Localization: MAP in SE(3)

Karel Zimmermann

Problem definition

Complete states: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

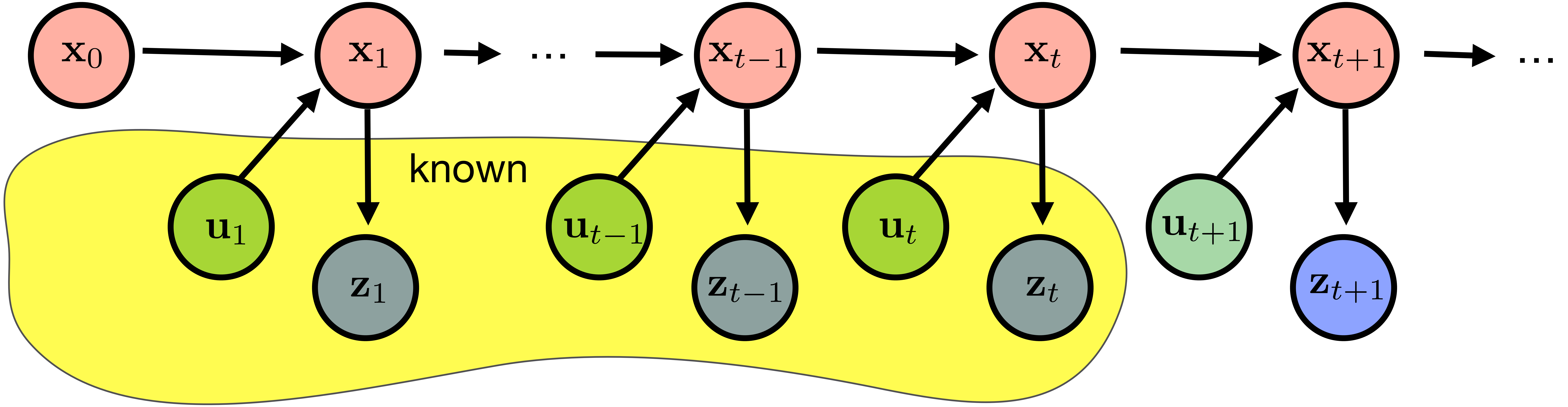
Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau \sim \pi}\{\sum_{r_t \sim \tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi J_\pi$

Algorithm: $\mathbf{z}_0, \mathbf{u}_1, \mathbf{z}_1, \ldots$ => estimate $p(\mathbf{x}_t | \mathbf{z}_{1:t}, \mathbf{u}_{1:t})$ $\overset{\pi(\mathbf{x}_t)}{=>}$ decide following action $\mathbf{u}_{t+1}$

perception (local, SLAM, object detection) control (planning, RL, opt.control



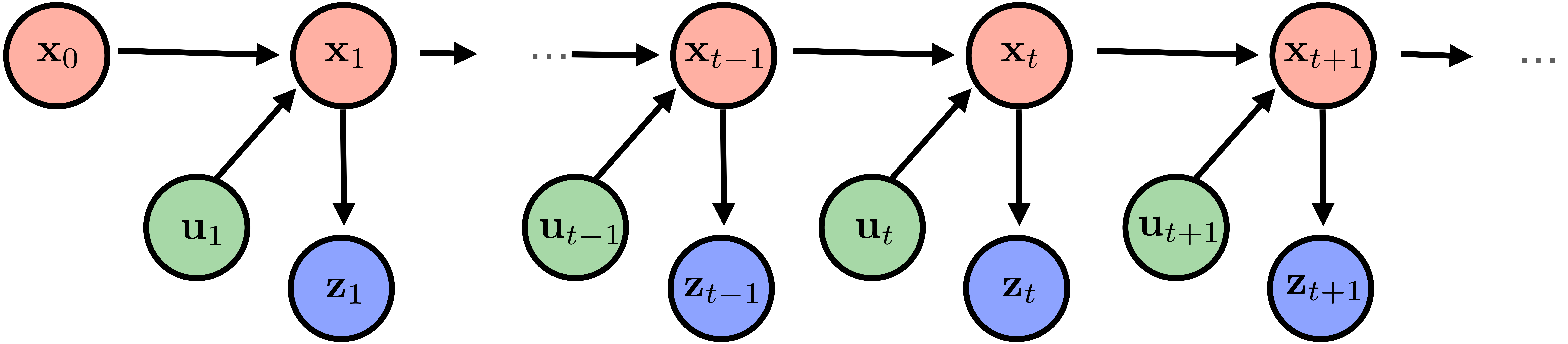
Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$ …. 6DOF robot's poses (no map for now)

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$ …. generated by external source

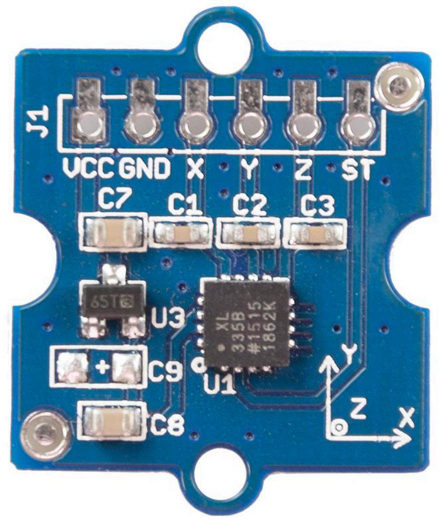
Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$ …. comes from variety of sensors

Goal:
- estimate most probable $\mathbf{x}_0 \ldots \mathbf{x}_t$
- or just $\mathbf{x}_t$
- or full distribution $p(\mathbf{x}_t)$ or even $p(\mathbf{x}_0 \ldots \mathbf{x}_t)$

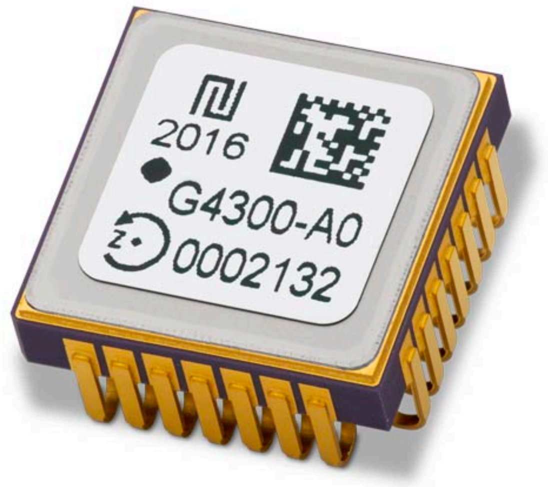


Sensors for localisation (odometry)

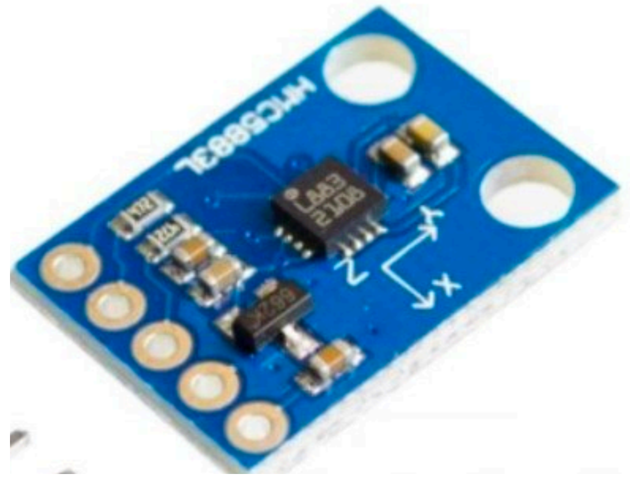
Motor encoders (wheel/joint position/velocity)



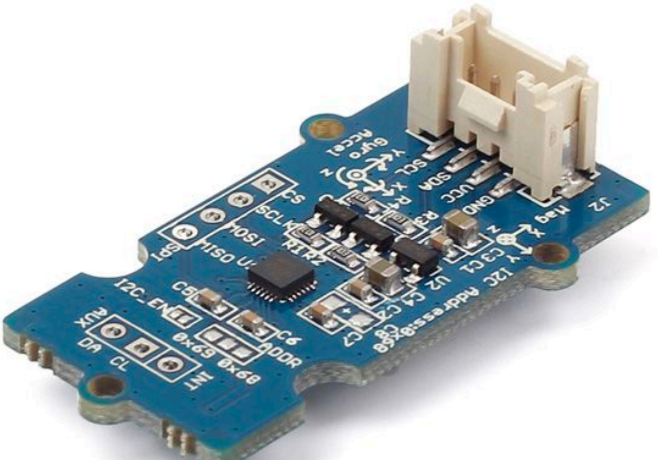
Accelerometer (linear acceleration)



Gyroscope (angular velocity)



Magnetometer (angle to magnetic north)



IMU: Accelerometer+Gyroscope+Magnetometer (9DOF measurements)

Sensors for localisation (exteroceptive)

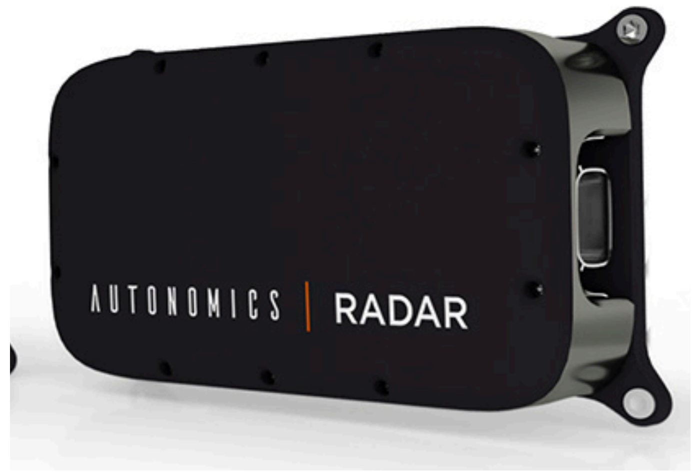
Camera (RGB images - spectral responses projected on image plane)

Stereo camera

RGBD camera (kinect, real sense, …)

Lidar

Sonar



Radar

Satelite navigation (GPS/GNSS)

SONARDYNE beacons

UWB

Sensor measurements

- Noise characteristic (GPS vs camera for localisation)
- Operates in its own coordinate frame
- Spatiotemporal (and spectral) resolution
  (i.e. number of pixels/channels in image, number of measurements per second)
- Absolute/relative measurements wrt a reference coordinate frame
  (e.g. GPS/IMU) and integrating the relative measurements does not work!

Consequence: Need a reasonable probabilistic approach that fuses all measurements in order to estimate the most probable pose(s)

Localisation problem definition

Previous lecture only 1D/2D translations (no rotations)

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$ .... ~~6DOF~~ robot's poses (no map for now)

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$ .... generated by external source

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$ .... comes from variety of sensors

Unknown

MAP: $\mathbf{x}^* = \arg\max_{\mathbf{x}} p(\mathbf{x} \,|\, \mathbf{z}, \mathbf{u}) = \arg\max_{\mathbf{x}_0 \ldots \mathbf{x}_t} p(\mathbf{x}_0 \ldots \mathbf{x}_t \,|\, \mathbf{z}_1 \ldots \mathbf{z}_t, \mathbf{u}_1 \ldots \mathbf{u}_t)$

1. Construct p(x|z)
2. Optimize poses

Localisation problem definition

Today only 2D translations + 1D rotation

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$ .... ~~6DOF~~ robot's poses (no map for now)

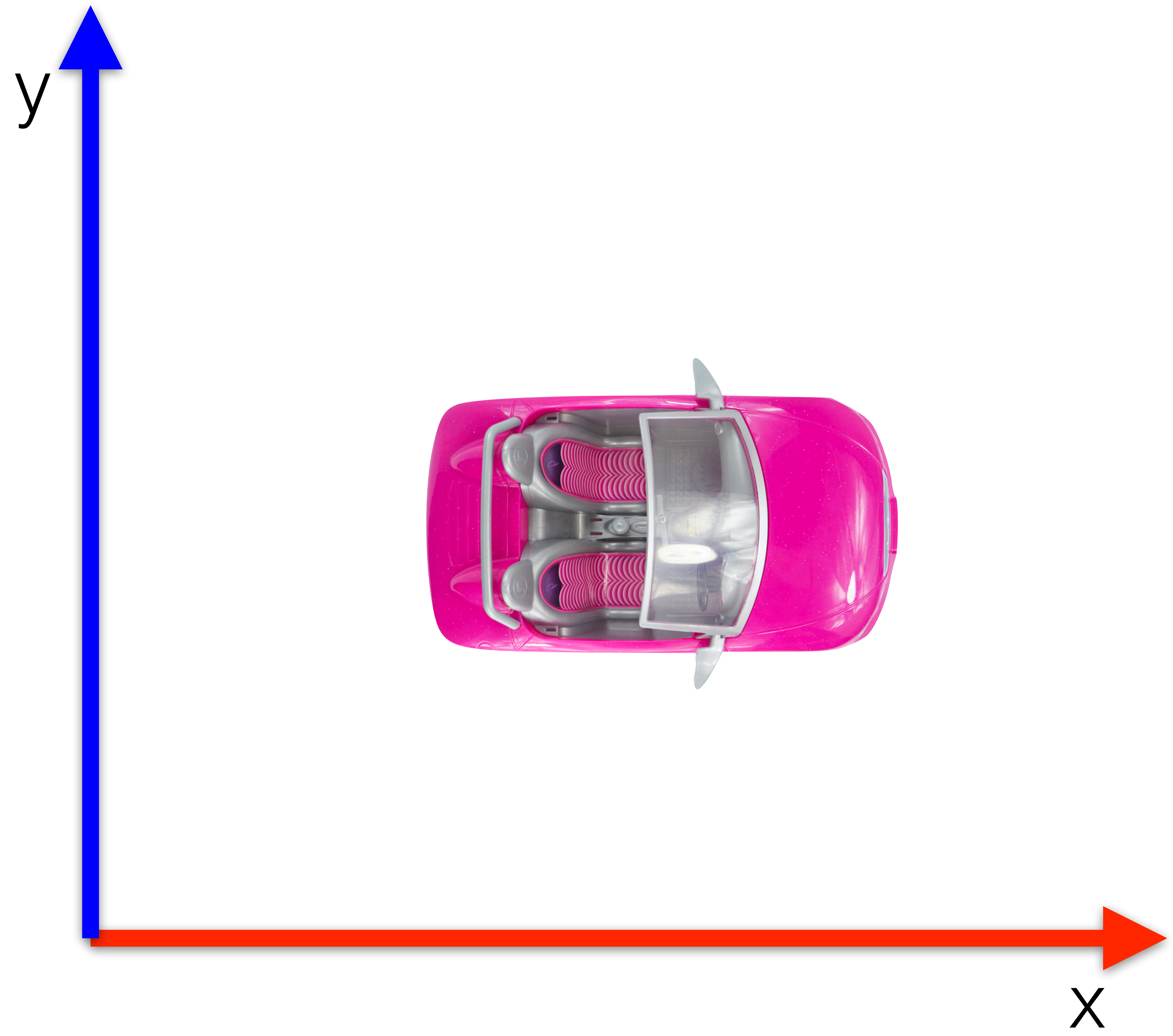
Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$ .... generated by external source

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$ .... comes from variety of sensors

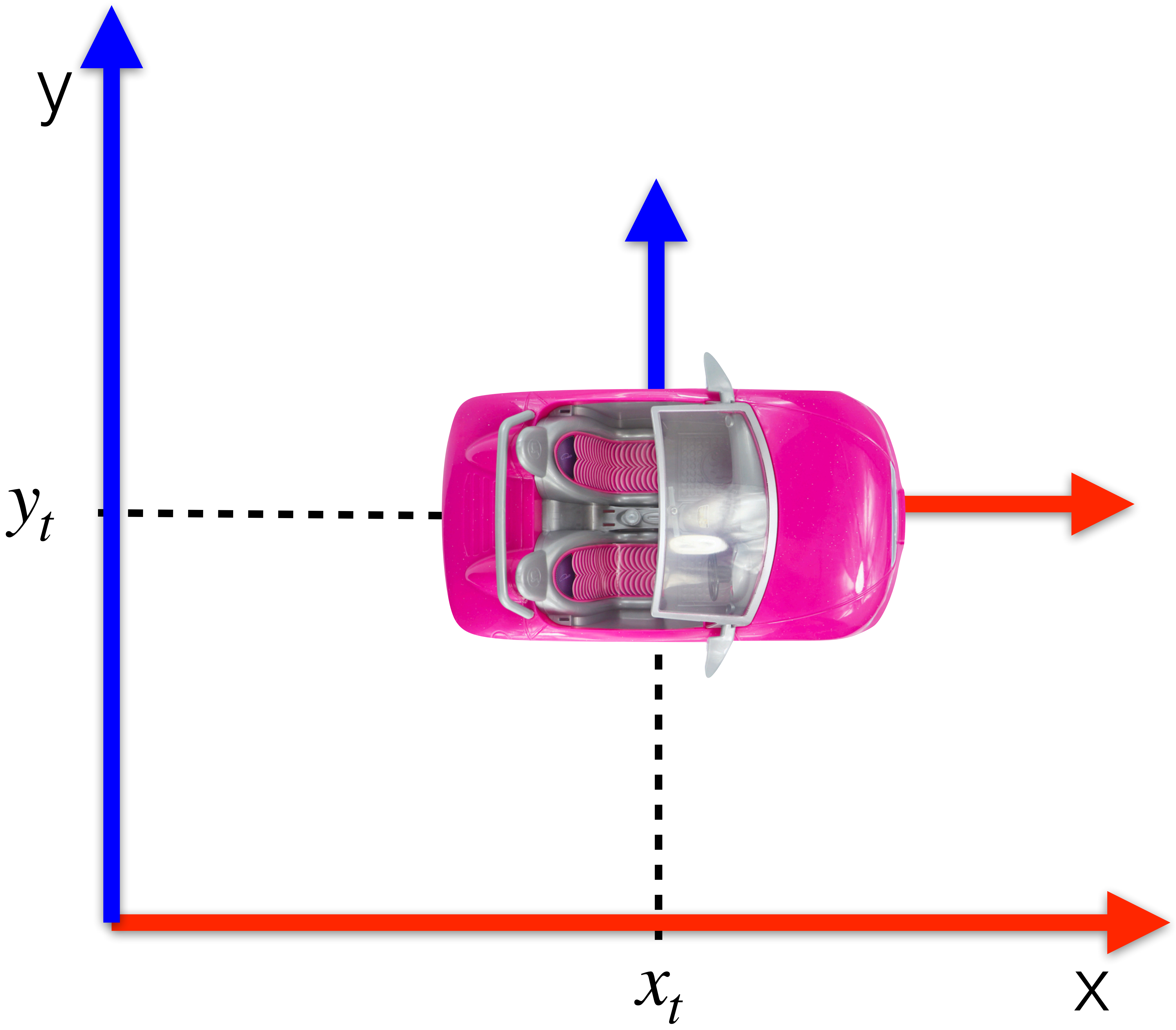
Unknown

MAP: $\mathbf{x}^* = \arg\max_{\mathbf{x}} p(\mathbf{x} \,|\, \mathbf{z}, \mathbf{u}) = \arg\max_{\mathbf{x}_0 \ldots \mathbf{x}_t} p(\mathbf{x}_0 \ldots \mathbf{x}_t \,|\, \mathbf{z}_1 \ldots \mathbf{z}_t, \mathbf{u}_1 \ldots \mathbf{u}_t)$

1. Construct p(x|z)
2. Optimize poses

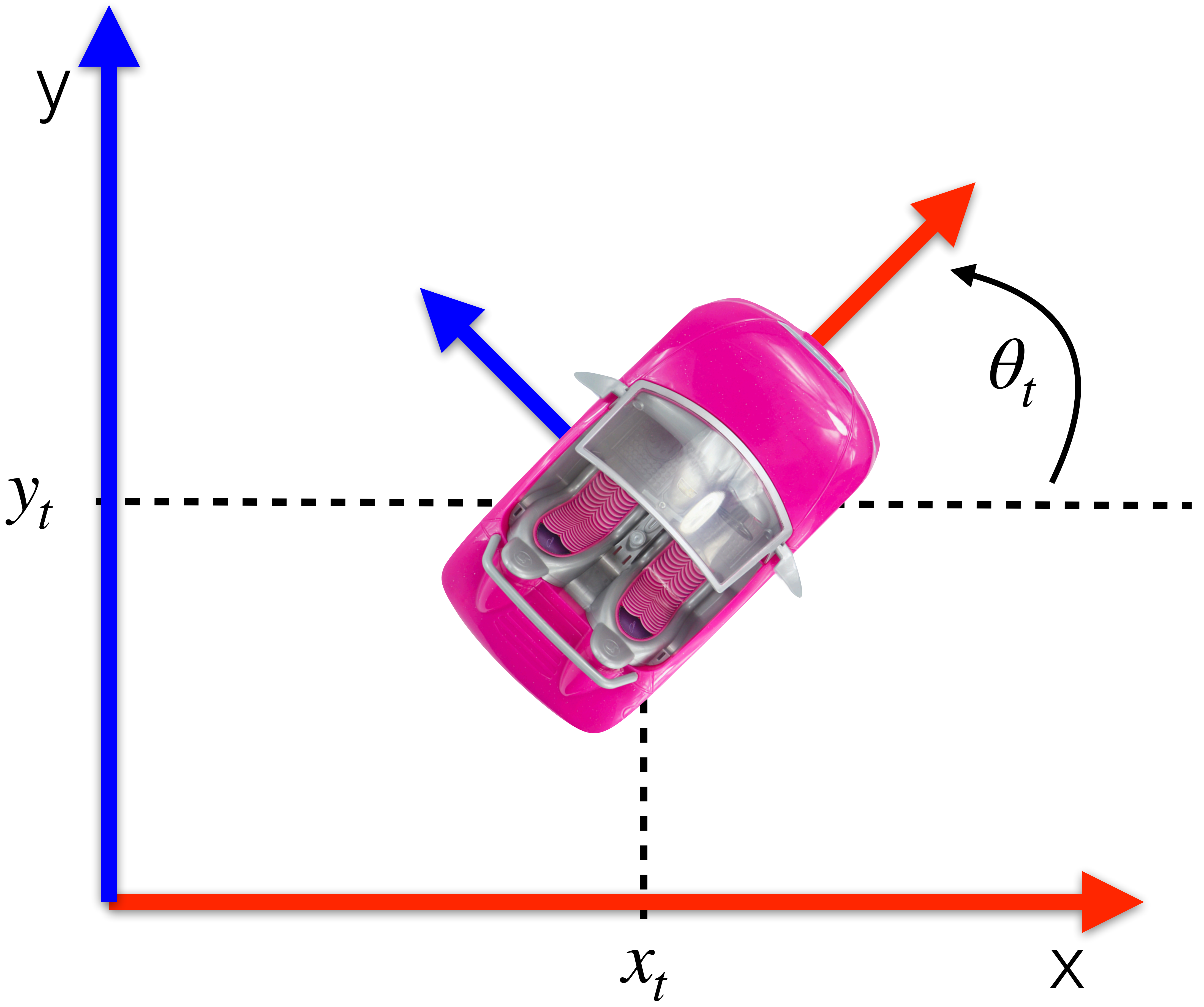


World coordinate frame (wcf)



Robot coordinate frame (rcf)
i.e. pose of robot in wcf

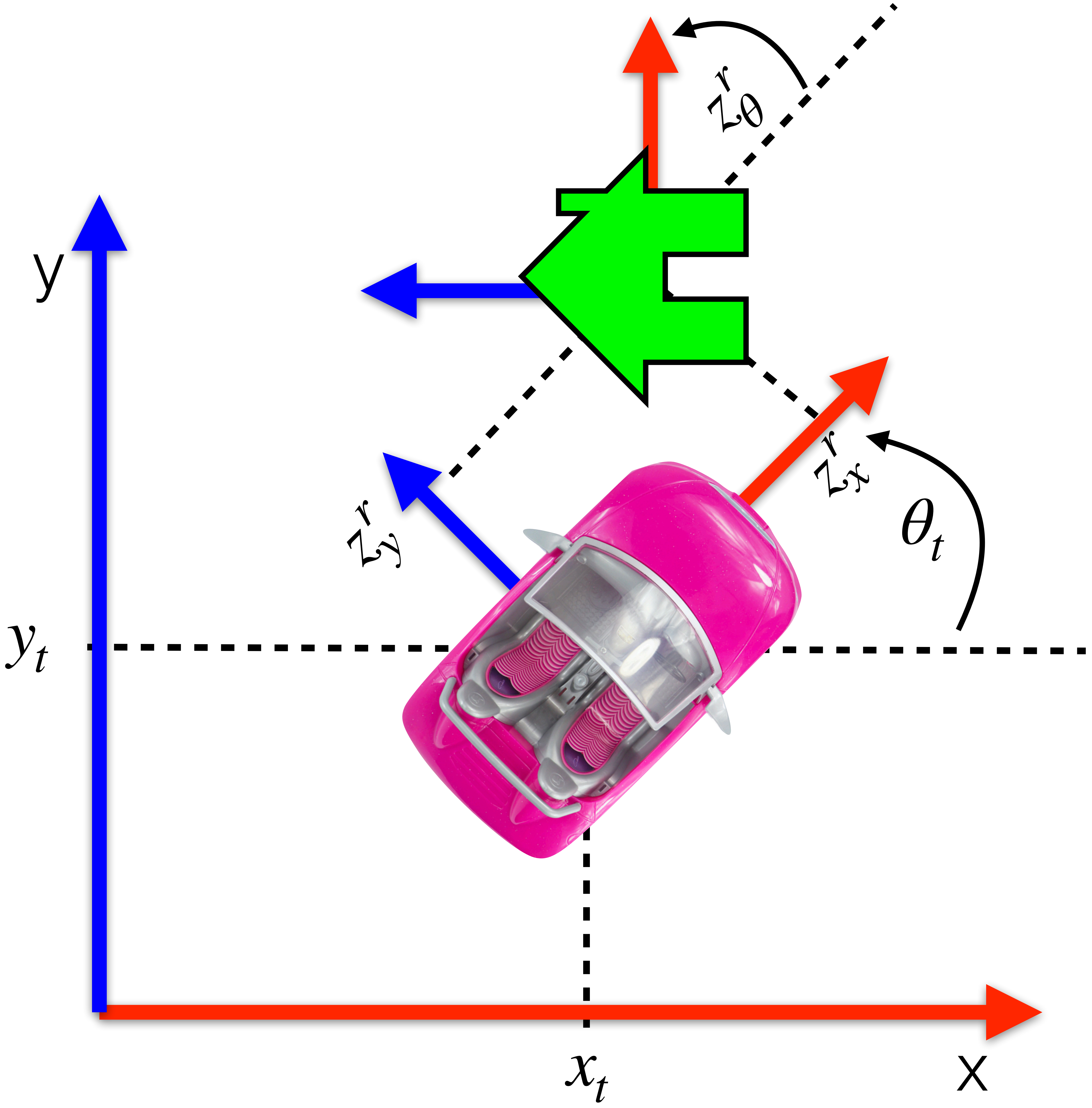
World coordinate frame (wcf)



Robot coordinate frame (rcf)
i.e. pose of robot in wcf

$$\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$$

World coordinate frame (wcf)



Robot sees (measures) house in rcf
i.e. pose of house in rcf

$$\mathbf{z} = \begin{bmatrix} z^r_x \\ z^r_y \\ z^r_\theta \end{bmatrix}$$

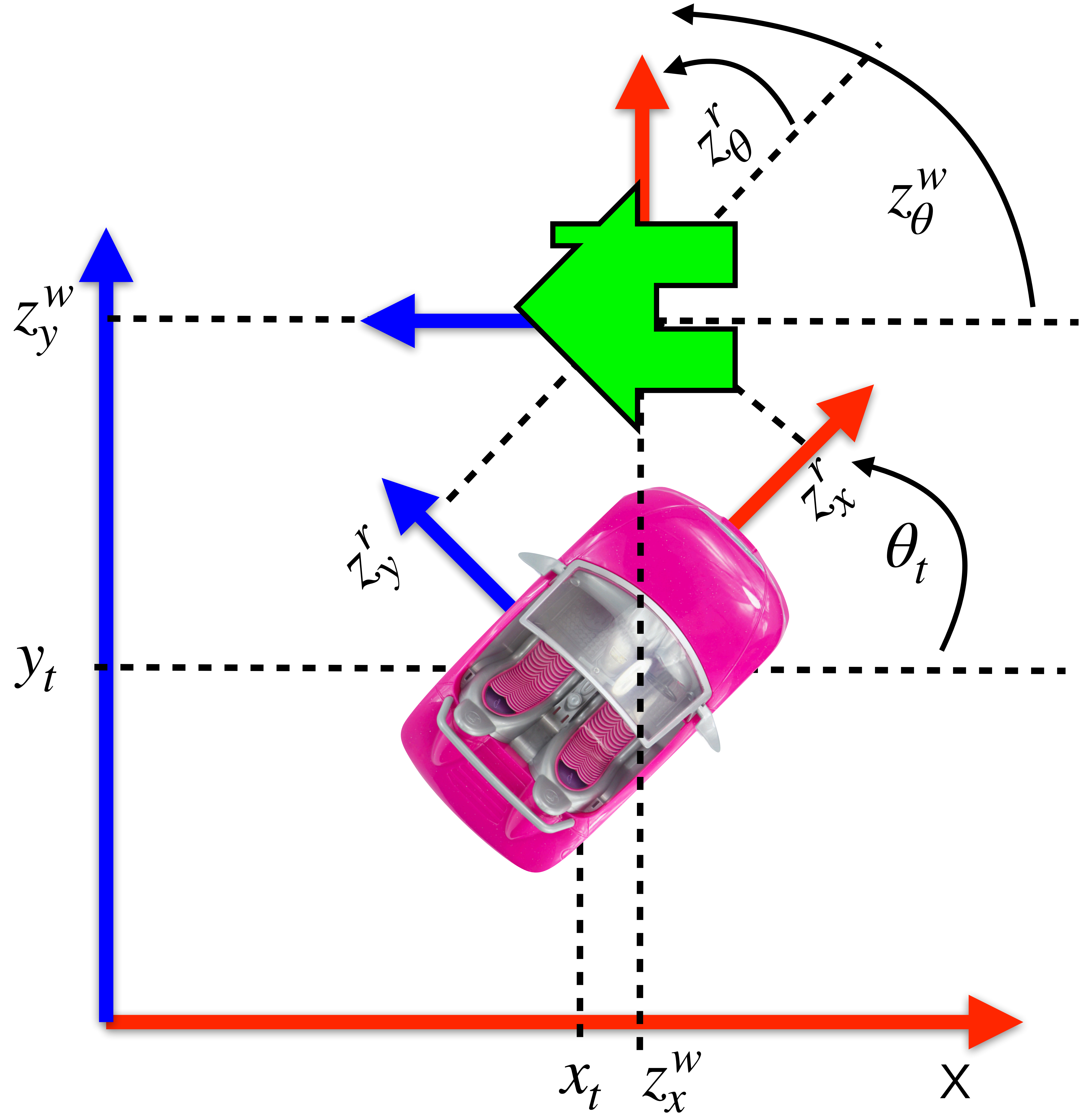
Robot coordinate frame (rcf)
i.e. pose of robot in wcf

$$\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$$

World coordinate frame (wcf)

Pose of house in wcf:

$$\mathbf{z}^w = \begin{bmatrix} z_x^w \\ z_y^w \\ z_\theta^w \end{bmatrix} = \begin{bmatrix} \cos\theta_t & -\sin\theta_t & 0 \\ \sin\theta_t & \cos\theta_t & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} z_x^r \\ z_y^r \\ z_\theta^r \end{bmatrix} + \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$$



Robot sees (measures) house in rcf
i.e. pose of house in rcf

$$\mathbf{z} = \begin{bmatrix} z_x^r \\ z_y^r \\ z_\theta^r \end{bmatrix}$$

Robot coordinate frame (rcf)
i.e. pose of robot in wcf

$$\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$$

World coordinate frame (wcf)

Pose of the house transformed from rcf to wcf:

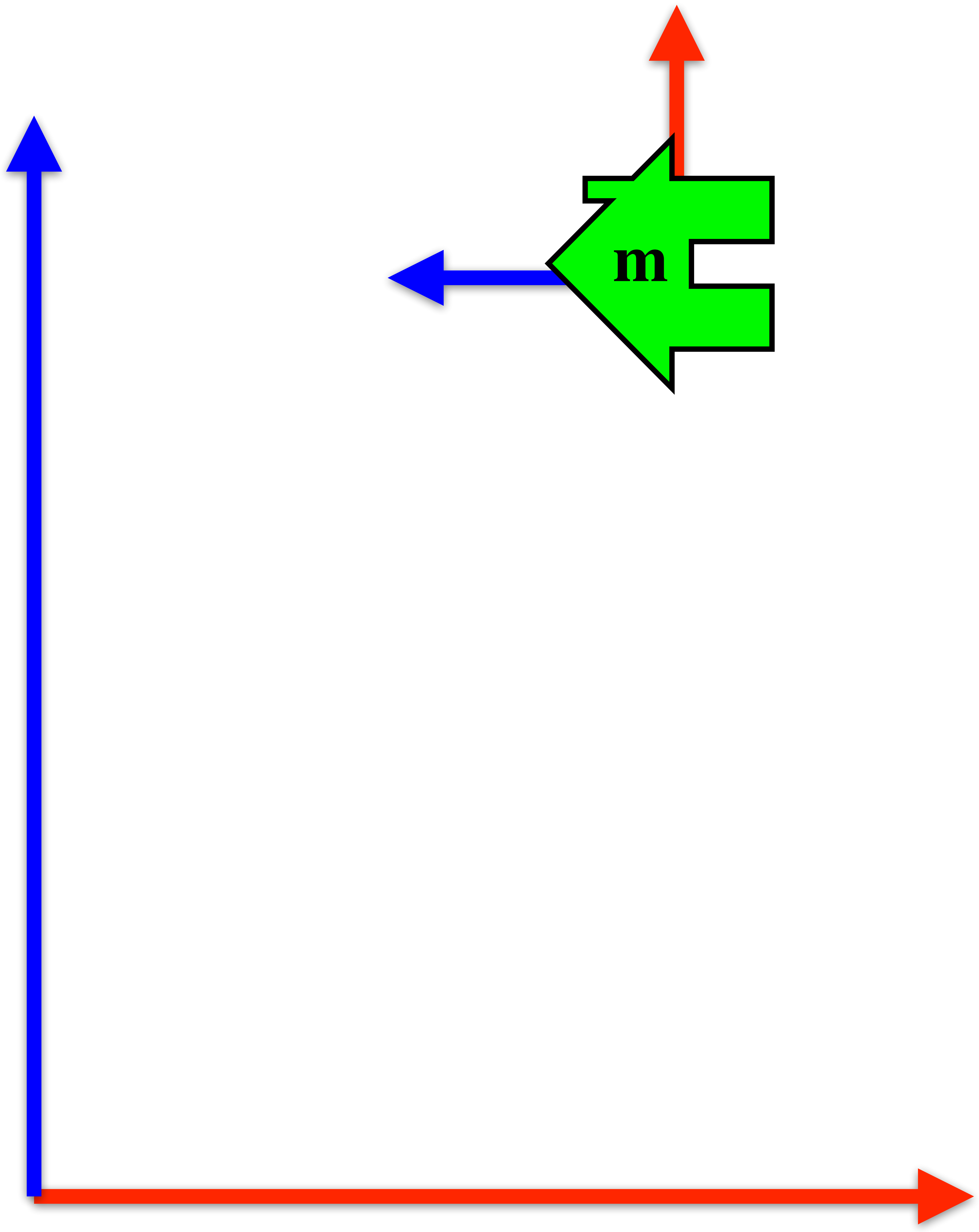
$$\mathbf{z}^w = \begin{bmatrix} z_x^w \\ z_y^w \\ z_\theta^w \end{bmatrix} = \begin{bmatrix} \cos\theta_t & -\sin\theta_t & 0 \\ \sin\theta_t & \cos\theta_t & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} z_x^r \\ z_y^r \\ z_\theta^r \end{bmatrix} + \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} = \begin{bmatrix} R(\theta_t) & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{bmatrix} \mathbf{z}^r + \mathbf{x}_t = T(\mathbf{z}^r, \mathbf{x}_t)$$

Pose of the house transformed from wcf to rcf:

$$\mathbf{z}^r = \begin{bmatrix} R(\theta_t)^\top & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{bmatrix} (\mathbf{z}^w - \mathbf{x}_t) = T^{-1}(\mathbf{z}^w, \mathbf{x}_t)$$

Localization of robot in wcf from known position of the house

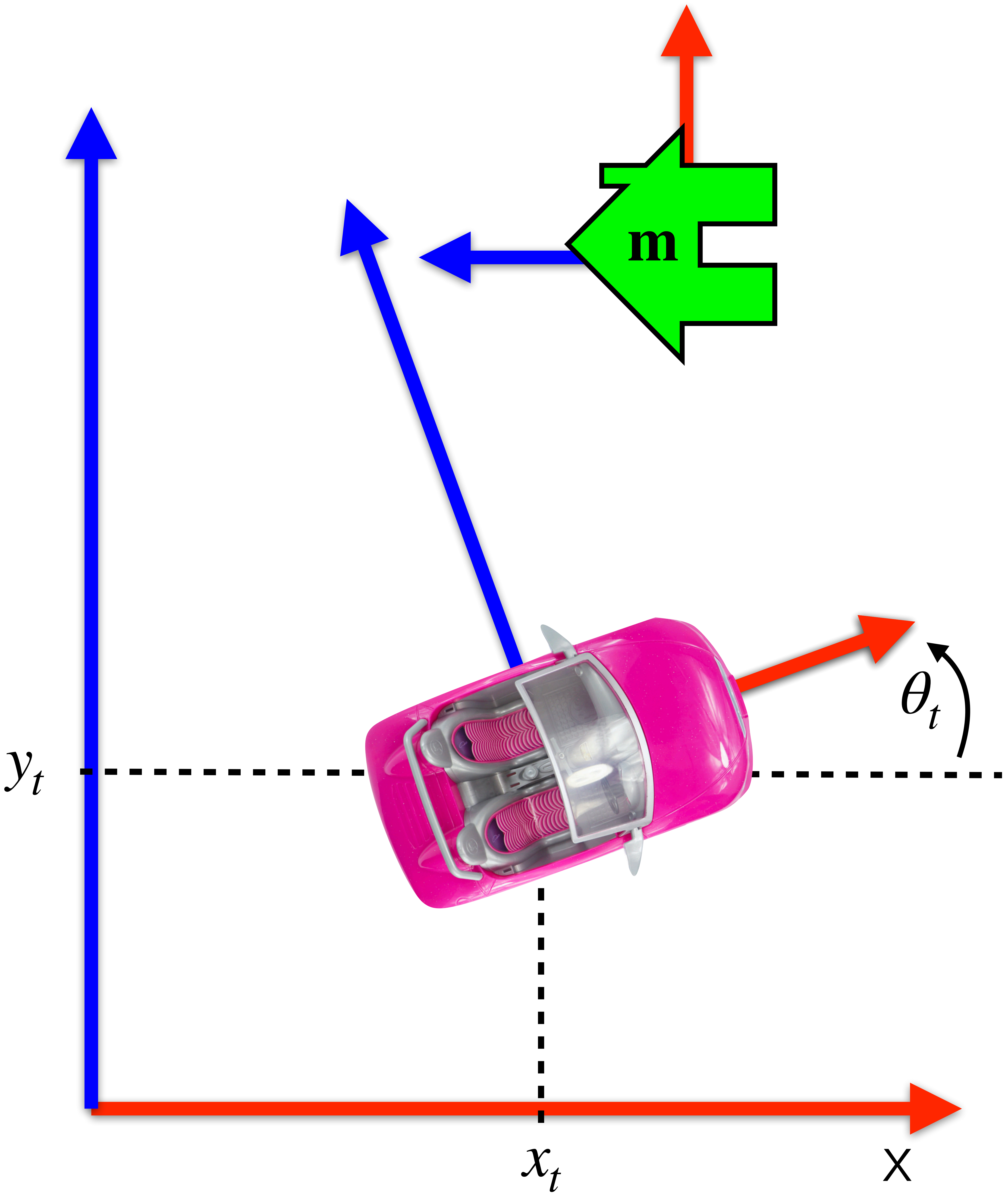
Assume that house pose in wcf is known **m**



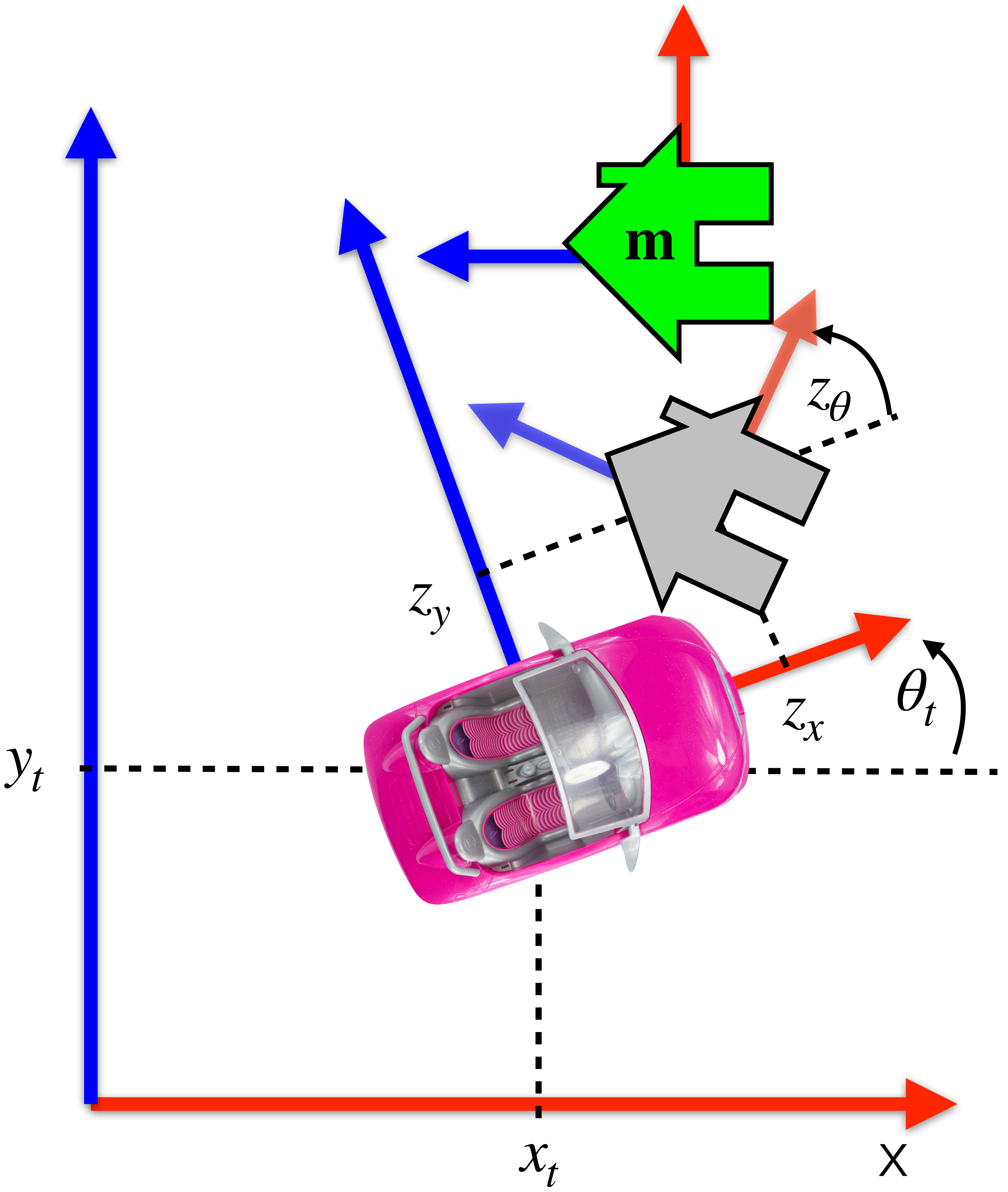
Localization of robot in wcf from known position of the house

Assume that marker pose in wcf is known $\mathbf{m}$

We assume that robot pose in wcf $\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$



Localization of robot in wcf from known position of the house

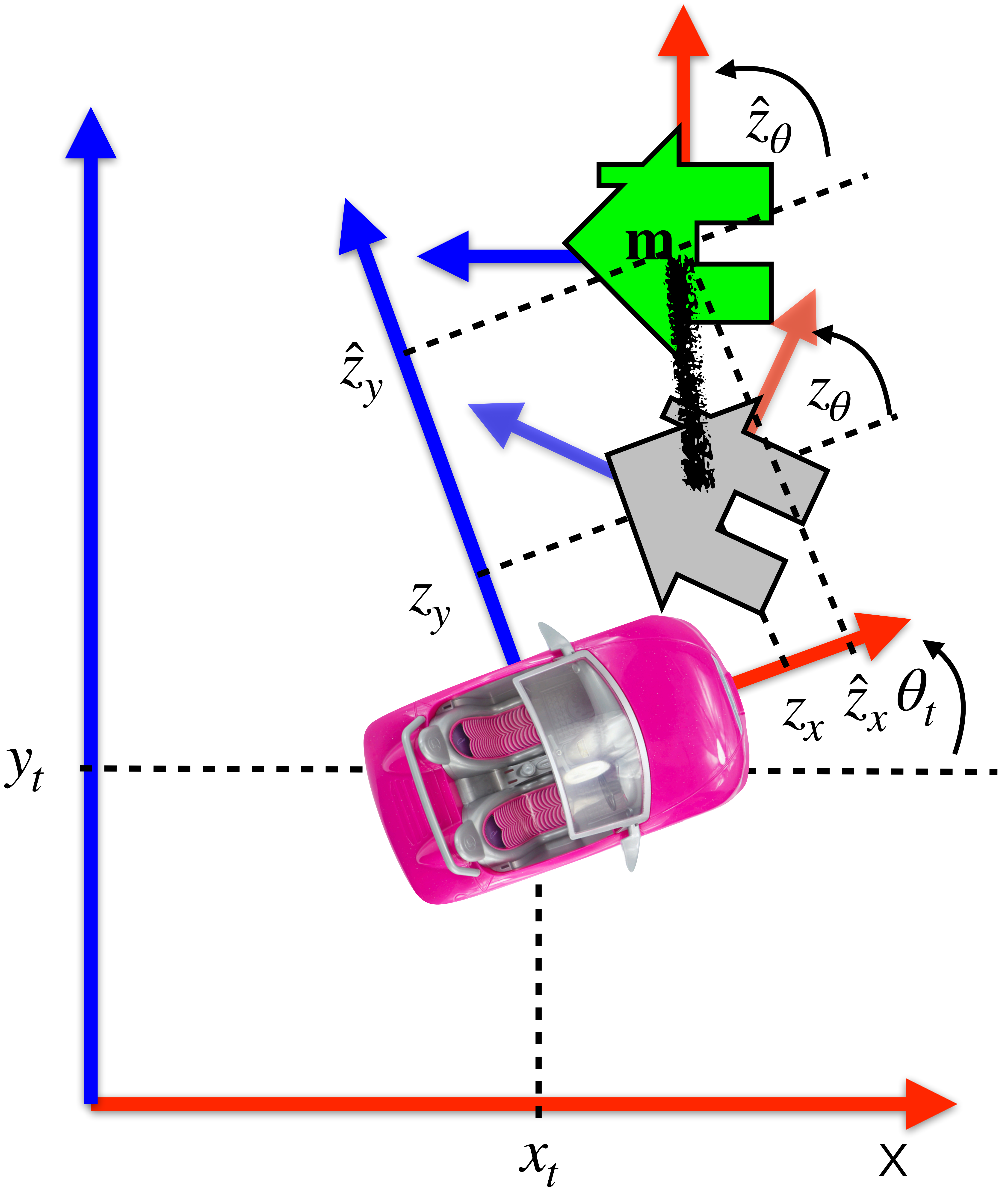


Assume that marker pose in wcf is known $\mathbf{m}$

We assume that robot pose in wcf $\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$

Robot measures the house in rcf $\mathbf{z} = \begin{bmatrix} z_x^r \\ z_y^r \\ z_\theta^r \end{bmatrix}$

Localization of robot in wcf from known position of the house



Assume that marker pose in wcf is known $\mathbf{m}$

We assume that robot pose in wcf $\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$

Robot measures the house in rcf $\mathbf{z} = \begin{bmatrix} z_x^r \\ z_y^r \\ z_\theta^r \end{bmatrix}$

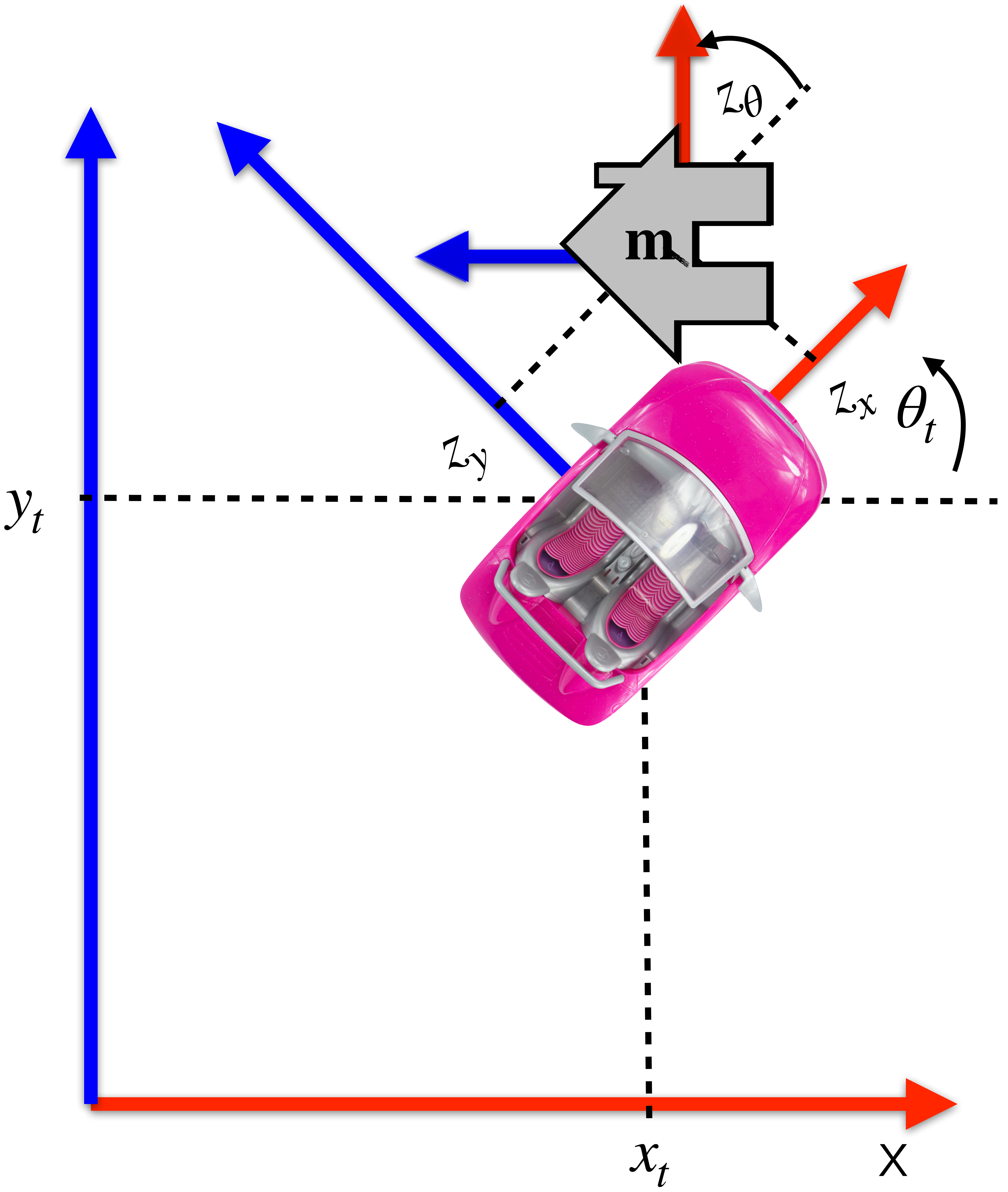
Marker pose in rcf $\hat{\mathbf{z}} = T^{-1}(\mathbf{m}, \mathbf{x}_t)$

Since pose is incorrect $\mathbf{z} \neq \hat{\mathbf{z}}$

Find the correct pose

$$\mathbf{x}_t^\star = \arg\min_{\mathbf{x}_t} \|T^{-1}(\mathbf{m}, \mathbf{x}_t) - \mathbf{z}\|^2$$

Localization of robot in wcf from known position of the house



Assume that marker pose in wcf is known $\mathbf{m}$

We assume that robot pose in wcf $\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$

Robot measures the house in rcf $\mathbf{z} = \begin{bmatrix} z_x^r \\ z_y^r \\ z_\theta^r \end{bmatrix}$

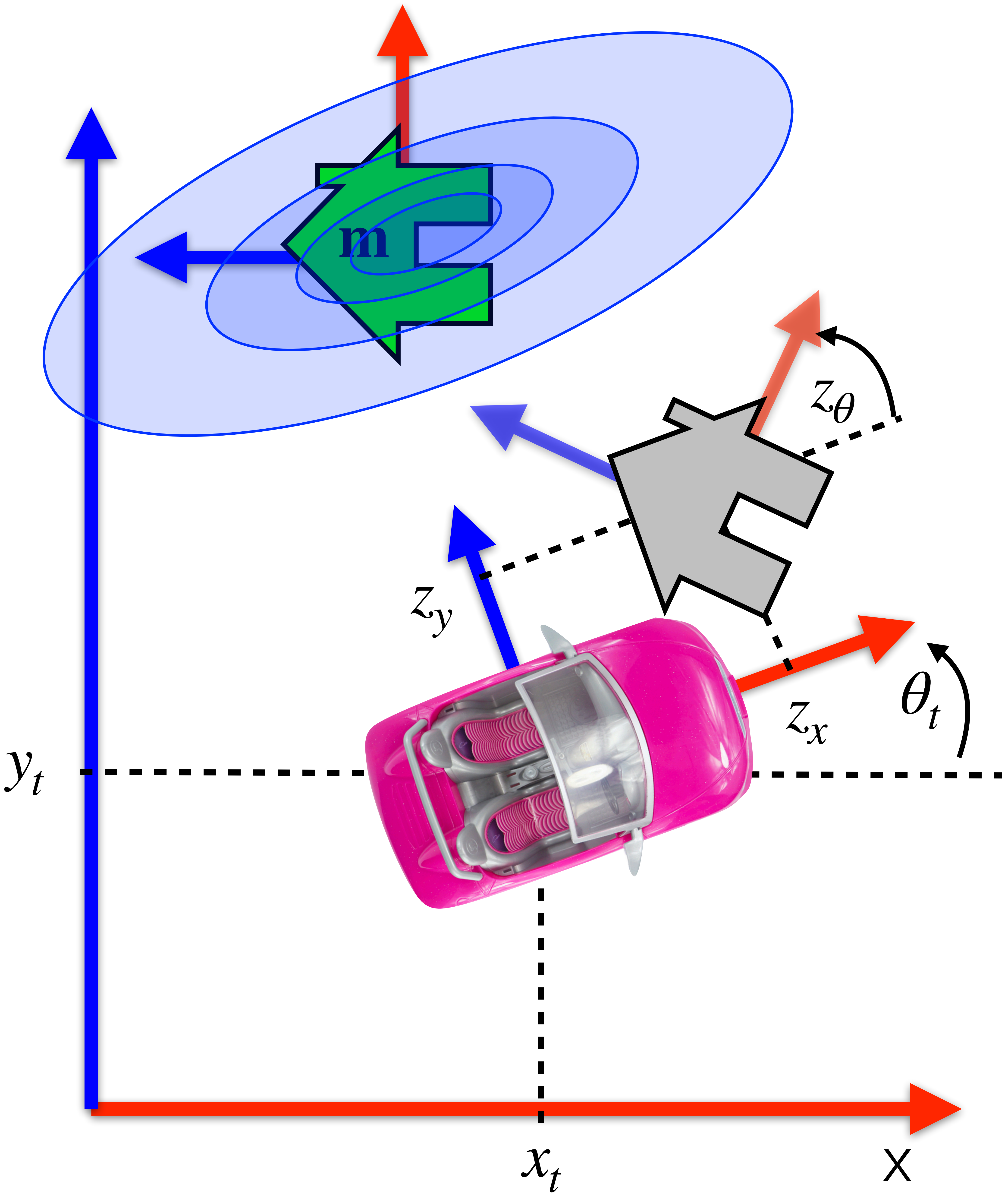
Marker pose in rcf $\hat{\mathbf{z}} = T^{-1}(\mathbf{m}, \mathbf{x}_t)$

Since pose is incorrect $\mathbf{z} \neq \hat{\mathbf{z}}$

Find the correct pose

$$\mathbf{x}_t^\star = \arg\min_{\mathbf{x}_t} \|T^{-1}(\mathbf{m}, \mathbf{x}_t) - \mathbf{z}\|^2$$

Localization of robot in wcf from known position of the house



Assume that marker pose in wcf is known $\mathbf{m}$

We assume that robot pose in wcf $\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$

Robot measures the house in rcf $\mathbf{z} = \begin{bmatrix} z_x^r \\ z_y^r \\ z_\theta^r \end{bmatrix}$

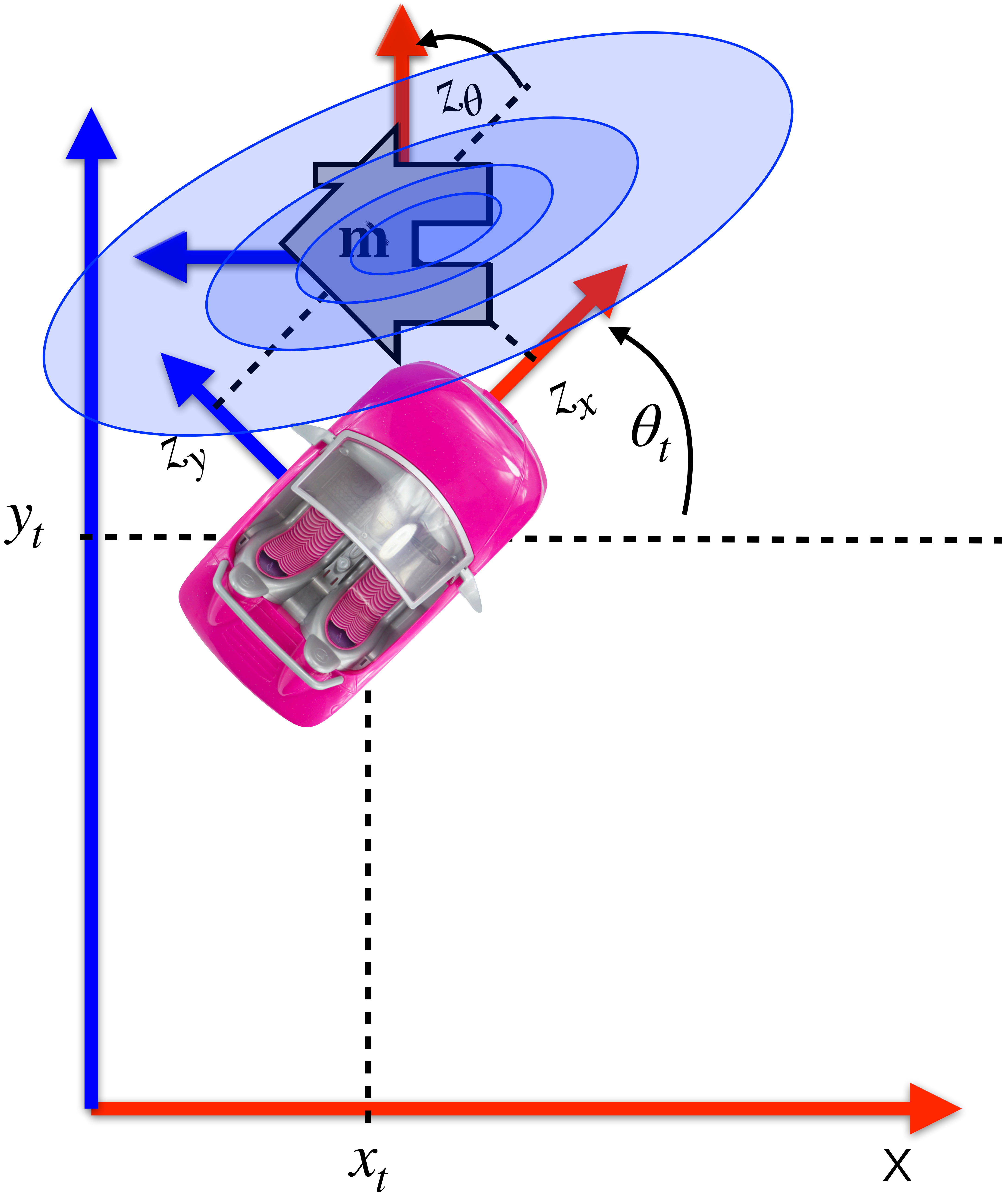
Marker pose in rcf $\hat{\mathbf{z}} = T^{-1}(\mathbf{m}, \mathbf{x}_t)$

Since pose is incorrect $\mathbf{z} \neq \hat{\mathbf{z}}$

Find the correct pose

$$\mathbf{x}_t^\star = \arg\max_{\mathbf{x}_t} \mathcal{N}(\mathbf{z};\ T^{-1}(\mathbf{m}, \mathbf{x}_t), \Sigma)$$
$$= \arg\min_{\mathbf{x}_t} \|T^{-1}(\mathbf{m}, \mathbf{x}_t) - \mathbf{z}\|_\Sigma^2$$

Localization of robot in wcf from known position of the house



Assume that marker pose in wcf is known $\mathbf{m}$

We assume that robot pose in wcf $\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$

Robot measures the house in rcf $\mathbf{z} = \begin{bmatrix} z_x^r \\ z_y^r \\ z_\theta^r \end{bmatrix}$

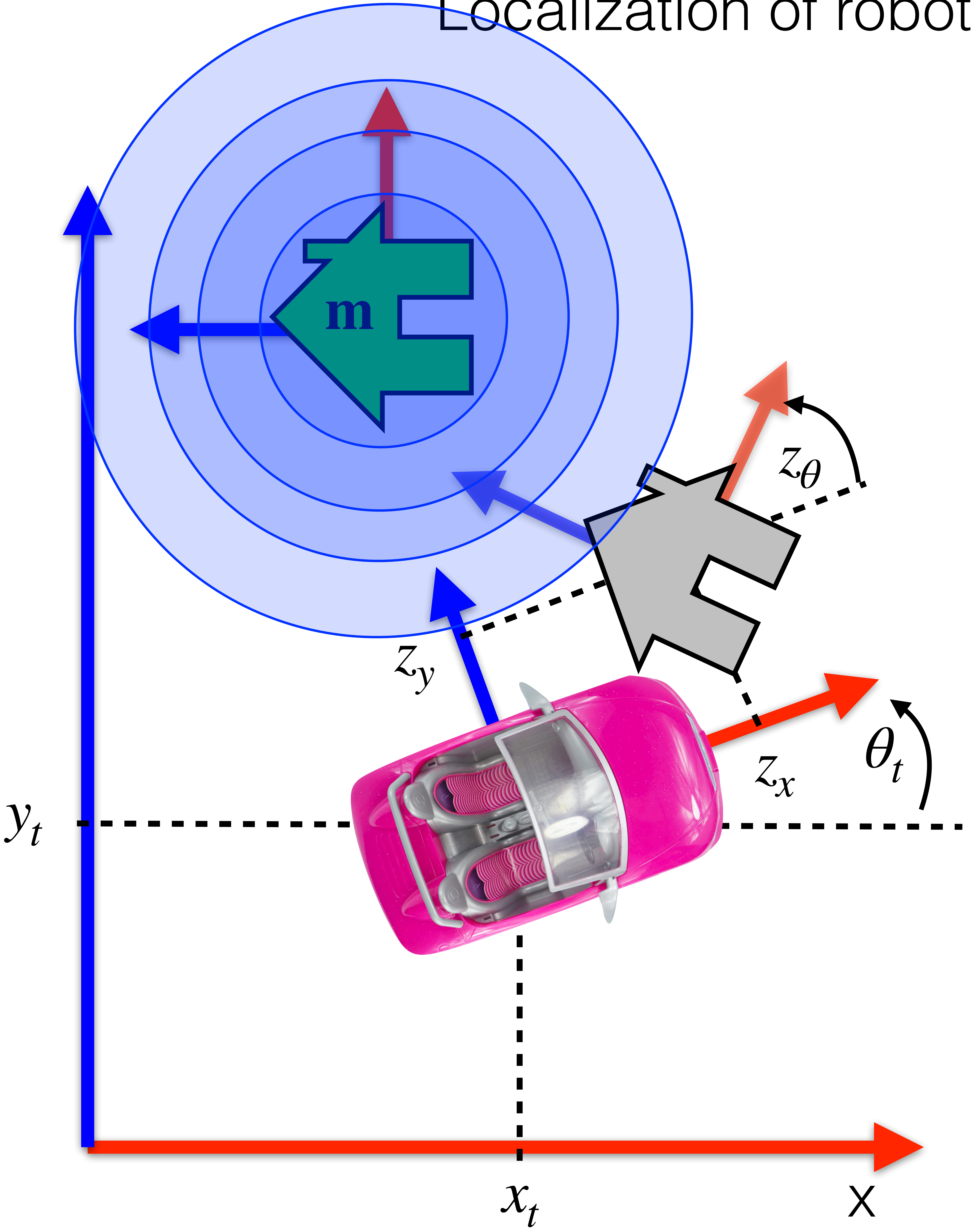
Marker pose in rcf $\hat{\mathbf{z}} = T^{-1}(\mathbf{m}, \mathbf{x}_t)$

Since pose is incorrect $\mathbf{z} \neq \hat{\mathbf{z}}$

Find the correct pose

$$\mathbf{x}_t^\star = \arg\max_{\mathbf{x}_t} \mathcal{N}(\mathbf{z};\ T^{-1}(\mathbf{m}, \mathbf{x}_t), \Sigma)$$
$$= \arg\min_{\mathbf{x}_t} \|T^{-1}(\mathbf{m}, \mathbf{x}_t) - \mathbf{z}\|_\Sigma^2$$

Localization of robot in wcf from known position of the house



Marker pose in rcf $\hat{\mathbf{z}} = T^{-1}(\mathbf{m}, \mathbf{x}_t)$

Since pose is incorrect $\mathbf{z} \neq \hat{\mathbf{z}}$

Find the correct pose

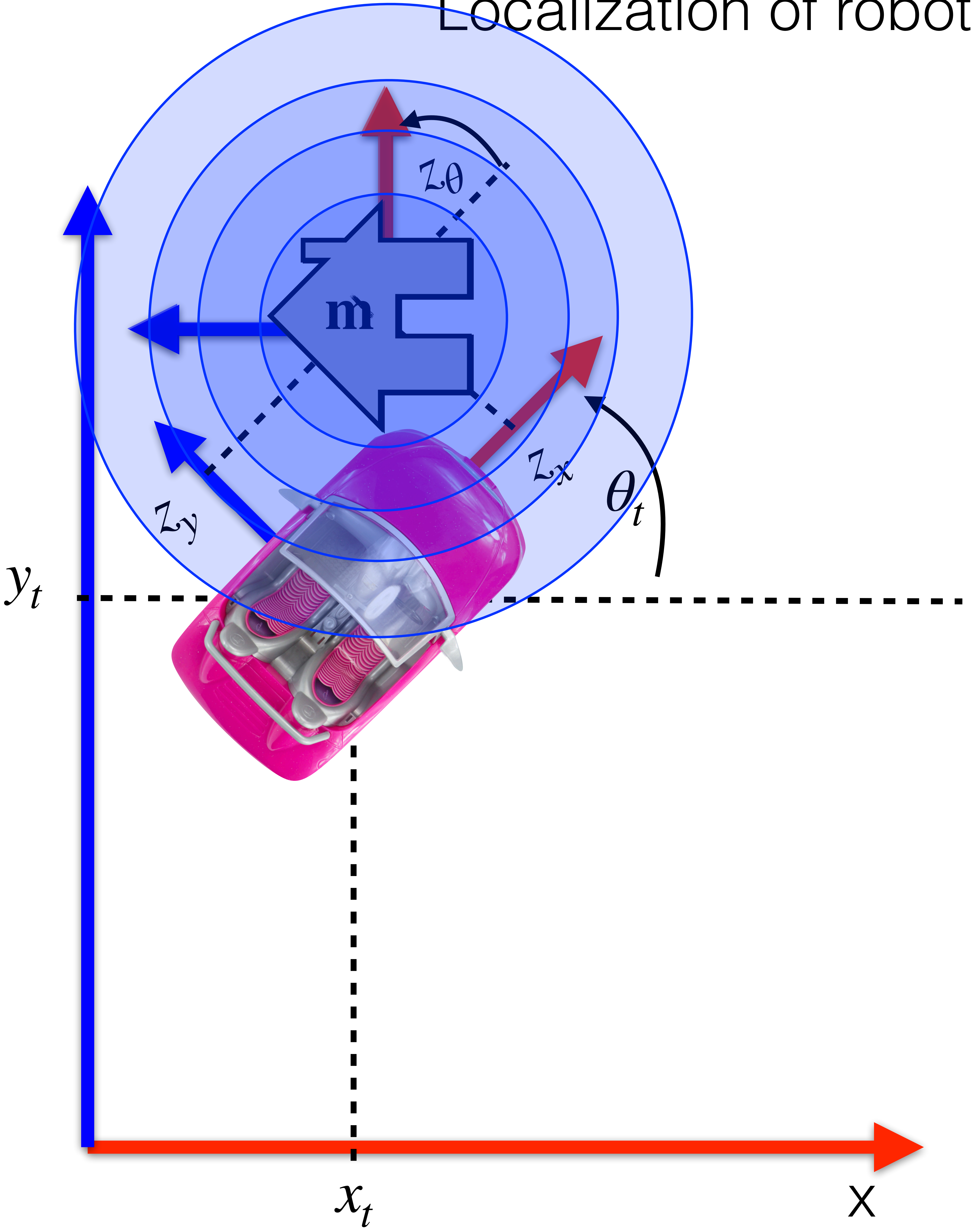
$$\mathbf{x}_t^\star = \arg\min_{\mathbf{x}_t} \|T^{-1}(\mathbf{m}, \mathbf{x}_t) - \mathbf{z}\|_\Sigma^2$$

If $\Sigma = \frac{1}{c} \cdot \mathbf{I} = \begin{bmatrix} 1/c & 0 & 0 \\ 0 & 1/c & 0 \\ 0 & 0 & 1/c \end{bmatrix}$

$$\mathbf{x}_t^\star = \arg\min_{\mathbf{x}_t} \; c \cdot \|(T^{-1}(\mathbf{m}, \mathbf{x}_t) - \mathbf{z})\|^2 \qquad \text{rcf}$$

$$= \arg\min_{\mathbf{x}_t} c \cdot \|T(\mathbf{z}, \mathbf{x}_t) - \mathbf{m}\|^2 \qquad \text{wcf}$$

Localization of robot in wcf from known position of the house



Marker pose in rcf $\hat{\mathbf{z}} = T^{-1}(\mathbf{m}, \mathbf{x}_t)$

Since pose is incorrect $\mathbf{z} \neq \hat{\mathbf{z}}$

Find the correct pose

$$\mathbf{x}_t^\star = \arg\min_{\mathbf{x}_t} \|T^{-1}(\mathbf{m}, \mathbf{x}_t) - \mathbf{z}\|_\Sigma^2$$

If $\Sigma = \frac{1}{c} \cdot \mathbf{I} = \begin{bmatrix} 1/c & 0 & 0 \\ 0 & 1/c & 0 \\ 0 & 0 & 1/c \end{bmatrix}$

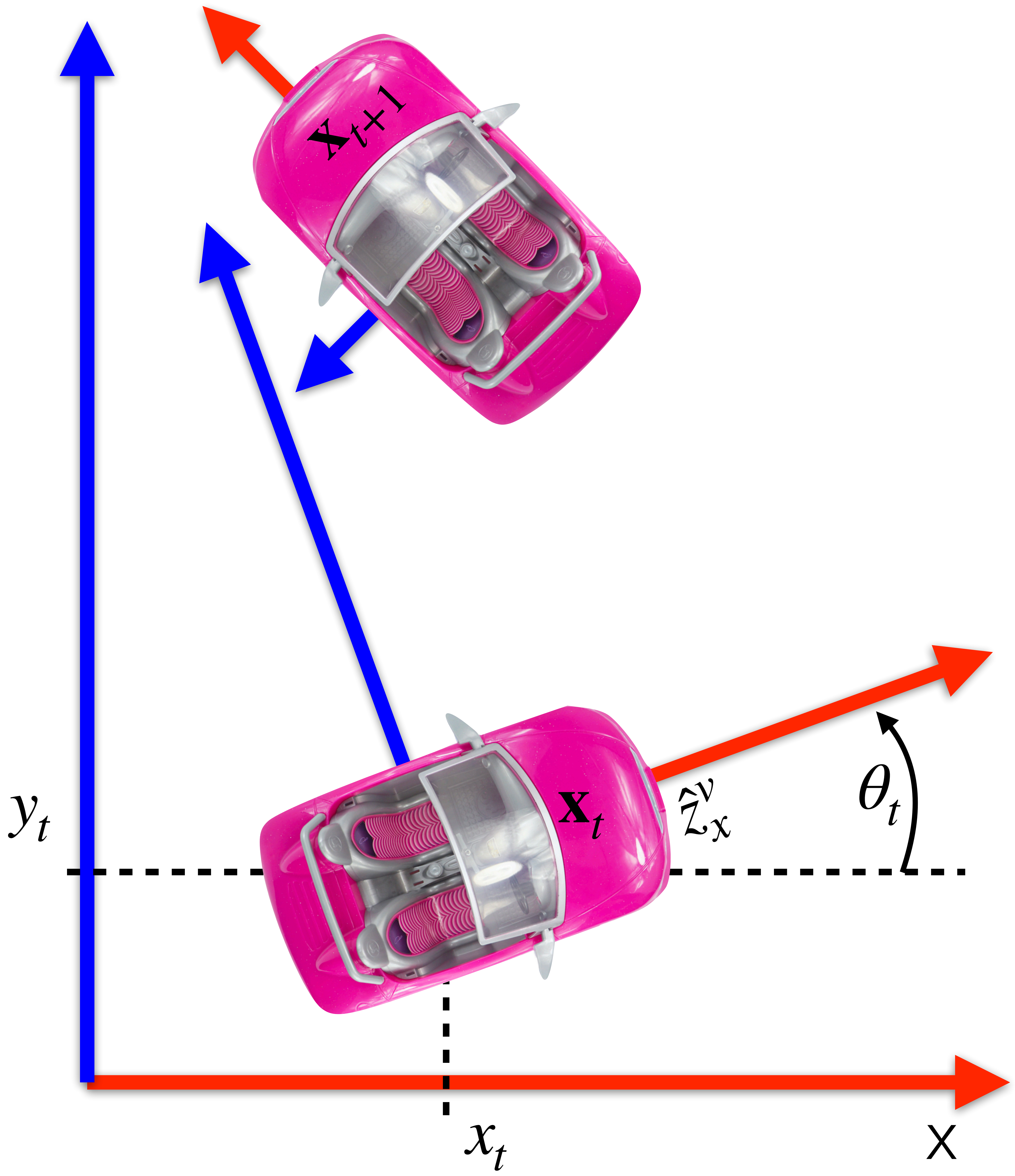
$$\mathbf{x}_t^\star = \arg\min_{\mathbf{x}_t} \; c \cdot \|(T^{-1}(\mathbf{m}, \mathbf{x}_t) - \mathbf{z})\|^2 \qquad \text{rcf}$$

$$= \arg\min_{\mathbf{x}_t} c \cdot \|T(\mathbf{z}, \mathbf{x}_t) - \mathbf{m}\|^2 \qquad \text{wcf}$$

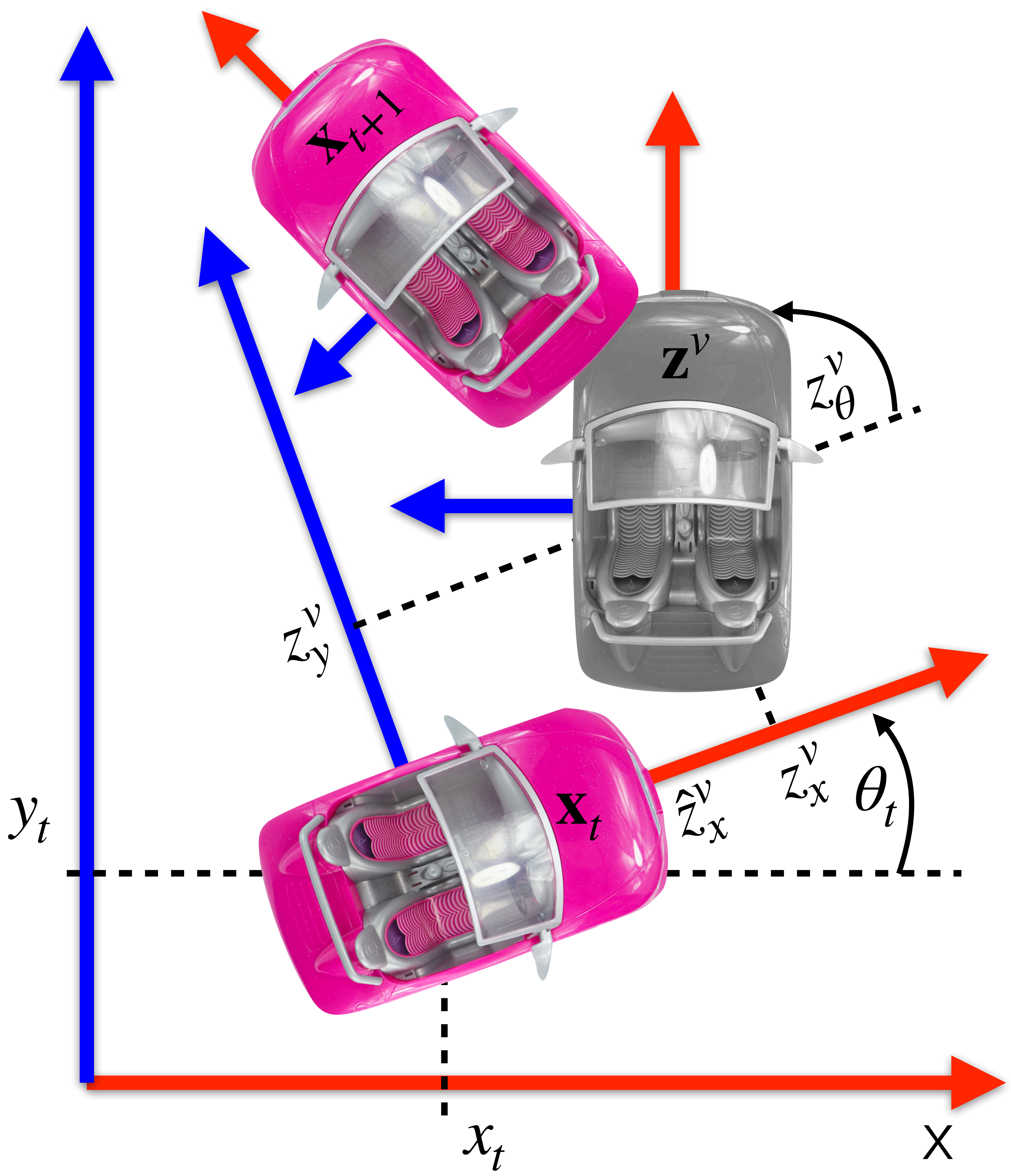
Localization of robot in wcf from known odometry

Odometry represented by linear+angular velocity

Robot poses in wcf: $\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} \quad \mathbf{x}_{t+1} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix}$



Localization of robot in wcf from known odometry

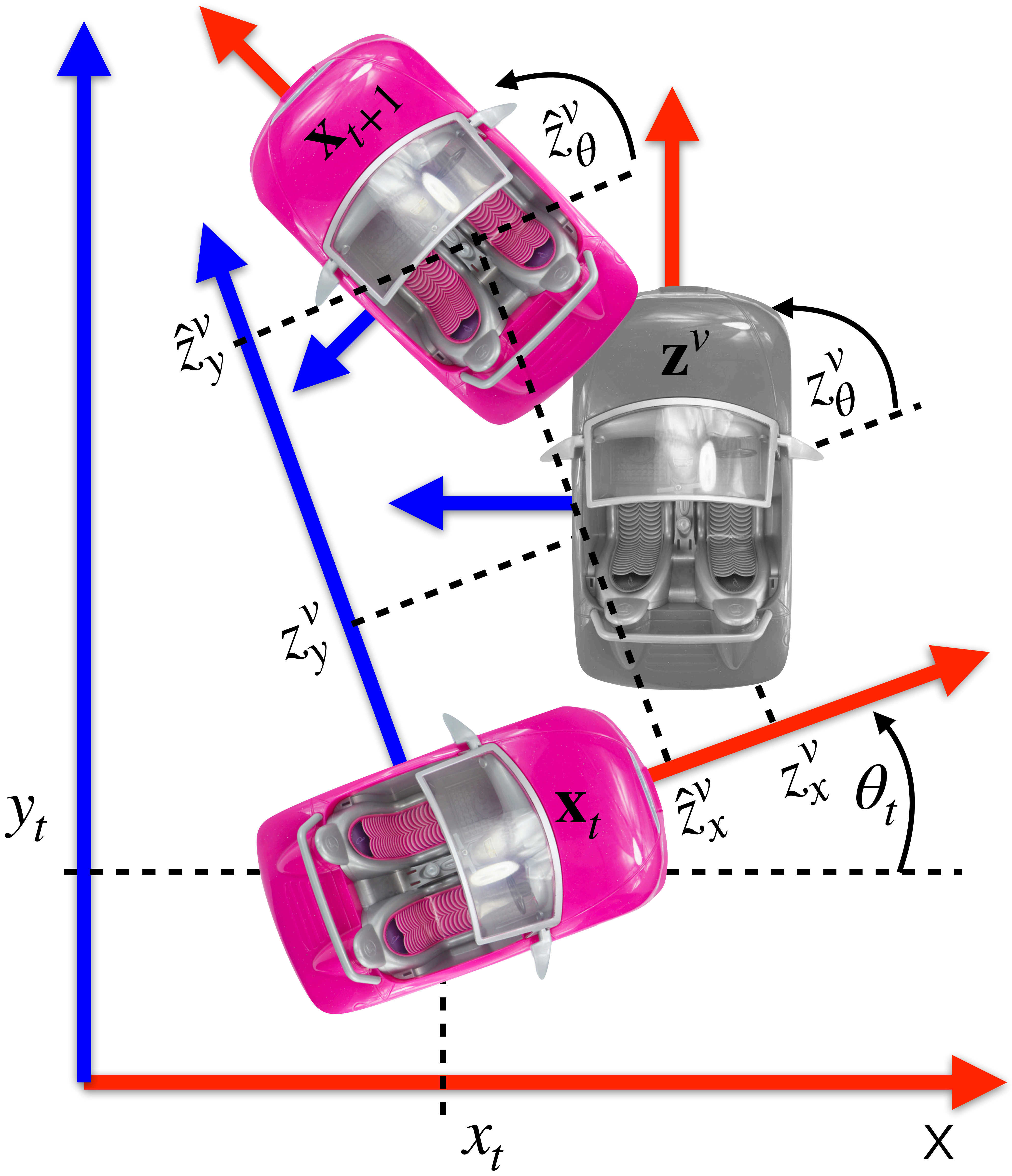


Odometry represented by linear+angular velocity

Robot poses in wcf: $\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} \quad \mathbf{x}_{t+1} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix}$

Robot measures velocity in $\mathbf{x}_t$-rcf: $\mathbf{z}^v = \begin{bmatrix} z^v_x \\ z^v_y \\ z^v_\theta \end{bmatrix}$

Localization of robot in wcf from known odometry



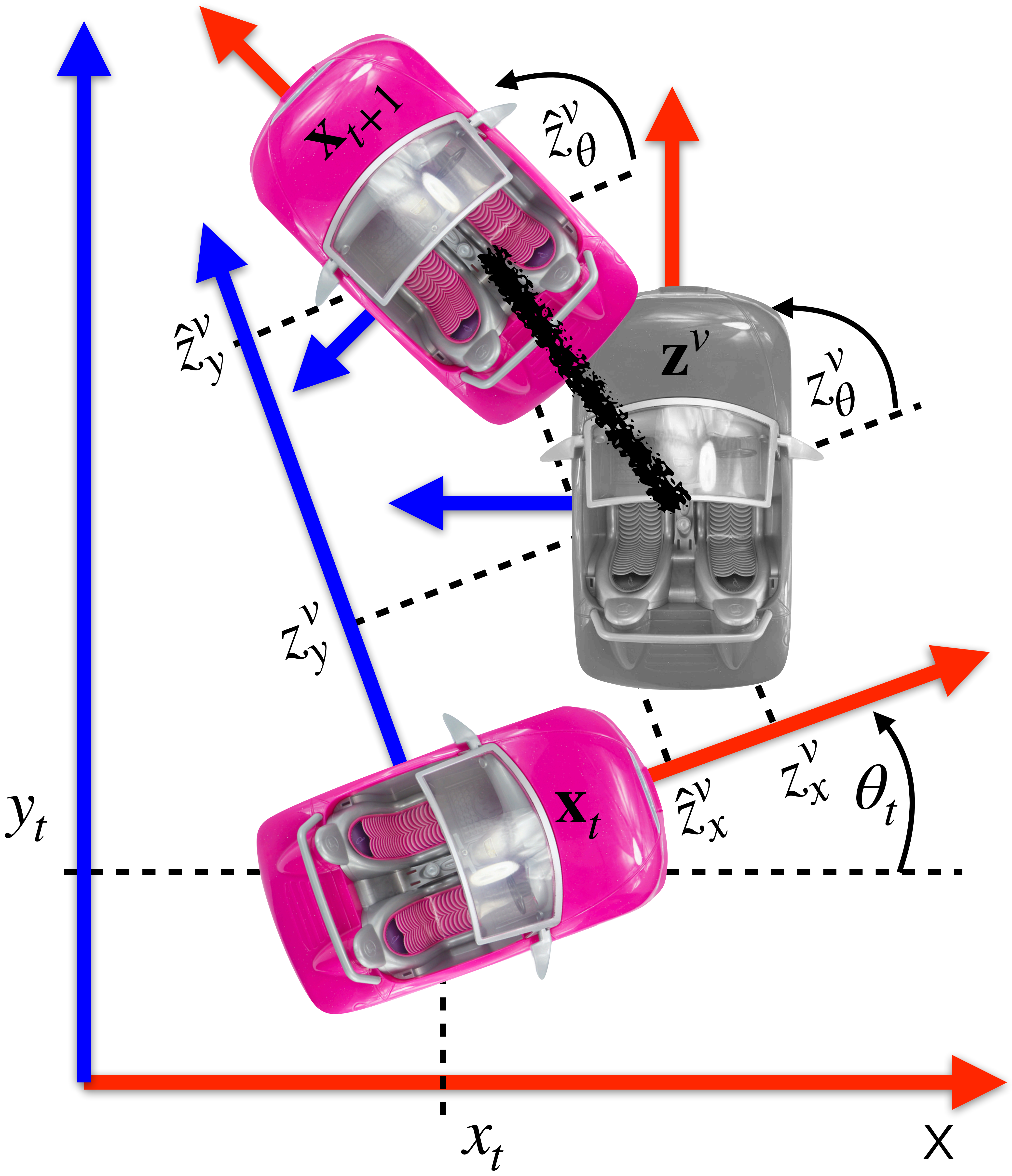
Odometry represented by linear+angular velocity

Robot poses in wcf: $\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} \quad \mathbf{x}_{t+1} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix}$

Robot measures velocity in $\mathbf{x}_t$-rcf: $\mathbf{z}^v = \begin{bmatrix} z^v_x \\ z^v_y \\ z^v_\theta \end{bmatrix}$

Next pose $\mathbf{x}_{t+1}$ in $\mathbf{x}_t$-rcf: $\hat{\mathbf{z}}^v = T^{-1}(\mathbf{x}_{t+1}, \mathbf{x}_t)$

Localization of robot in wcf from known odometry



Odometry represented by linear+angular velocity

Robot poses in wcf: $\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} \quad \mathbf{x}_{t+1} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix}$

Robot measures velocity in $\mathbf{x}_t$-rcf: $\mathbf{z}^v = \begin{bmatrix} z^v_x \\ z^v_y \\ z^v_\theta \end{bmatrix}$

Next pose $\mathbf{x}_{t+1}$ in $\mathbf{x}_t$-rcf: $\hat{\mathbf{z}}^v = T^{-1}(\mathbf{x}_{t+1}, \mathbf{x}_t)$

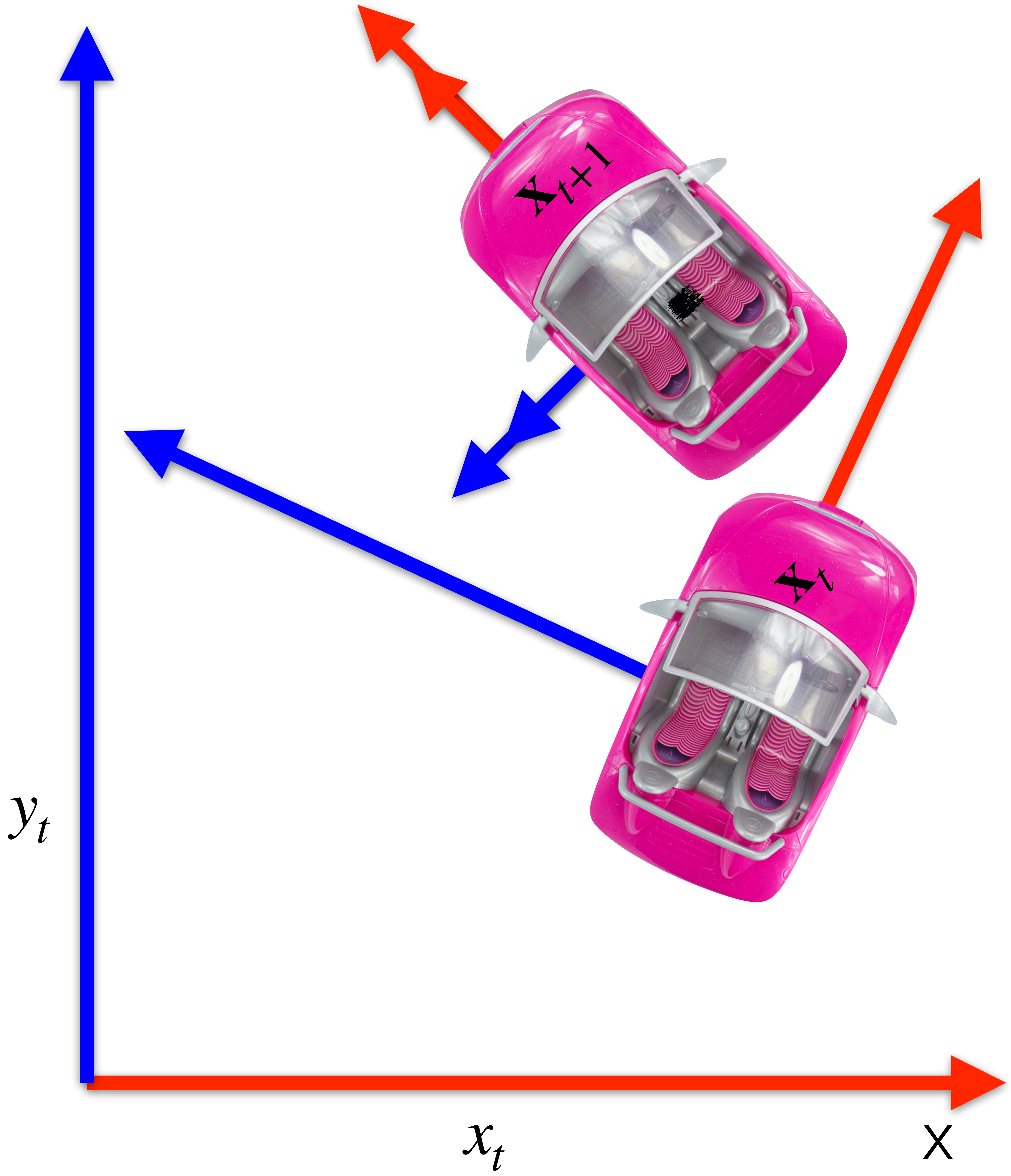
Since poses are incorrect $\mathbf{z}^v \neq \hat{\mathbf{z}}^v$

Find the correct poses

$$\mathbf{x}^\star_t, \mathbf{x}^\star_{t+1} = \arg\max_{\mathbf{x}_t, \mathbf{x}_{t+1}} \mathcal{N}(\mathbf{z}^v;\ T^{-1}(\mathbf{x}_{t+1}, \mathbf{x}_t), \Sigma^v)$$

$$= \arg\min_{\mathbf{x}_t, \mathbf{x}_{t+1}} \|T^{-1}(\mathbf{x}_{t+1}, \mathbf{x}_t) - \mathbf{z}^v\|^v_\Sigma$$

Localization of robot in wcf from known odometry



Odometry represented by linear+angular velocity

Robot poses in wcf: $\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} \quad \mathbf{x}_{t+1} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix}$

Robot measures velocity in $\mathbf{x}_t$-rcf: $\mathbf{z}^v = \begin{bmatrix} z_x^v \\ z_y^v \\ z_\theta^v \end{bmatrix}$

Next pose $\mathbf{x}_{t+1}$ in $\mathbf{x}_t$-rcf: $\hat{\mathbf{z}}^v = T^{-1}(\mathbf{x}_{t+1}, \mathbf{x}_t)$

Since poses are incorrect $\mathbf{z}^v \neq \hat{\mathbf{z}}^v$

Find the correct poses

$$\mathbf{x}_t^\star, \mathbf{x}_{t+1}^\star = \arg\max_{\mathbf{x}_t, \mathbf{x}_{t+1}} \mathcal{N}(\mathbf{z}^v;\ T^{-1}(\mathbf{x}_{t+1}, \mathbf{x}_t), \Sigma^v)$$
$$= \arg\min_{\mathbf{x}_t, \mathbf{x}_{t+1}} \|T^{-1}(\mathbf{x}_{t+1}, \mathbf{x}_t) - \mathbf{z}^v\|_\Sigma^v$$

Localization of robot in wcf from known marker pose, odometry and GPS

GPS | odometry | marker

$$\mathbf{x}^\star = \arg\max_{\mathbf{x}_0,\dots\mathbf{x}_t} \prod_t p(\mathbf{z}_t^{gps}\,|\,\mathbf{x}_t) \cdot \prod_t p(\mathbf{z}_t^{v}\,|\,\mathbf{x}_t, \mathbf{x}_{t-1}) \cdot \prod_t p(\mathbf{z}_t^{m}\,|\,\mathbf{x}_t, \mathbf{m})$$

$$= \arg\max_{\mathbf{x}_0,\dots\mathbf{x}_t} \prod_t \mathcal{N}(\mathbf{z}^{gps};\ \mathbf{x}_t, \Sigma_t^{gps}) \cdot \prod_t \mathcal{N}(\mathbf{z}^{v};\ T^{-1}(\mathbf{x}_{t+1}, \mathbf{x}_t), \Sigma_t^{v}) \cdot \prod_t \mathcal{N}(\mathbf{z}^{m};\ T^{-1}(\mathbf{m}, \mathbf{x}_t), \Sigma_t^{m})$$

$$= \arg\min_{\mathbf{x}_0,\dots\mathbf{x}_T} \sum_t \|\mathbf{x}_t - \mathbf{z}_t^{gps}\|^2_{\Sigma_t^{gps}} + \sum_t \|T^{-1}(\mathbf{x}_{t+1}, \mathbf{x}_t) - \mathbf{z}_t^{v}\|^2_{\Sigma_t^{v}} + \sum_t \|T^{-1}(\mathbf{m}, \mathbf{x}_t) - \mathbf{z}^{m}\|^2_{\Sigma_t^{m}}$$

Straightforward extensions

GPS | odometry | marker

$$= \arg\min_{\mathbf{x}_0,\ldots\mathbf{x}_T} \sum_t \|\mathbf{x}_t - \mathbf{z}_t^{gps}\|^2_{\Sigma_t^{gps}} + \sum_t \|T^{-1}(\mathbf{x}_{t+1}, \mathbf{x}_t) - \mathbf{z}_t^{v}\|^2_{\Sigma_t^{v}} + \sum_t \|T^{-1}(\mathbf{m}, \mathbf{x}_t) - \mathbf{z}^{m}\|^2_{\Sigma_t^{m}}$$

Straightforward extensions

GPS, odometry, marker(s), priors, motion model, loop-closures

$$= \arg\min_{\substack{\mathbf{x}_0,\dots\mathbf{x}_T \\ \mathbf{m}^1\dots\mathbf{m}^J}} \underbrace{\sum_t \|\mathbf{x}_t - \mathbf{z}_t^{gps}\|^2_{\Sigma_t^{gps}}}_{\text{GPS}} + \underbrace{\sum_t \|T^{-1}(\mathbf{x}_{t+1}, \mathbf{x}_t) - \mathbf{z}^v\|^2_{\Sigma_t^v}}_{\text{odometry}} + \underbrace{\sum_j \sum_t \|T^{-1}(\mathbf{m}^j, \mathbf{x}_t) - \mathbf{z}\|^2_{\Sigma_t^{m}}}_{\text{marker(s)}}$$

$$+ \underbrace{\sum_t \|\mathbf{x}_t - \mathbf{x}_t^{prior}\|^2_{\Sigma_t^{prior}}}_{\text{priors}} + \underbrace{\sum_t \|g(\mathbf{x}_{t-1}, \mathbf{u}_t) - \mathbf{x}_t\|^2_{\Sigma_t^g}}_{\text{motion model}} + \underbrace{\sum_t \|T^{-1}(\mathbf{x}_0, \mathbf{x}_T)\|^2_{\Sigma_t^{lc}}}_{\text{loop-closures}}$$

Localization => SLAM

Replace gaussian by other exponential distribution
(i.e. minimize L1-norm, robust loss Huber-norm)

Diagonal covariance

GPS odometry marker(s)

$$= \arg\min_{\substack{\mathbf{x}_0,\dots\mathbf{x}_T \\ \mathbf{m}^1\dots\mathbf{m}^J}} \sum_t \|\mathbf{x}_t - \mathbf{z}_t^{gps}\|^2_{\Sigma_t^{gps}} + \sum_t \|T^{-1}(\mathbf{x}_{t+1}, \mathbf{x}_t) - \mathbf{z}^v\|^2_{\Sigma_t^v} + \sum_j \sum_t \|T^{-1}(\mathbf{m}^j, \mathbf{x}_t) - \mathbf{z}\|^2_{\Sigma_t^{m_j}}$$

If $\Sigma_t = \frac{1}{c_t} \cdot \mathbf{I} = \begin{bmatrix} 1/c_t & 0 & 0 \\ 0 & 1/c_t & 0 \\ 0 & 0 & 1/c_t \end{bmatrix}$ then:

GPS odometry marker(s)

$$= \arg\min_{\substack{\mathbf{x}_0,\dots\mathbf{x}_T \\ \mathbf{m}^1\dots\mathbf{m}^J}} \sum_t c_t^{gps} \|\mathbf{x}_t - \mathbf{z}_t^{gps}\|^2 + \sum_t c_t^v \|T(\mathbf{z}^v, \mathbf{x}_t) - \mathbf{x}_{t+1}\|^2 + \sum_j \sum_t c_t^{m^j} \|(T(\mathbf{z}^{m^j}, \mathbf{x}_t) - \mathbf{m}^j)\|^2$$

Optimization

GPS — odometry — marker(s)

$$= \arg\min_{\substack{\mathbf{x}_0,\dots\mathbf{x}_T \\ \mathbf{m}^1\dots\mathbf{m}^J}} \sum_t c_t^{gps}\|\mathbf{x}_t - \mathbf{z}_t^{gps}\|^2 + \sum_t c_t^{v}\|T(\mathbf{z}^v, \mathbf{x}_t) - \mathbf{x}_{t+1}\|^2 + \sum_j \sum_t c_t^{m^j}\|(T(\mathbf{z}^{m^j}, \mathbf{x}_t) - \mathbf{m}^j)\|^2$$

$$= \arg\min_{\mathbf{x}} \sum_i \|f_i(\mathbf{x})\|^2 = \arg\min_{\mathbf{x}} \left\| \begin{matrix} f_1(\mathbf{x}) \\ \vdots \\ f_N(\mathbf{x}) \end{matrix} \right\|^2 = \arg\min_{\mathbf{x}} \|f(\mathbf{x})\|^2$$

where $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^m$

$f'(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^{m\times n}$

Alternative formulation: $\arg\min_{\Delta\mathbf{x}} \|f(\mathbf{x}_k + \Delta\mathbf{x})\|^2$ where $\mathbf{x}_k$ is an initial solution

$$\approx \arg\min_{\Delta\mathbf{x}} \|f(\mathbf{x}_k) + f'(\mathbf{x}_k)\Delta\mathbf{x}\|^2 = -[f'(\mathbf{x}_k)]^+ f(\mathbf{x}_k)$$

GN: $\mathbf{x}_{k+1} = \mathbf{x}_k - [f'(\mathbf{x}_k)]^+ f(\mathbf{x}_k)$

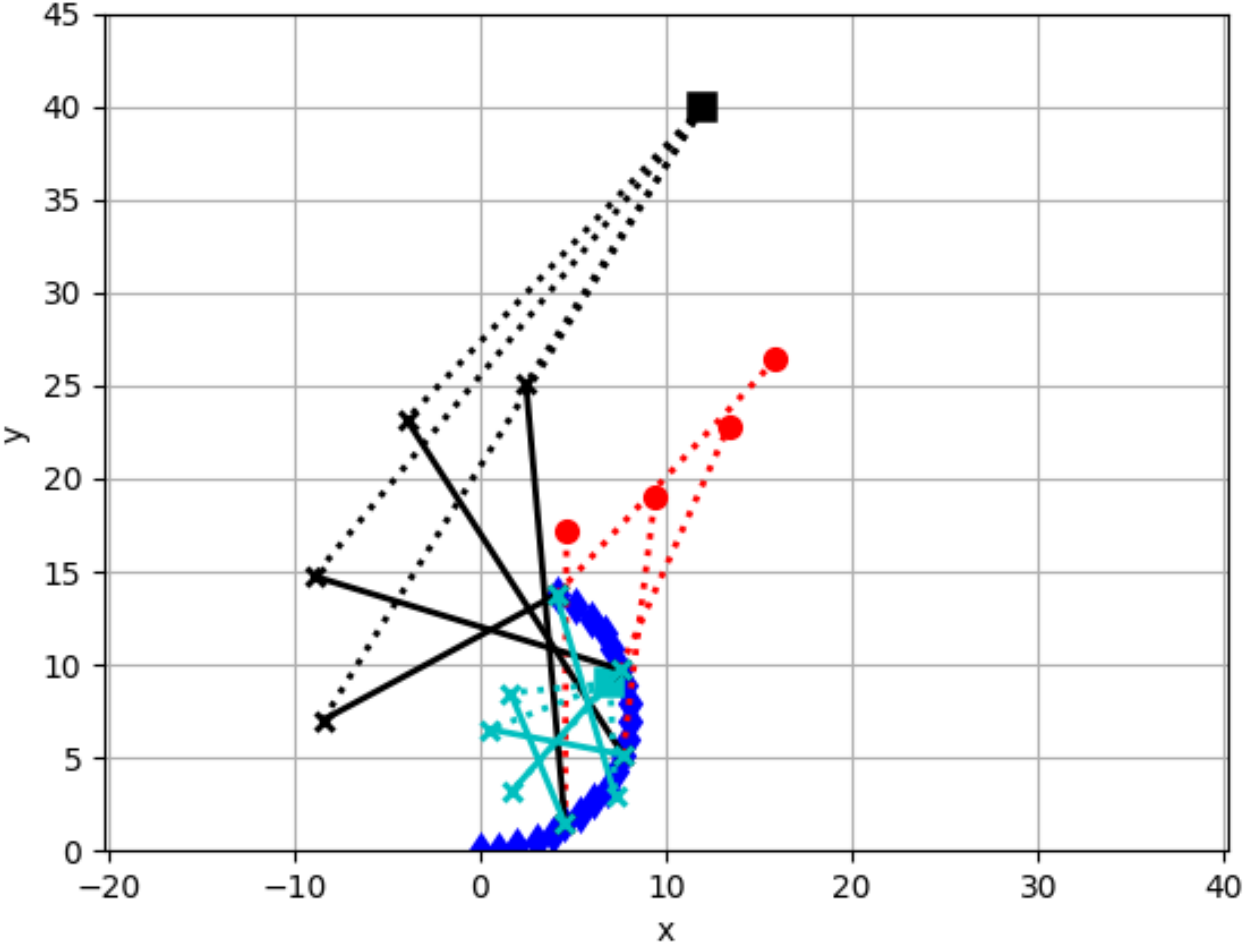
$$\approx \arg\min_{\Delta\mathbf{x}} \|f(\mathbf{x}_k) + f'(\mathbf{x}_k)\Delta\mathbf{x}\|^2 = -[f'(\mathbf{x}_k) + \lambda\mathbf{I}]^+ f(\mathbf{x}_k)$$

subject to $\|\Delta\mathbf{x}\|^2 \leq c$

LM:
TR: $\mathbf{x}_{k+1} = \mathbf{x}_k - [f'(\mathbf{x}_k) + \lambda\mathbf{I}]^+ f(\mathbf{x}_k)$

```
scipy.optimize.least_squares(fun, x0, jac, method='lm')
```

Optimization in SE(2) manifold trajectory length 21



- absolute marker
- relative marker
- odometry
- some optimised poses
- some ground truth poses (not used in optimisation)

noise:

- odom 0.2m / 0.2rad
- markers 0.3m / 0.3rad

Optimization in SE(2) manifold trajectory length 21

noise:

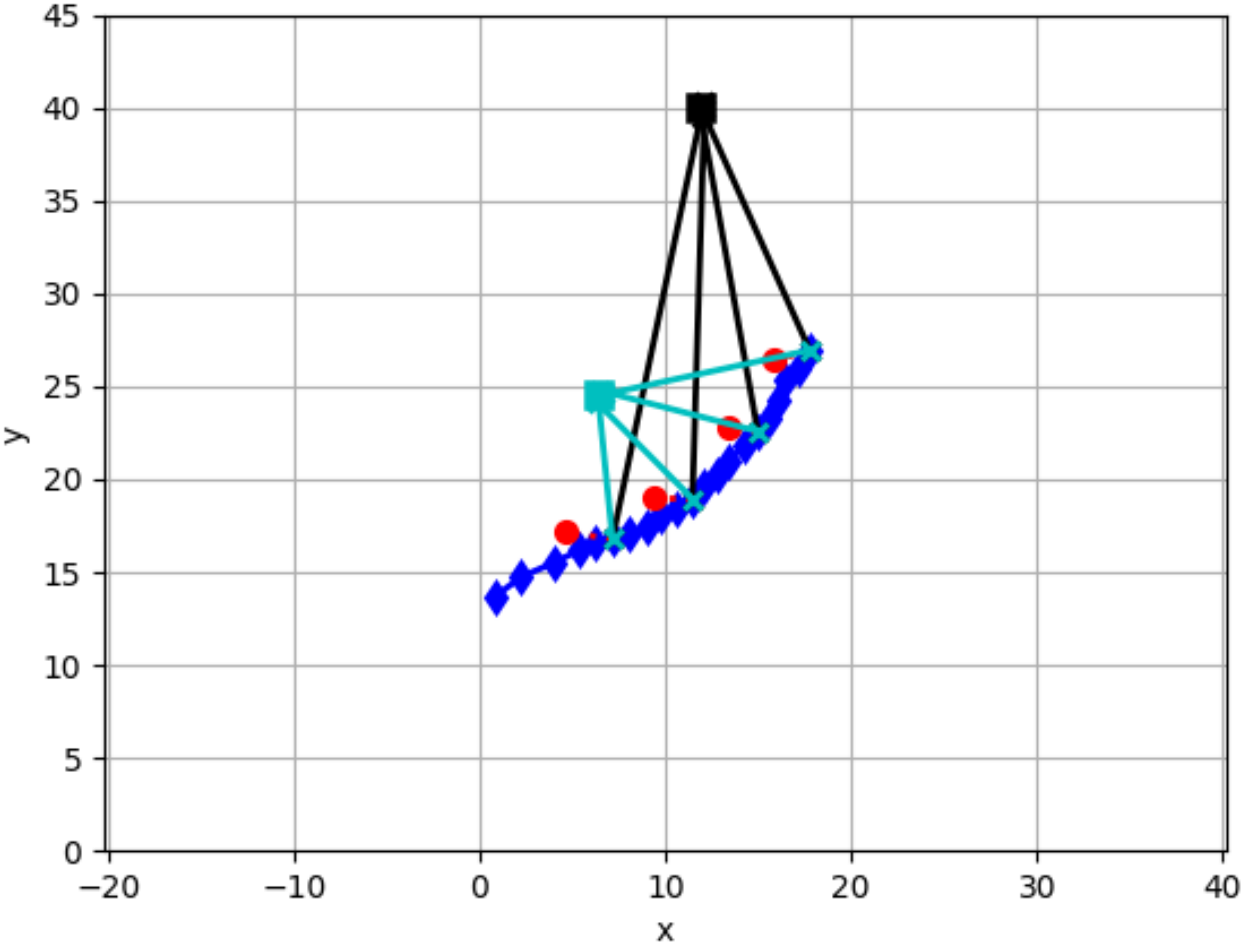
- odom 0.2m / 0.2rad
- markers 0.3m / 0.3rad

Optimization in SE(2) manifold trajectory length 21

noise:

- odom 0.2m / 0.2rad
- markers 0.3m / 0.3rad

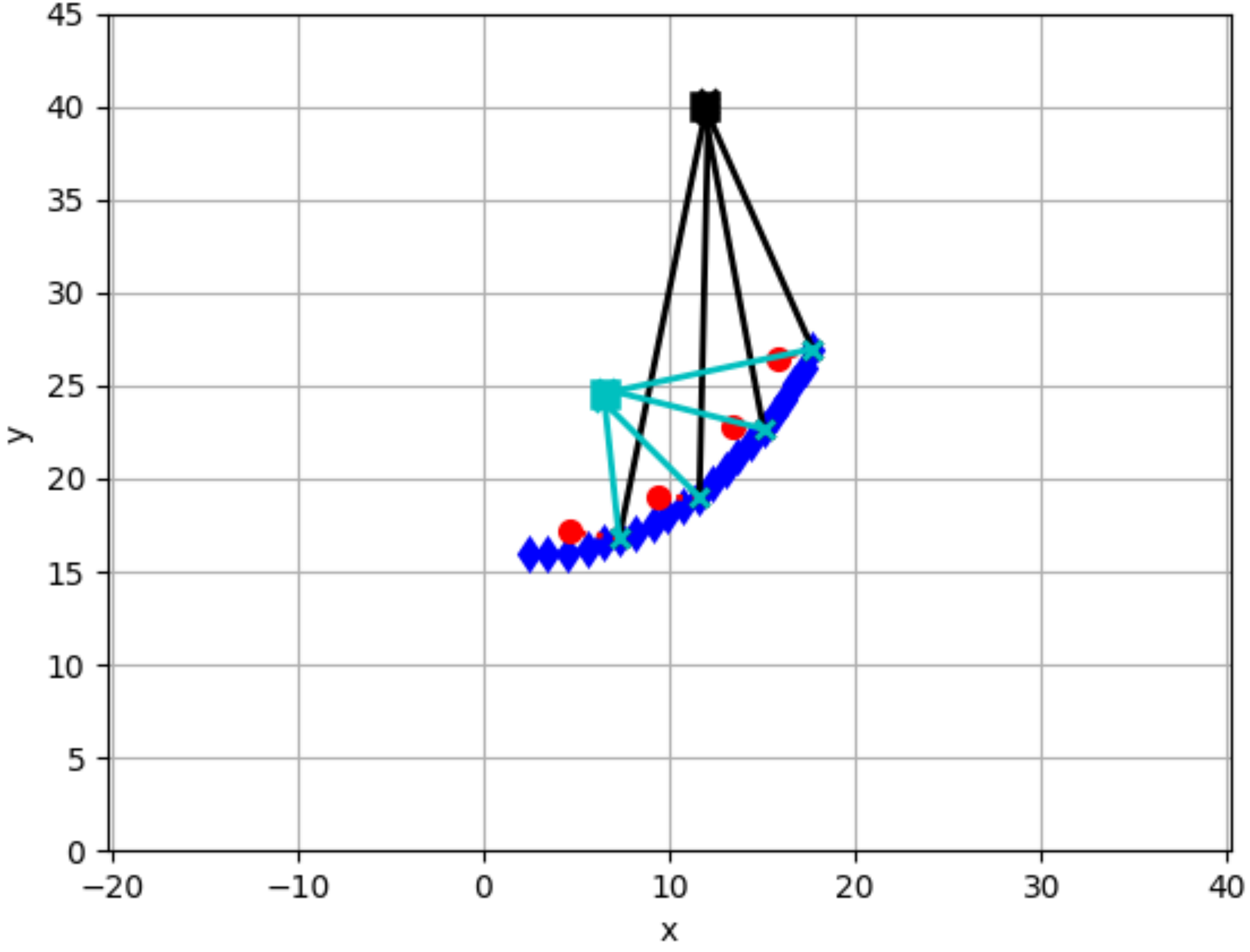
Optimization in SE(2) manifold trajectory length 21



noise:

- odom 0.2m / 0.2rad
- markers 0.3m / 0.3rad

Optimization in SE(2) manifold trajectory length 21



noise:

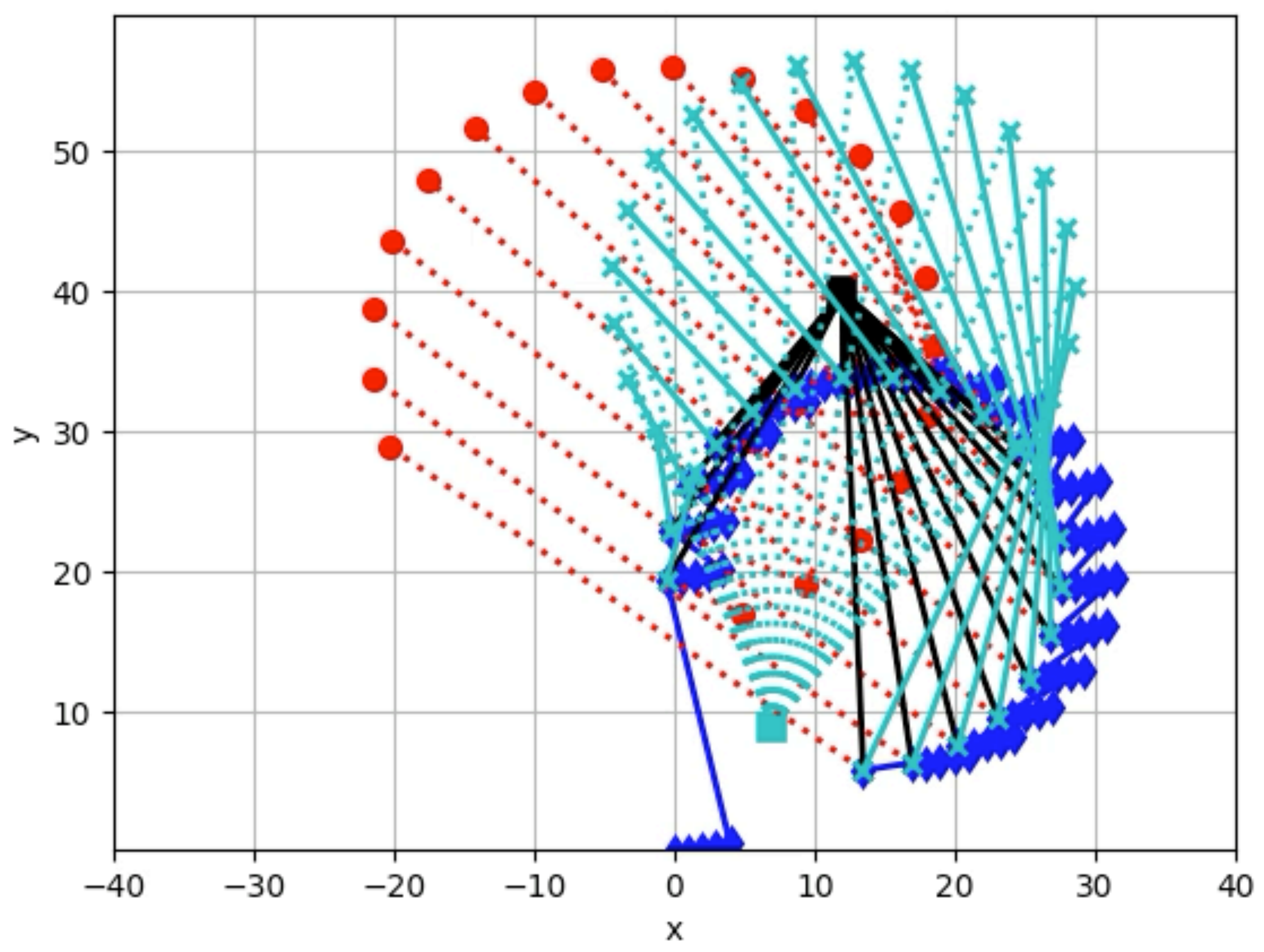
- odom 0.2m / 0.2rad
- markers 0.3m / 0.3rad

Optimization in SE(2) manifold trajectory length 101

What will break it????

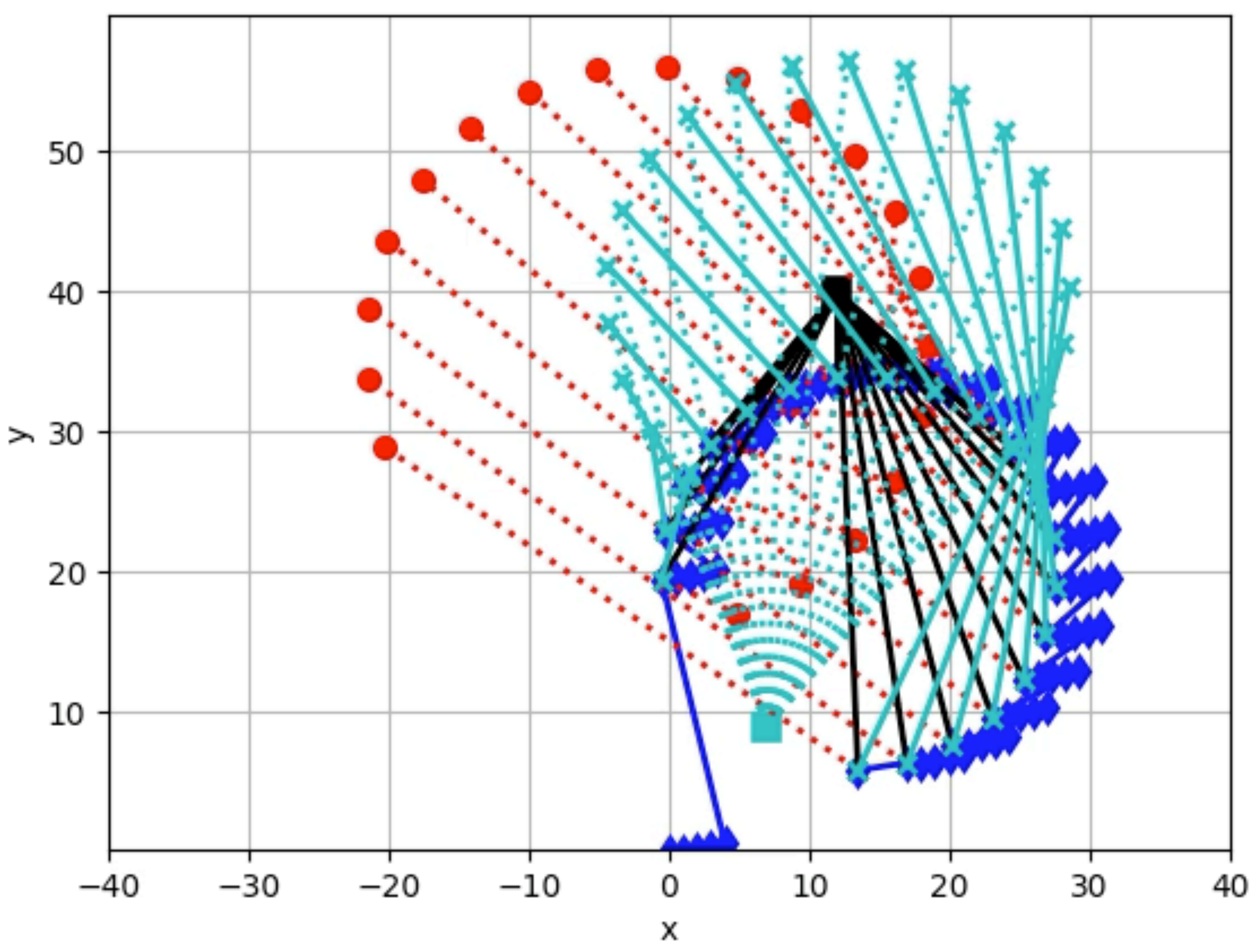
Optimization in SE(2) manifold trajectory length 101

noise 0

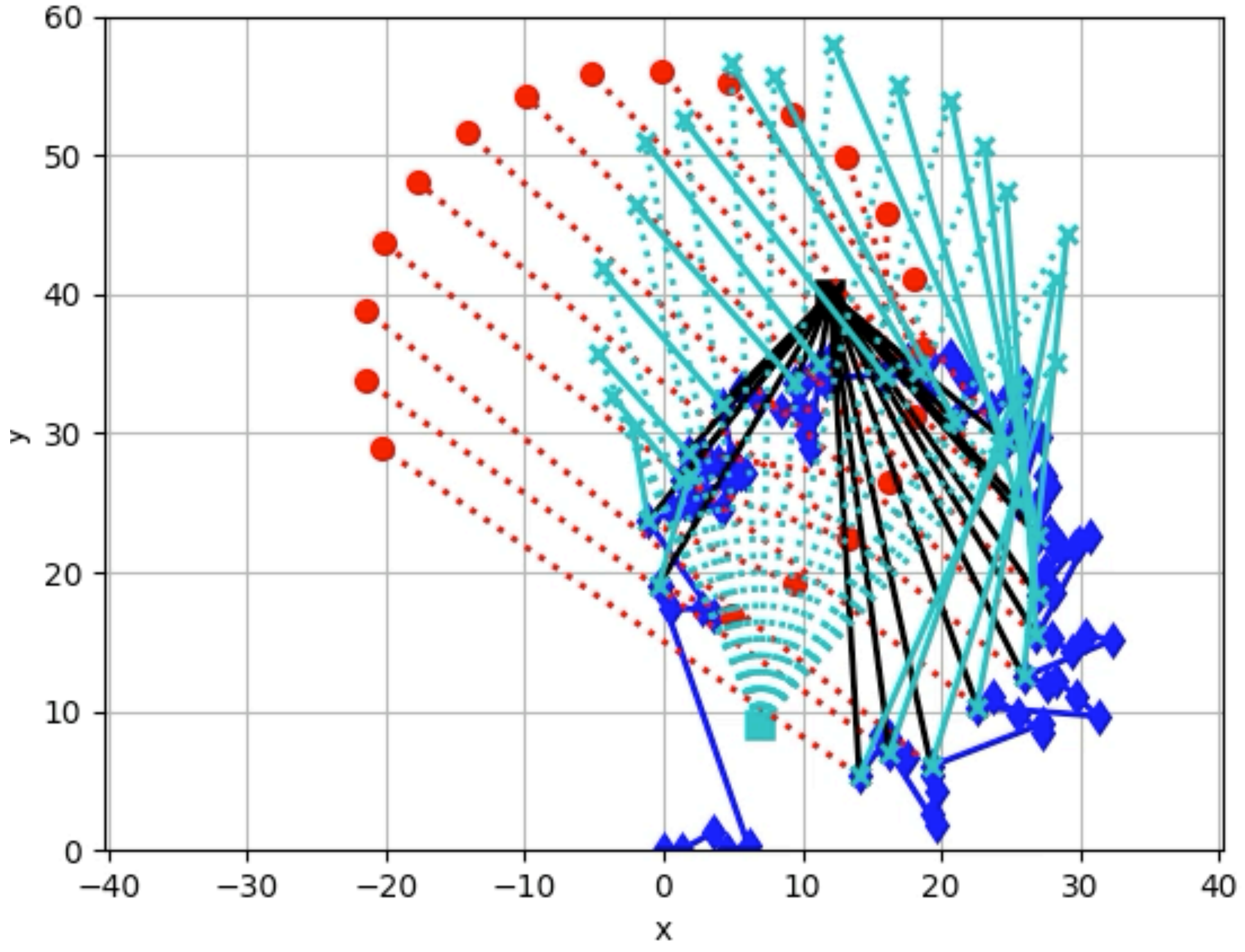


Optimization in SE(2) manifold trajectory length 101

noise 0

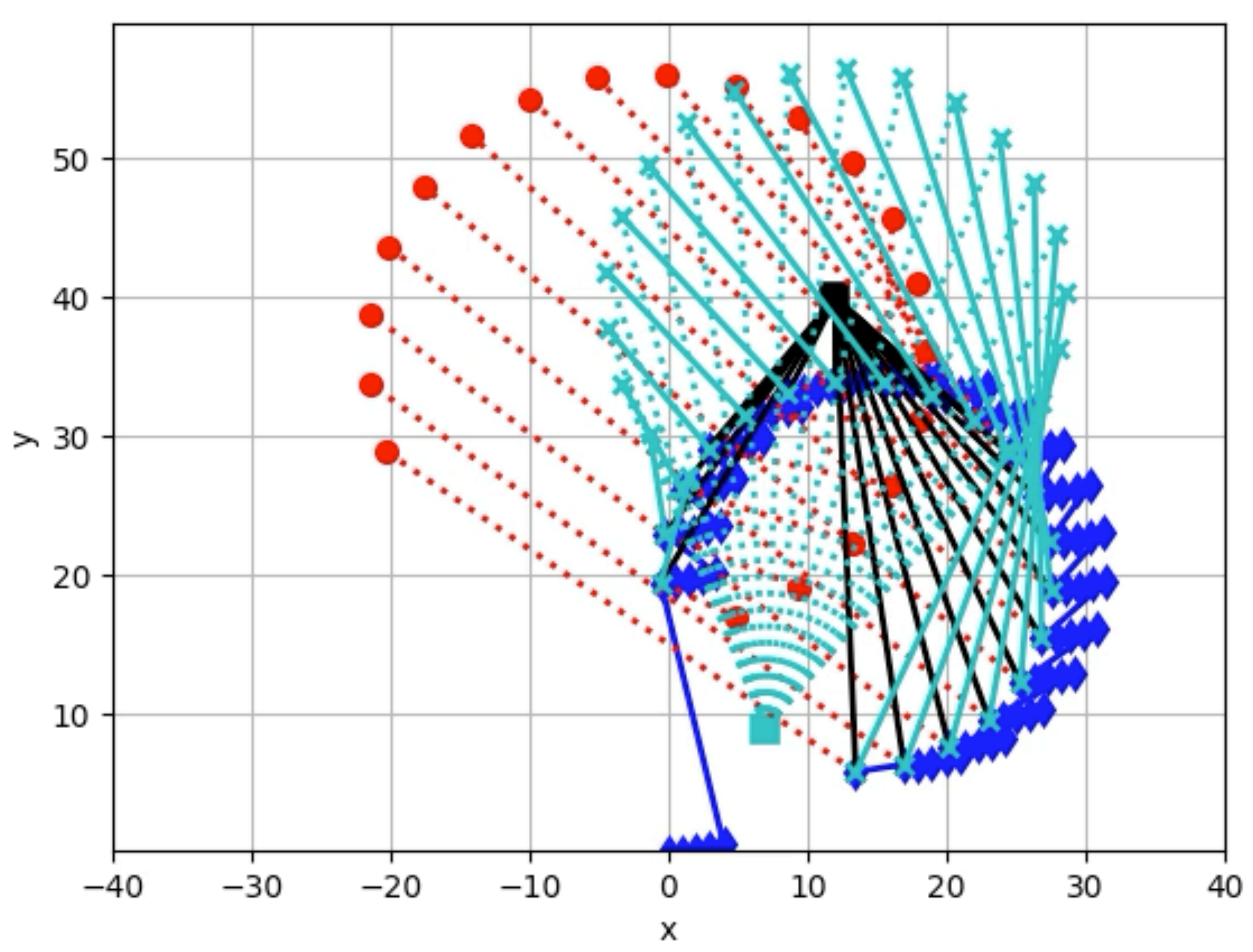


noise 0.7

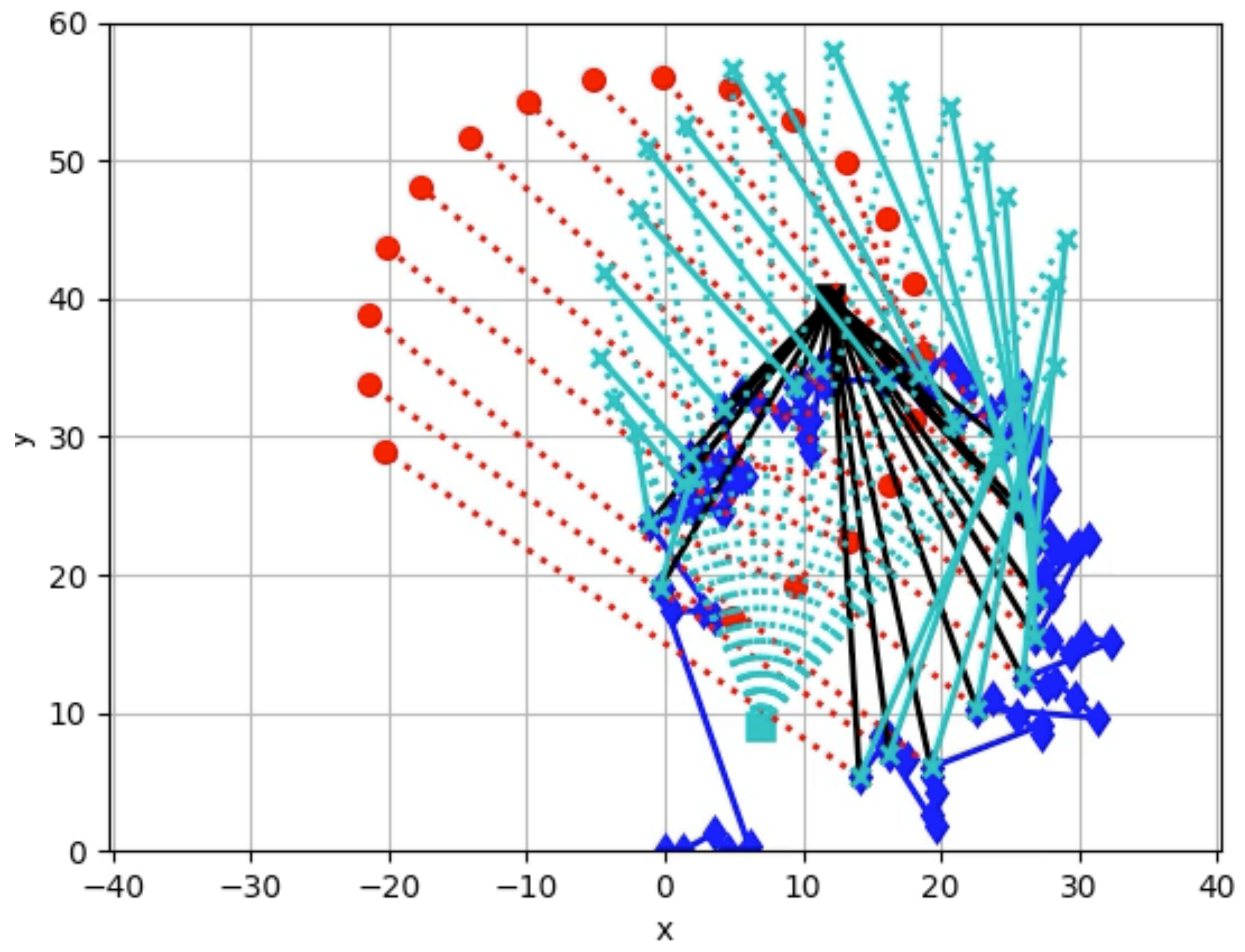


Optimization in SE(2) manifold trajectory length 101

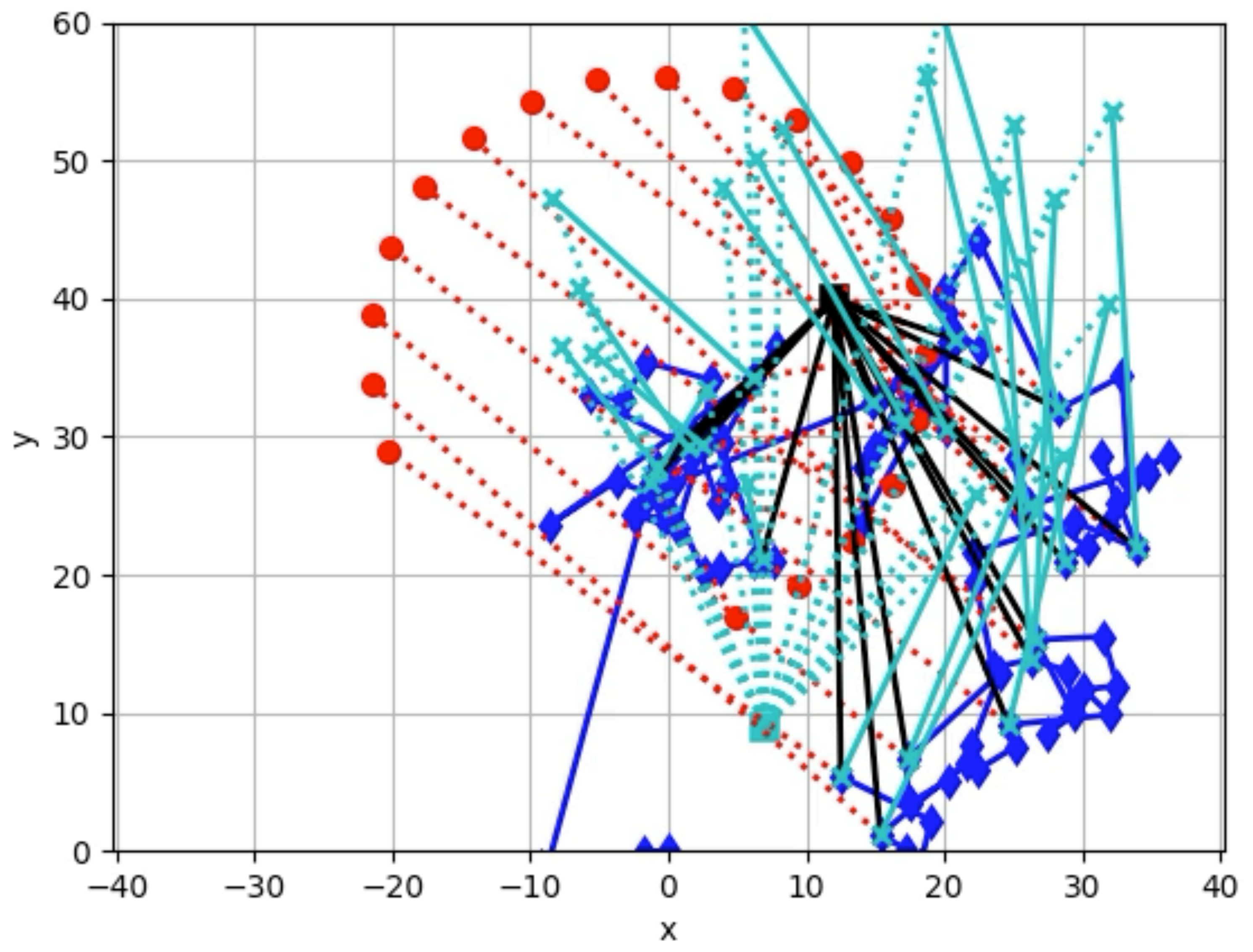
noise 0



noise 0.7



noise 2.5

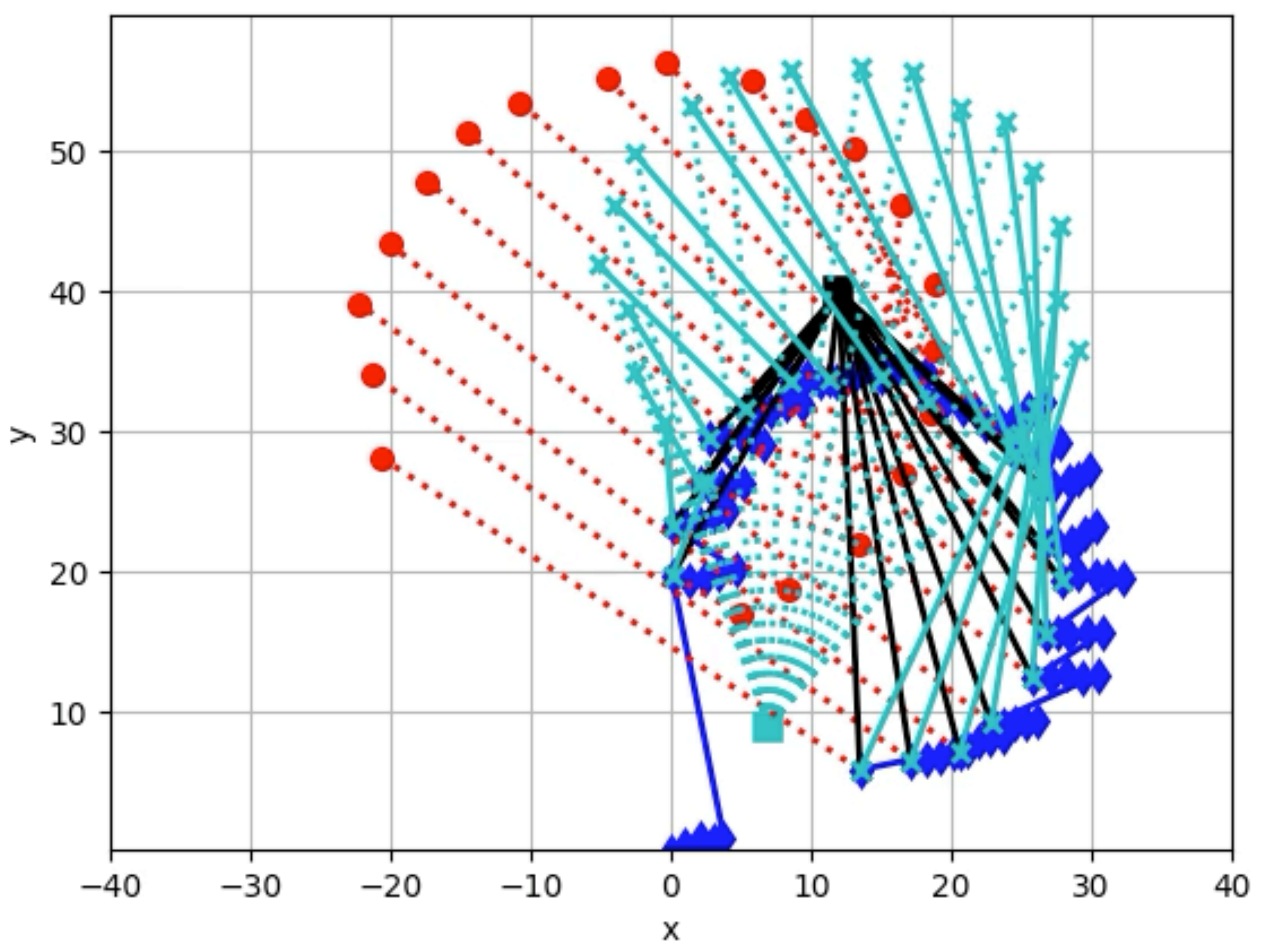


noise 0.7 Optimization in SE(2) manifold trajectory length 101

c_odom=**0**
c_ma=1
c_mr=1

noise 0.7 Optimization in SE(2) manifold trajectory length 101

c_odom=**0**
c_ma=1
c_mr=1

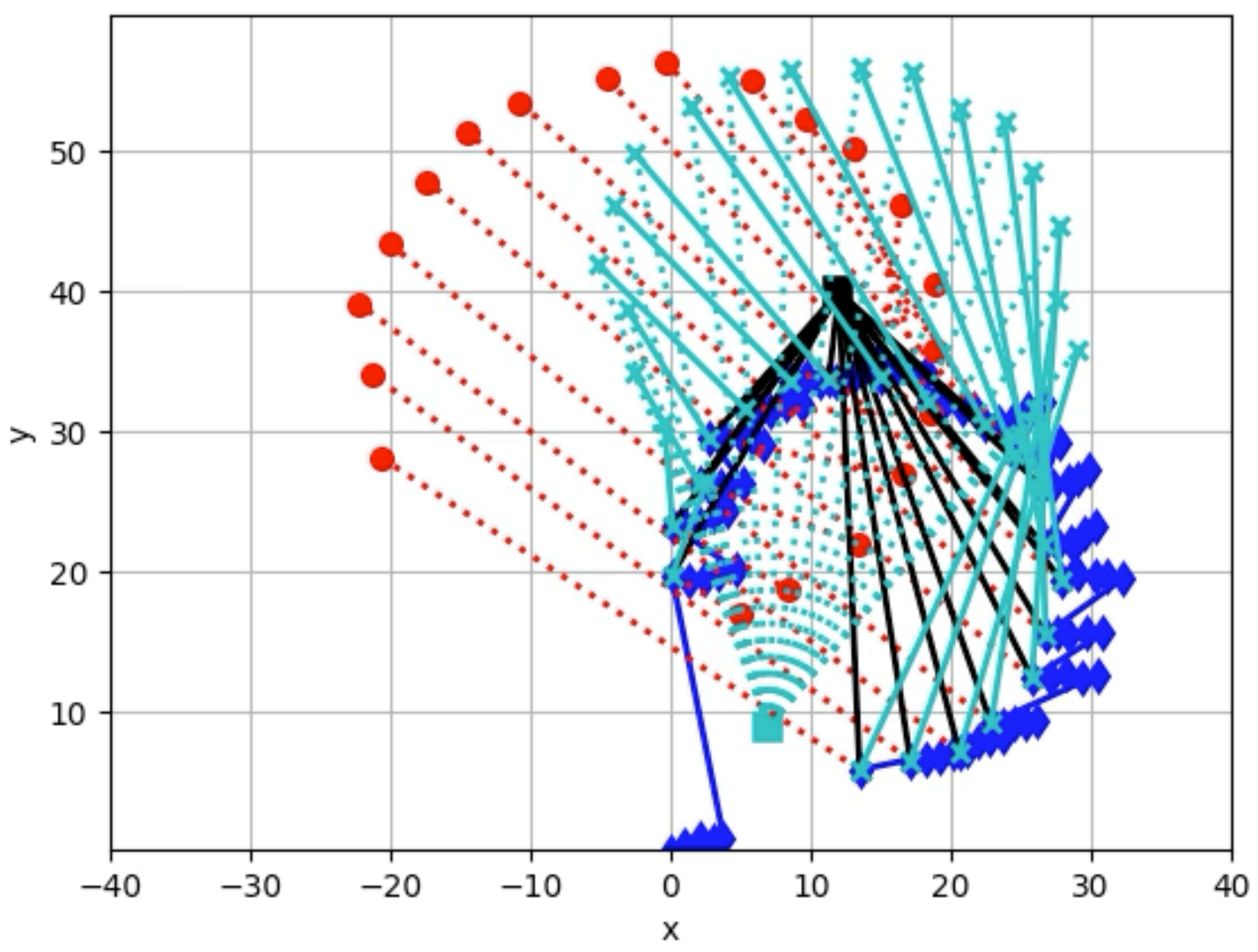


noise 0.7

Optimization in SE(2) manifold trajectory length 101

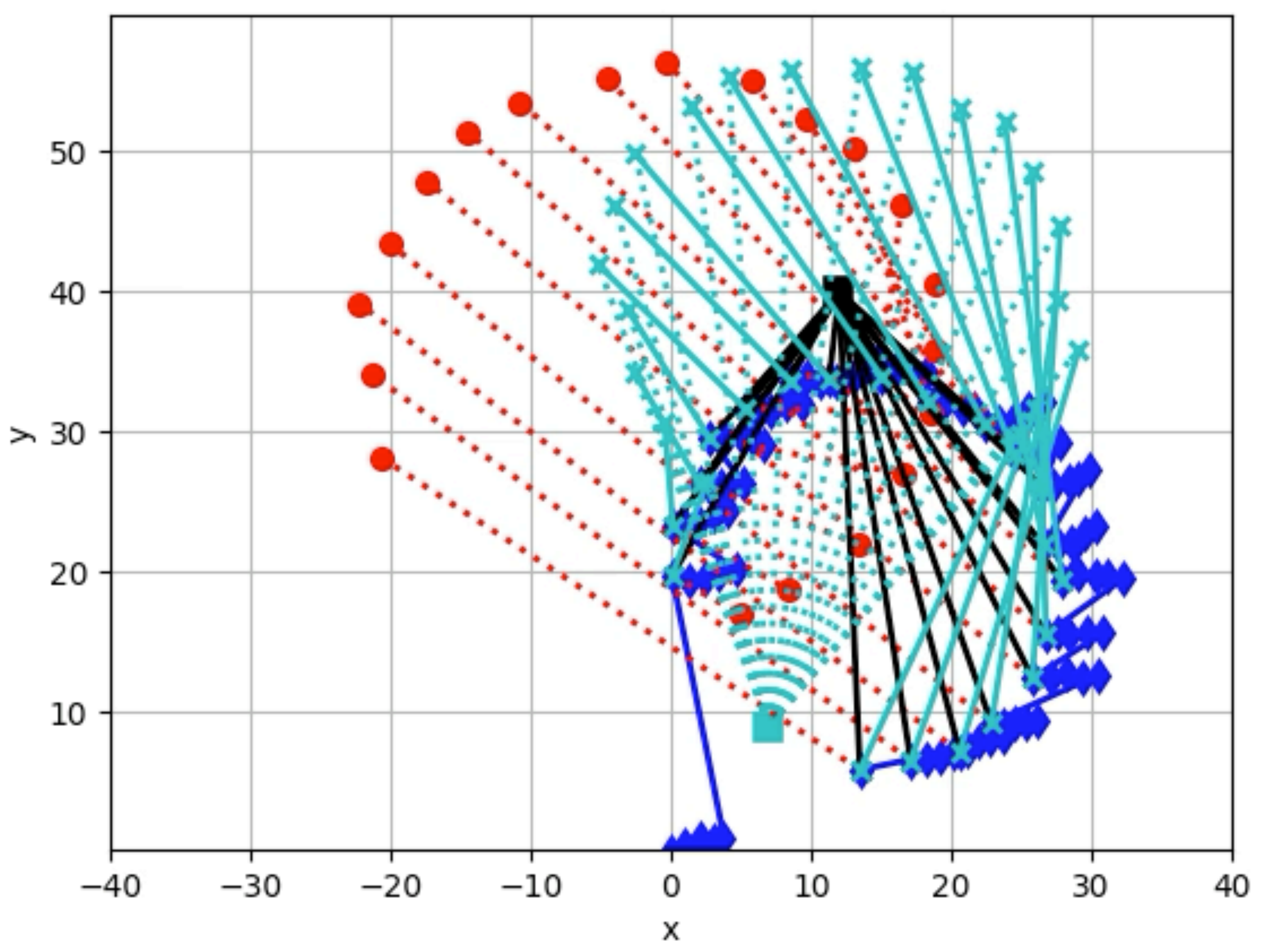
c_odom=**0**
c_ma=1
c_mr=1

c_odom=1
c_ma=**0**
c_mr=1

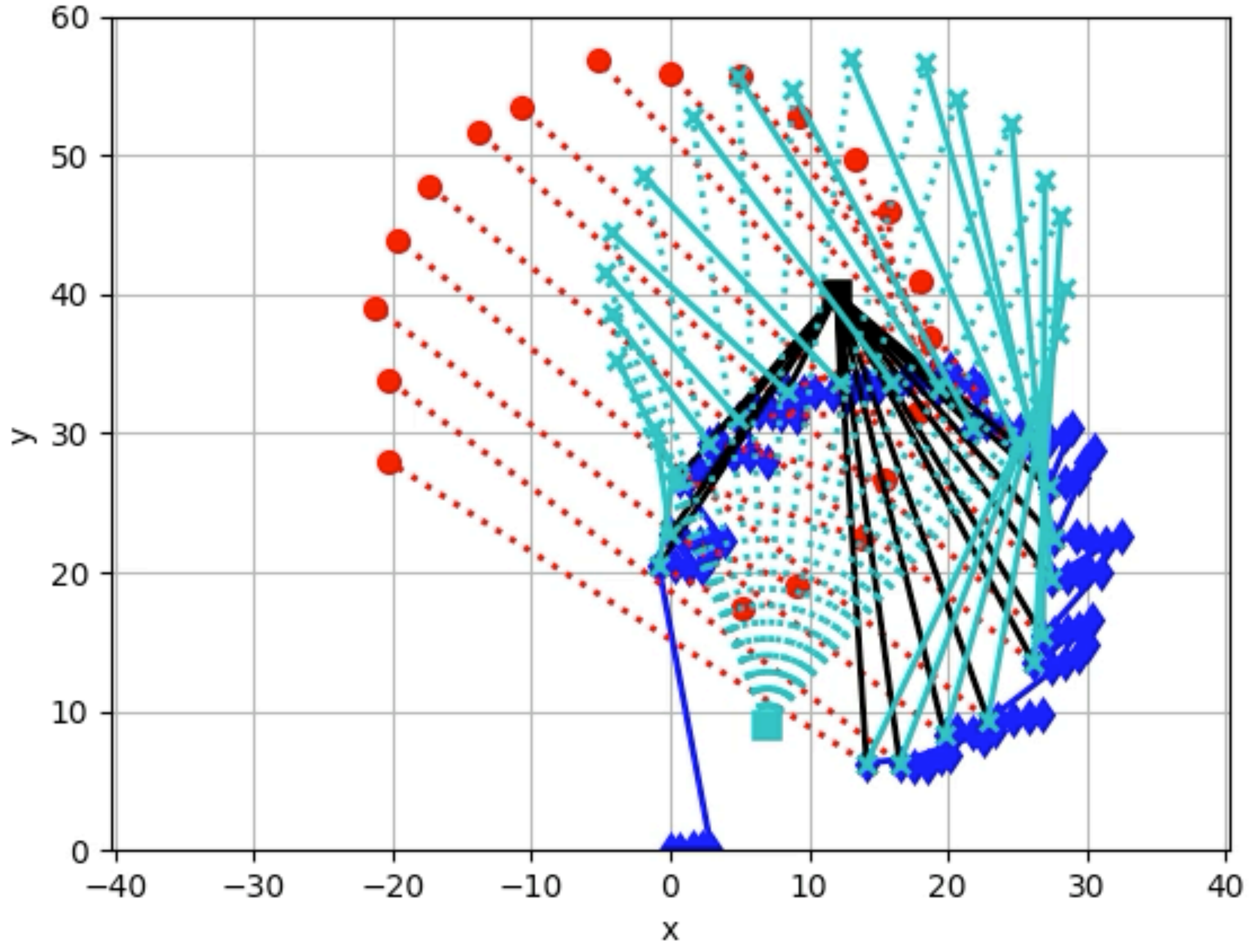


noise 0.7 Optimization in SE(2) manifold trajectory length 101

c_odom=**0**
c_ma=1
c_mr=1



c_odom=1
c_ma=**0**
c_mr=1

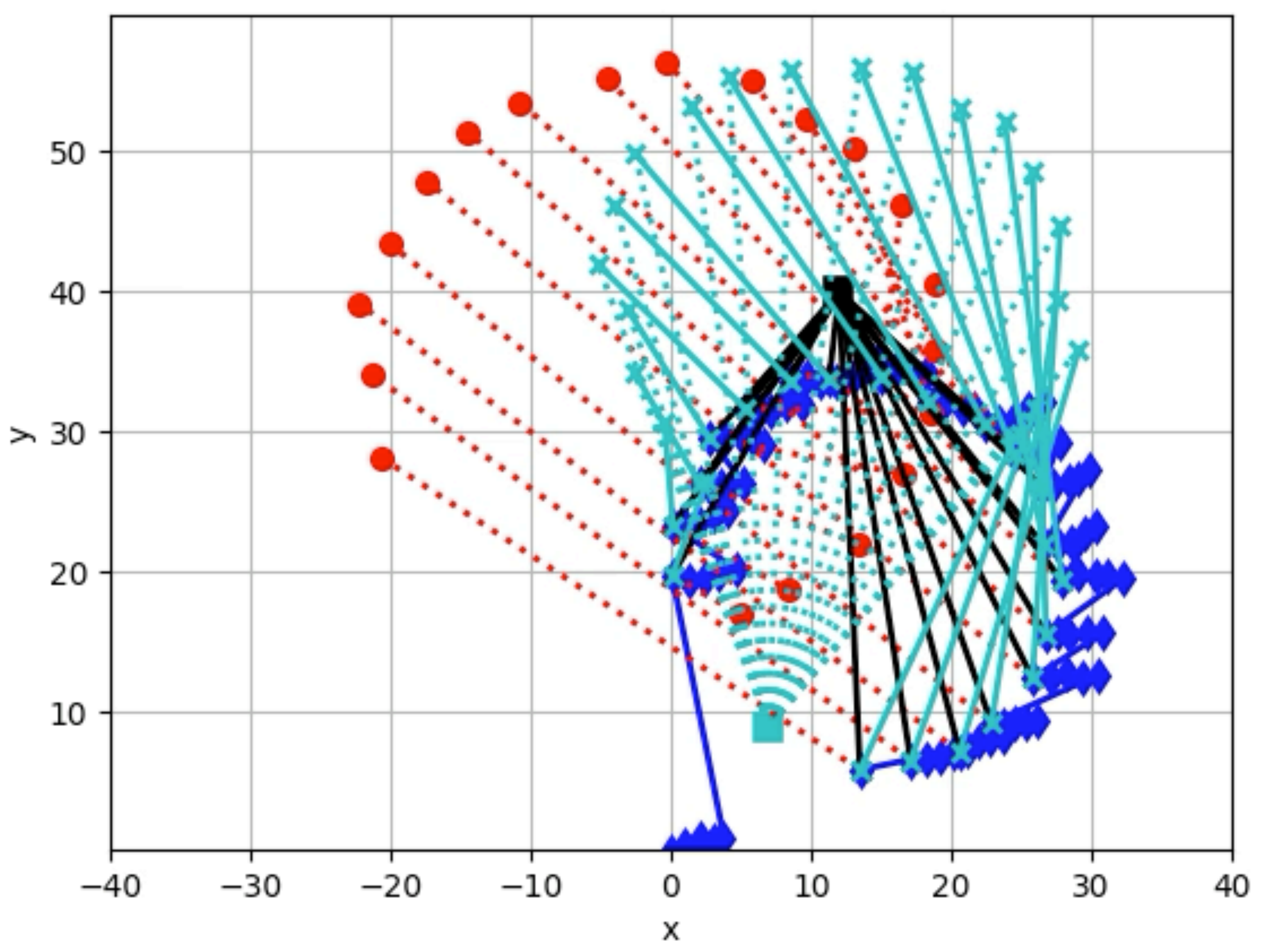


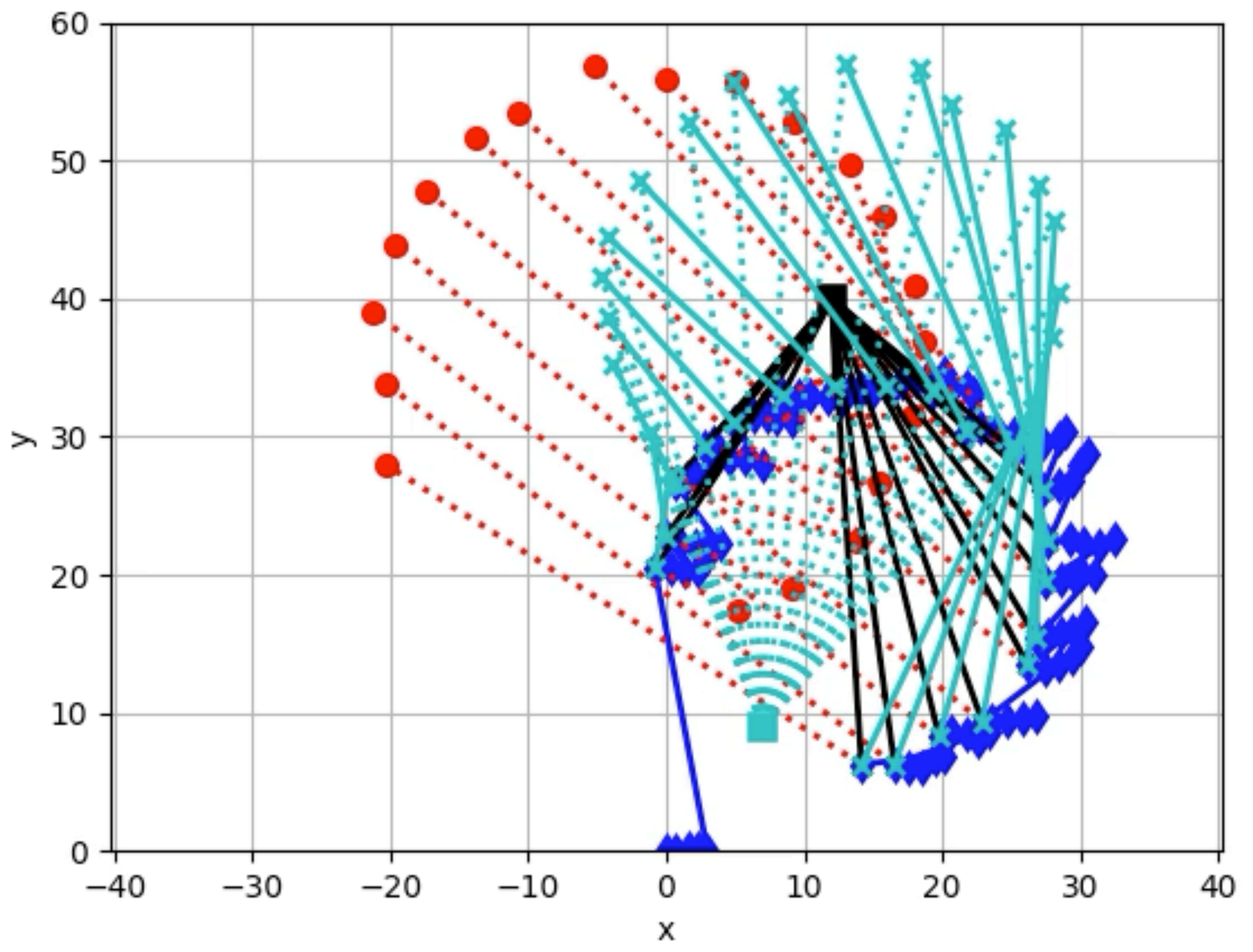
noise 0.7 Optimization in SE(2) manifold trajectory length 101

c_odom=**0**
c_ma=1
c_mr=1

c_odom=1
c_ma=**0**
c_mr=1

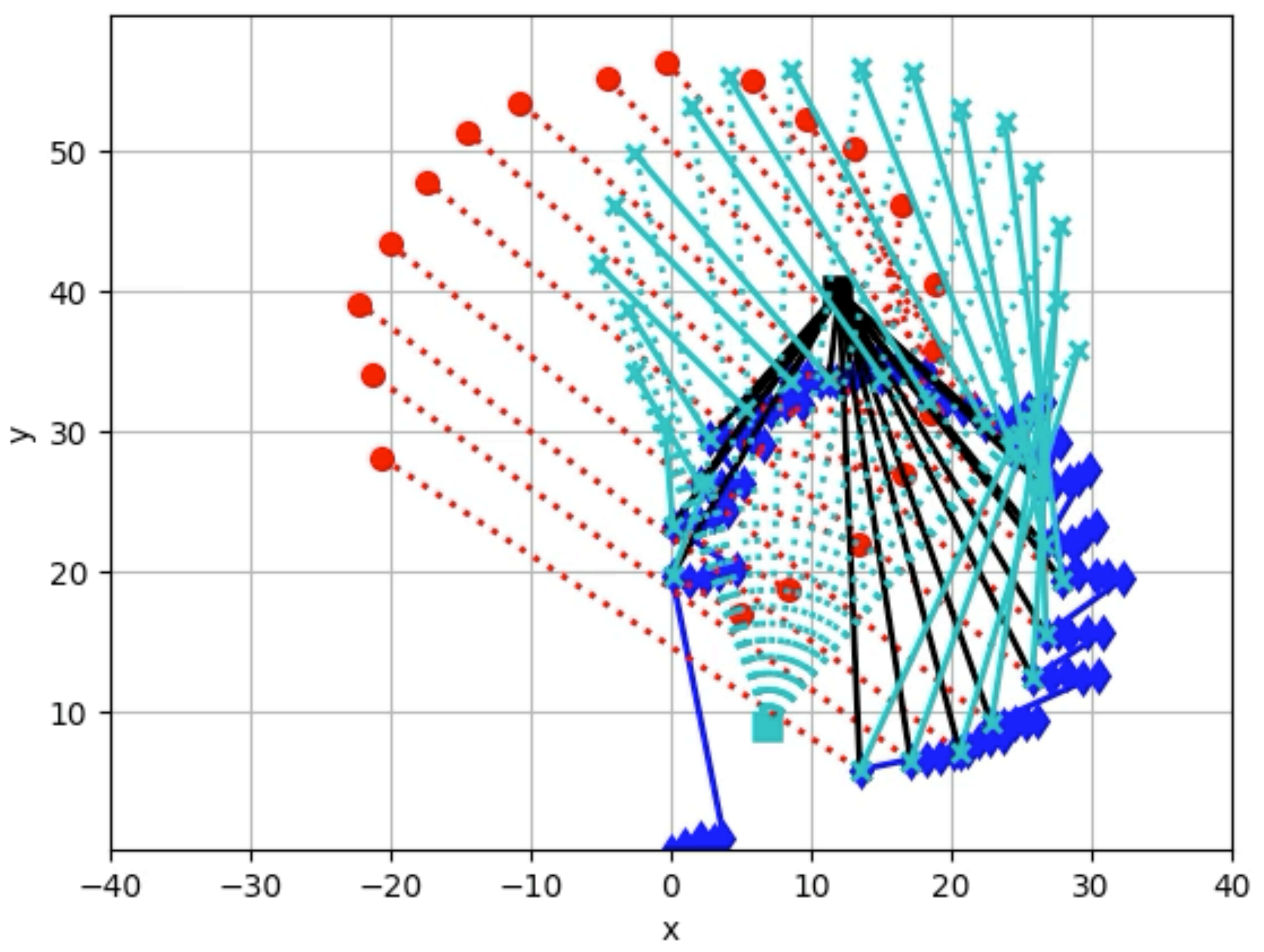
c_odom=1
c_ma=1
c_mr=**0**



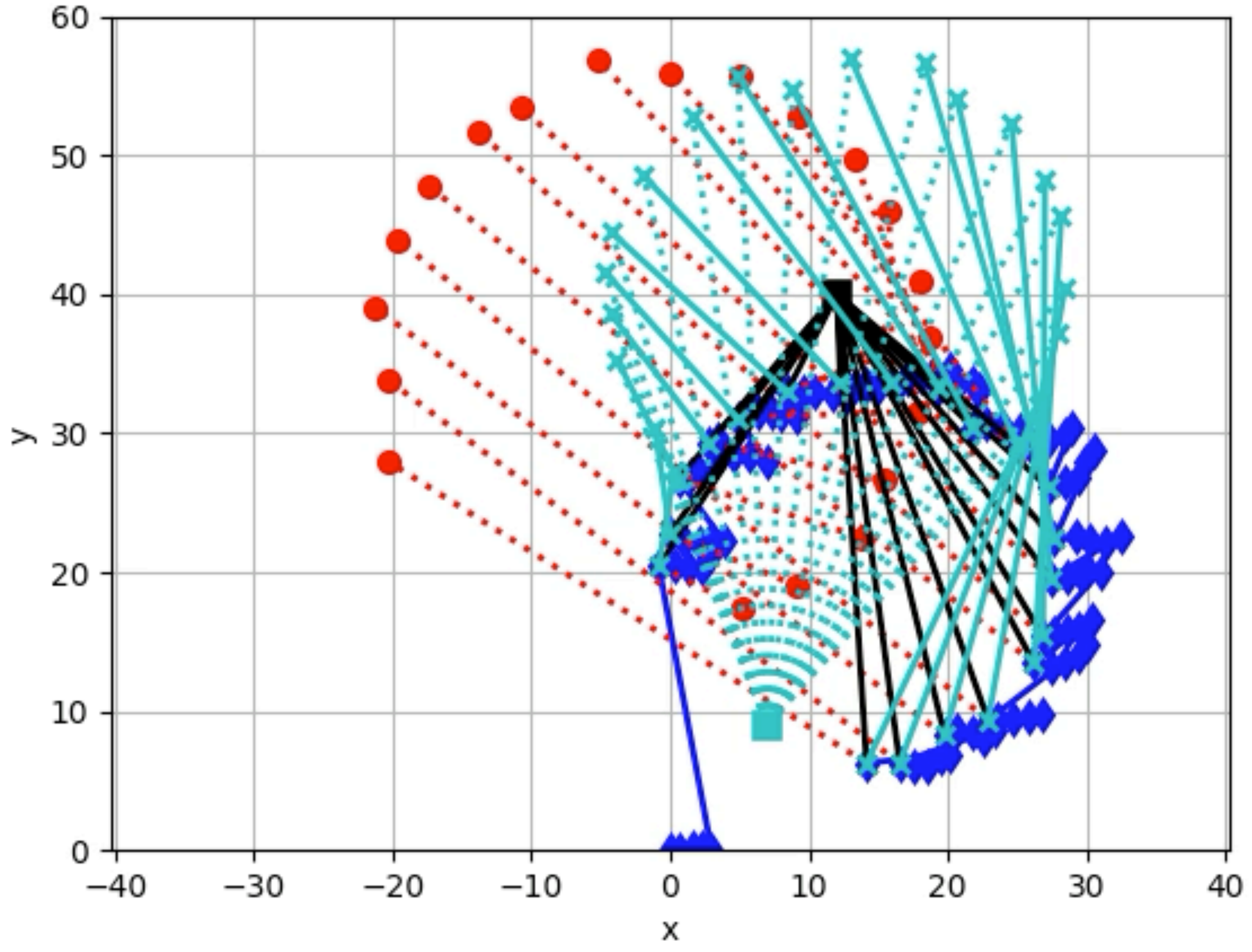


noise 0.7 Optimization in SE(2) manifold trajectory length 101

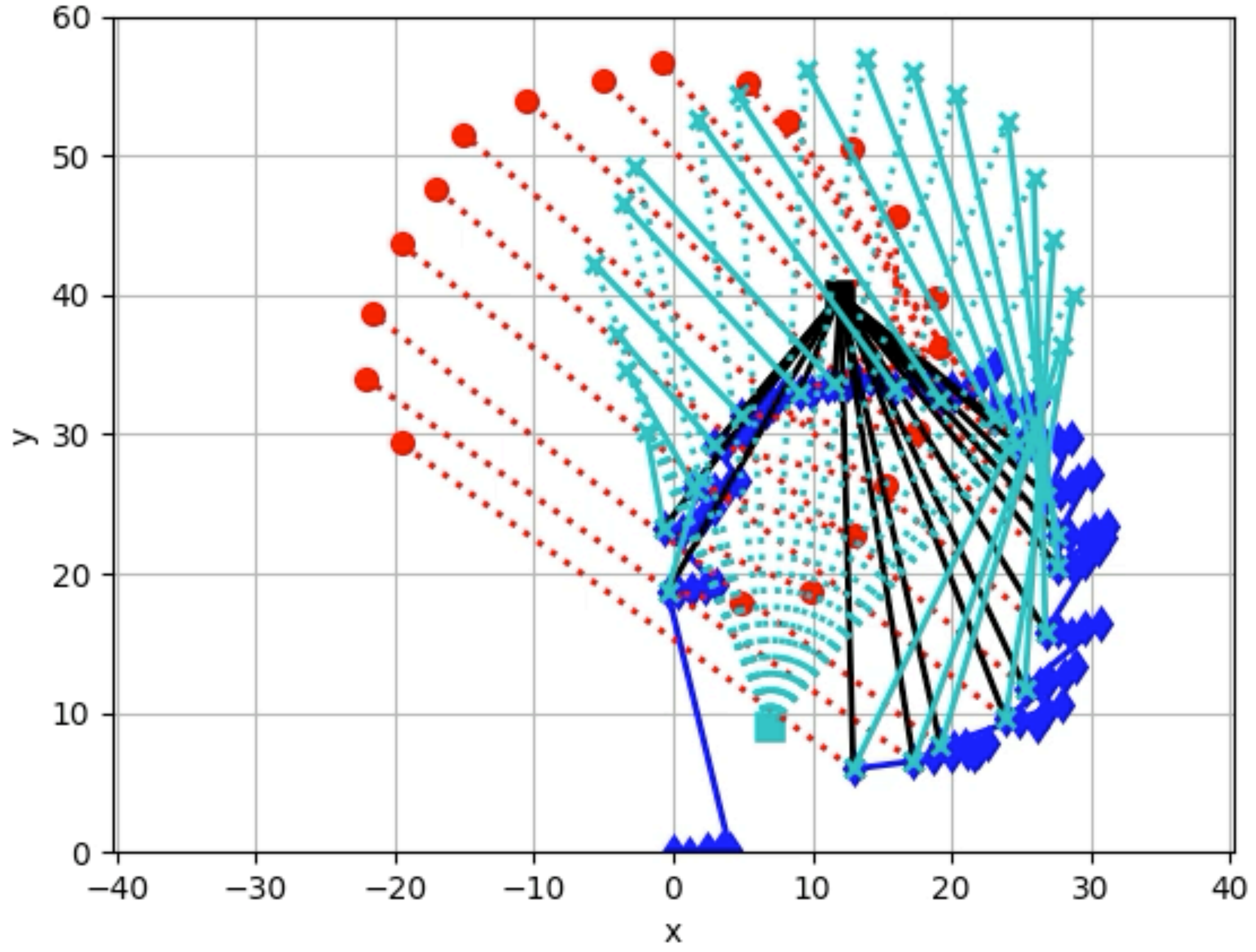
c_odom=**0**
c_ma=1
c_mr=1



c_odom=1
c_ma=**0**
c_mr=1

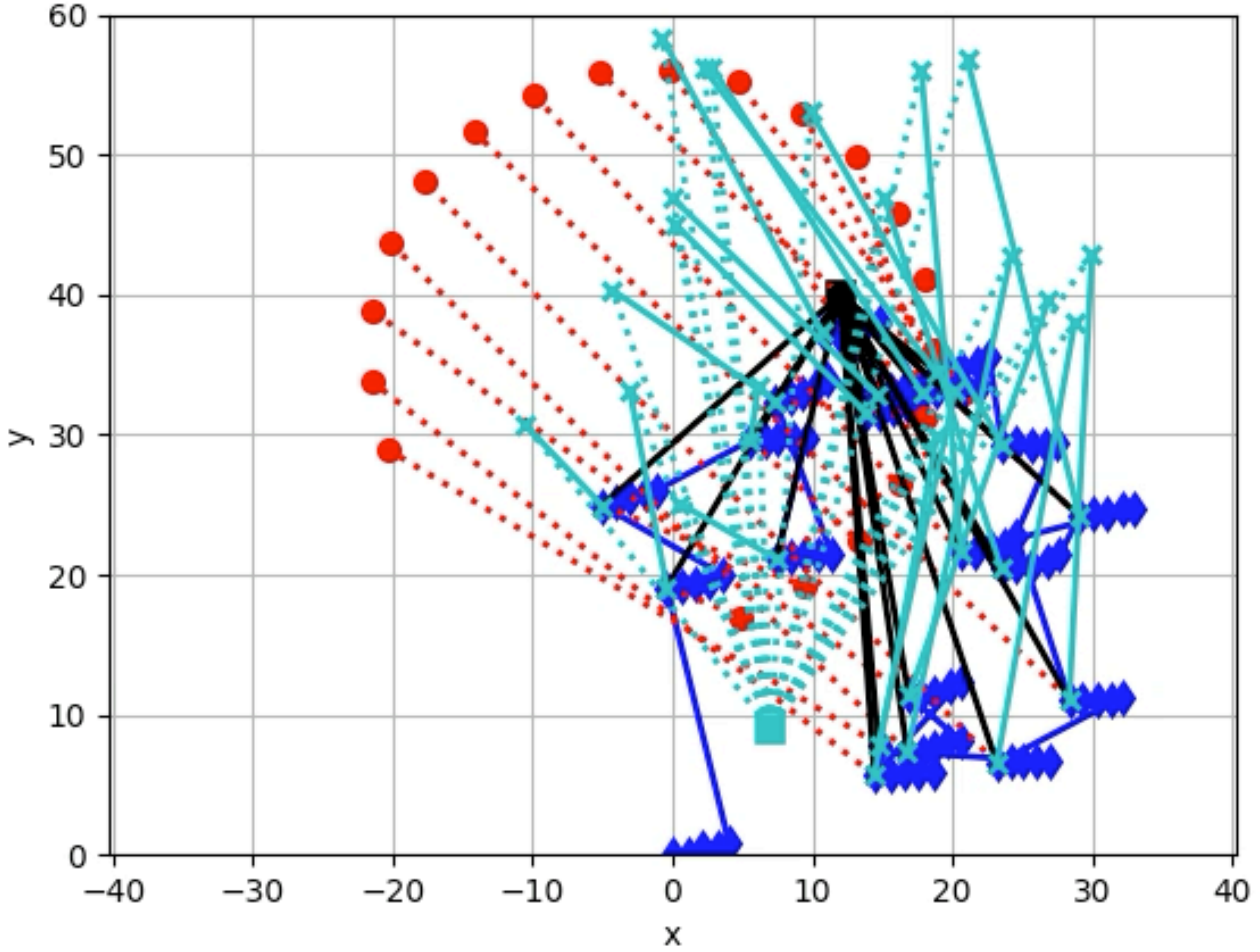


c_odom=1
c_ma=1
c_mr=**0**

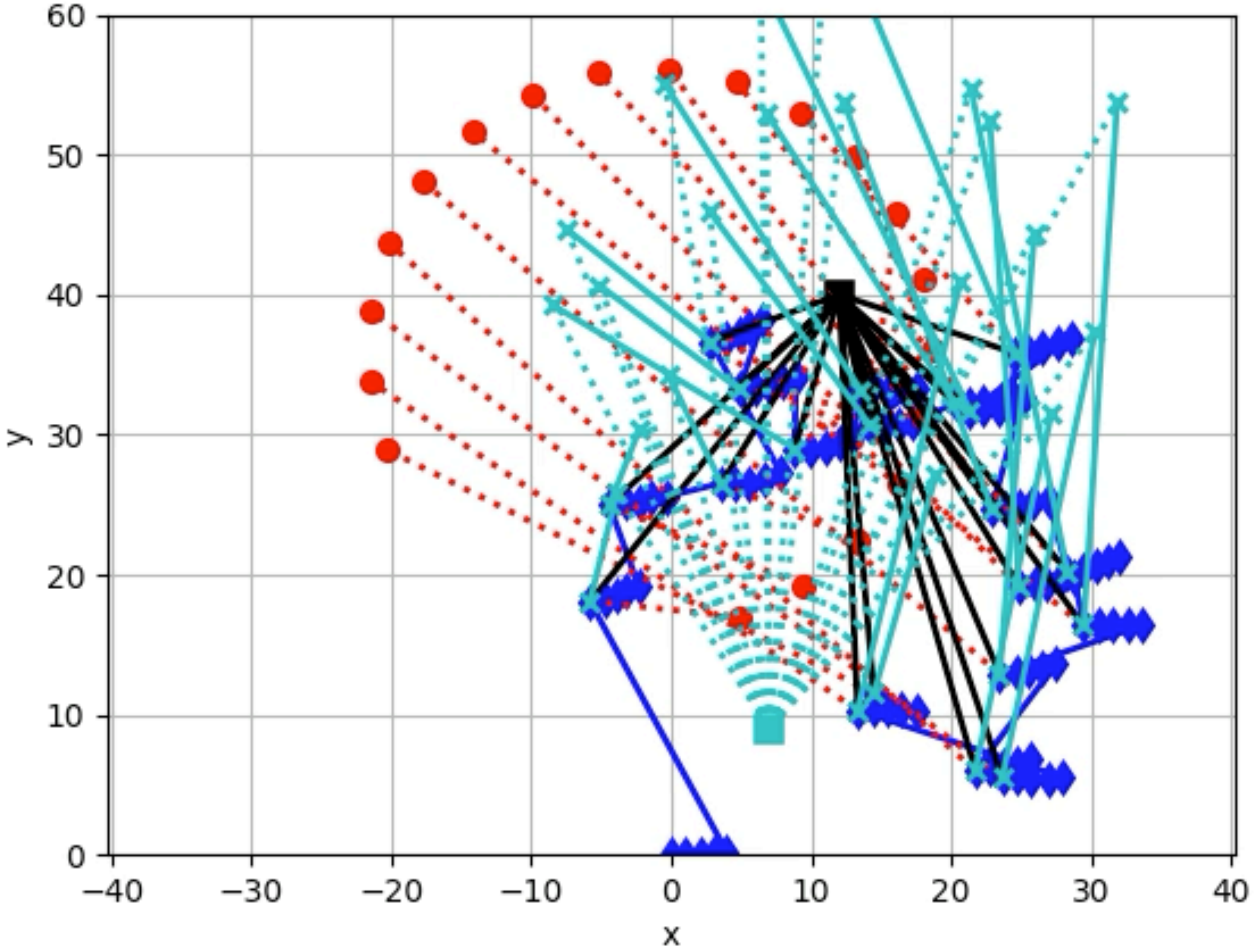


Optimization in SE(2) manifold trajectory length 101

noise_odom=0.1 c_odom=**1**
noise_ma=3 c_ma=1
noise_mr=3 c_mr=1



noise_odom=0.1 c_odom=**5**
noise_ma=3 c_ma=1
noise_mr=3 c_mr=1

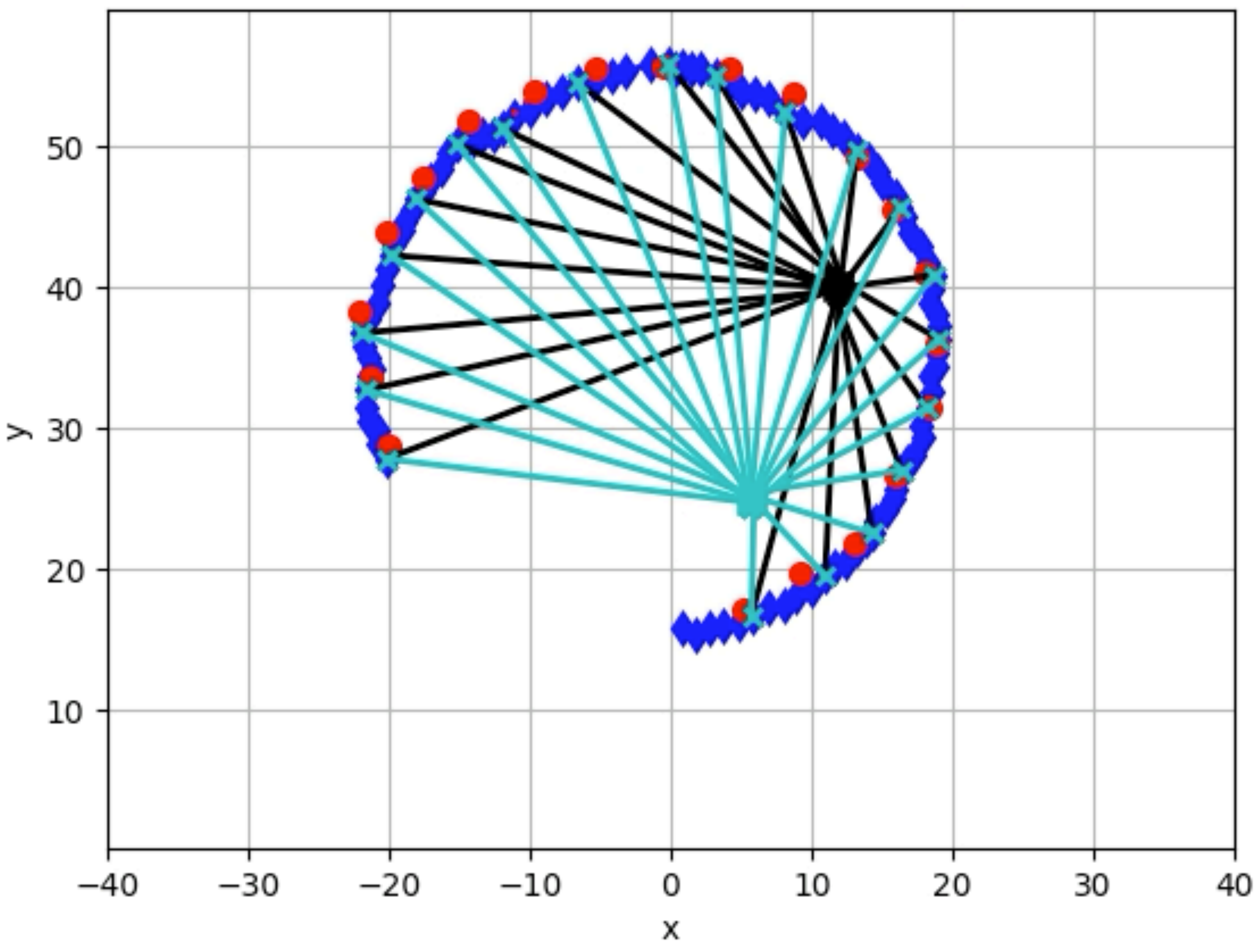


It is good to know, what you can rely on

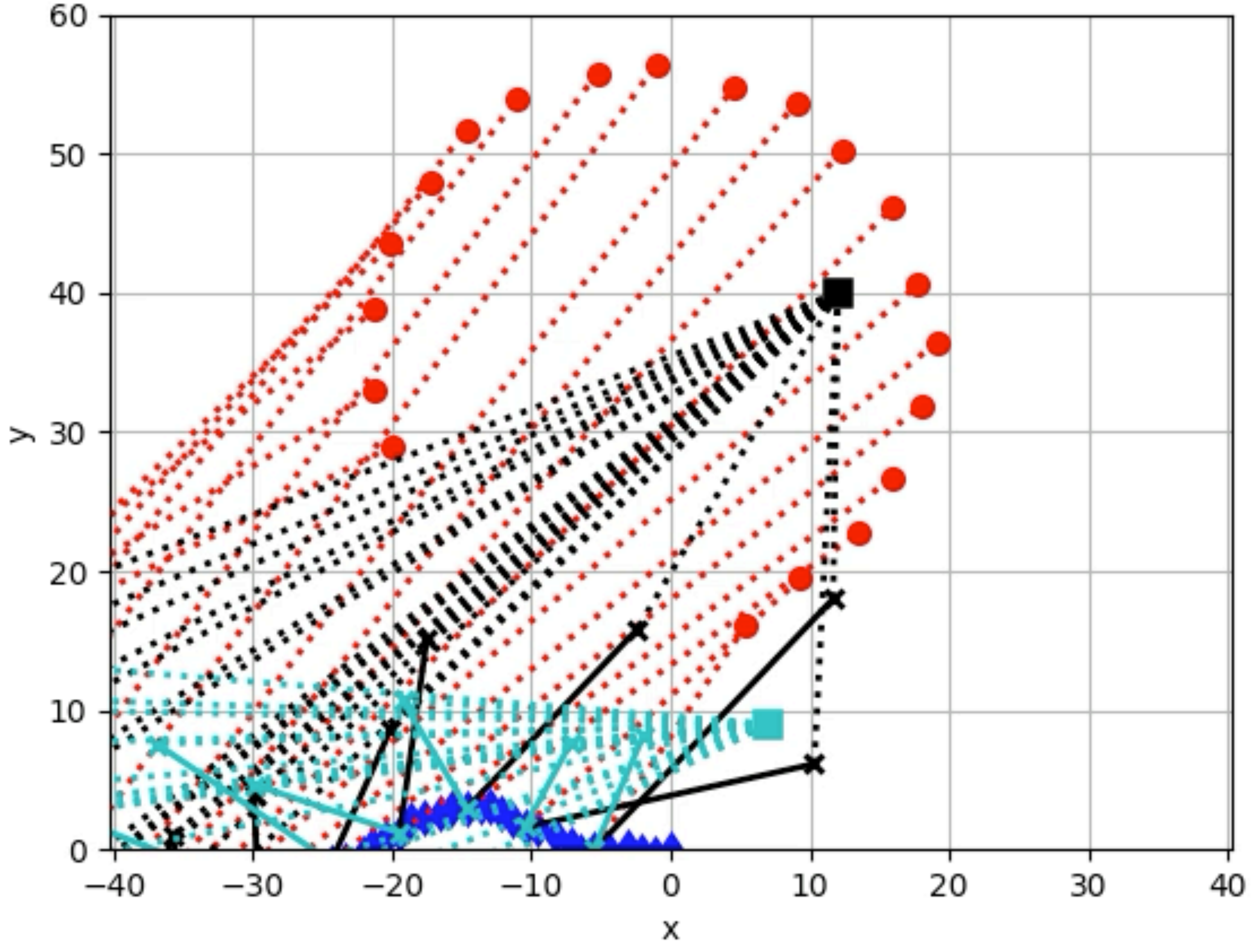
Optimization in SE(2) manifold trajectory length 101

noise 0.5
c_odom=0.2
c_ma=1
c_mr=1

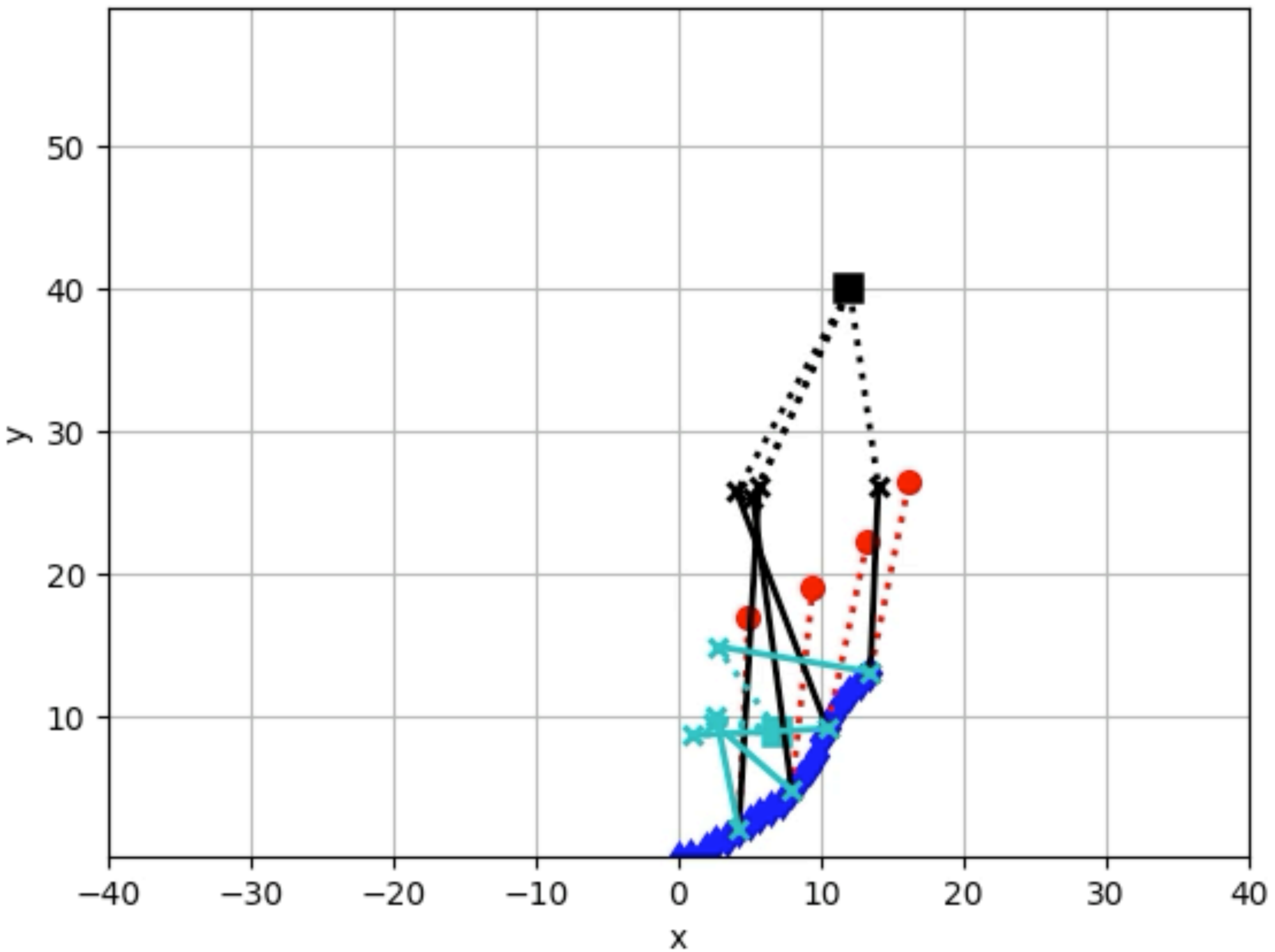
odom/marker ini



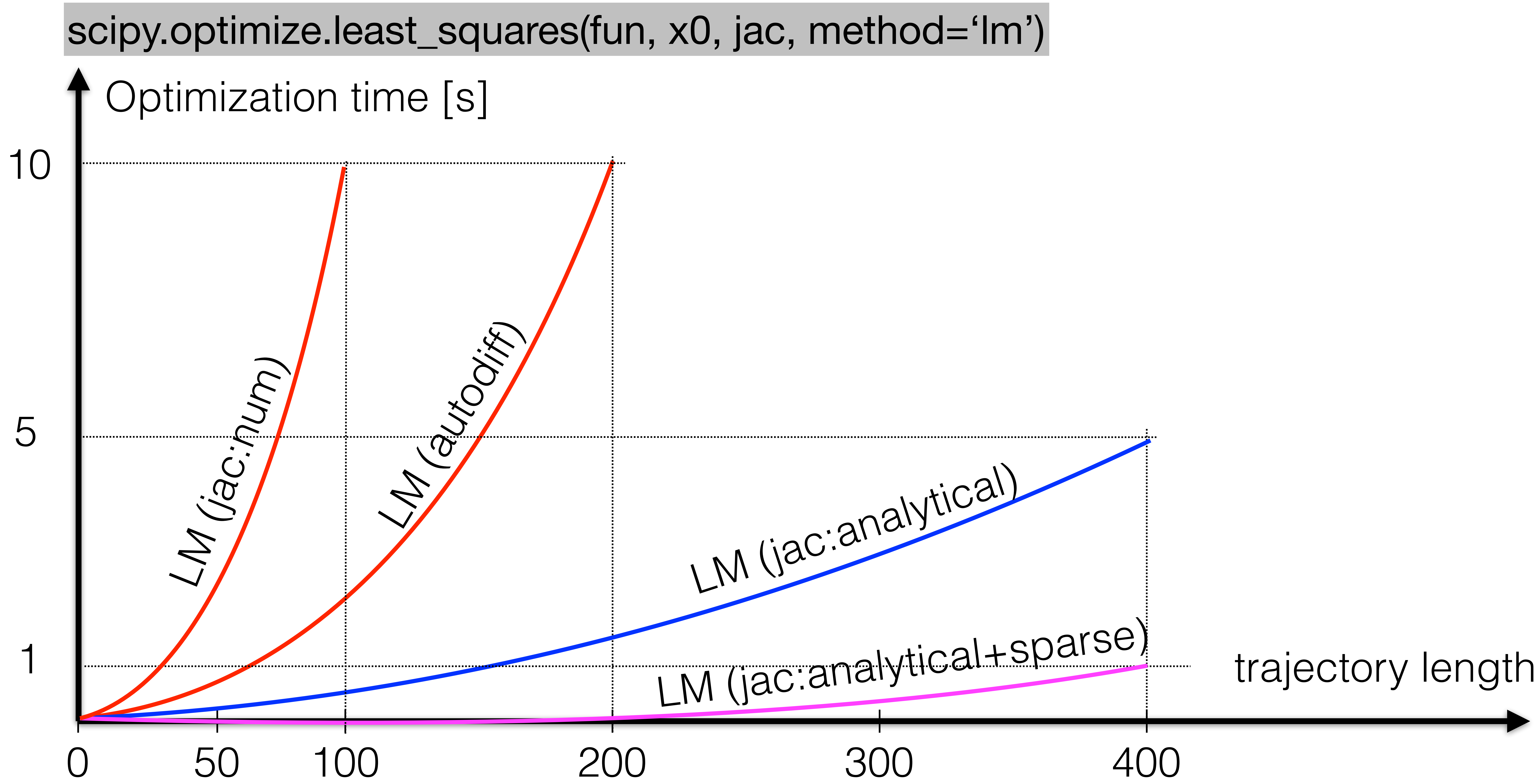
adversarial ini



Optimization in SE(2) manifold trajectory length 401
successive optimization with incoming measurements
noise_markers = 0.5   noise_odom = 0.1



Optimization



Optimization

```
scipy.optimize.least_squares(fun, x0, jac, method='lm')
```

Solution time grows fast with:

- problem dimensionality (e.g. DOFxT+M)
- number of residual terms (e.g. number of measurements)

In practise you introduce simplifications:

- jacobian is extremely sparse => use sparse matrix to represent it
- when new measurement comes only a sub-graph is optimized
- pre-integrate some factor (e.g. sum up odometry measurements over 0.5s)
- sparsification of old factor graph
- to tackle the real-time requirements frontend and backend optimizers used
- limited temporal horizon considered
- if factor graph is tree, efficient solution via dynamic programming (Kalman filter)

Summary

- **Understand** SLAM problem in SE(2)
- **Write down optimisation** criterion in negative log-space for gaussian prob. distr.
- **Solve** underlying opt. problem using non-linear least squares
- **Issues:**
    - covariance delivered by sensors is really bad
    - measurements are strongly correlated
    - gradient optimization converges to a local minimum
    - noise often non-gaussian => if modeled optimization issues
    - factor graph keep growing to infinity vs realtime requirements
- **Next lecture:** Adds lidar’s measurement probability