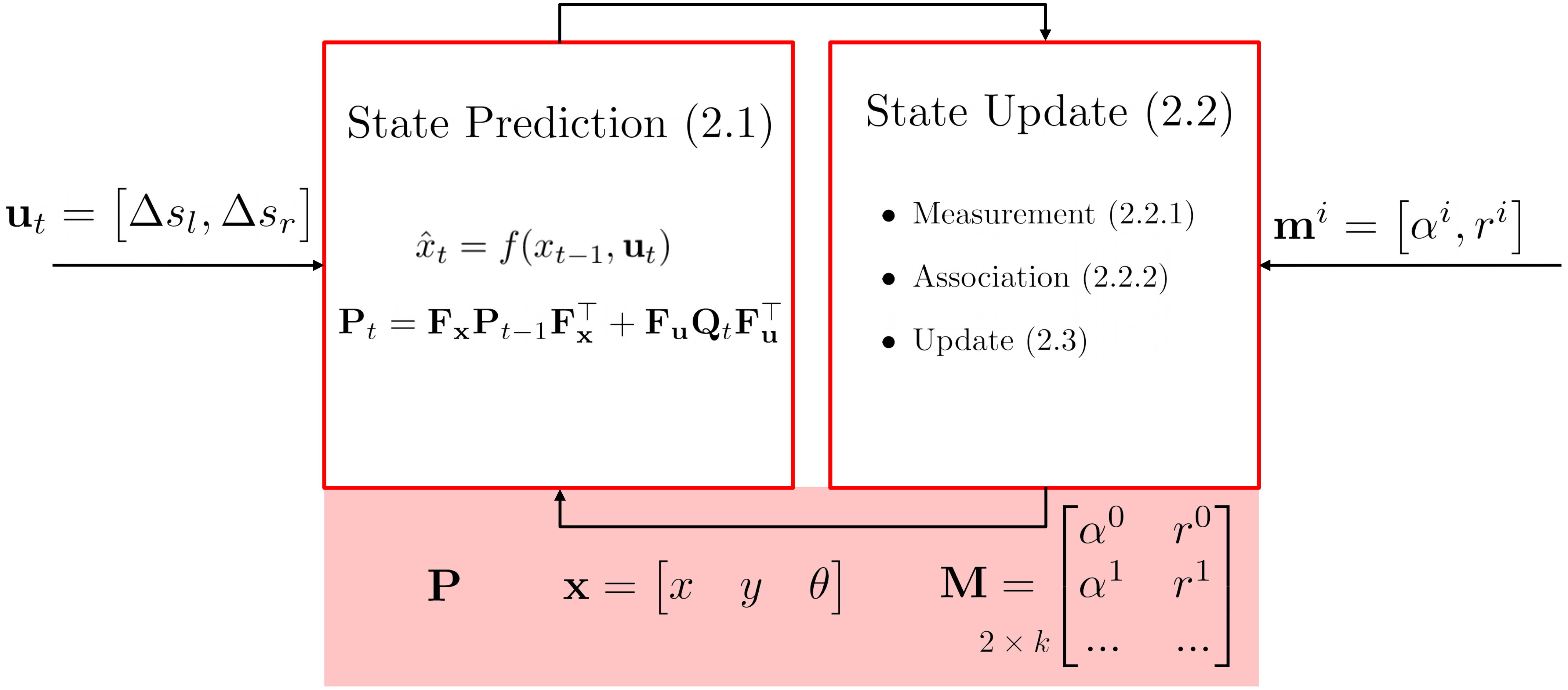
Exercise 4
- EKF
- Localization: Line extraction, given map
- Wheel odometry

Exercise 5
- Simultaneous Localization and Mapping (SLAM)
- Unknown environment (a-priori)



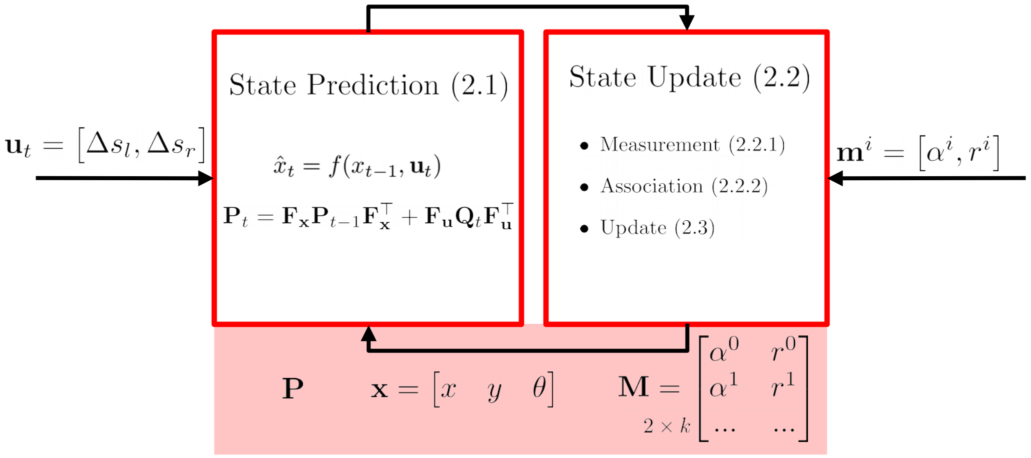


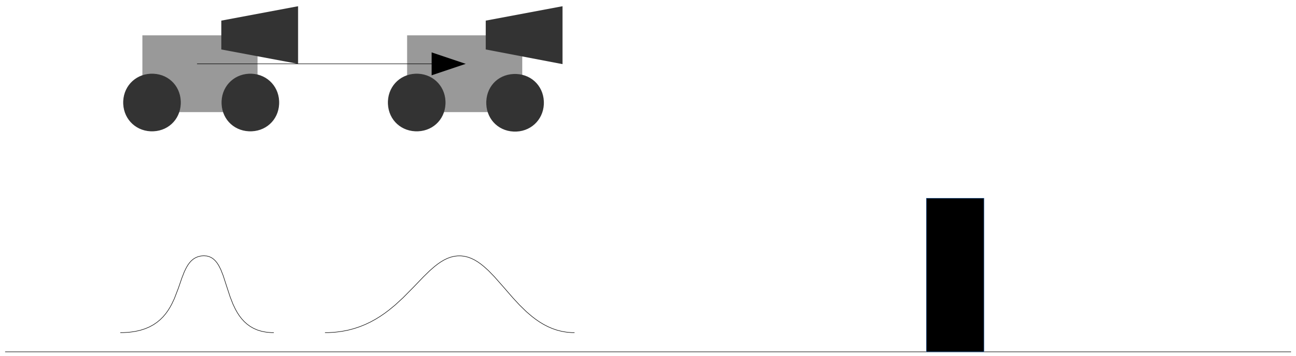
Kalman Filtering Overview



Example: EKF Localization

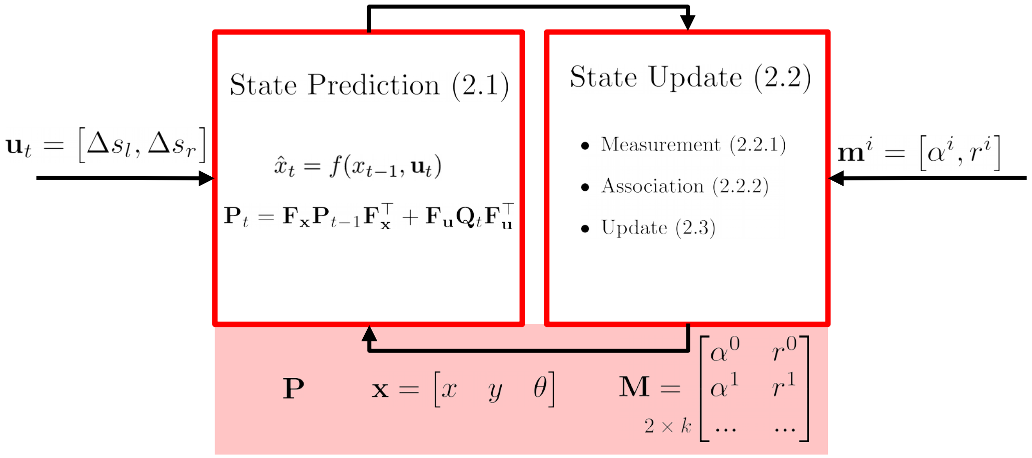
1. State Prediction

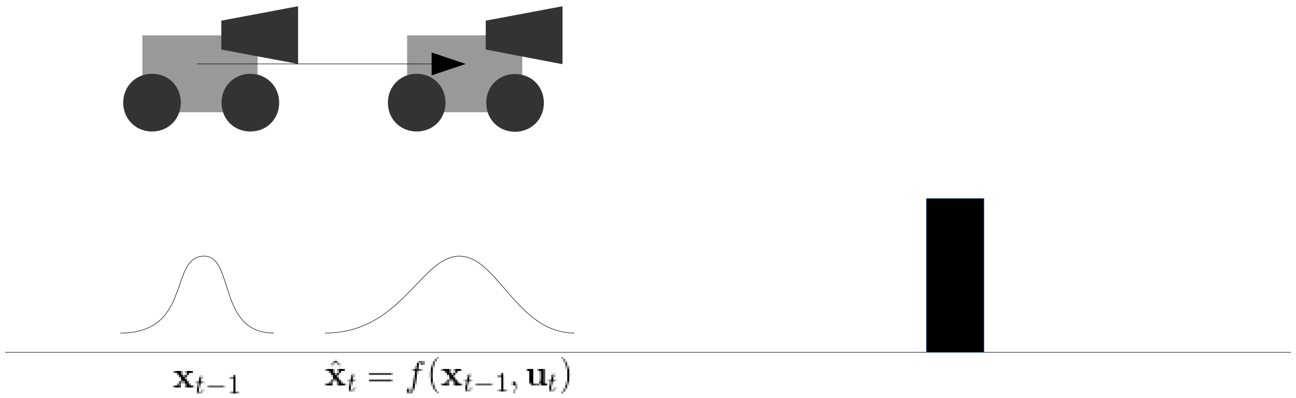




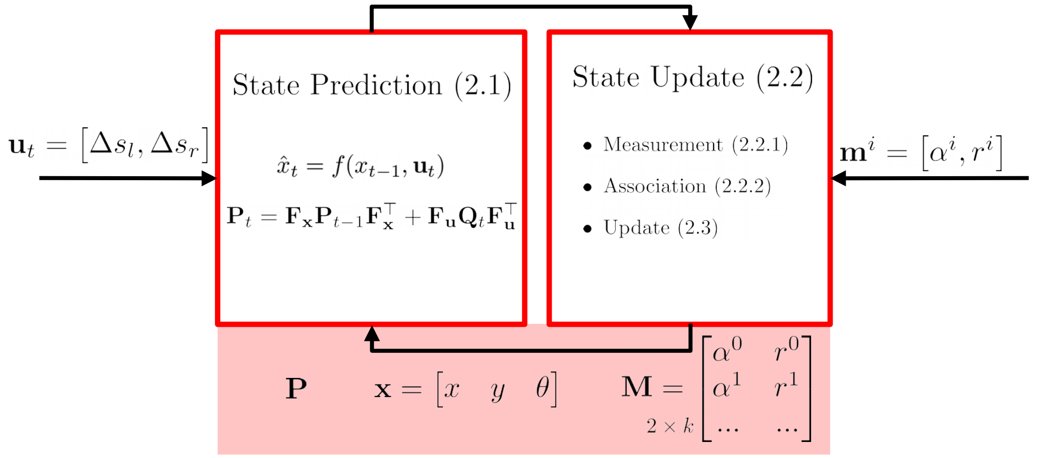
Example: EKF Localization

1. State Prediction

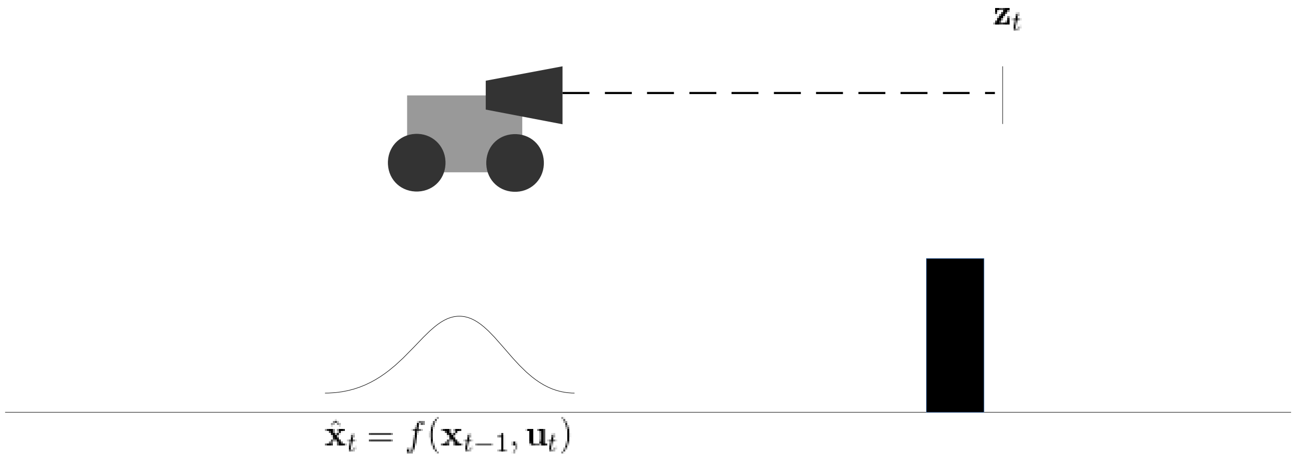




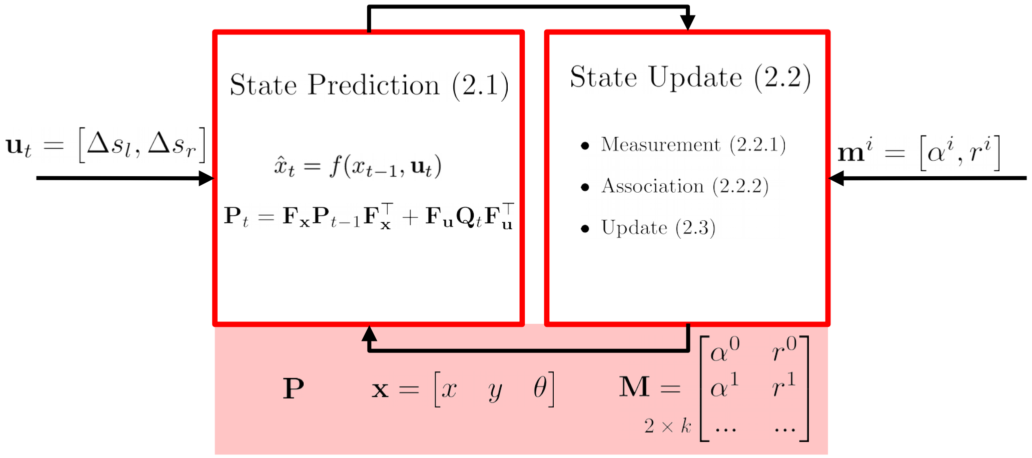
Example: EKF Localization



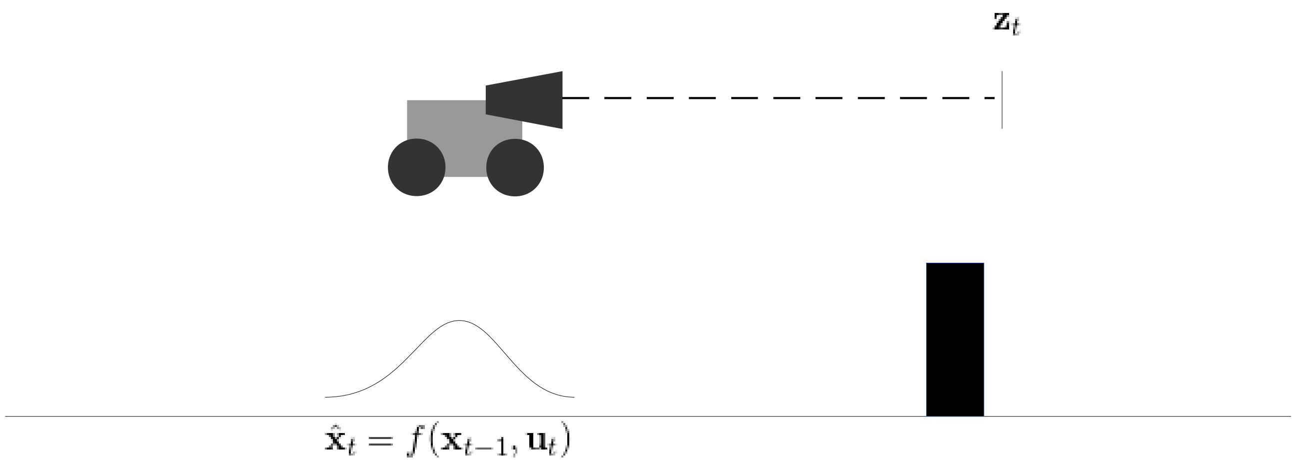
1. State Prediction
2. **Measurement**



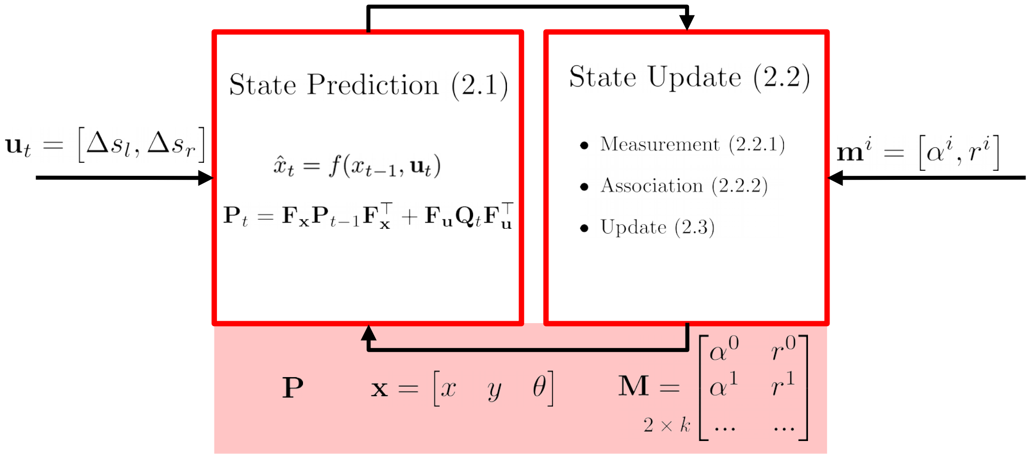
Example: EKF Localization



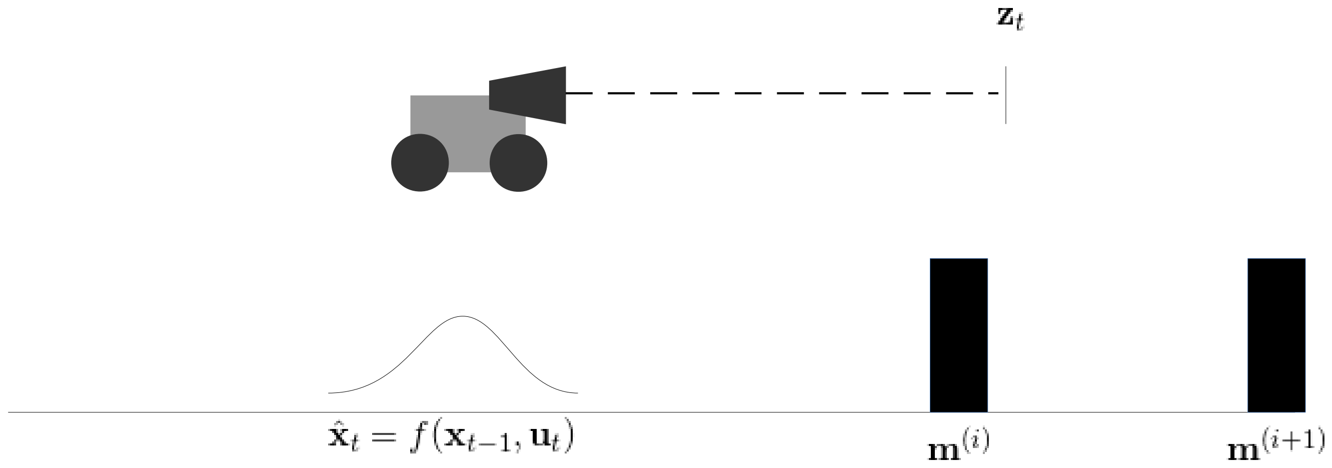
1. State Prediction
2. Measurement
3. **Association**



Example: EKF Localization

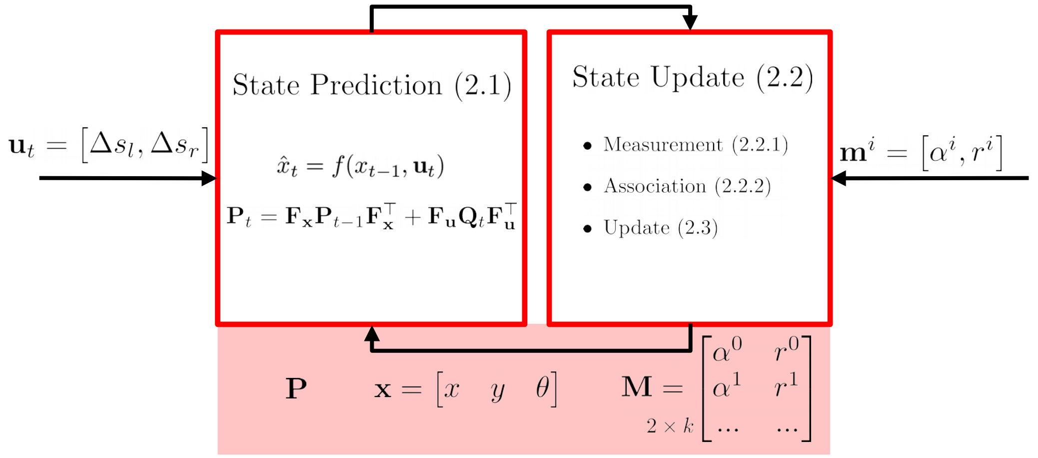


1. State Prediction
2. Measurement
3. **Association**

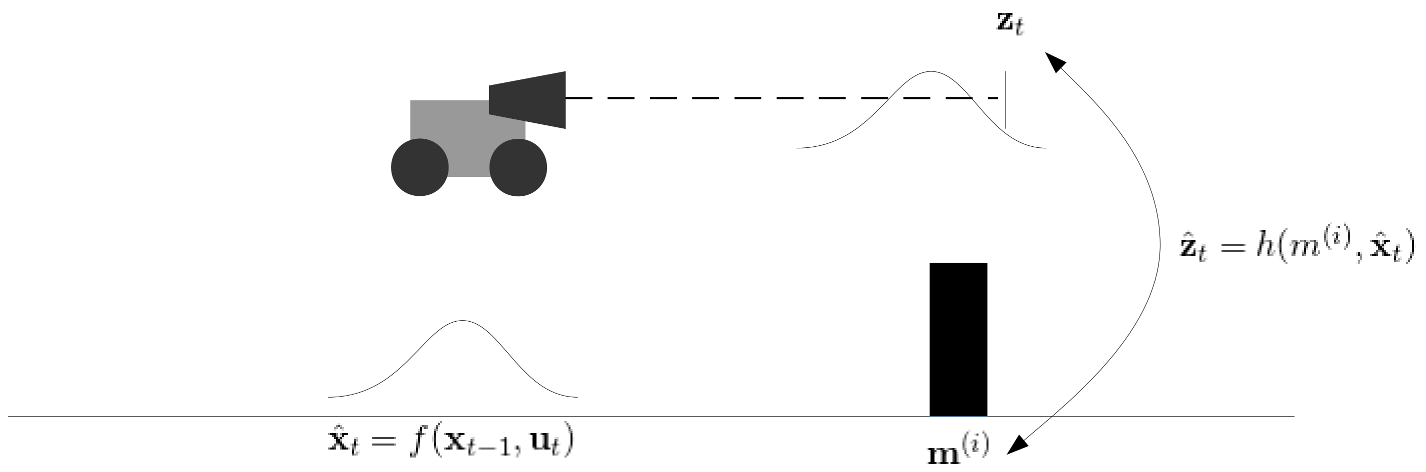


Which association is more likely? How can we measure this?

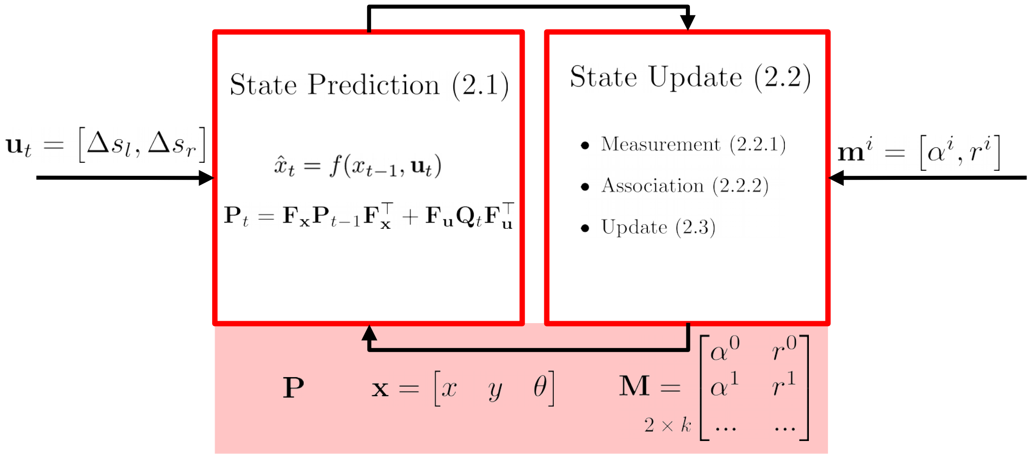
Example: EKF Localization



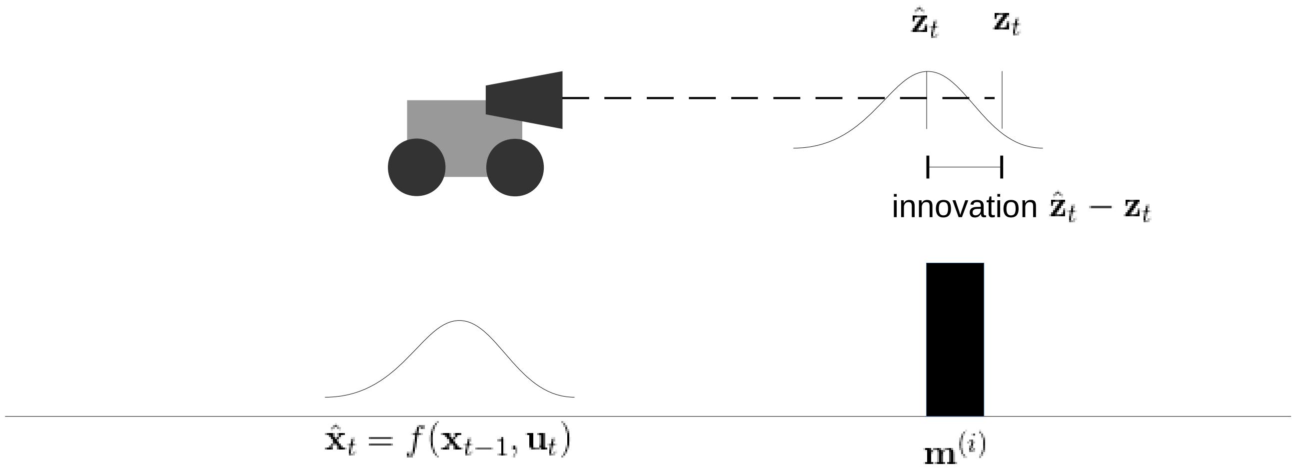
1. State Prediction
2. Measurement
3. **Association**



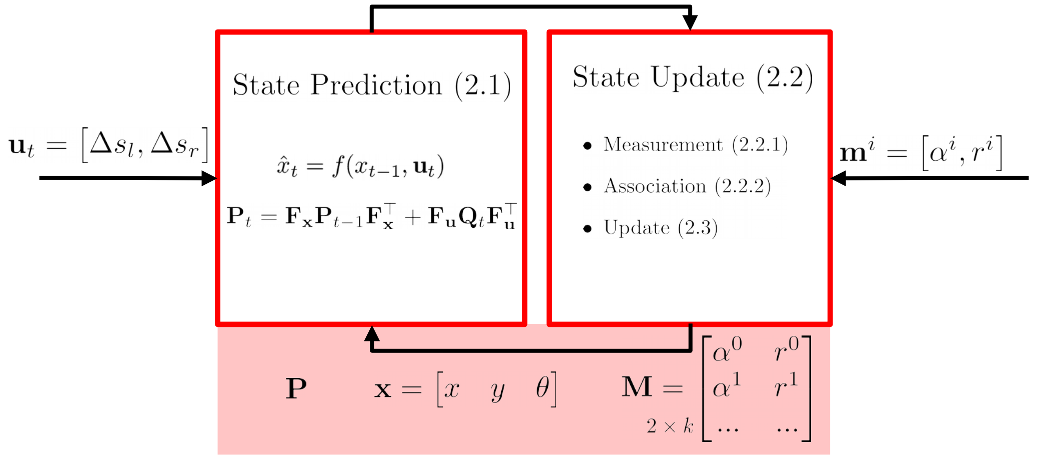
Example: EKF Localization



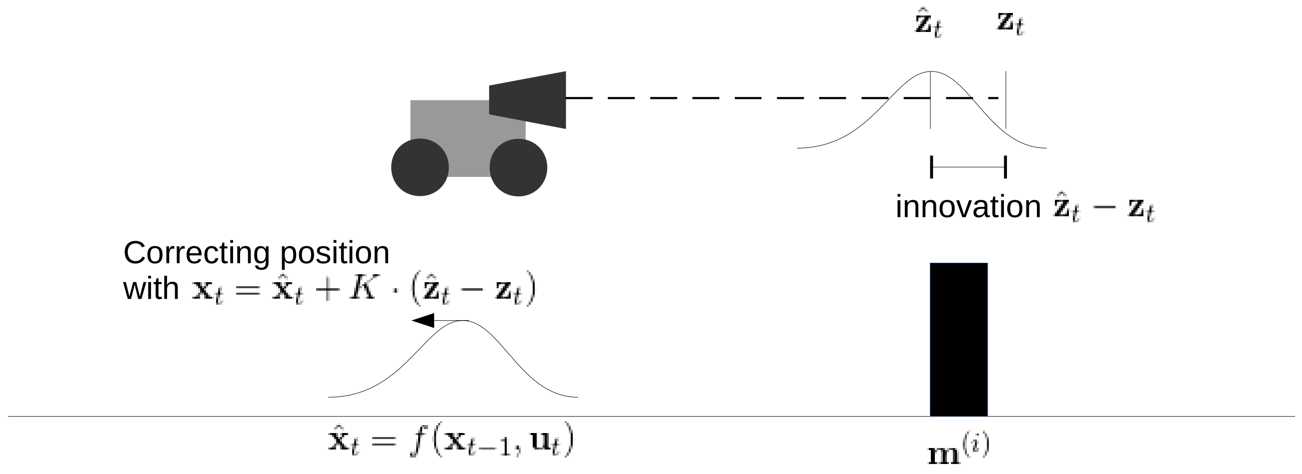
1. State Prediction
2. Measurement
3. Association
4. **Measurement Update**



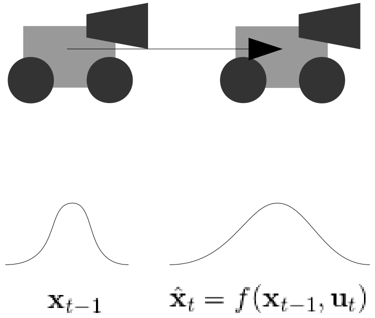
Example: EKF Localization



1. State Prediction
2. Measurement
3. Association
4. **Measurement Update**



Prediction/Time Update



- Propagate the system's state to the next time step

$$\hat{x}_t = f(x_{t-1}, \mathbf{u}_t) = x_{t-1} + \begin{bmatrix} \frac{\Delta s_l + \Delta s_r}{2} \cos\left(\theta_{t-1} + \frac{\Delta s_r - \Delta s_l}{2b}\right) \\ \frac{\Delta s_l + \Delta s_r}{2} \sin\left(\theta_{t-1} + \frac{\Delta s_r - \Delta s_l}{2b}\right) \\ \frac{\Delta s_r - \Delta s_l}{b} \end{bmatrix}$$

- Propagate the covariance

$$\text{A-Priori Cov.:}\quad \mathbf{P}_t = \mathbf{F_x} \mathbf{P}_{t-1} \mathbf{F_x}^\top + \mathbf{F_u} \mathbf{Q}_t \mathbf{F_u}^\top$$

$$\mathbf{F_x} = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial \theta} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial \theta} \\ \frac{\partial f_3}{\partial x} & \frac{\partial f_3}{\partial y} & \frac{\partial f_3}{\partial \theta} \end{bmatrix}$$

Jac. motion model wrt of state

$$\mathbf{F_u} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} \\ \frac{\partial f_3}{\partial u_1} & \frac{\partial f_3}{\partial u_2} \end{bmatrix}$$

Jac. of motion model wrt control input

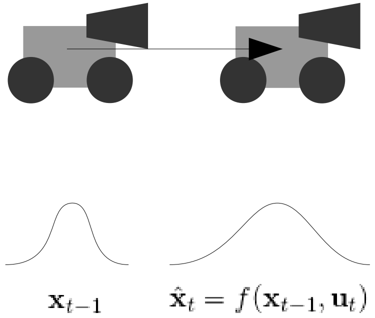
$$\mathbf{Q} = \begin{bmatrix} k|\Delta s_l| & 0 \\ 0 & k|\Delta s_r| \end{bmatrix}$$

State & Covariance Prediction/Propagation

$$\hat{x}_t = f(x_{t-1}, \mathbf{u}_t) = x_{t-1} + \begin{bmatrix} \frac{\Delta s_l + \Delta s_r}{2} \cos\left(\theta_{t-1} + \frac{\Delta s_r - \Delta s_l}{2b}\right) \\ \frac{\Delta s_l + \Delta s_r}{2} \sin\left(\theta_{t-1} + \frac{\Delta s_r - \Delta s_l}{2b}\right) \\ \frac{\Delta s_r - \Delta s_l}{b} \end{bmatrix} \qquad \mathbf{P}_t = \mathbf{F_x} \mathbf{P}_{t-1} \mathbf{F_x}^\top + \mathbf{F_u} \mathbf{Q}_t \mathbf{F_u}^\top$$

$$\mathbf{F_x} = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial \theta} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial \theta} \\ \frac{\partial f_3}{\partial x} & \frac{\partial f_3}{\partial y} & \frac{\partial f_3}{\partial \theta} \end{bmatrix} \quad \mathbf{F_u} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} \\ \frac{\partial f_3}{\partial u_1} & \frac{\partial f_3}{\partial u_2} \end{bmatrix} \quad \mathbf{Q} = \begin{bmatrix} k|\Delta s_l| & 0 \\ 0 & k|\Delta s_r| \end{bmatrix}$$

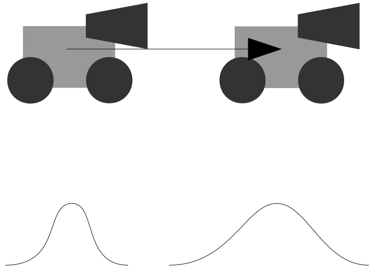
$$\mathbf{u}_t = [\Delta s_l, \Delta s_r]$$



- Task 1:
  - $[\hat{\mathbf{x}}_t, \hat{\mathbf{F}}_x, \hat{\mathbf{F}}_u] = transitionFunction(\mathbf{x}_{t-1}, \mathbf{u}_t, b)$
  - Derive Jacobian analytically
  - $validateTransitionFunction()$

State & Covariance Prediction/Propagation

$$\hat{x}_t = f(x_{t-1}, \mathbf{u}_t) = x_{t-1} + \begin{bmatrix} \frac{\Delta s_l + \Delta s_r}{2} \cos\left(\theta_{t-1} + \frac{\Delta s_r - \Delta s_l}{2b}\right) \\ \frac{\Delta s_l + \Delta s_r}{2} \sin\left(\theta_{t-1} + \frac{\Delta s_r - \Delta s_l}{2b}\right) \\ \frac{\Delta s_r - \Delta s_l}{b} \end{bmatrix} \qquad \mathbf{P}_t = \mathbf{F_x} \mathbf{P}_{t-1} \mathbf{F_x}^\top + \mathbf{F_u} \mathbf{Q}_t \mathbf{F_u}^\top$$



$$\mathbf{F_x} = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial \theta} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial \theta} \\ \frac{\partial f_3}{\partial x} & \frac{\partial f_3}{\partial y} & \frac{\partial f_3}{\partial \theta} \end{bmatrix} \quad \mathbf{F_u} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} \\ \frac{\partial f_3}{\partial u_1} & \frac{\partial f_3}{\partial u_2} \end{bmatrix} \quad \mathbf{Q} = \begin{bmatrix} k|\Delta s_l| & 0 \\ 0 & k|\Delta s_r| \end{bmatrix}$$

$$\mathbf{u}_t = [\Delta s_l, \Delta s_r]$$

$$\mathbf{F_x} = \begin{bmatrix} 1 & 0 & -\frac{u_1+u_2}{2} \sin(\theta + \frac{u_2-u_1}{2b}) \\ 0 & 1 & \frac{u_1+u_2}{2} \cos(\theta + \frac{u_2-u_1}{2b}) \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{F_u^{11}} = \frac{1}{2}\cos(\theta + \frac{u_2 - u_1}{2b}) + \frac{1}{2b}\sin(\theta + \frac{u_2 - u_1}{2b})\frac{u_1 + u_2}{2}$$

$$\mathbf{F_u^{12}} = \frac{1}{2}\cos(\theta + \frac{u_2 - u_1}{2b}) - \frac{1}{2b}\sin(\theta + \frac{u_2 - u_1}{2b})\frac{u_1 + u_2}{2}$$

$$\mathbf{F_u^{21}} = \frac{1}{2}\sin(\theta + \frac{u_2 - u_1}{2b}) - \frac{1}{2b}\cos(\theta + \frac{u_2 - u_1}{2b})\frac{u_1 + u_2}{2}$$

$$\mathbf{F_u^{22}} = \frac{1}{2}\sin(\theta + \frac{u_2 - u_1}{2b}) + \frac{1}{2b}\cos(\theta + \frac{u_2 - u_1}{2b})\frac{u_1 + u_2}{2}$$

$$\mathbf{F_u^{31}} = -1/b$$

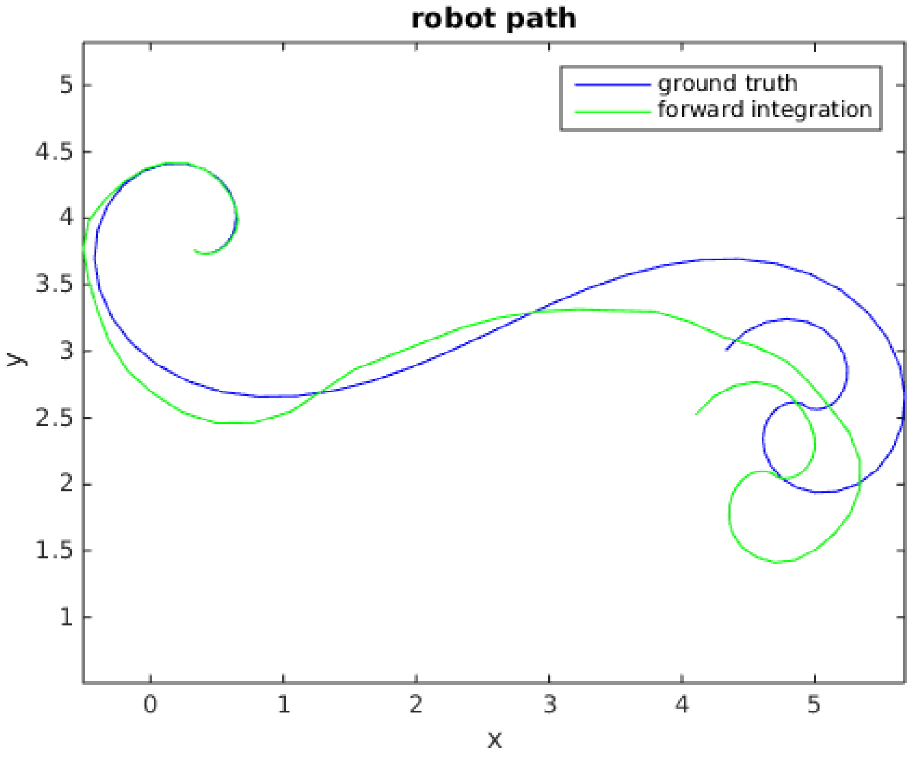
$$\mathbf{F_u^{32}} = 1/b$$

Task 1: Prediction of State and Covariance

```
f = x + [ ...
    (u(1)+u(2))/2 * cos(x(3) + (u(2)-u(1))/(2*b) )
    (u(1)+u(2))/2 * sin(x(3) + (u(2)-u(1))/(2*b) )
    (u(2)-u(1))/b ...
    ];

F_x = [...
    1, 0, -sin(x(3) - (u(1) - u(2))/(2*b))*(u(1)/2 + u(2)/2)
    0, 1,  cos(x(3) - (u(1) - u(2))/(2*b))*(u(1)/2 + u(2)/2)
    0, 0,  1 ...
    ];

F_u = [...
    cos(x(3) - (u(1) - u(2))/(2*b))/2 + (sin(x(3) - (u(1) - u(2))/(2*b))*(u(1)/2 + u(2)/2))/(2*b),
    cos(x(3) - (u(1) - u(2))/(2*b))/2 - (sin(x(3) - (u(1) - u(2))/(2*b))*(u(1)/2 + u(2)/2))/(2*b)
    sin(x(3) - (u(1) - u(2))/(2*b))/2 - (cos(x(3) - (u(1) - u(2))/(2*b))*(u(1)/2 + u(2)/2))/(2*b),
    sin(x(3) - (u(1) - u(2))/(2*b))/2 + (cos(x(3) - (u(1) - u(2))/(2*b))*(u(1)/2 + u(2)/2))/(2*b)
    -1/b,
    1/b ...
    ];
```

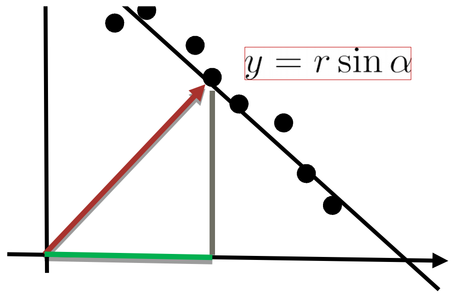
State & Covariance Prediction/Propagation

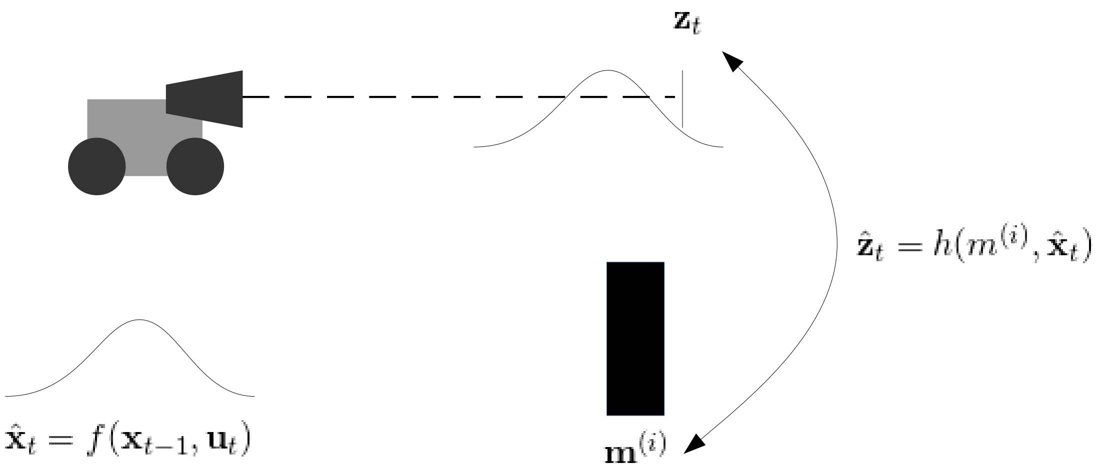
- We typically accumulate an error (drift) if we only do prediction
  - How does it affect the uncertainty of the state?



State & Covariance Update: Measurement Function

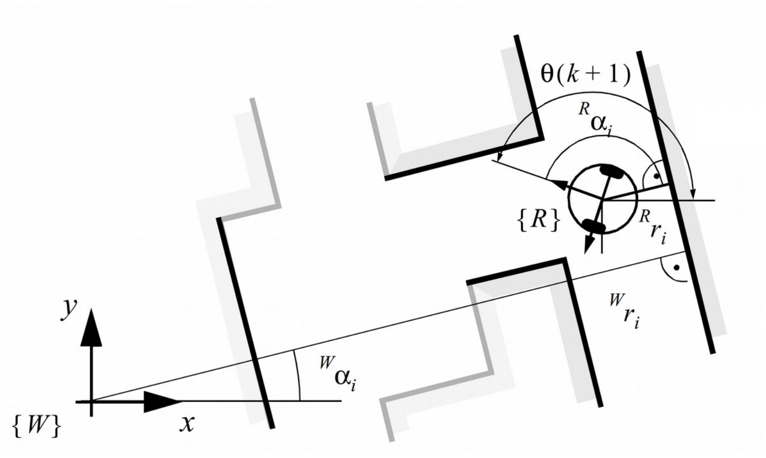
- Line parametrization: $m^i = \left[\alpha^i \; r^i\right]^\top$

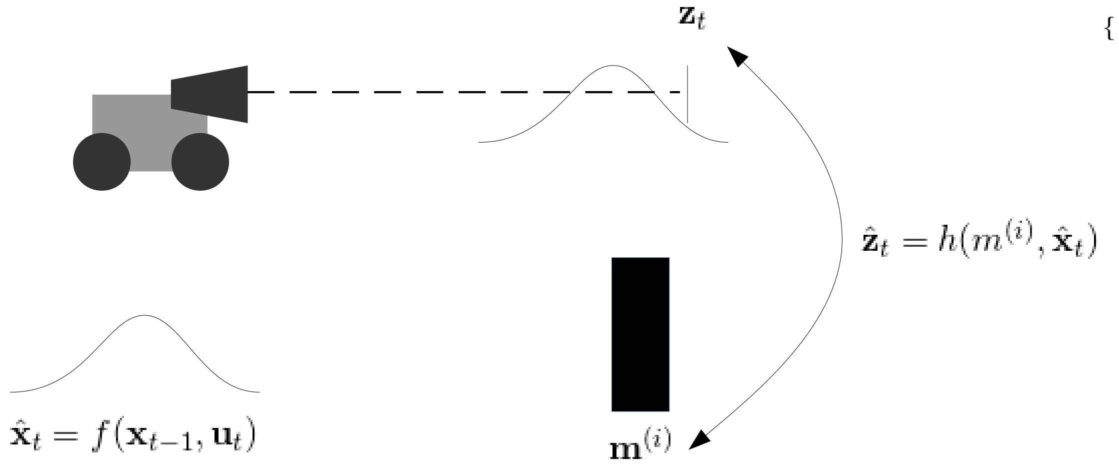




State & Covariance Update: Measurement Function

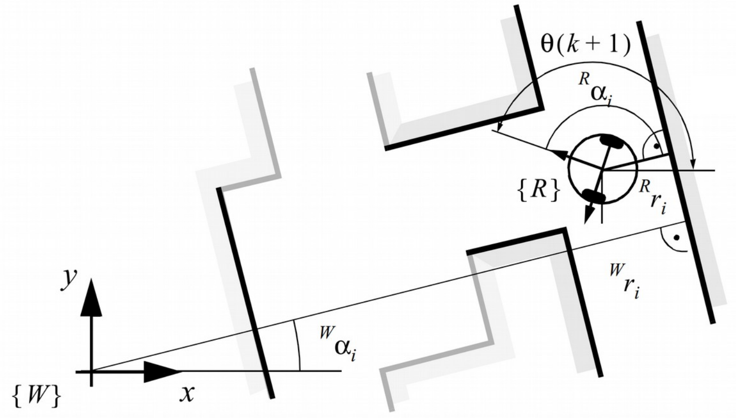
- Line parametrization: $m^i = \left[\alpha^i \; r^i\right]^\top$
  - Lines in the map w.r.t. the world frame
  - Lines from the sensor w.r.t. to the body frame





State & Covariance Update: Measurement Function

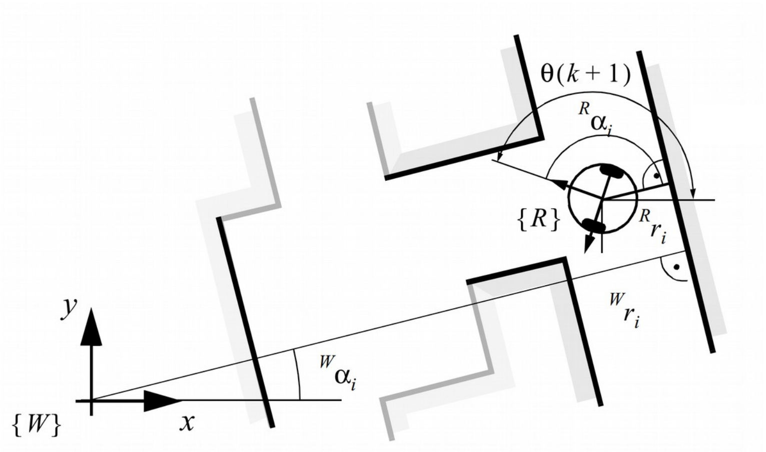
- Task 2:



- $[\hat{\mathbf{z}}_t, \hat{\mathbf{H}}_\mathbf{t}] = measurementFunction(\hat{\mathbf{x}}_t, \mathbf{m}^i)$
- $\hat{\mathbf{z}}_t$: Predicted observation
- $\hat{\mathbf{H}}_\mathbf{t}$: Jacobian of measurement model with respect to state
- $validateMeasurementFunction()$

State & Covariance Update: Measurement Function

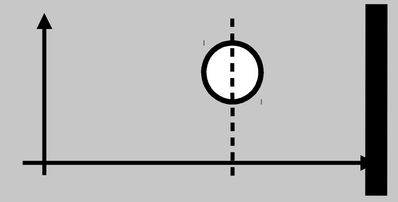
- Line parametrization: $m^i = \left[\alpha^i \; r^i\right]^\top$
  - Lines in the map w.r.t. the world frame
  - Lines from the sensor w.r.t. to the body frame



- Predicted measurement

$$\hat{z}^j = \begin{bmatrix} {}^R\hat{\alpha}^j \\ {}^R\hat{r}^j \end{bmatrix} = h^j(\hat{x}_t, m^j) = \begin{bmatrix} {}^W\alpha^j - \hat{\theta}_t \\ {}^Wr^j - (\hat{x}_t \cos({}^W\alpha^j) + \hat{y}_t \sin({}^W\alpha^j)) \end{bmatrix}$$

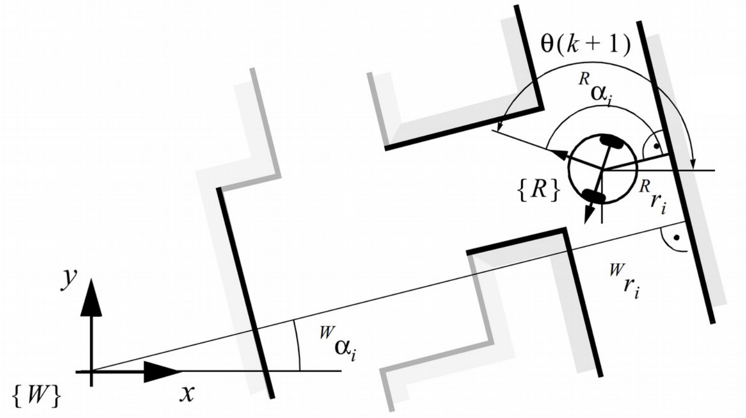
$${}^W\alpha^j = 0$$



$$\hat{z}^j = \begin{bmatrix} \hat{\alpha}^j \\ \hat{r}_j \end{bmatrix} = \begin{bmatrix} -\hat{\theta} \\ {}^Wr^j - \hat{x} \end{bmatrix}$$

State & Covariance Update: Measurement Function

- Task 2: derivation of the measurement Jacobian



$$\hat{z}^j = \begin{bmatrix} {}^R\hat{\alpha}^j \\ {}^R\hat{r}^j \end{bmatrix} = h^j(\hat{x}_t, m^j) = \begin{bmatrix} {}^W\alpha^j - \hat{\theta}_t \\ {}^W r^j - (\hat{x}_t \cos({}^W\alpha^j) + \hat{y}_t \sin({}^W\alpha^j)) \end{bmatrix}$$

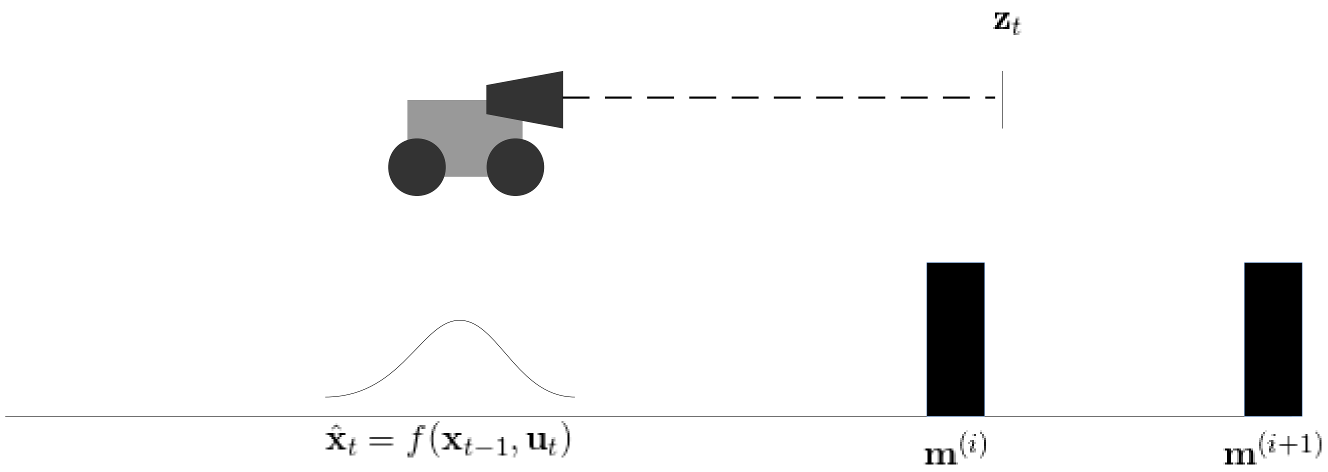
$$\mathbf{H} = \begin{bmatrix} \frac{\partial h_1}{\partial x} & \frac{\partial h_1}{\partial y} & \frac{\partial h_1}{\partial \theta} \\ \frac{\partial h_2}{\partial x} & \frac{\partial h_2}{\partial y} & \frac{\partial h_2}{\partial \theta} \end{bmatrix} = \begin{bmatrix} 0 & 0 & -1 \\ -\cos({}^W\alpha^j) & -\sin({}^W\alpha^j) & 0 \end{bmatrix}$$

Task 2: Measurement Model

```
h = [...
    m(1) - x(3)
    m(2) - (x(1)*cos(m(1)) + x(2)*sin(m(1)))
    ];

H_x = [...
    0,           0,          -1
    -cos(m(1)), -sin(m(1)),   0
    ];
```

State & Covariance Update: Measurement Association

- Task 3:
  - $[\hat{\mathbf{v}}_t, \hat{\mathbf{H}}_\mathbf{t}, \mathbf{R}_t] = associateMeasurements(\hat{\mathbf{x}}_t, \hat{\mathbf{P}}_t, \mathbf{Z}_t, \mathbf{R}_t, \mathbf{M}, g)$
  - $validateAssociations()$



Which association is more likely? How can we measure this?

State & Covariance Update: Measurement Association

- Task 3:
  - $[\hat{\mathbf{v}}_t, \hat{\mathbf{H}}_{\mathbf{t}}, \mathbf{R}_t] = associateMeasurements(\hat{\mathbf{x}}_t, \hat{\mathbf{P}}_t, \mathbf{Z}_t, \mathbf{R}_t, \mathbf{M}, g)$
  - $validateAssociations()$

$$\mathbf{v}_t^{ij} = \mathbf{z}_t^j - \hat{\mathbf{z}}_t^i$$

Predicted Measurement

Innovation

Observed Measurement

State & Covariance Update: Measurement Association

- Task 3:
  - $[\hat{\mathbf{v}}_t, \hat{\mathbf{H}}_\mathbf{t}, \mathbf{R}_t] = associateMeasurements(\hat{\mathbf{x}}_t, \hat{\mathbf{P}}_t, \mathbf{Z}_t, \mathbf{R}_t, \mathbf{M}, g)$
  - $validateAssociations()$

$$\mathbf{v}_t^{ij} = \mathbf{z}_t^j - \hat{\mathbf{z}}_t^i$$

Innovation

Predicted Measurement

Observed Measurement

$$\Sigma_{IN_t}^{ij} = \hat{\mathbf{H}}_t^i \, \hat{\mathbf{P}}_t (\hat{\mathbf{H}}_t^i)^\top + \mathbf{R}_t^j$$

Innovation covariance

Measurement Covariance

State & Covariance Update: Measurement Association

- Task 3:
  - $[\hat{\mathbf{v}}_t, \hat{\mathbf{H}}_\mathbf{t}, \mathbf{R}_t] = associateMeasurements(\hat{\mathbf{x}}_t, \hat{\mathbf{P}}_t, \mathbf{Z}_t, \mathbf{R}_t, \mathbf{M}, g)$
  - $validateAssociations()$

$$\mathbf{v}_t^{ij} = \mathbf{z}_t^j - \hat{\mathbf{z}}_t^i$$

Innovation

Predicted Measurement

Observed Measurement

$$\Sigma_{IN_t}^{ij} = \hat{\mathbf{H}}_t^i \; \hat{\mathbf{P}}_t (\hat{\mathbf{H}}_t^i)^\top + \mathbf{R}_t^j$$

Innovation covariance

Measurement Covariance

$$d_t^{ij} = (\mathbf{v}_t^{ij})^\top (\Sigma_{IN_t}^{ij})^{-1} \mathbf{v}_t^{ij}$$

Mahalanobis distance

$$d_t^{ij} < g^2$$

Validation gate

Task 3: Associate Measurements

```
nMeasurements = size(Z,2);
nMapEntries = size(M,2);
d = zeros(nMeasurements, nMapEntries);
v = zeros(2, nMeasurements * nMapEntries);
H = zeros(2, 3, nMeasurements * nMapEntries);
for i = 1 : nMeasurements
    for j = 1 : nMapEntries
        [z_priori, H(:, :, j + (i-1) * nMapEntries)] = measurementFunction(x, M(:,j));
        v(:,j + (i-1) * nMapEntries) = Z(:,i) - z_priori;
        W = H(:, :, j + (i-1) * nMapEntries) * P * H(:, :, j + (i-1) * nMapEntries)' + R(:,:,i);
        d(i,j) = v(:,j + (i-1) * nMapEntries)' * inv(W) * v(:,j + (i-1) * nMapEntries);
    end
end

% line feature matching (pp. 341)

% association of each measurement to the map point with minimal distance
[minima, map_index] = min(d');
[measurement_index] = find(minima < g^2);
map_index = map_index(measurement_index);

v = v(:, map_index + (measurement_index-1)*nMapEntries);
H = H(:, :, map_index + (measurement_index-1)*nMapEntries);
R = R(:, :, measurement_index);
```

State & Covariance Update: Filtering

- Task 4:
  - $[\mathbf{x}_t, \mathbf{P_t}] = filterStep(\mathbf{x}_{t-1}, \mathbf{P}_{t-1}, \mathbf{u}_t, \mathbf{Z}_t, \mathbf{R}_t, \mathbf{M}, g, b)$
  - $validateFilter()$
  - $incrementalLocalization()$

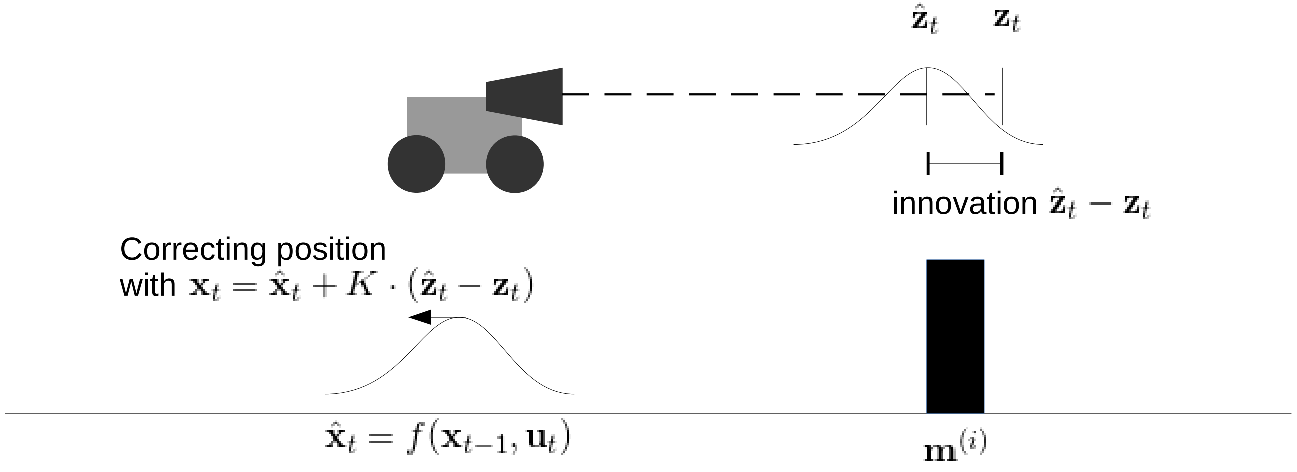
$$\mathbf{S} = \mathbf{H}\mathbf{P}_{t-1}\mathbf{H}^\top + \mathbf{R}$$

$$\boxed{\mathbf{K}} = \mathbf{P}_t\mathbf{H}^\top\mathbf{S}^{-1}$$

Kalman Gain

$$\mathbf{P}_t = (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{P}_{t-1}$$

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{K}\mathbf{v}$$



State & Covariance Update: Filtering

- Task 4:
  - $[\mathbf{x}_t, \mathbf{P_t}] = filterStep(\mathbf{x}_{t-1}, \mathbf{P}_{t-1}, \mathbf{u}_t, \mathbf{Z}_t, \mathbf{R}_t, \mathbf{M}, g, b)$
  - $validateFilter()$
  - $incrementalLocalization()$

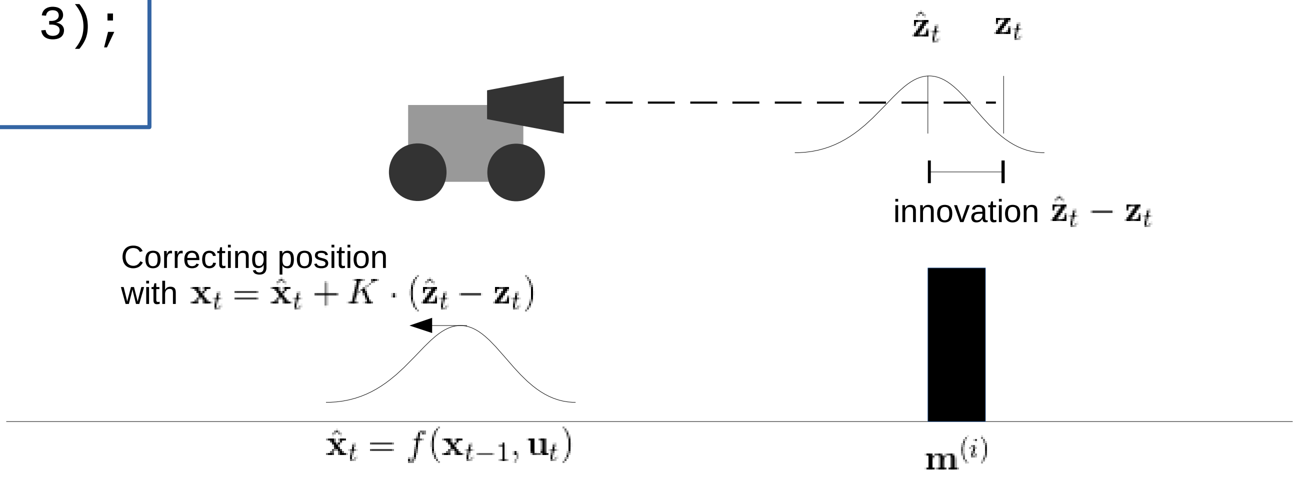
$$\mathbf{S} = \mathbf{H}\mathbf{P}_{t-1}\mathbf{H}^\top + \mathbf{R}$$

$$\underbrace{\mathbf{K}}_{\text{Kalman Gain}} = \mathbf{P}_t\mathbf{H}^\top\mathbf{S}^{-1}$$

$$\mathbf{P}_t = (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{P}_{t-1}$$

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{K}\mathbf{v}$$

```
y = reshape(v, [], 1);
H = reshape(permute(H, [1,3,2]), [], 3);
R = blockDiagonal(R);
```

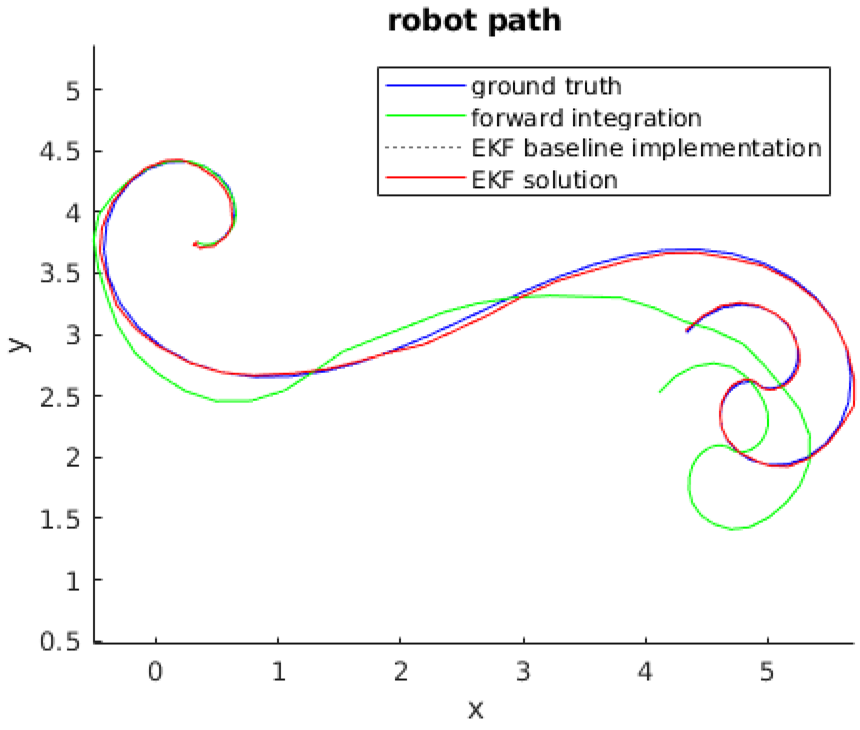


Task 4: Filtering

```
% propagate the state (p. 337)
Q = k*diag(abs(u));

[x_priori, F_x, F_u] = transitionFunction(x, u, b);
P_priori = F_x * P * F_x' + F_u * Q * F_u';

if size(Z,2) == 0
    x_posteriori = x_priori;
    P_posteriori = P_priori;
    return;
end

[v, H, R] = associateMeasurements(x_priori, P_priori, Z, R, M, g);

y = reshape(v, [], 1);
H = reshape(permute(H, [1,3,2]), [], 3);
R = blockDiagonal(R);

% update state estimates (pp. 335)
S = H * P_priori * H' + R;
K = P_priori * (H' / S);
P_posteriori = (eye(size(P_priori)) - K*H) * P_priori;
x_posteriori = x_priori + K * y;
```

State & Covariance Update: Filtering

- Using the update state we can correct the drift



Task 5: V-REP

```
[z, R, ~] = extractLinesPolar(S(1,:), S(2,:), C_TR, params);

figure(2), cla, hold on;
z_prior = zeros(size(M));
for k = 1:size(M,2)
    z_prior(:,k) = measurementFunction(x, M(:,k));
end
plot(z(1,:), z(2,:),'bo');
plot(z_prior(1,:), z_prior(2,:),'rx');

xlabel('angle [rad]'); ylabel('distance [m]')
legend('measurement','prior')
drawnow;

% estimate robot pose
[x_posterori, P_posterori] = filterStep(x, P, u, z, R, M, k, g, b);
```