Kalman Filter Examples

Static State Estimation

- recall the static state estimation problem we have been studying
  - the process or plant model

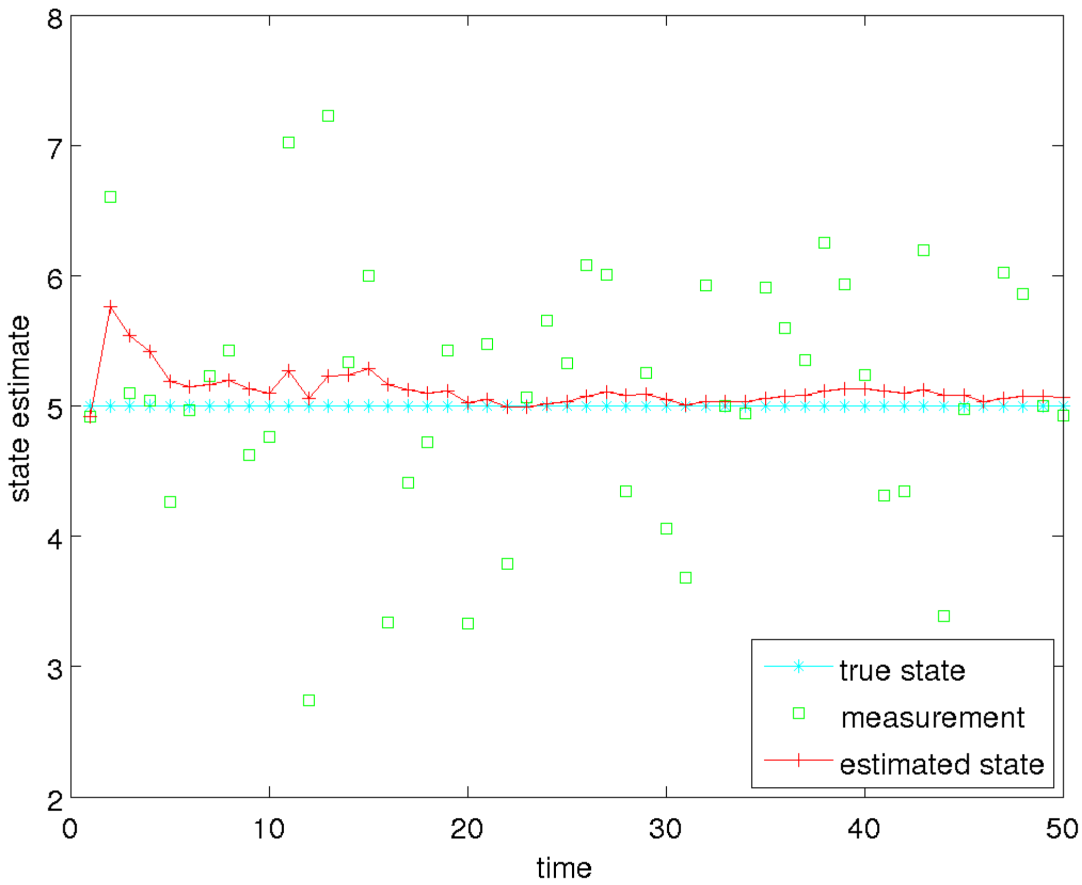
$$A_t = 1, \quad B_t = 0, \quad R_t = 0 \qquad \begin{aligned} x_t &= A_t x_{t-1} + B_t u_t + \varepsilon_t \\ &= x_{t-1} \end{aligned}$$

  - the observation model

$$C_t = 1, \quad Q_t = \sigma_t^2 \qquad z_t = x_t + \delta_t$$

Static State Estimation

- how well does the Kalman filter work

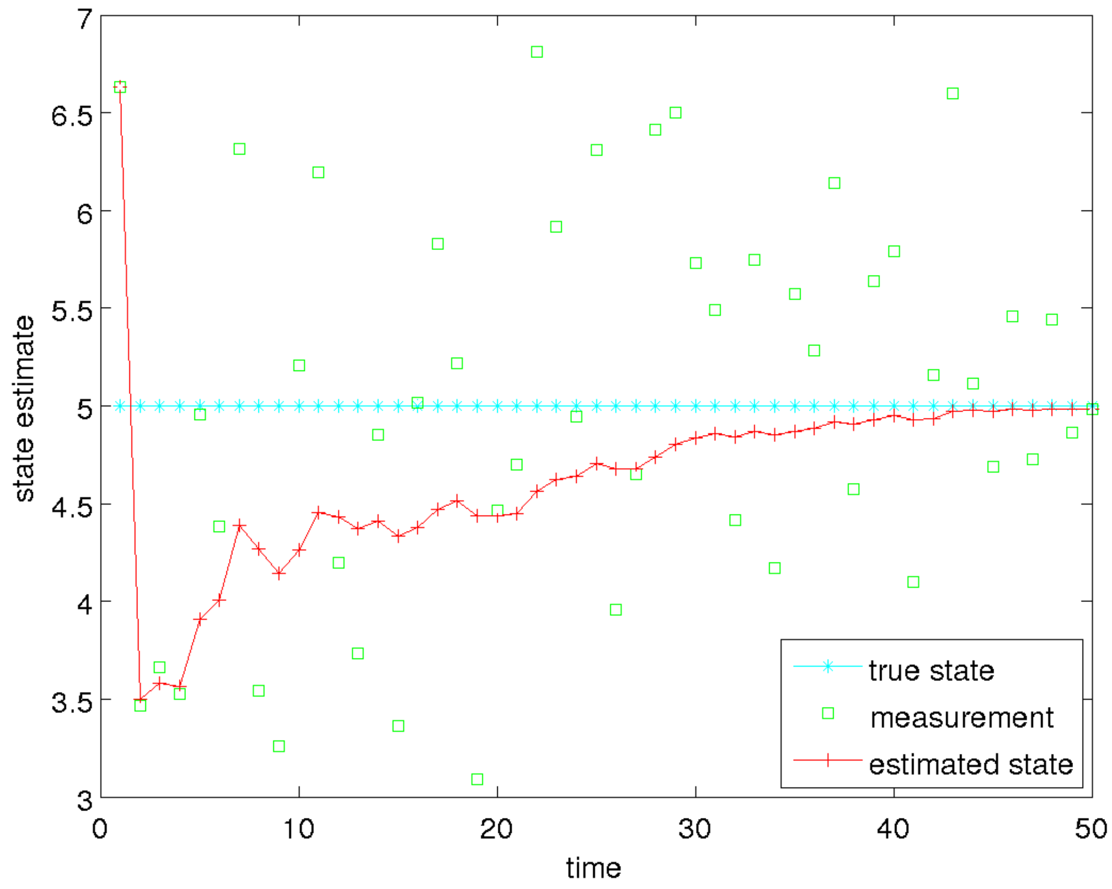


Static State Estimation

- notice that we need to specify the measurement noise covariance Q_t
- how sensitive is the Kalman filter to Q_t ?
  - e.g., what if we use a Q_t that is much smaller than the actual measurement noise?
  - e.g., what if we use a Q_t that is much larger than the actual measurement noise?

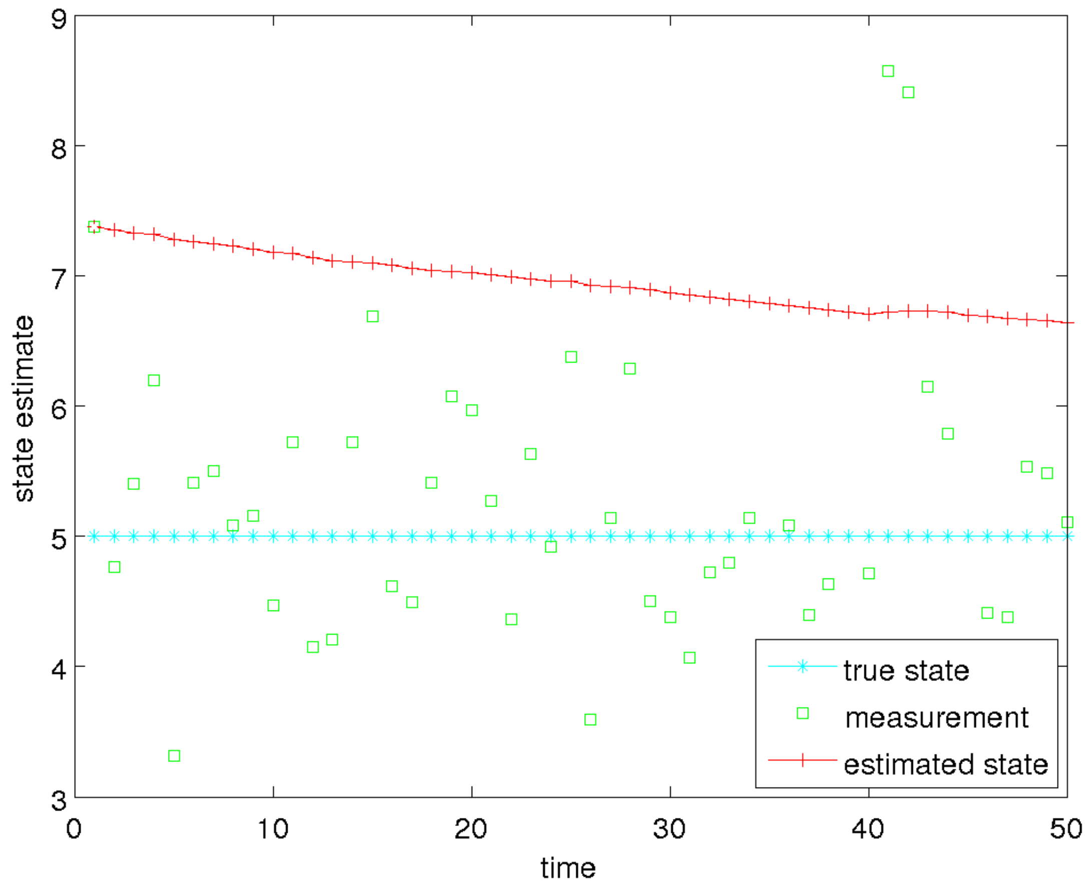
Static State Estimation

- specified Q_t = 0.01 * actual Q_t



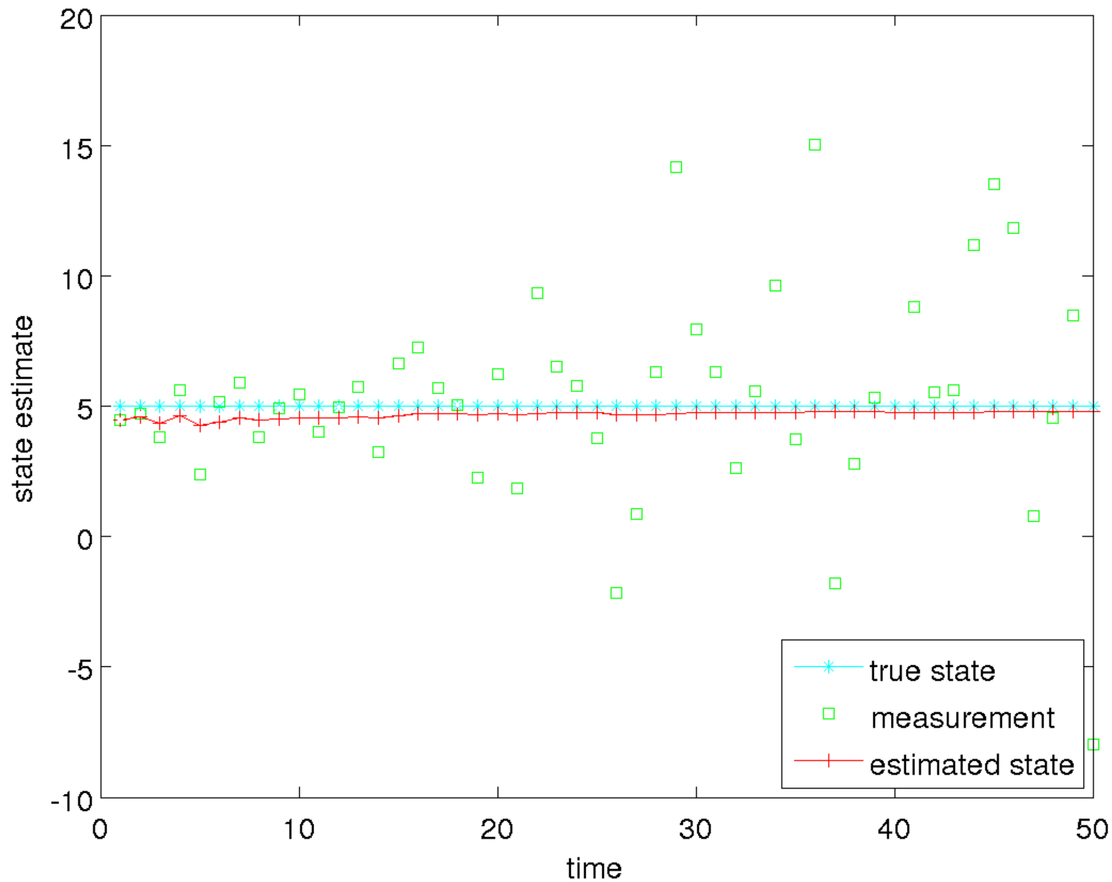
Static State Estimation

- specified Q_t = 100 * actual Q_t



Static State Estimation

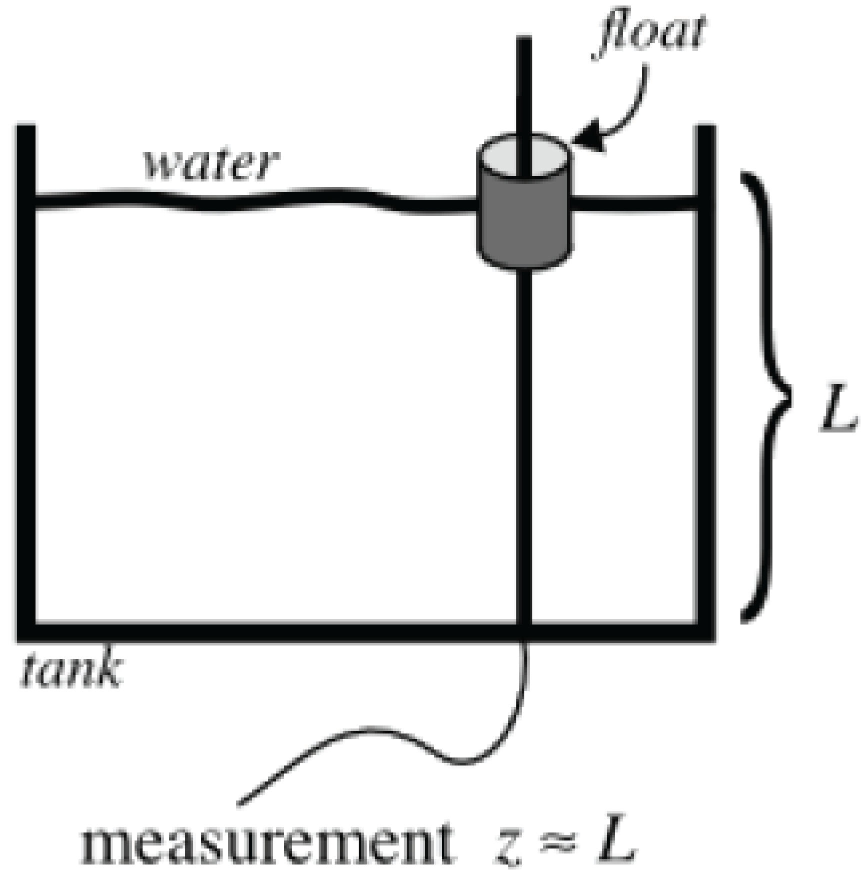
- suppose our measurements get progressively noisier over time



noise variance increases 10% for each successive measurement

Tank of Water

- estimate the level of water in the tank; the water could be
  - static, filling, or emptying
  - not sloshing or sloshing



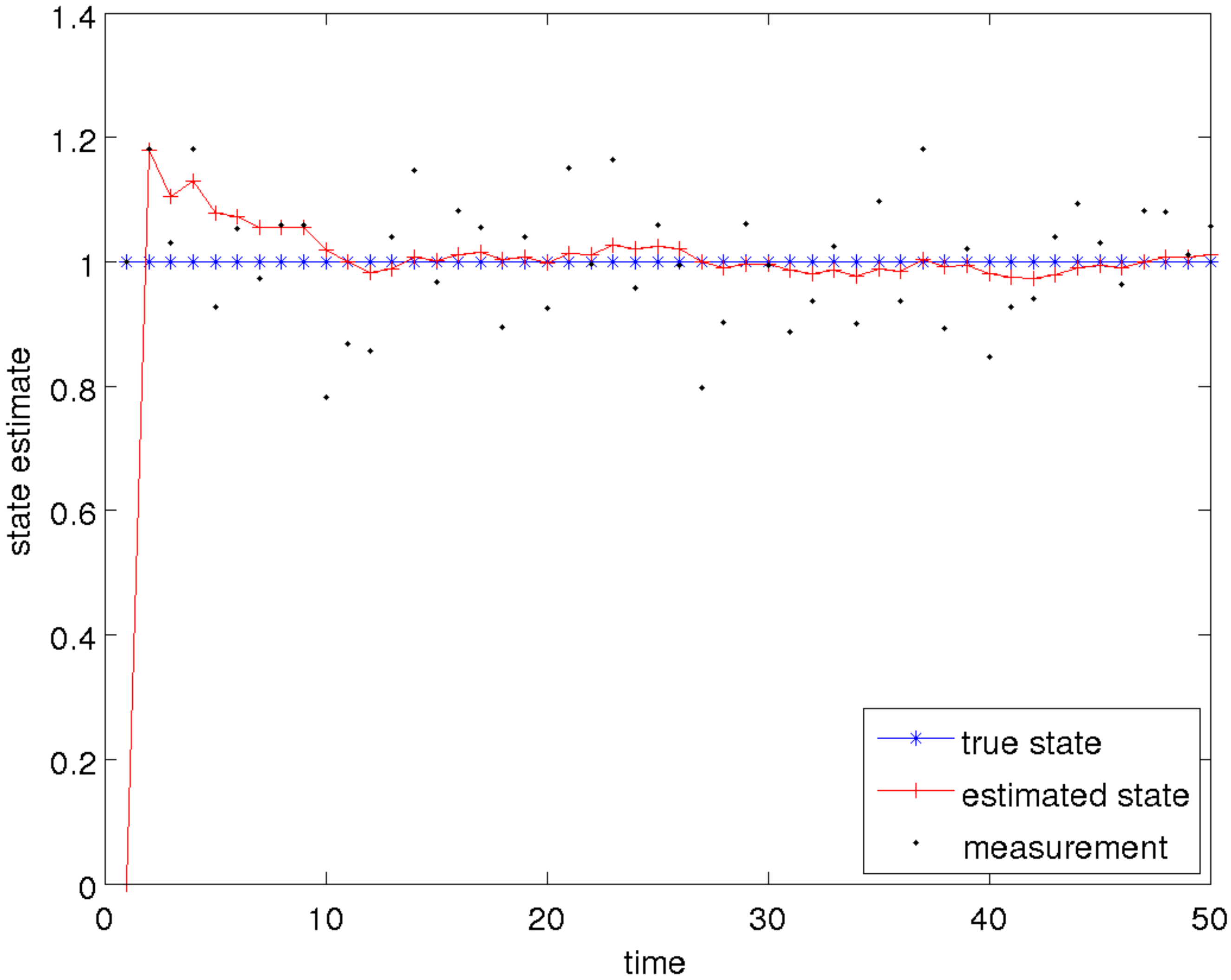
Tank of Water

- static level

plant model $$x_t = x_{t-1}$$

measurement model $$z_t = x_t + \delta_t$$

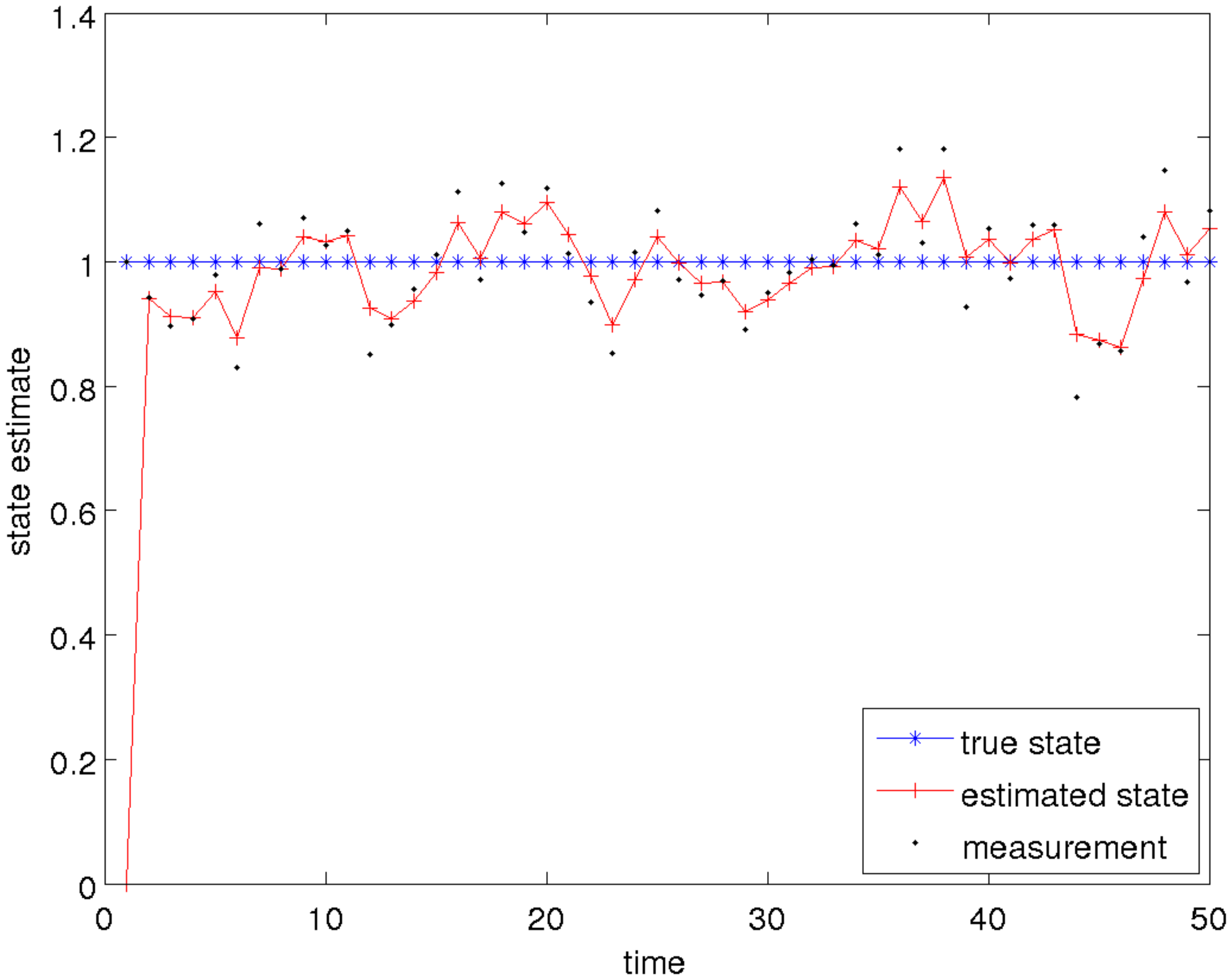
Tank of Water: Static and Not Sloshing



Tank of Water: Static and Not Sloshing

- notice that in this case the Kalman filter tends towards estimating a constant level because the plant noise covariance is small compared to the measurement noise covariance
  - the estimated state is much smoother than the measurements
- what happens if we increase the plant noise covariance?

Tank of Water: Filling and Not Sloshing



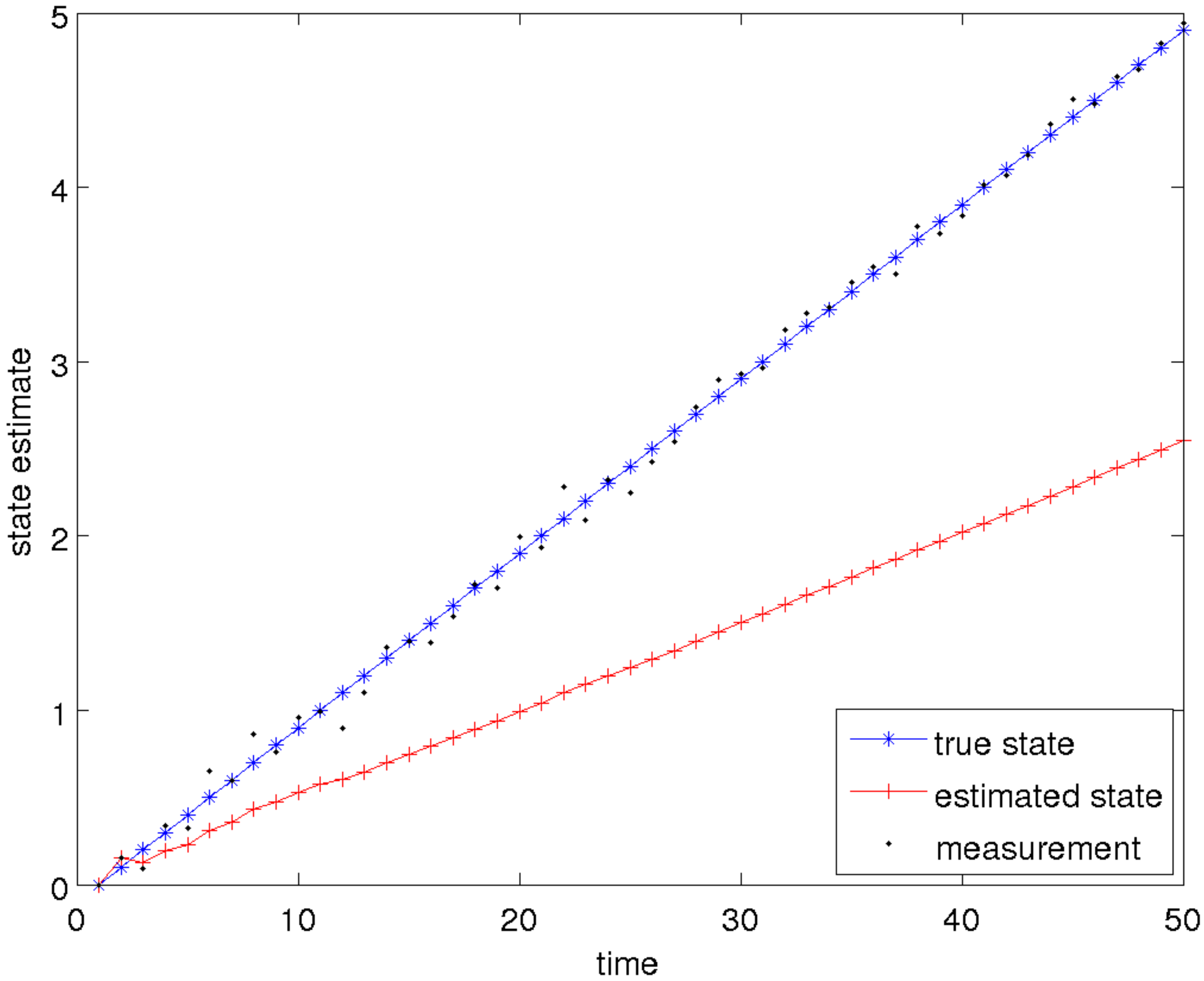
Tank of Water: Static and Not Sloshing

- notice that in this case the Kalman filter tends towards estimating values that are closer to the measurements
- increasing the plant noise covariance causes the filter to place more weight on the measurements

Tank of Water: Filling and not Sloshing

- suppose the true situation is that the tank is filling at a constant rate but we use the static tank plant model
  - i.e., we have a plant model that does not accurately model the state transition

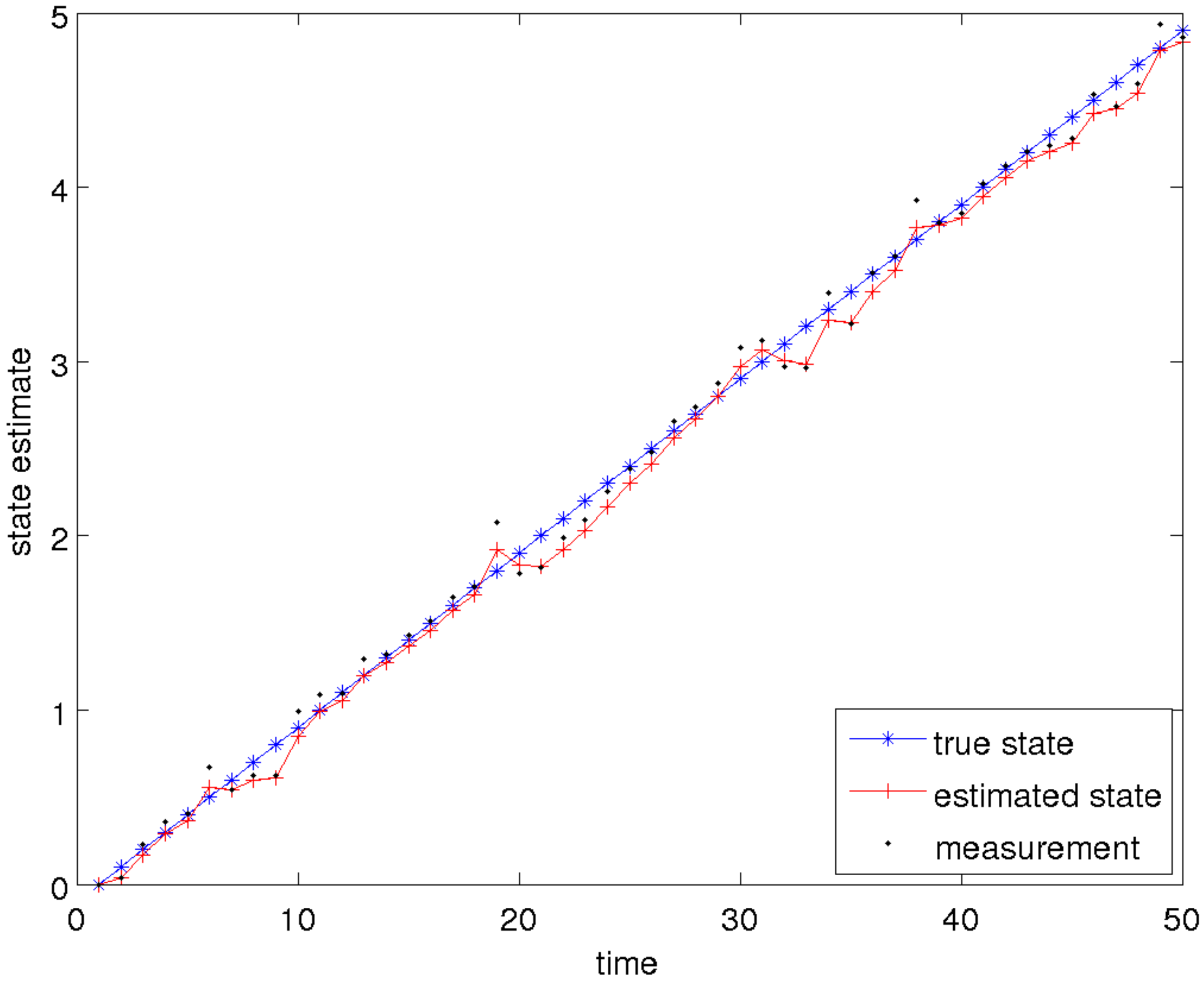
Tank of Water: Filling and not Sloshing



Tank of Water: Filling and not Sloshing

- notice that in this case the estimated state trails behind the true level
  - estimated state has a much greater error than the noisy measurements
- if the plant model does not accurately model reality than you can expect poor results
  - however, increasing the plant noise covariance will allow the filter to weight the measurements more heavily in the estimation…

Tank of Water: Filling and not Sloshing



Tank of Water: Filling and not Sloshing

- it is not clear if we have gained anything in this case
  - the estimated state is reasonable but it is not clear if it is more accurate than the measurements
- what happens if we change the plant model to more accurately reflect the filling?

Tank of Water

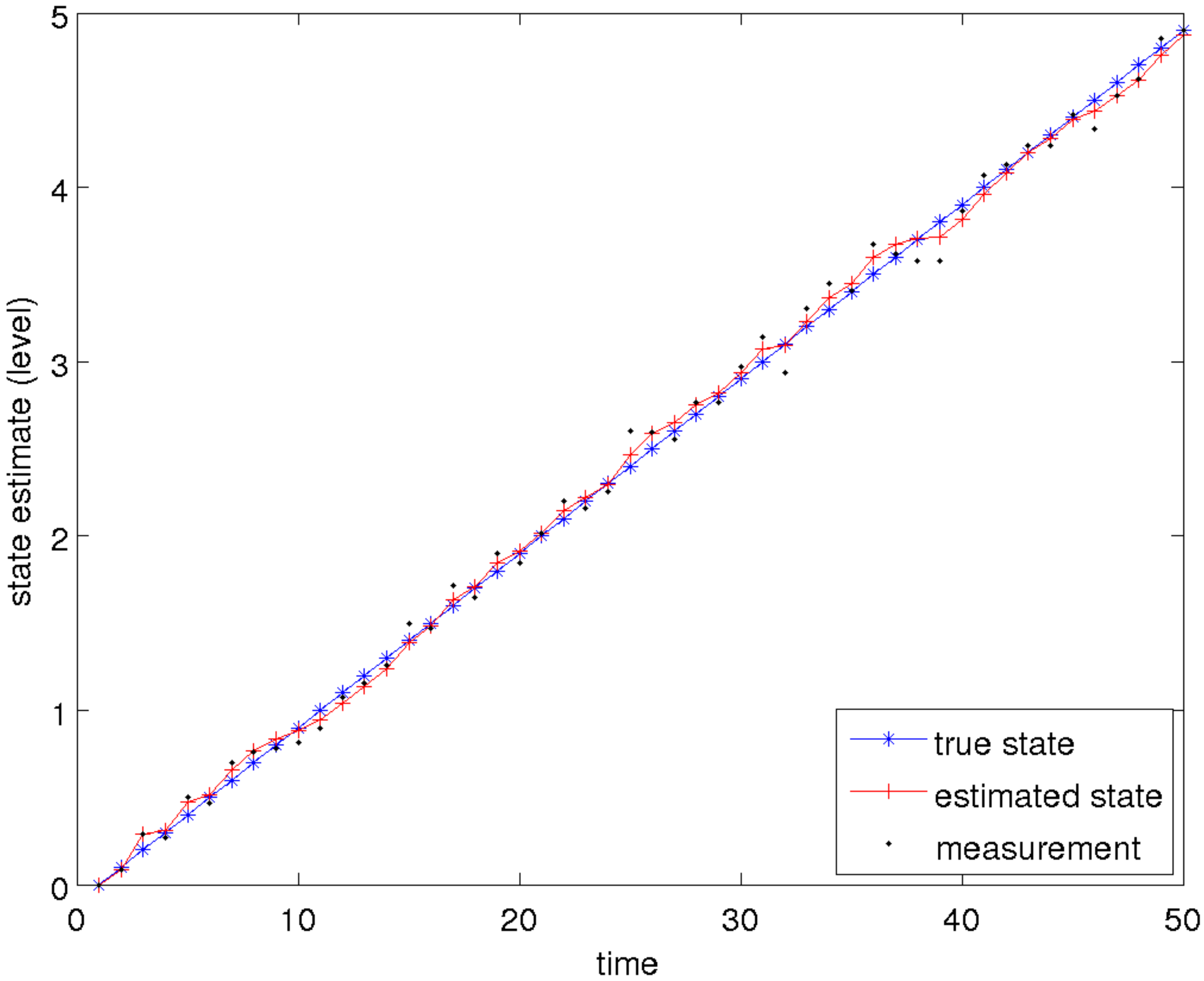
- filling at a (noisy) constant rate and we do not care about the rate

plant model $$x_t = x_{L,t-1} + \underbrace{\Delta x_L}_{u_t} + \varepsilon_t$$

measurement model $$z_t = x_t + \delta_t$$

- u_t is the change in the water level that occurred from time t-1 to t

Tank of Water: Filling and not Sloshing



Tank of Water: Filling and not Sloshing

- notice that the estimated state is more accurate and smoother than the measurements
- what about the filling rate?

Tank of Water

- filling at a (noisy) constant rate and we want to estimate the rate

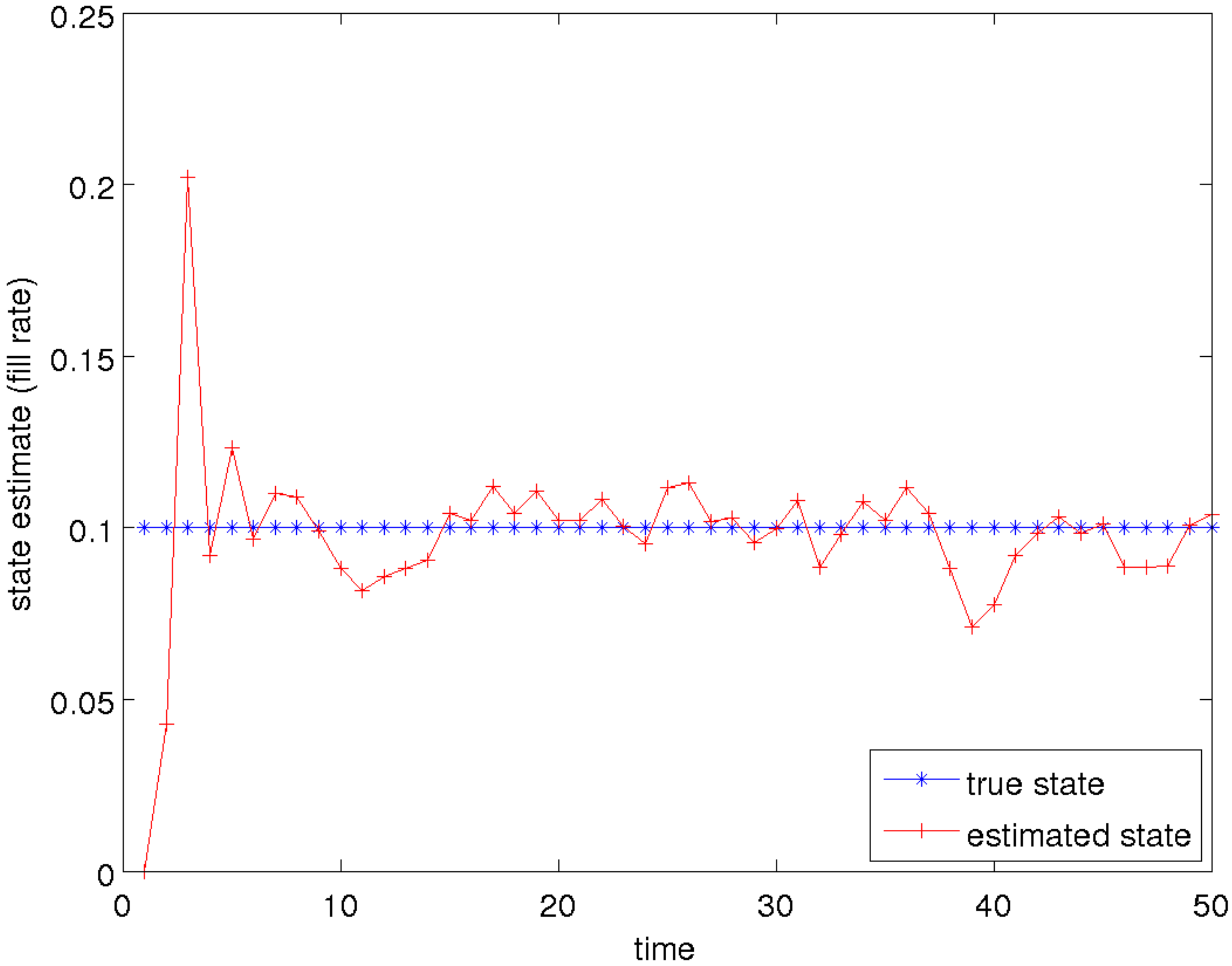
plant model

$$x_t = \underbrace{\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}}_{A_t} \underbrace{\begin{bmatrix} x_L \\ \Delta x_L \end{bmatrix}_{t-1}}_{x_{t-1}} + \varepsilon_t$$

measurement model

$$z_t = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C_t} x_t + \delta_t$$

Tank of Water: Filling and not Sloshing



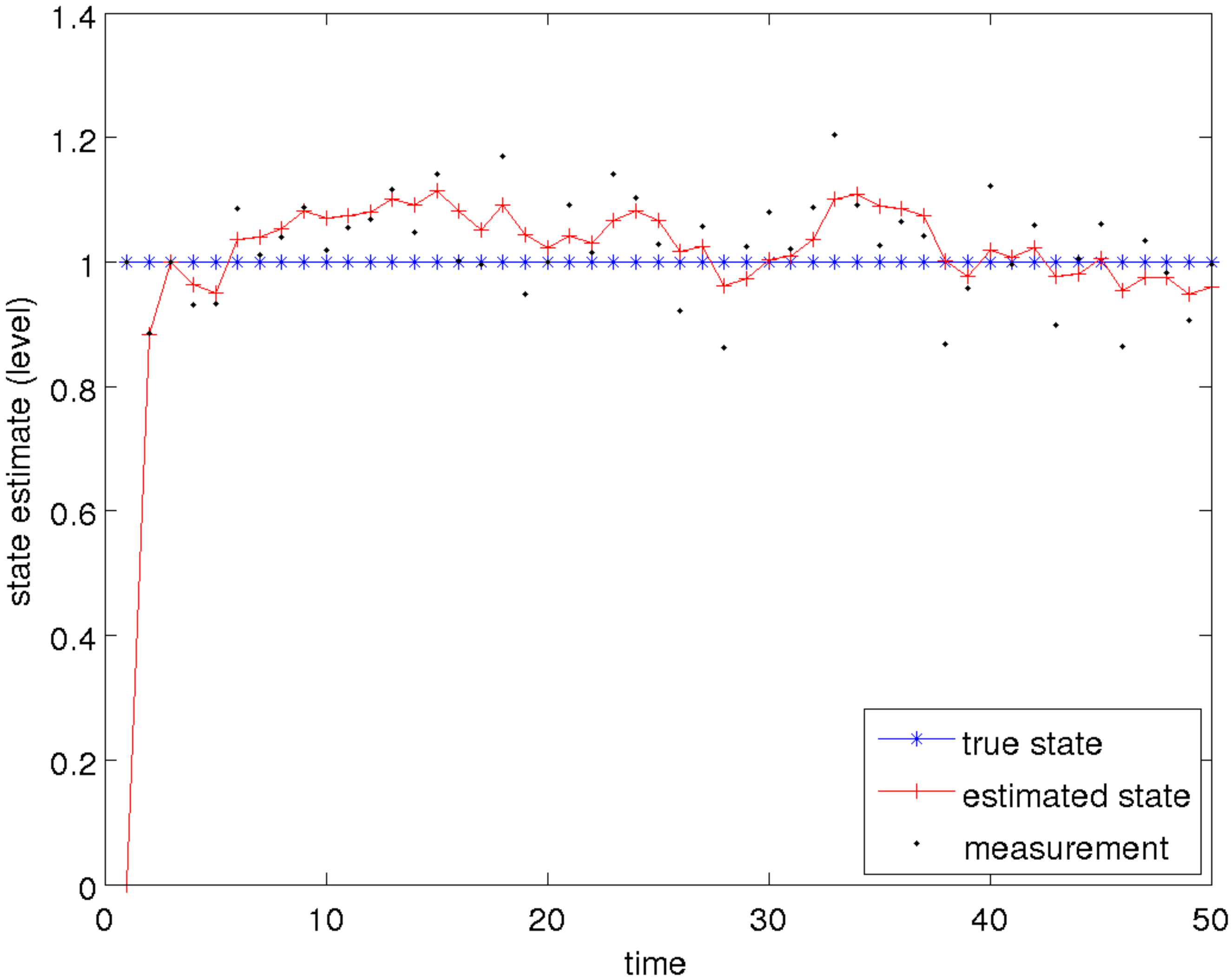
Tank of Water: Filling and not Sloshing

- notice that the estimated filling rate seems to jump more than the estimated level
  - this should not be surprising as we never actually measure the filling rate directly
    - it is being inferred from the measured level (which is quite noisy)

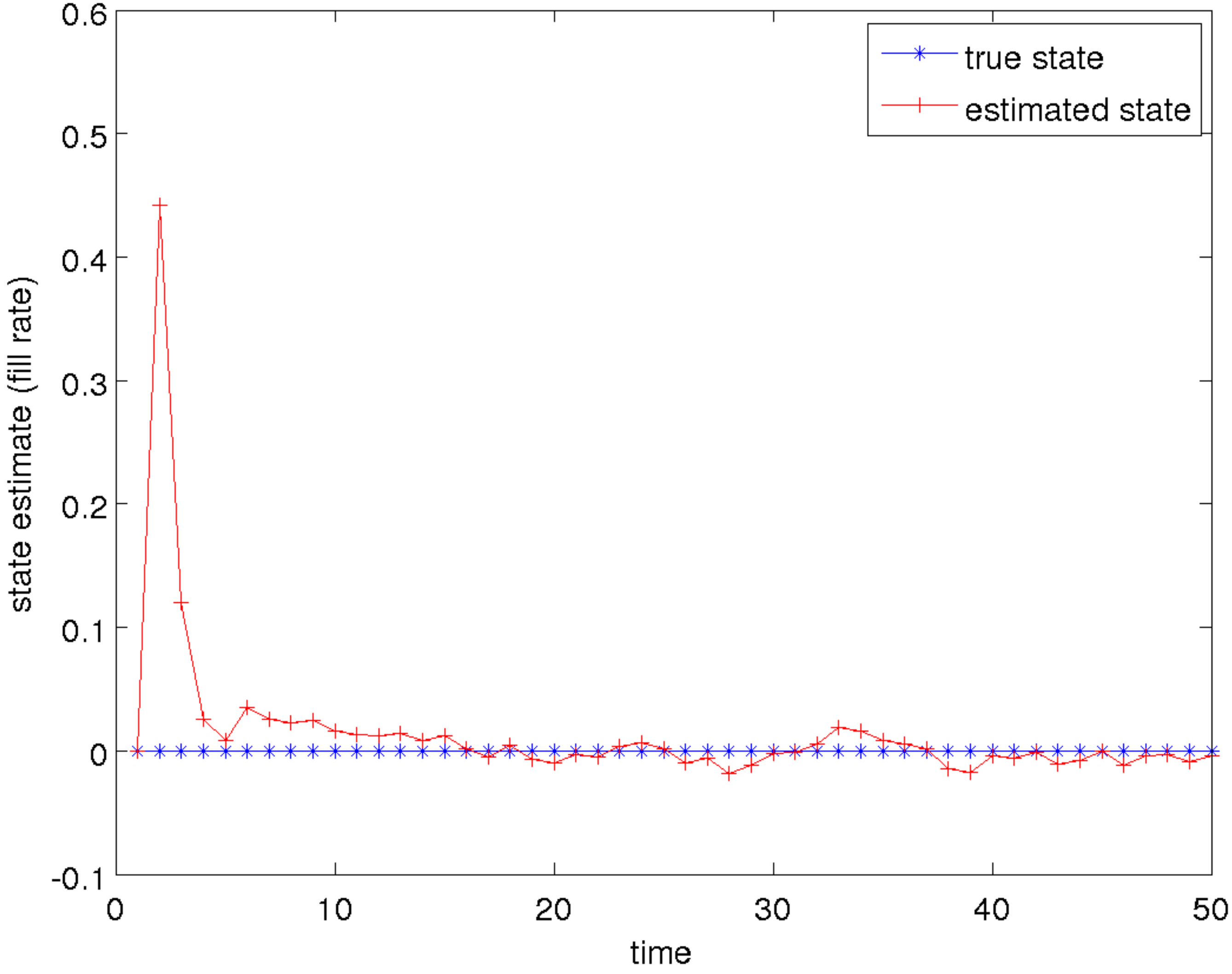
Tank of Water: Static and not Sloshing

- can we trick the filter by using the filling plant model when the level is static?
  - hopefully not, as the filter should converge to a fill rate of zero!

Tank of Water: Static and not Sloshing

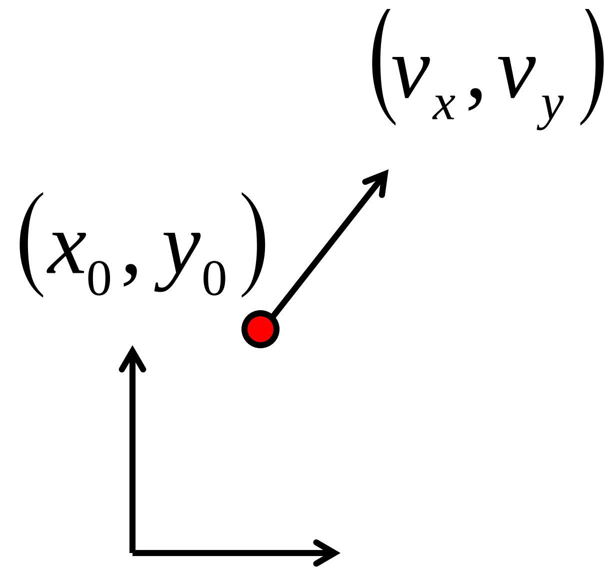


Tank of Water: Static and not Sloshing



Projectile Motion

- projectile launched from some initial point with some initial velocity under the influence of gravity (no drag)



$$x(t) = x_0 + v_x t$$

$$y(t) = y_0 + v_y t - \tfrac{1}{2} g t^2$$

$$v_x(t) = v_x$$

$$v_y(t) = v_y - gt$$

Projectile Motion

- convert the continuous time equations to discrete recurrence relations for some time step Δt

$$x_t = x_{t-1} + v_{x,t-1}\Delta t$$

$$y_t = y_{t-1} + v_{y,t-1}\Delta t - \tfrac{1}{2} g\Delta t^2$$

$$v_{x,t} = v_{x,t-1}$$

$$v_{y,t} = v_{y,t-1} - g\Delta t$$

Projectile Motion

- rewrite in matrix form

$$\underbrace{\begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_t}_{x_t} = \underbrace{\begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{A_t} \underbrace{\begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_{t-1}}_{x_{t-1}} + \underbrace{\begin{bmatrix} 0 \\ -\frac{1}{2} g \Delta t^2 \\ 0 \\ -gt \end{bmatrix}}_{u_t}$$

Omnidirectional Robot

- an omnidirectional robot is a robot that can move in any direction (constrained in the ground plane)
  - http://www.youtube.com/watch?v=DPz-ullMOqc
  - http://www.engadget.com/2011/07/09/curtis-boirums-robotic-car-makes-omnidirectional-dreams-come-tr/

- if we are not interested in the orientation of the robot then its state is simply its location

$$x_t = \begin{bmatrix} x \\ y \end{bmatrix}_t$$

Omnidirectional Robot

- a possible choice of motion control is simply a change in the location of the robot

$$x_t = \underbrace{\begin{bmatrix} x \\ y \end{bmatrix}_{t-1}}_{x_{t-1}} + \underbrace{\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}_{t}}_{u_t}$$

- with noisy control inputs

$$x_t = \underbrace{\begin{bmatrix} x \\ y \end{bmatrix}_{t-1}}_{x_{t-1}} + \underbrace{\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}_{t}}_{u_t} + \varepsilon_t$$

Differential Drive

- recall that we developed two motion models for a differential drive
  - using the velocity model, the control inputs are

$$u_t = \begin{pmatrix} v_t \\ \omega_t \end{pmatrix} + \begin{pmatrix} \varepsilon_{\alpha_1 v_t^2 + \alpha_2 \omega_t^2} \\ \varepsilon_{\alpha_3 v_t^2 + \alpha_4 \omega_t^2} \end{pmatrix}$$

Differential Drive

- using the velocity motion model the discrete time forward kinematics are

$$x_t = \begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x_c + \frac{v}{\omega}\sin(\theta + \omega\Delta t) \\ y_c - \frac{v}{\omega}\cos(\theta + \omega\Delta t) \\ \theta + \omega\Delta t \end{pmatrix}$$

$$= \begin{pmatrix} x - \frac{v}{\omega}\sin\theta + \frac{v}{\omega}\sin(\theta + \omega\Delta t) \\ y + \frac{v}{\omega}\cos\theta - \frac{v}{\omega}\cos(\theta + \omega\Delta t) \\ \theta + \omega\Delta t \end{pmatrix} \quad \text{Eqs 5.9}$$

Differential Drive

- there are two problems when trying to use the velocity motion model in a Kalman filter
  1. the plant model is not linear in the state and control

$$x_t = \begin{pmatrix} x - \frac{v_t}{\omega_t}\sin\theta + \frac{v_t}{\omega_t}\sin(\theta + \omega_t\,\Delta t) \\ y + \frac{v_t}{\omega_t}\cos\theta - \frac{v_t}{\omega_t}\cos(\theta + \omega_t\,\Delta t) \\ \theta + \omega_t\,\Delta t \end{pmatrix}$$

  2. it is not clear how to describe the control noises as a plant covariance matrix

$$u_t = \begin{pmatrix} v_t \\ \omega_t \end{pmatrix} + \begin{pmatrix} \varepsilon_{\alpha_1 v_t^2 + \alpha_2 \omega_t^2} \\ \varepsilon_{\alpha_3 v_t^2 + \alpha_4 \omega_t^2} \end{pmatrix}$$

Measurement Model

- there are potentially other problems
  - any non-trivial measurement model will be non-linear in terms of the state
- consider using the known locations of landmarks in a measurement model

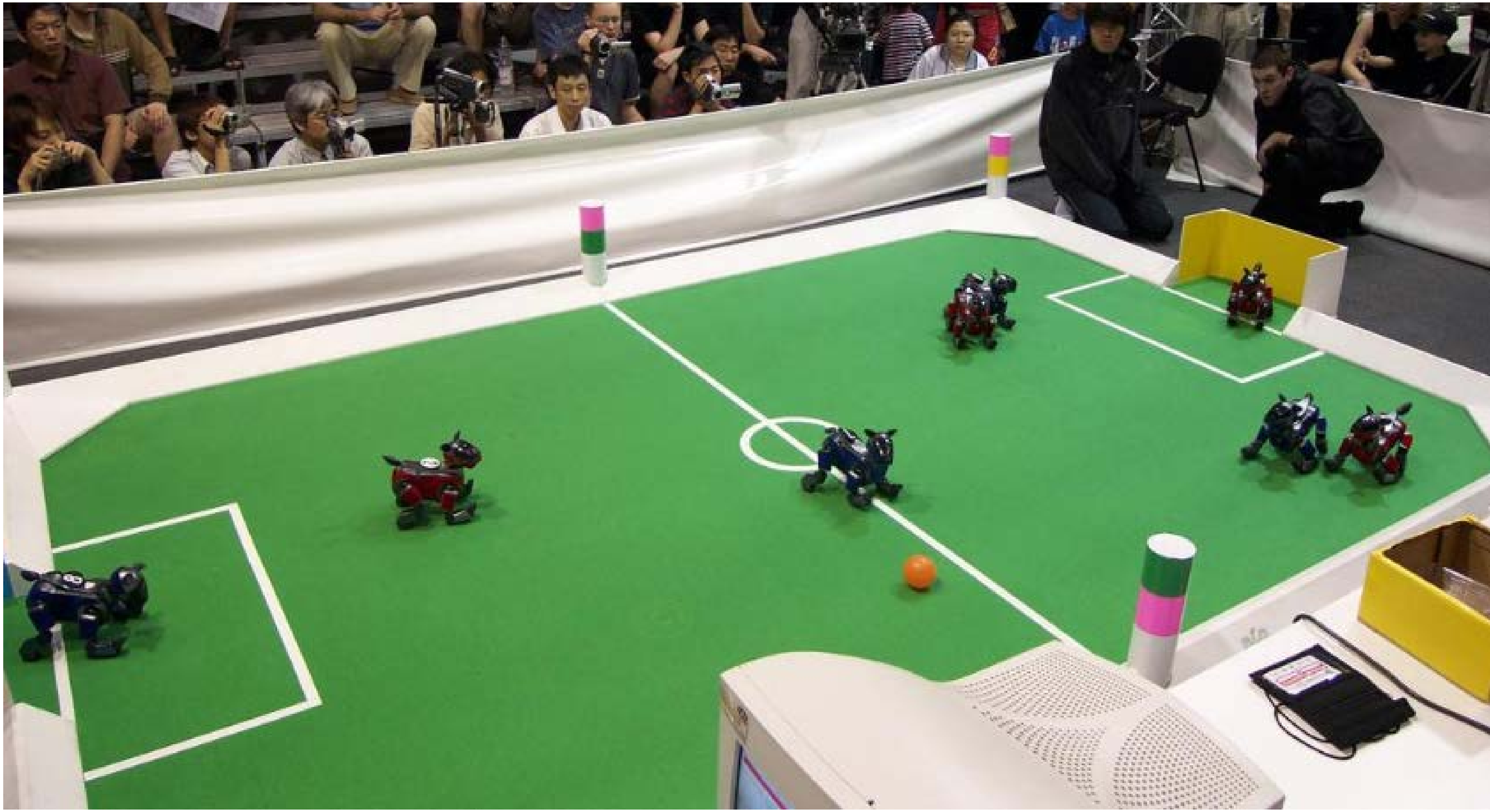
Landmarks

- a landmark is literally a prominent geographic feature of the landscape that marks a known location
- in common usage, landmarks now include any fixed easily recognizable objects
  - e.g., buildings, street intersections, monuments
- for mobile robots, a landmark is any fixed object that can be sensed

Landmarks for Mobile Robots

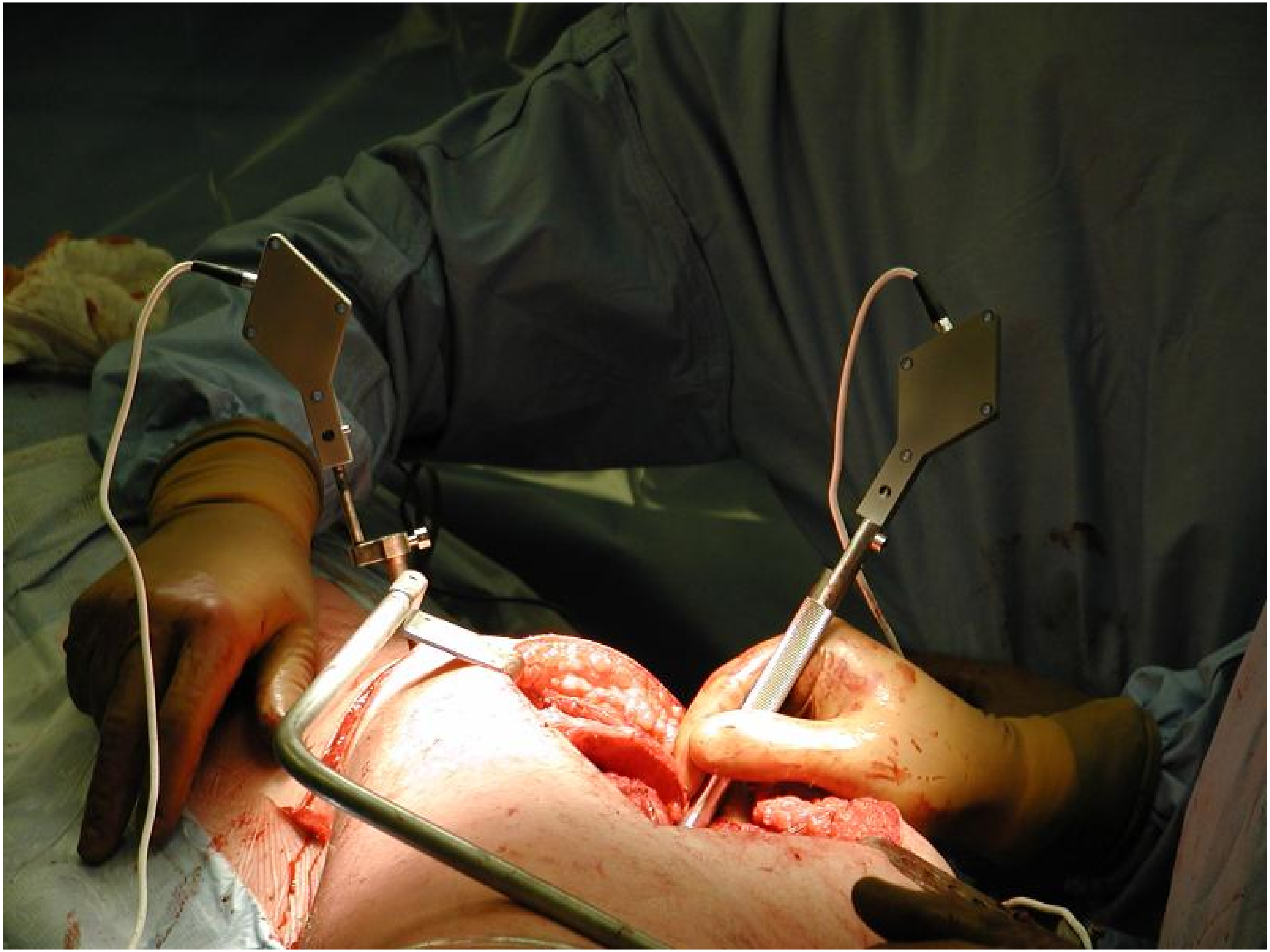
- visual
  - artificial or natural
- retro-reflective
- beacons
  - LORAN (Long Range Navigation): terrestrial radio; now being phased out
  - GPS: satellite radio
- acoustic
- scent?

Landmarks: RoboSoccer



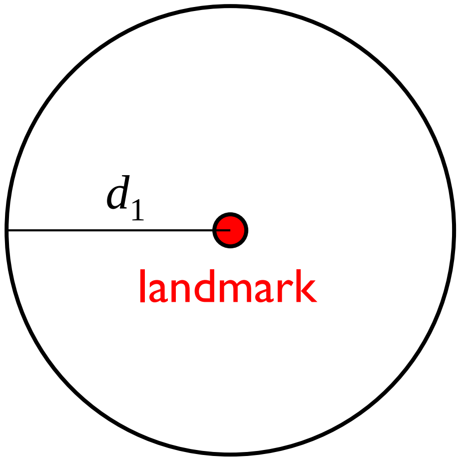
Landmarks: Retroreflector

Landmarks: Active Light



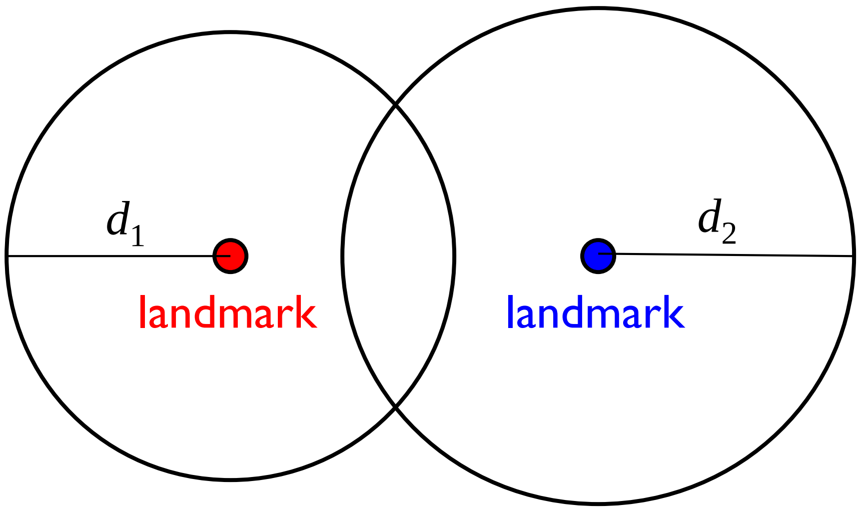
Trilateration

- uses distance measurements to two or more landmarks
- suppose a robot measures the distance d_1 to a landmark
  - the robot can be anywhere on a circle of radius d_1 around the landmark



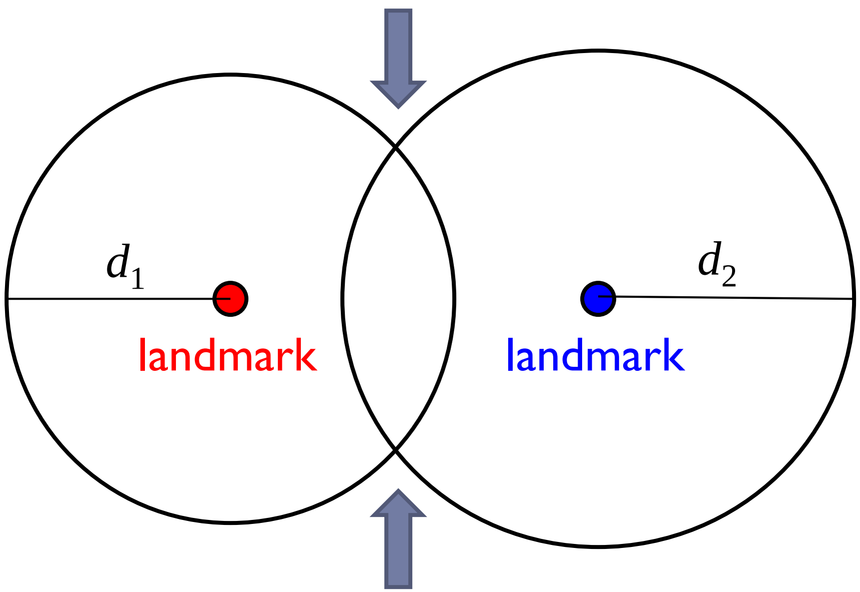
Trilateration

- without moving, suppose the robot measures the distance d_2 to a second landmark
  - the robot can be anywhere on a circle of radius d_2 around the second landmark



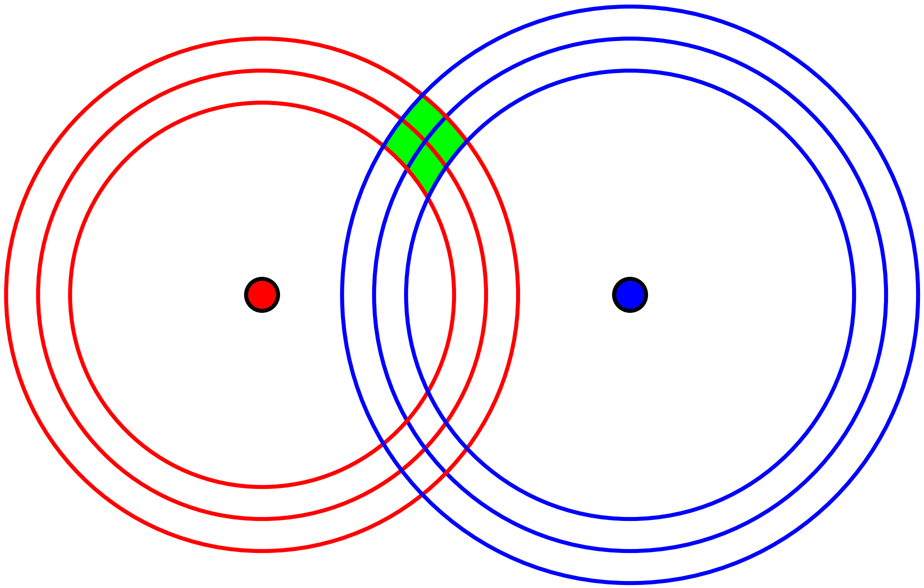
Trilateration

- the robot must be located at one of the two intersection points of the circles
  - tie can be broken if other information is known



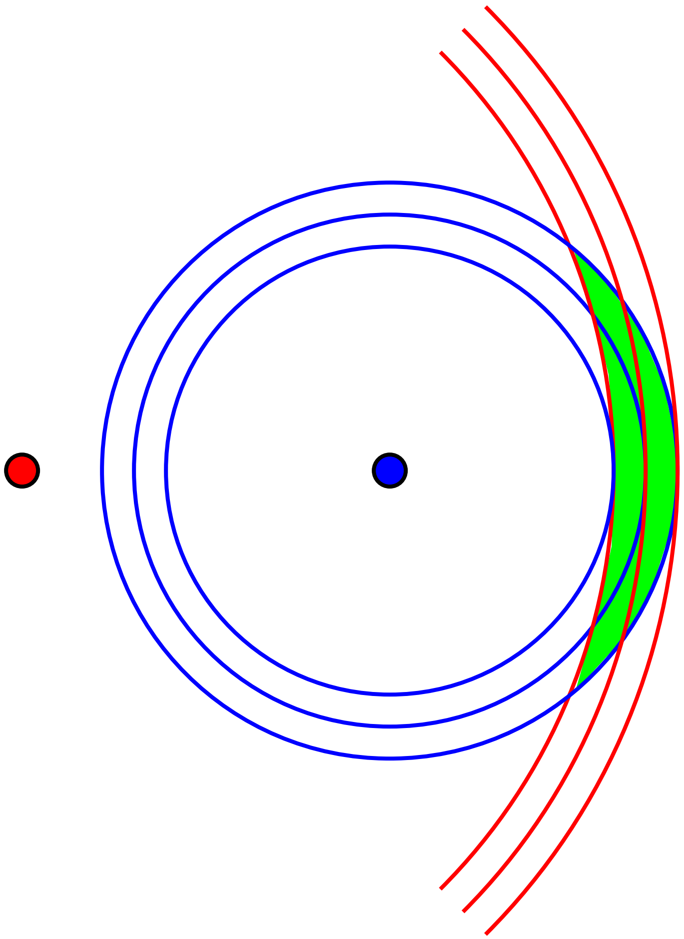
Trilateration

- if the distance measurements are noisy then there will be some uncertainty in the location of the robot



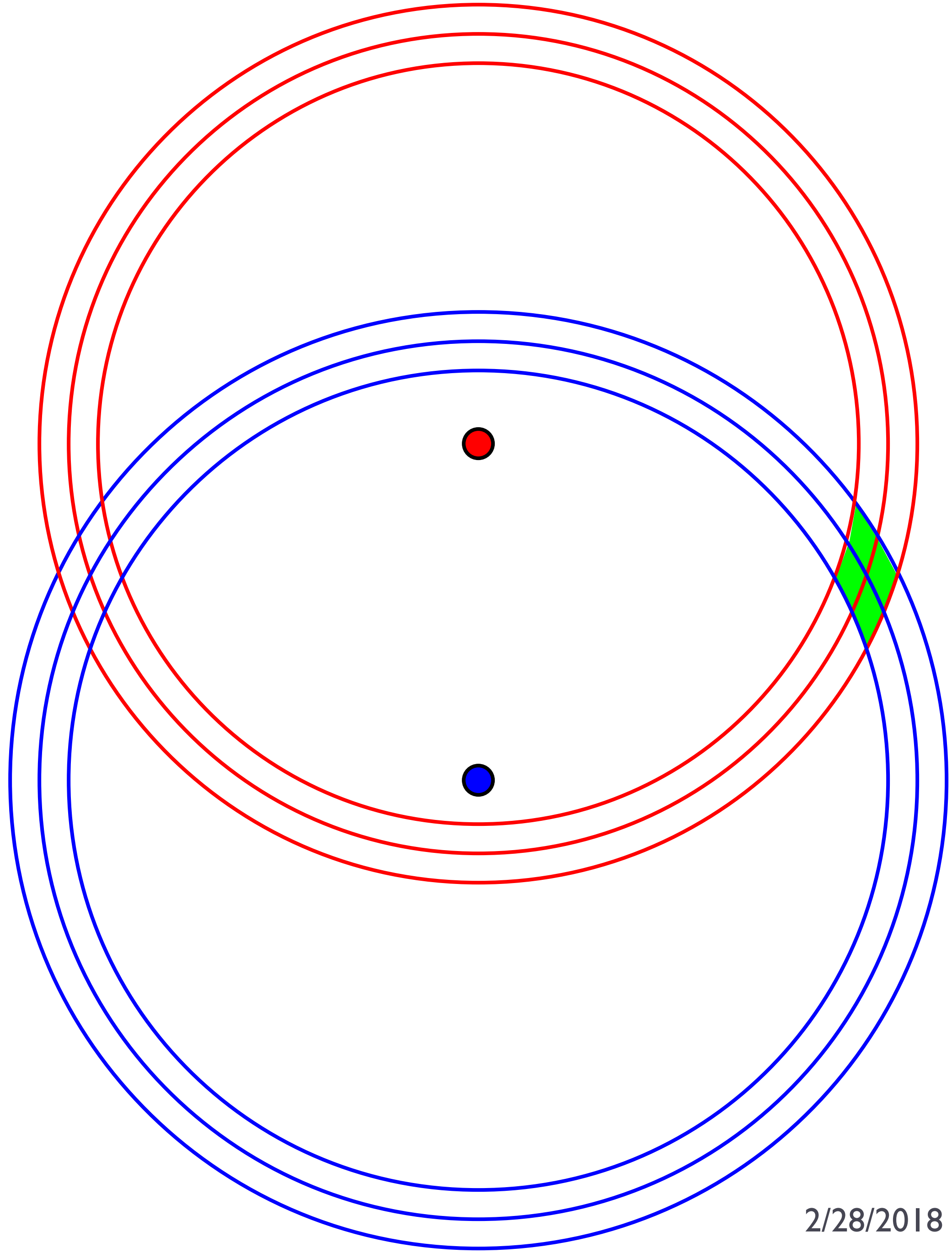
Trilateration

- notice that the uncertainty changes depending on where the robot is relative to the landmarks
- uncertainty grows quickly if the robot is in line with the landmarks



Trilateration

- uncertainty grows as the robot moves farther away from the landmarks
  - but not as dramatically as the previously slide

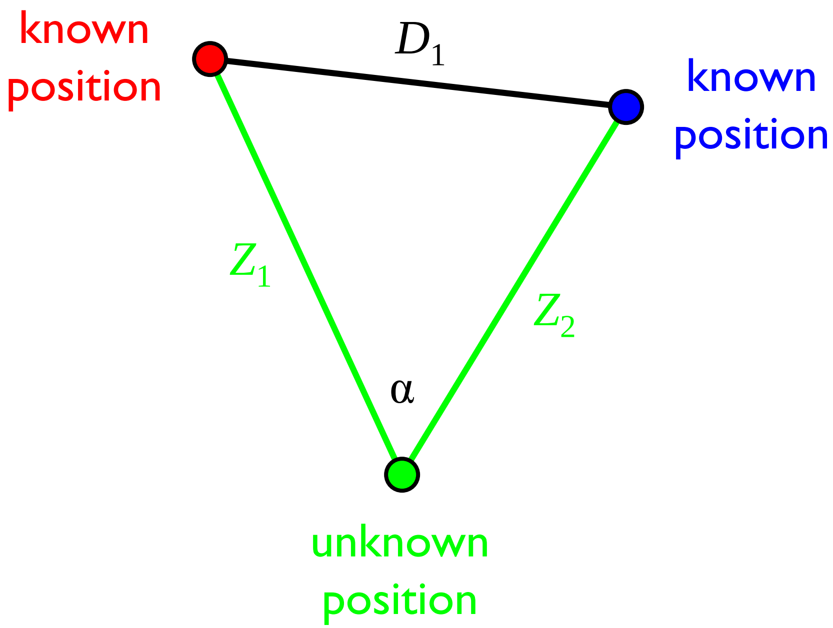


Triangulation

- triangulation uses angular information to infer position
  - http://longhamscouts.org.uk/content/view/52/38/

Triangulation

- in robotics the problem often appears as something like:
  - suppose the robot has a (calibrated) camera that detects two landmarks (with known location)
    - then we can determine the angular separation, or relative bearing, α between the two landmarks



Triangulation

- the unknown position must lie somewhere on a circle arc
  - Euclid proved that any point on the shown circular arc forms an inscribed triangle with angle α
    - we need at least one more beacon to estimate the robot's location