COLUMBIA UNIVERSITY EEME E6911 FALL '25

TOPICS IN CONTROL : PROBABILISTIC ROBOTICS

EKF LOCALIZATION

Instructor: Ilija Hadzic

Summary

- Use motion model for prediction:
  - System input (throttle, commands, etc.)
  - Velocity estimate (form odometry)
  - Accumulated odometry
- Use measurement model for update:
  - Direct pose measurement (e.g .GPS, etc.)
  - Raw sensor signals
  - Known or unknown correspondences
- Linearize both models using Jacobians
- Apply Kalman filter equations

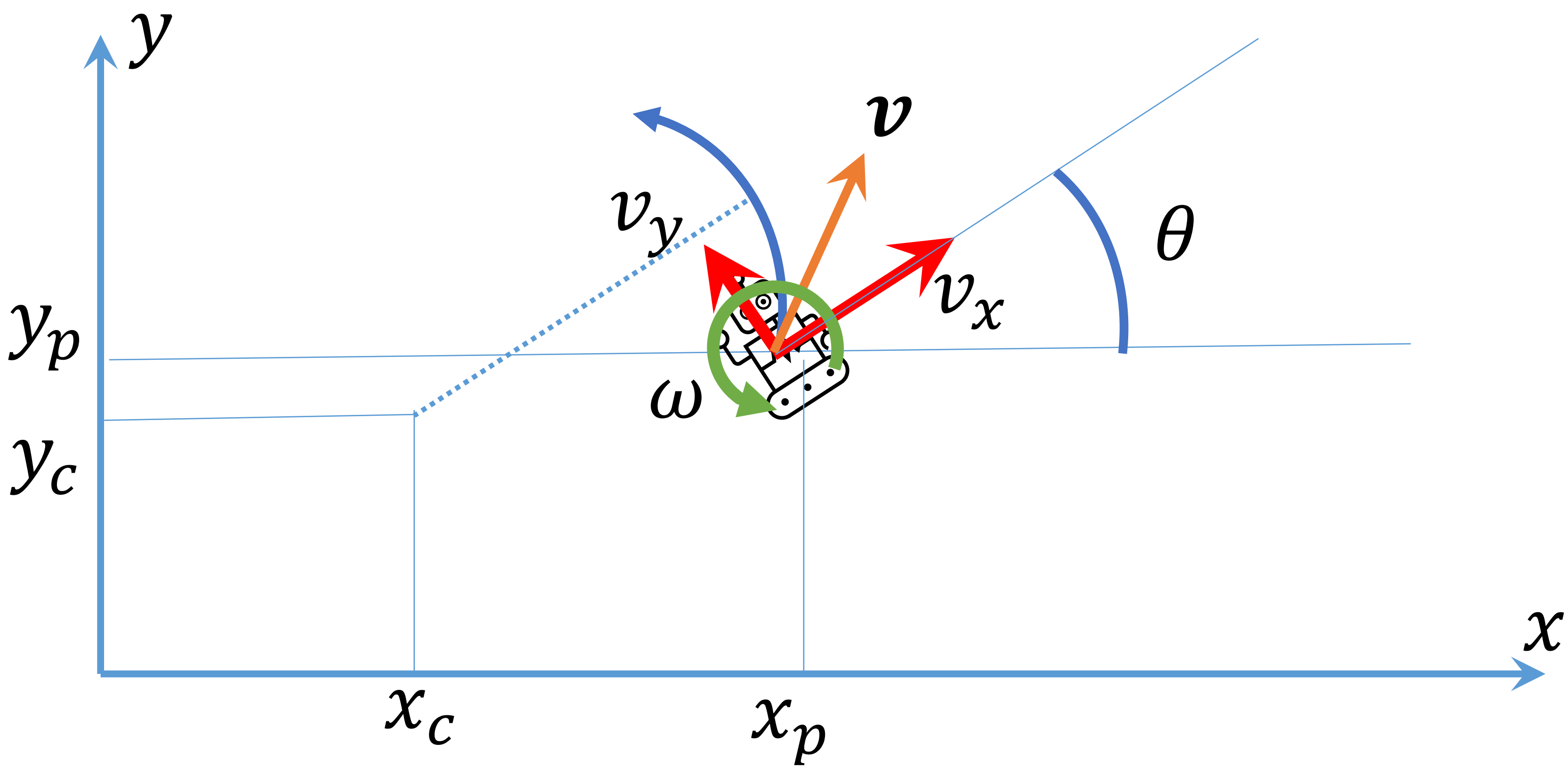
Motion Model

- Recall 2D model from odometry class:

$$\begin{bmatrix} \theta[n] \\ x[n] \\ y[n] \end{bmatrix} = \begin{bmatrix} \theta[n-1] + \omega[n]\Delta t \\ v_{\mathrm{x}}[\mathrm{n}]\Delta \mathrm{t} \cdot \cos\theta[n-1] - v_y[n]\Delta \mathrm{t} \cdot \sin\theta[n-1] + x[n-1] \\ v_{\mathrm{x}}[\mathrm{n}]\Delta \mathrm{t} \cdot \sin\theta[n-1] + v_y[n]\Delta \mathrm{t} \cdot \cos\theta[n-1] + y[n-1] \end{bmatrix}$$

- We can use it for prediction if we know the velocity.
- What is the geometric interpretation of this model?

Motion Model – Tangent Motion

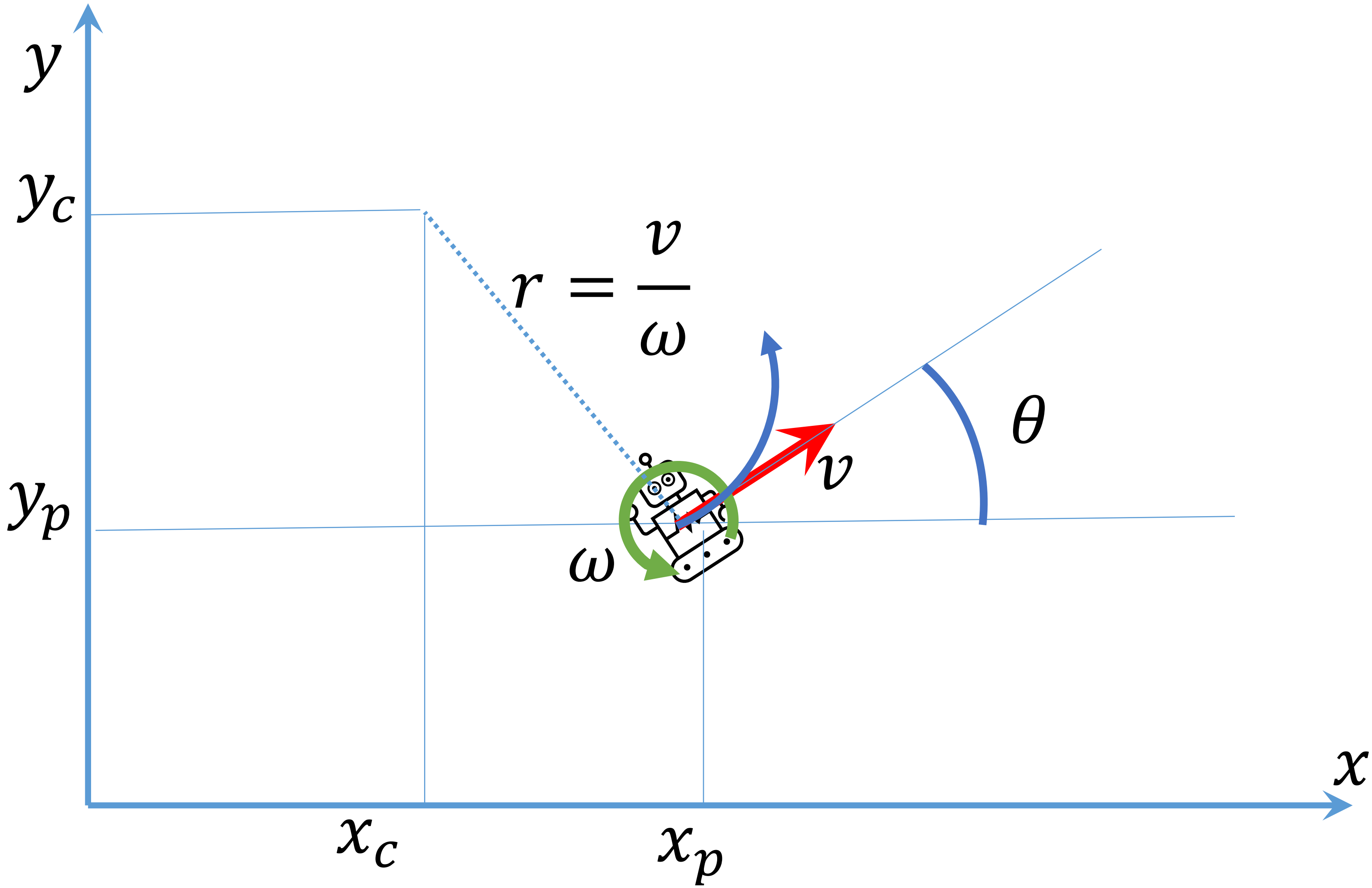


- Can we do better?
- Follow the motion curvature instead?
- Should we?

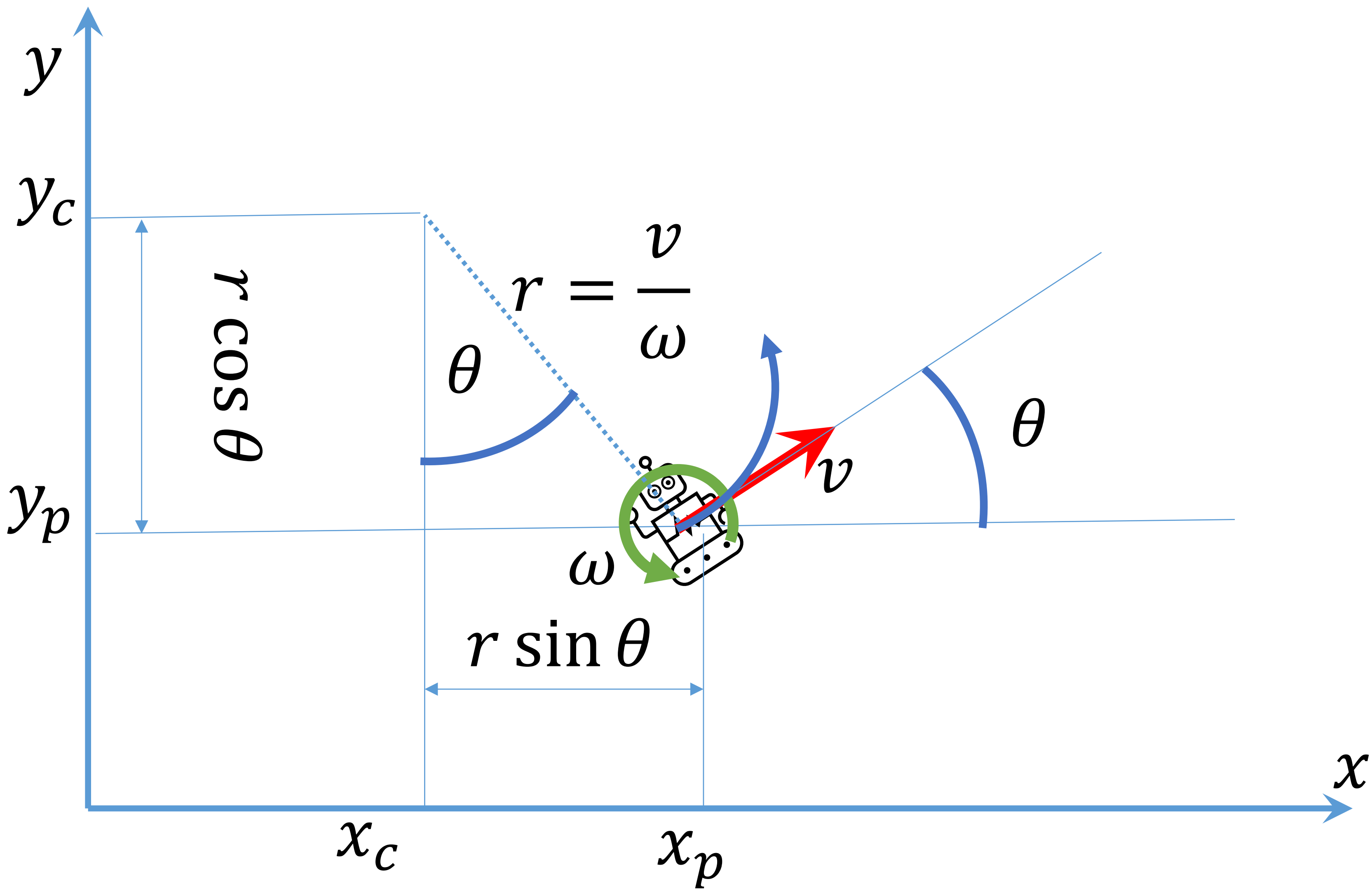
Motion Model – Arc Motion

- Follows the curvature of motion.
- Works better if Δt is long.
- Thrun, Section 5.3.3
- Careful:
  - Thrun derives the model for $v_y = 0$
  - Valid assumption for many drivetrains:
    - Differential, Skid-Steer, Ackermann, etc.
  - Does not work for omnidirectional drivetrains:
    - H-drive, X-drive, 4WIS-drive

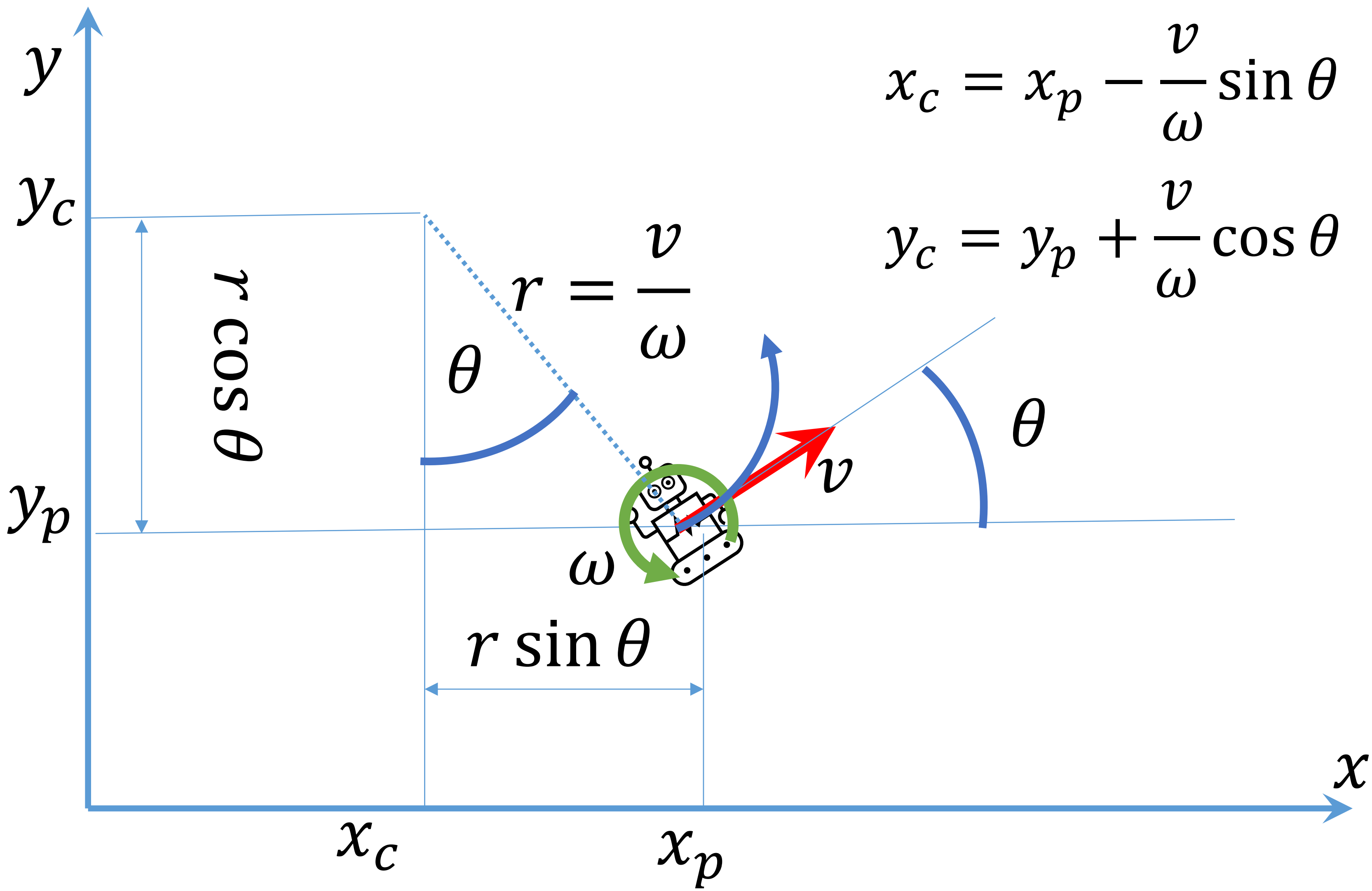
Motion Model – Arc Motion



Where is the center of rotation?



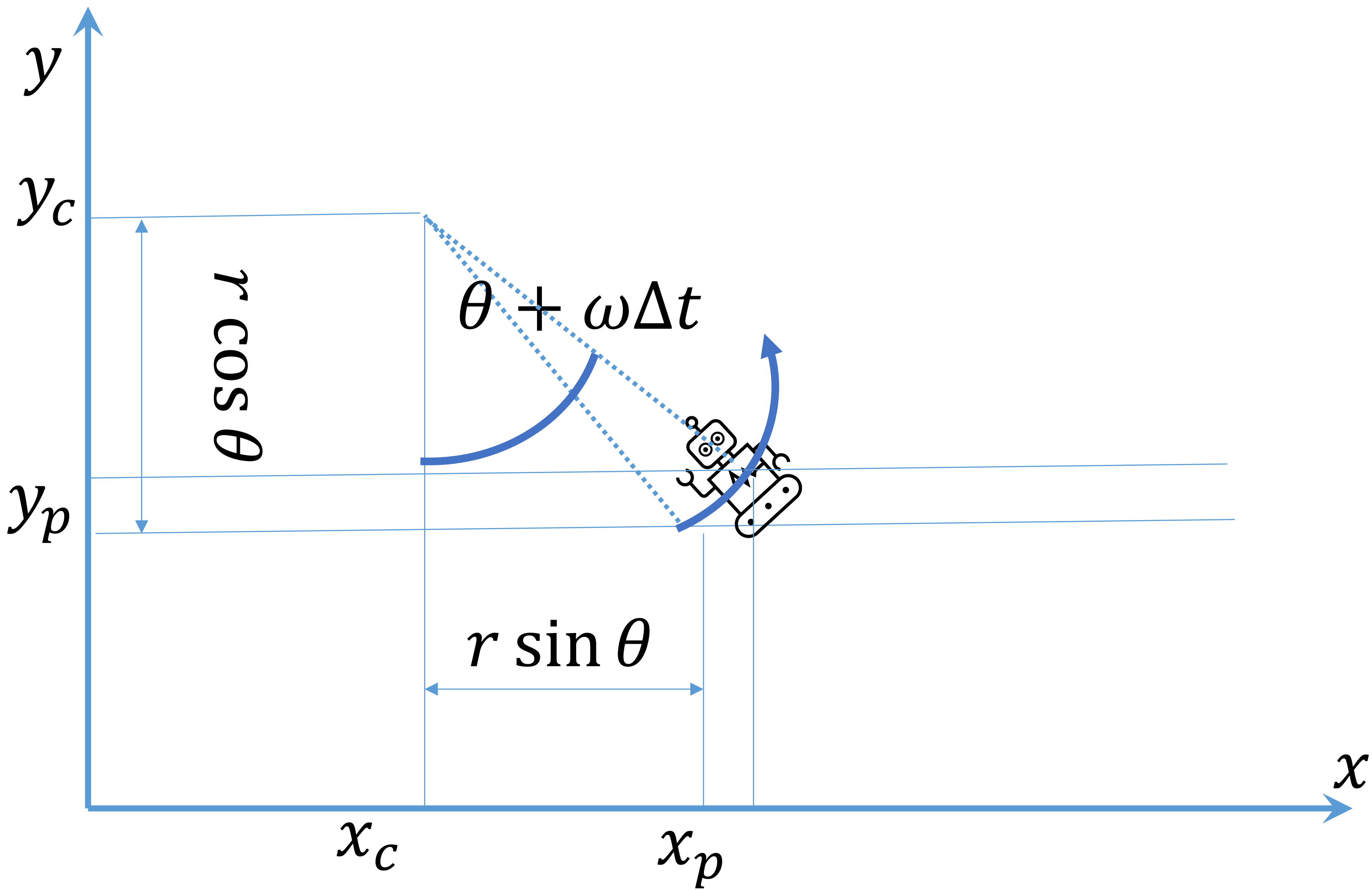
Where is the center of rotation?



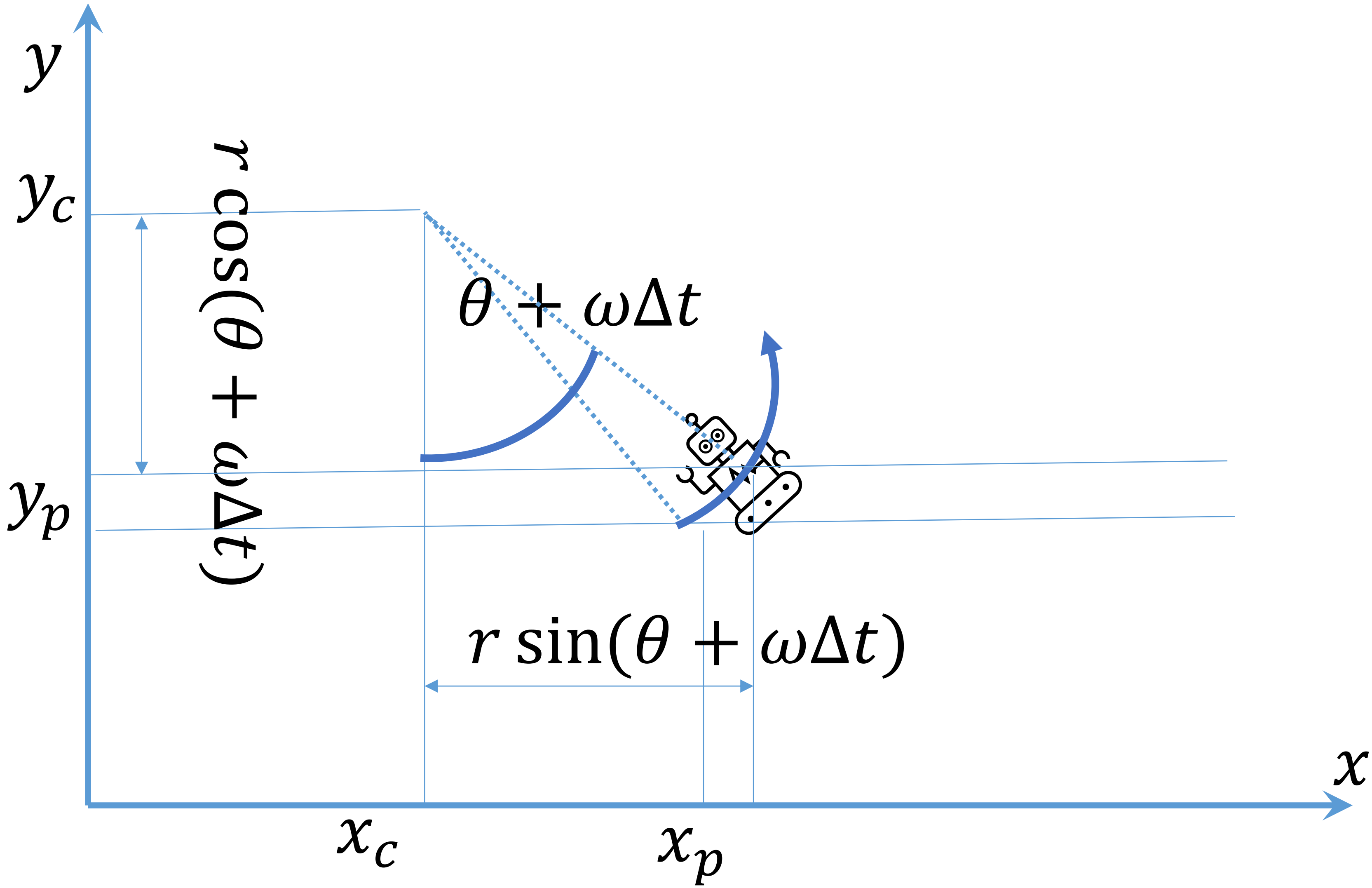
$$x_c = x_p - \frac{v}{\omega}\sin\theta$$

$$y_c = y_p + \frac{v}{\omega}\cos\theta$$

Do the arc motion!



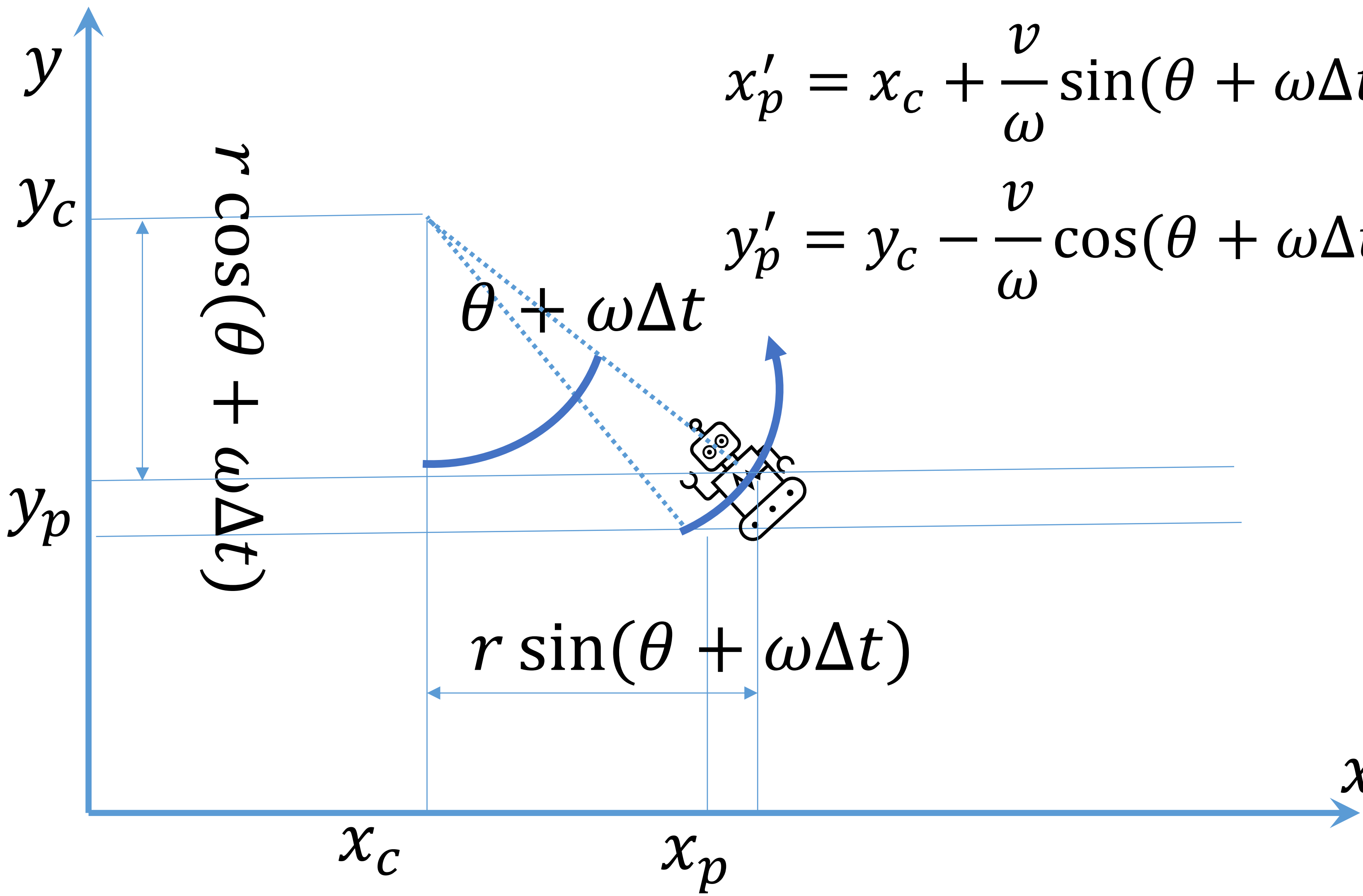
New coordinates relative to the center?



New coordinates relative to the center

$$x_p' = x_c + \frac{v}{\omega}\sin(\theta + \omega\Delta t)$$

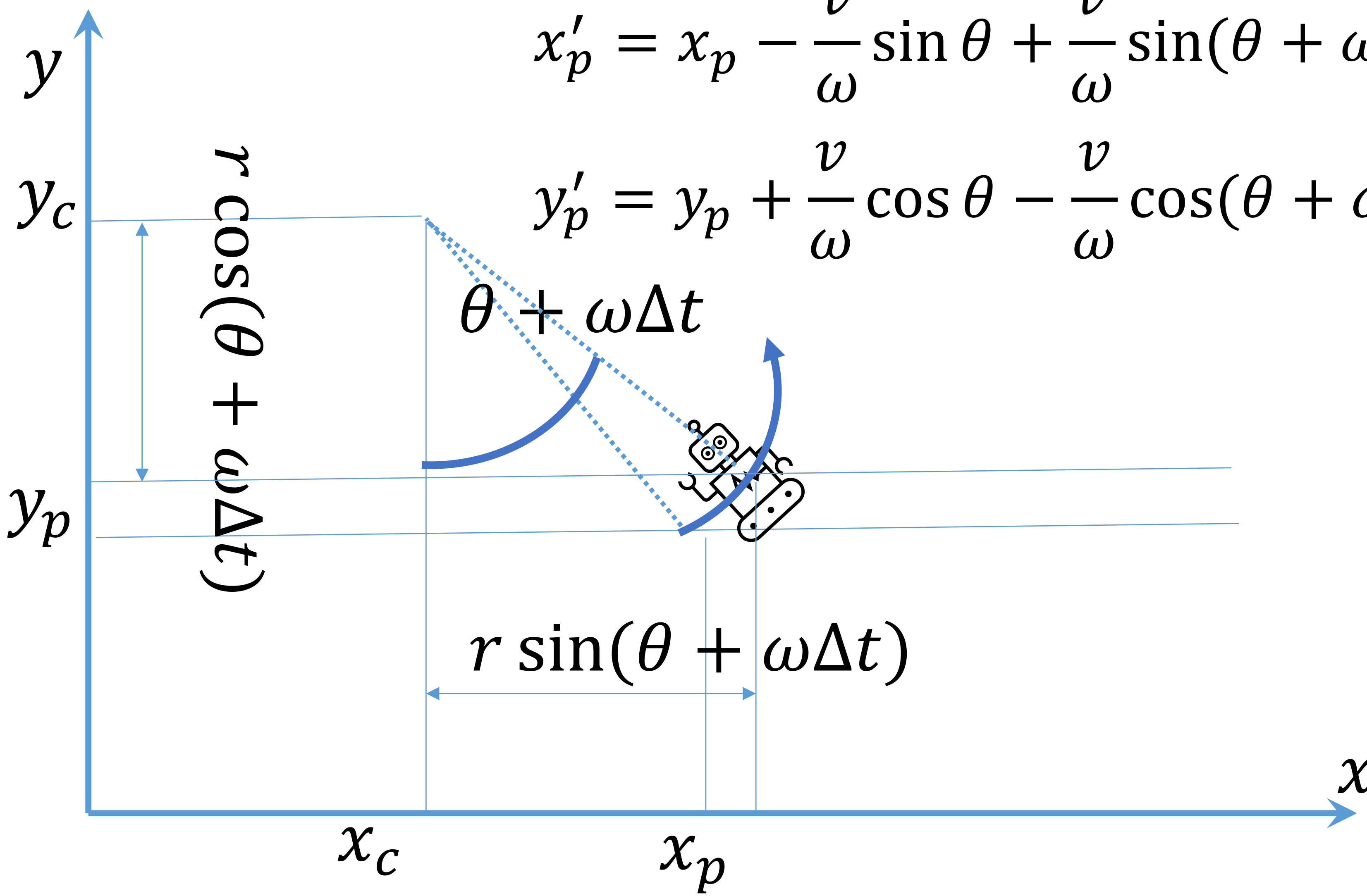
$$y_p' = y_c - \frac{v}{\omega}\cos(\theta + \omega\Delta t)$$



New coordinates in global frame

$$x_p' = x_p - \frac{v}{\omega}\sin\theta + \frac{v}{\omega}\sin(\theta + \omega\Delta t)$$

$$y_p' = y_p + \frac{v}{\omega}\cos\theta - \frac{v}{\omega}\cos(\theta + \omega\Delta t)$$



Motion Model – Arc Motion

$$\theta[n] = \theta[n-1] + \omega\Delta t$$

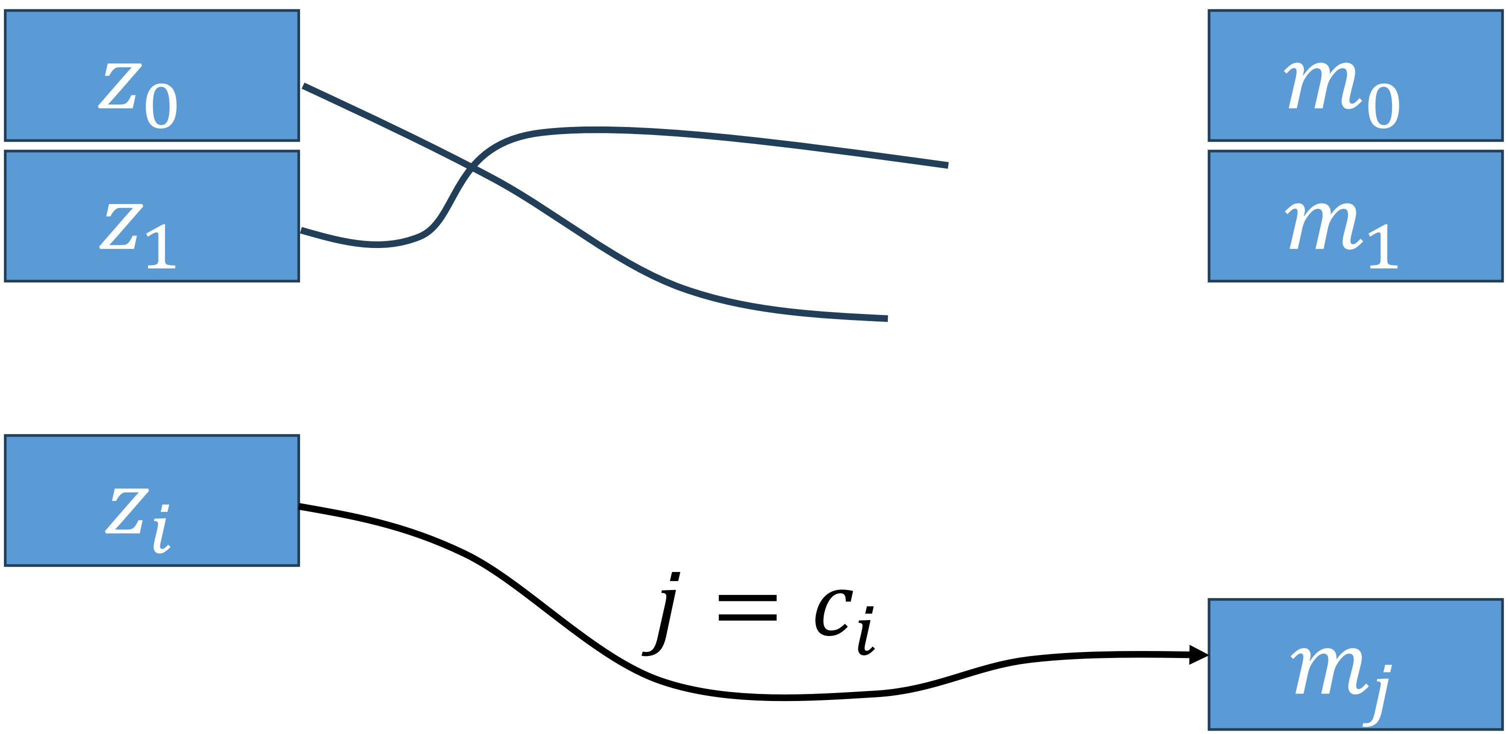
$$x[n] = x[n-1] - \frac{v}{\omega}\sin\theta + \frac{v}{\omega}\sin(\theta + \omega\Delta t)$$

$$\mathrm{y}[n] = y[n-1] + \frac{v}{\omega}\cos\theta - \frac{v}{\omega}\cos(\theta + \omega\Delta t)$$

- Find Jacobians
- Have everything needed for prediction
- Exercise: Derive motion model for $v_y \neq 0$

Correspondences

- Index table that relates measurements to the map
- At given time, we see different subset of features
- Subset changes in each time instant



Apply Measurements

- We have multiple measurements:

$$z_1, \Sigma_{z1}$$

$$z_2, \Sigma_{z2}$$

$$z_i, \Sigma_{zi}$$

- What are the options to incorporate them?

Apply Measurements

- We have multiple measurements:

$$\boldsymbol{z}_1, \boldsymbol{\Sigma}_{z1}$$

$$\boldsymbol{z}_2, \boldsymbol{\Sigma}_{z2}$$

$$\boldsymbol{z}_i, \boldsymbol{\Sigma}_{zi}$$

$$\boldsymbol{z} = [\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots \boldsymbol{z}_i, \ldots]^T$$

$$\boldsymbol{\Sigma}_z = \begin{bmatrix} \boldsymbol{\Sigma}_{z1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{z2} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Sigma}_{zi} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots \end{bmatrix}$$

- Construct one **big** measurement vector
- Yields one **big** Kalman gain matrix

Apply Measurements

- We have multiple measurements:

$$\boldsymbol{z}_1, \boldsymbol{\Sigma}_{z1}$$

$$\boldsymbol{z}_2, \boldsymbol{\Sigma}_{z2}$$

$$\boldsymbol{z}_i, \boldsymbol{\Sigma}_{zi}$$

$$\Rightarrow$$

$$\boldsymbol{z} = [\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots \boldsymbol{z}_i, \ldots]^T$$

$$\boldsymbol{\Sigma}_z = \begin{bmatrix} \boldsymbol{\Sigma}_{z1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{z2} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Sigma}_{zi} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots \end{bmatrix}$$

- Independent measurements → Sparse matrix
- Can we do better?

Apply Measurements

- We have multiple measurements:

$$\boldsymbol{z}_1, \boldsymbol{\Sigma}_{z1} \Rightarrow \boldsymbol{K}_1 = f(\boldsymbol{\Sigma}_{z1}, \boldsymbol{\Sigma}_{x0}) \Rightarrow \boldsymbol{x}_1, \boldsymbol{\Sigma}_{x1}$$

$$\boldsymbol{z}_2, \boldsymbol{\Sigma}_{z2}$$

$$\boldsymbol{z}_i, \boldsymbol{\Sigma}_{zi}$$

- Apply one measurement at the time
- Keep evolving the Kalman gain

Apply Measurements

- We have multiple measurements:

$\boldsymbol{z}_1, \boldsymbol{\Sigma}_{z1}$

$$\boldsymbol{z}_2, \boldsymbol{\Sigma}_{z2} \Rightarrow \boldsymbol{K}_2 = f(\boldsymbol{\Sigma}_{z2}, \boldsymbol{\Sigma}_{x1}) \Rightarrow \boldsymbol{x}_2, \boldsymbol{\Sigma}_{x2}$$

$\boldsymbol{z}_i, \boldsymbol{\Sigma}_{zi}$

- Apply one measurement at the time
- Keep evolving the Kalman gain

Apply Measurements

- We have multiple measurements:

$\boldsymbol{z}_1, \boldsymbol{\Sigma}_{z1}$

$\boldsymbol{z}_2, \boldsymbol{\Sigma}_{z2}$

$$\boldsymbol{z}_i, \boldsymbol{\Sigma}_{zi} \Rightarrow \boldsymbol{K}_i = f(\boldsymbol{\Sigma}_{zi}, \boldsymbol{\Sigma}_{xi-1}) \Rightarrow \boldsymbol{x_i}, \boldsymbol{\Sigma}_{xi}$$

- Apply one measurement at the time
- Keep evolving the Kalman gain

Careful!

- Measurements must be *independent*!
- Repeated measurements of the same landmark are *not* independent.
- Measurements of the same landmark with different sensor are independent.
- If you have measurement dependencies, you must use the “big matrix” method and figure out the cross-terms.

Full Algorithm

- Table 7.2, Page 204 in Thrun
- All steps should follow from the lecture
- Motion model is arc-motion for $v_y = 0$
- Few additional details:
  - Step 5: Assumption that uncertainty goes up with velocity
  - Step 21: Importance factor (matters for multi-state Kalman Filter).

Unknown Correspondences

- Recall from general EKF:

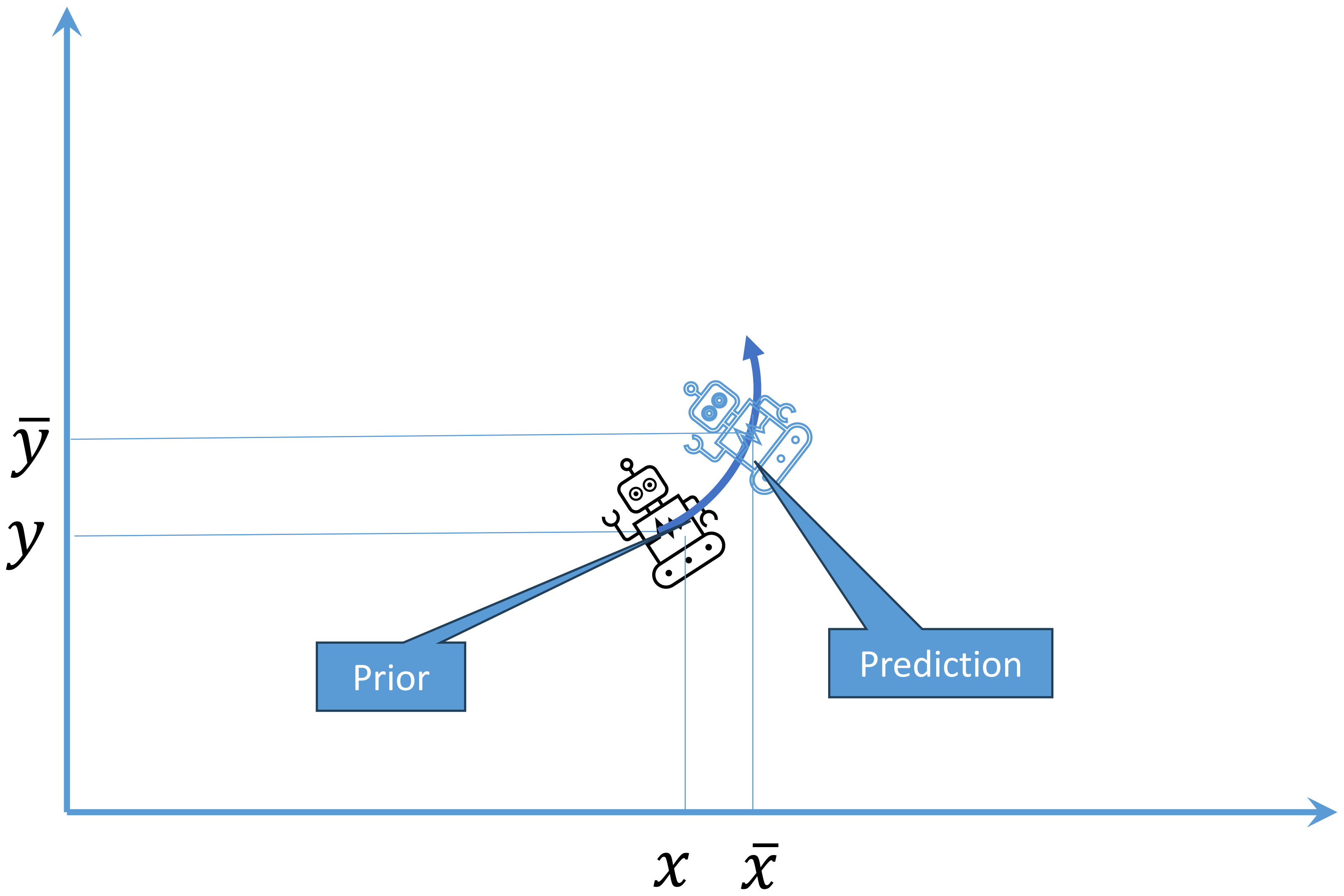
$$\boldsymbol{x}[n] = \overline{\boldsymbol{x}}[n] + \mathbf{K}(\boldsymbol{z}[n] - \boldsymbol{h}(\overline{\boldsymbol{x}}[\mathrm{n}]))$$

What we actually measured

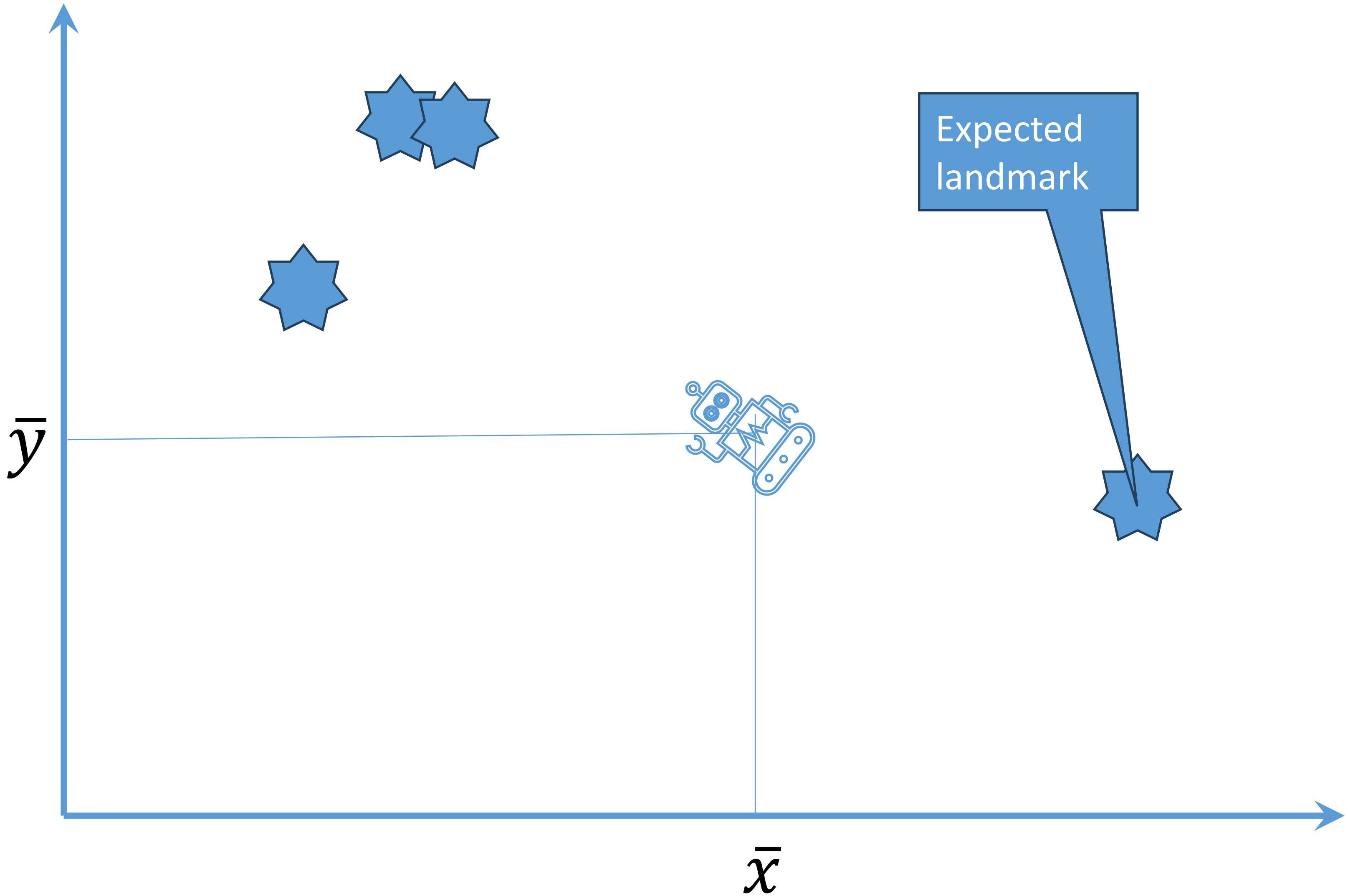
What we expect to measure

- How do we know which measurements to use?
  - Unique landmarks: we just know
  - Non-unique: must figure out from prior

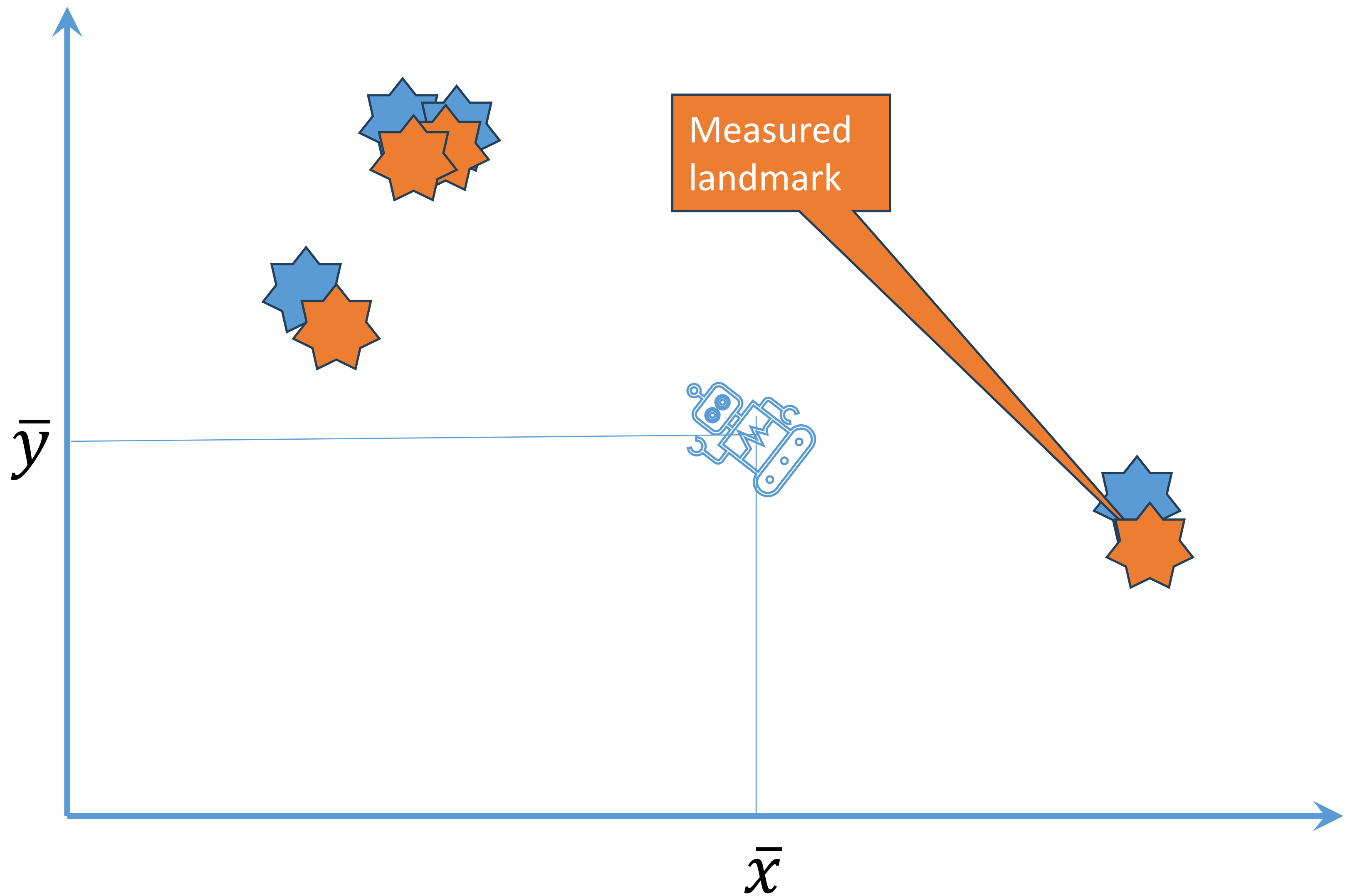
Closest Distance



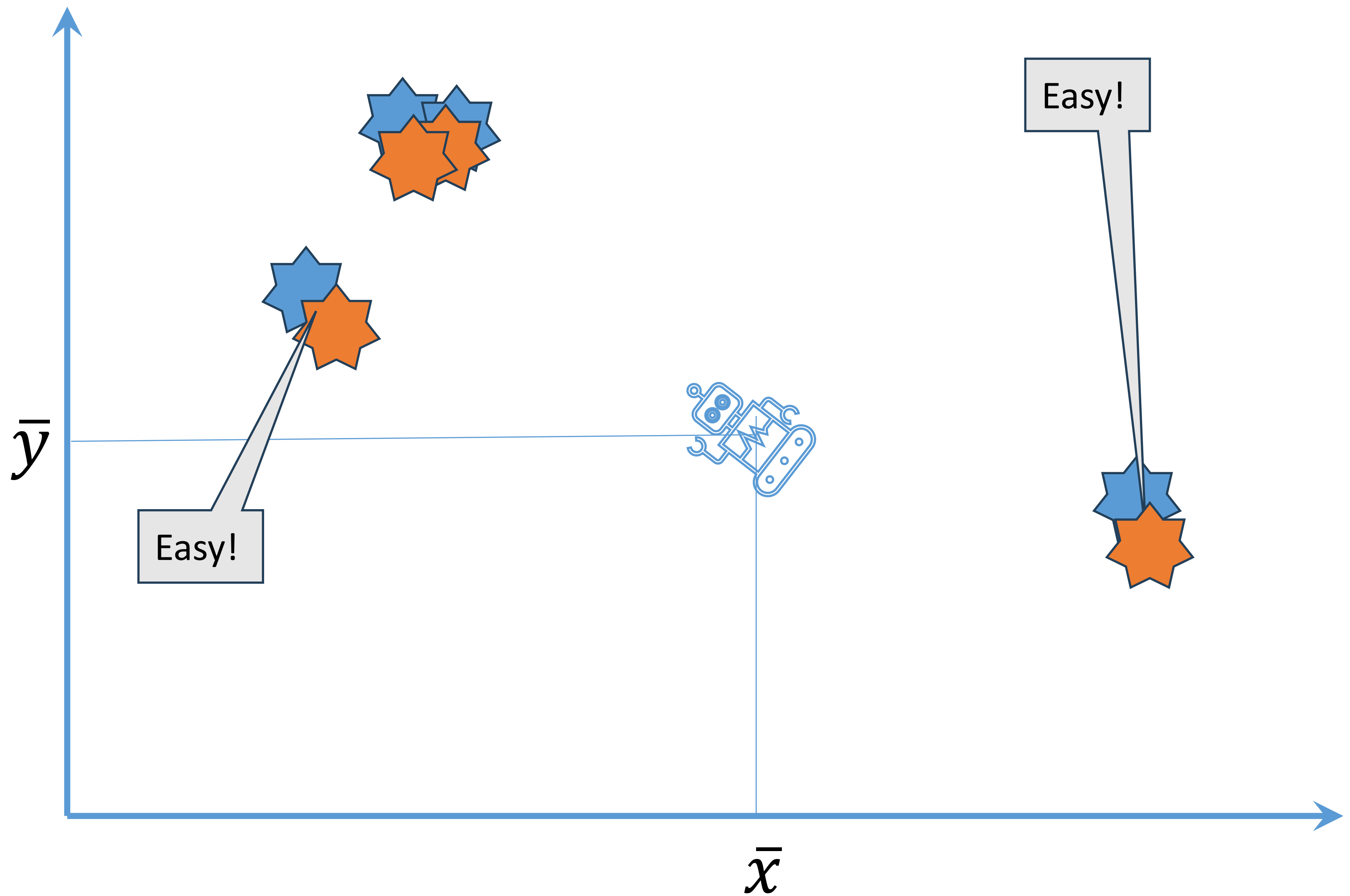
Closest Distance



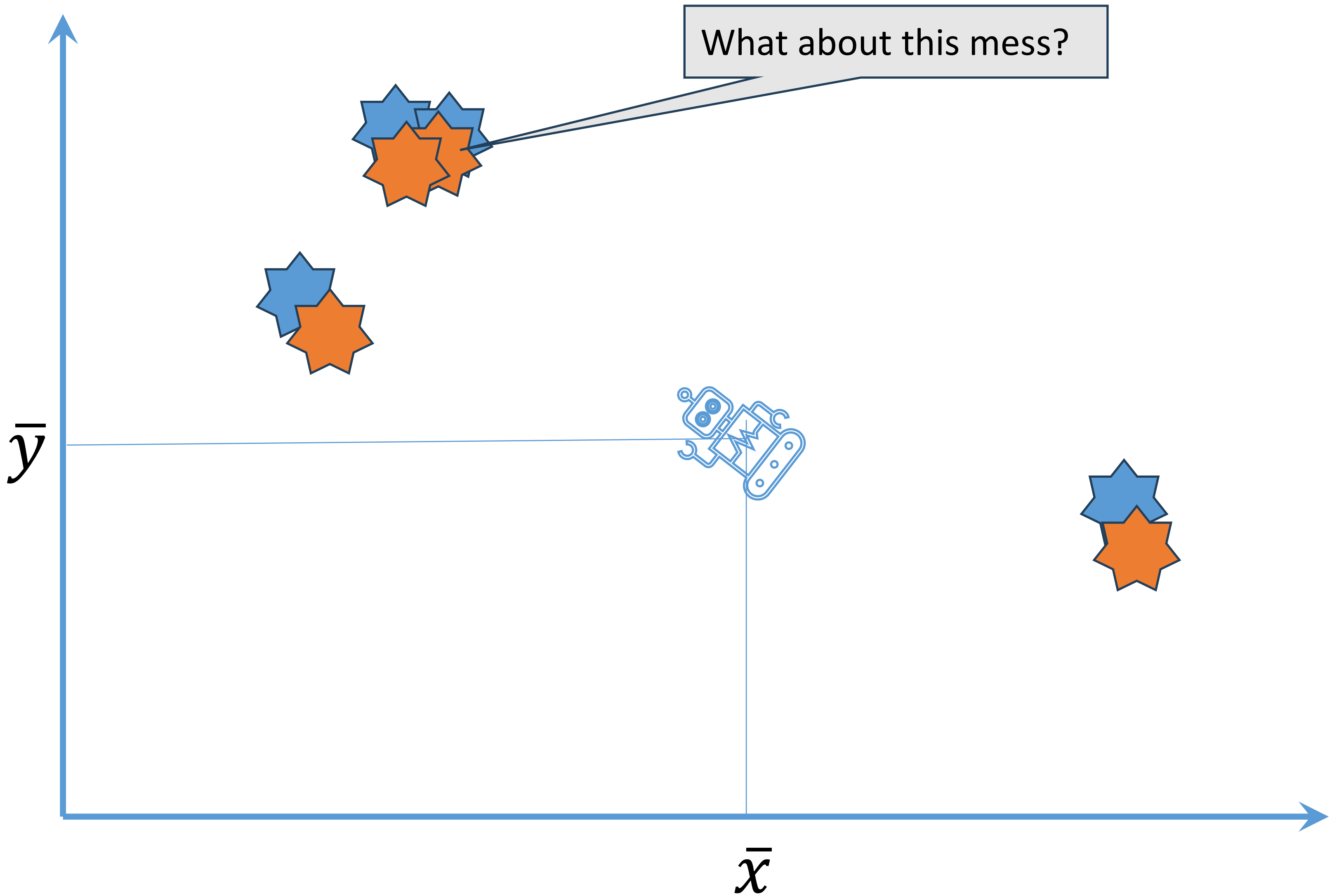
Closest Distance



Closest Distance



Closest Distance



Maximum Likelihood

- Use covariance when picking correspondence!
- Recall the Kalman gain:

$$\boldsymbol{K} = \overline{\boldsymbol{\Sigma}}_x[n]\boldsymbol{H}_{\boldsymbol{x}}^{\mathrm{T}}\underbrace{\left(\boldsymbol{H}_x\overline{\boldsymbol{\Sigma}}_x[n]\boldsymbol{H}_x^{\mathrm{T}} + \boldsymbol{\Sigma}_z[n]\right)}_{\text{Covariance of measurement difference!}}{}^{-1}$$

- Define for each landmark k:

$$\boldsymbol{S}^{(\boldsymbol{k})} = \boldsymbol{H}_x^{(k)}\overline{\boldsymbol{\Sigma}}_x[n]\boldsymbol{H}_x^{(\mathrm{k})^{\boldsymbol{T}}} + \boldsymbol{\Sigma}_z[n]$$

Maximum Likelihood

- For each landmark k, measurement difference:
  - Has normal distribution
  - Zero-mean
  - Covariance $\boldsymbol{S}^{(k)}$
- Hence:

$$c_i = \operatorname*{argmax}_k \frac{\exp\left(-\frac{1}{2}(\boldsymbol{z}_i - \boldsymbol{h}(\overline{\boldsymbol{x}}))\boldsymbol{S}^{(k)^{-1}}(\boldsymbol{z}_i - \boldsymbol{h}(\overline{\boldsymbol{x}}))^T\right)}{\sqrt{(2\pi)^d |\boldsymbol{S}^{(k)}|}}$$

Full Algorithm

- Table 7.3, Page 217 in Thrun
- Prediction is the same as for known correspondences
- For each measurement:
  - Calculate $\boldsymbol{S}^{(k)}$ for **all landmarks**
  - Pick the most likely
  - Proceed to calculate the Kalman gain
  - Update the (posterior) state