Sensor models

Fast Robots, ECE4160/5160, MAE 4190/5190

E. Farrell Helbling, 4/7/26
Slides adapted from Kirstin Petersen

Class Action Items

- I hope everyone got some rest!
- This week we start lab 9! Mapping.
  - You are going to slowly turn your car on its axis. You want to reliably get concurrent sensor readings from at least
  - If you do this open loop: max 5 points
  - If you do this closed loop: max 7.5 points

You are almost done!

- Lab 1-4: implement robot
- Lab 5-8: control and stunts
- Lab 9-12 localization and mapping
  - Lab 9: mapping
  - Flipped classroom April 10th: simulator
  - Lab 10: localization simulation (S/U)
  - Lab 11: localization on the real robot
  - Lab 12: navigation
- Lectures
  - Bayes filter recap/ SLAM
  - Ethics
  - Guest Lecture: ASML
  - Trivia and Robotics Day!

Bayes Filter

Lecture 17

- Odometry Model
- Velocity Model

1. **Algorithm Bayes_Filter** $(bel(x_{t-1}), u_t, z_t)$:
2. **for** all x_t do
3. $\overline{bel}(x_t) = \Sigma_{x_{t-1}} \; p(x_t \,|\, u_t, x_{t-1}) \; bel(x_{t-1})$ **(Prediction step)**
4. $bel(x_t) = \eta \; p(z_t \,|\, x_t) \; \overline{bel}(x_t)$ **(Update/measurement step)**
5. end **for**
6. **return** $bel(x_t)$

Transition probability/ action model

Measurement probability/ sensor model

Sensor Models $p(z_t \mid x_t)$

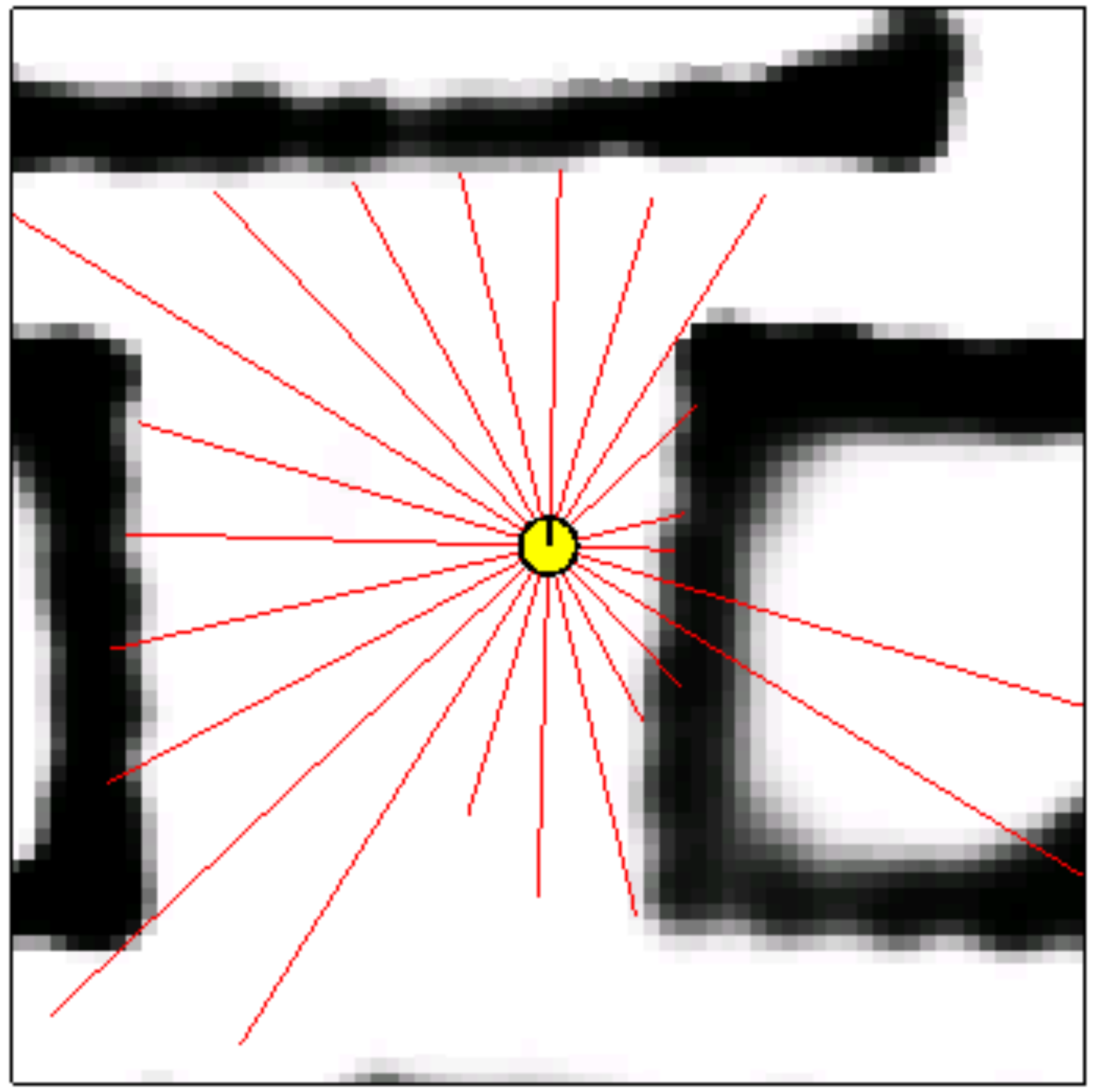
$$p(z_t \mid x_t, m)$$

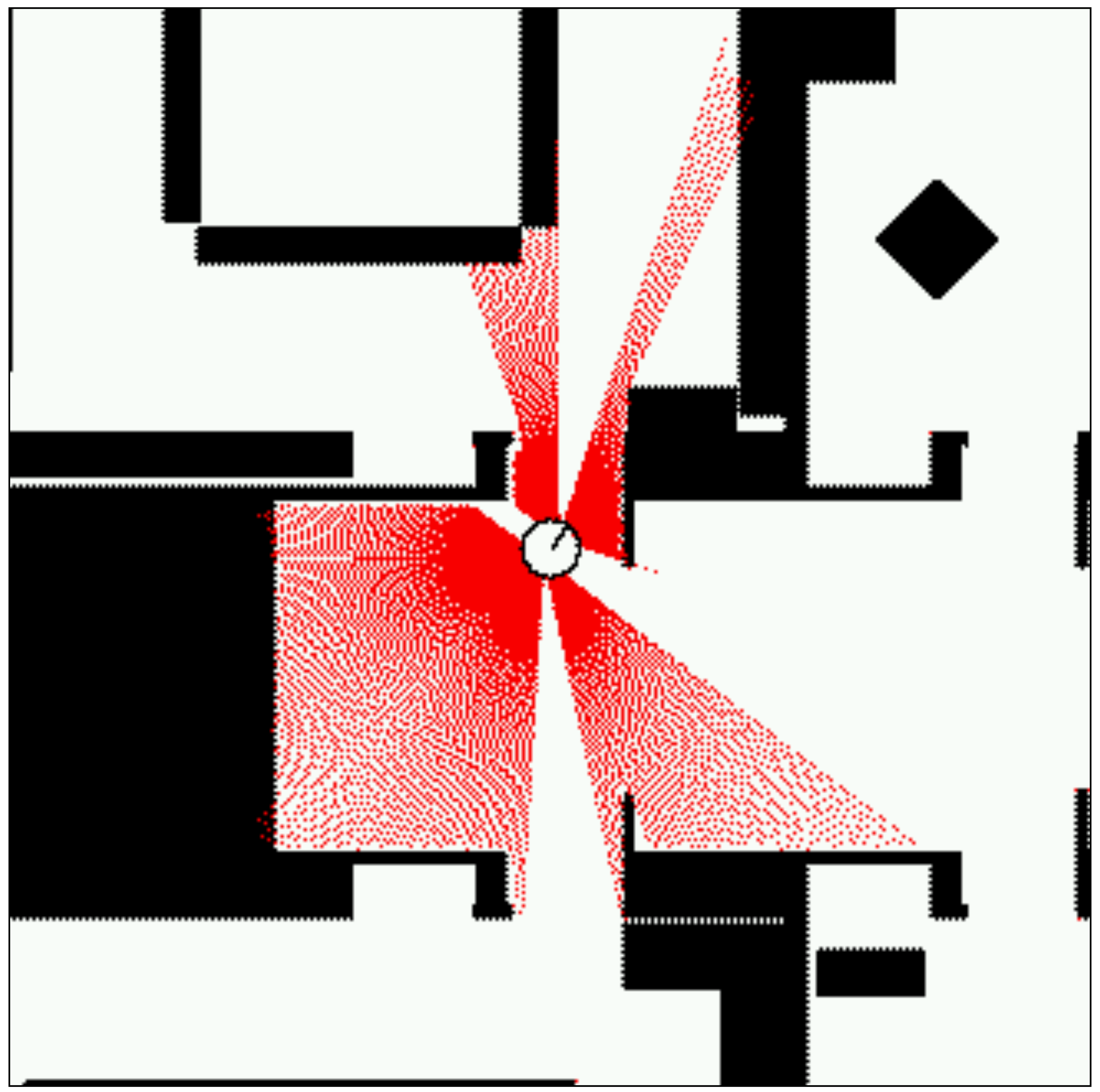
Sensors for Mobile Robots

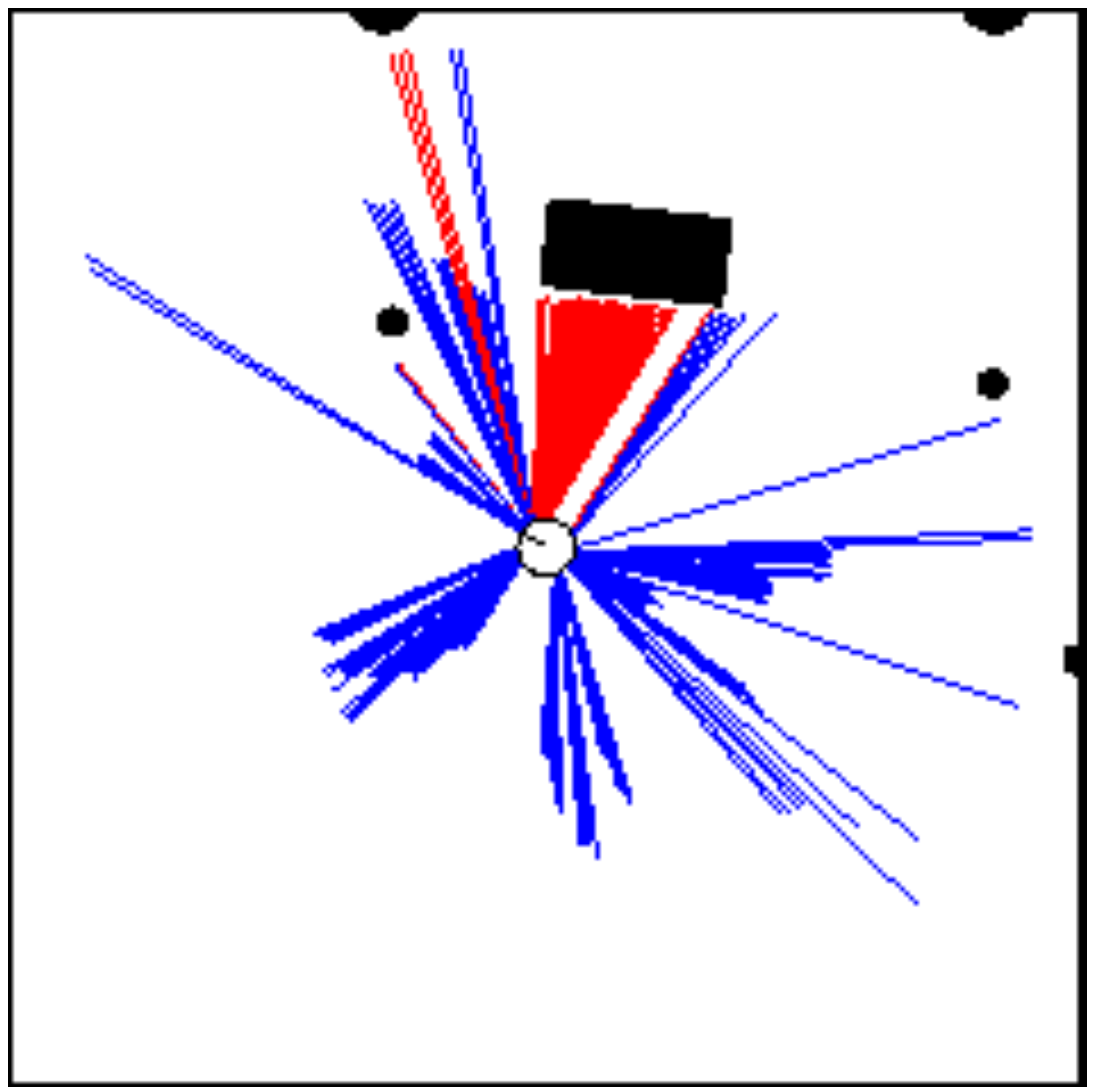
- **Contact Sensors**: bumpers
- **Internal/ Proprioceptive Sensors**:
  - Accelerometers (spring-mounted masses),
  - Gyroscopes (spinning mass, laser light),
  - Compasses, inclinometers (magnetic field, gravity)
- **Range Sensors**:
  - Infrared (intensity)
  - Sonar (time of flight)
  - Radar (phase and frequency)
  - Laser range finders (triangulation, ToF, phase)
- **Visual sensors**: Cameras
- **Satellite-based sensors:** GPS

Sensor Model

- Probabilistic robotics explicitly models the noise in exteroceptive sensor measurements
    - What about proprioceptive sensors?
- Where does the noise come from?

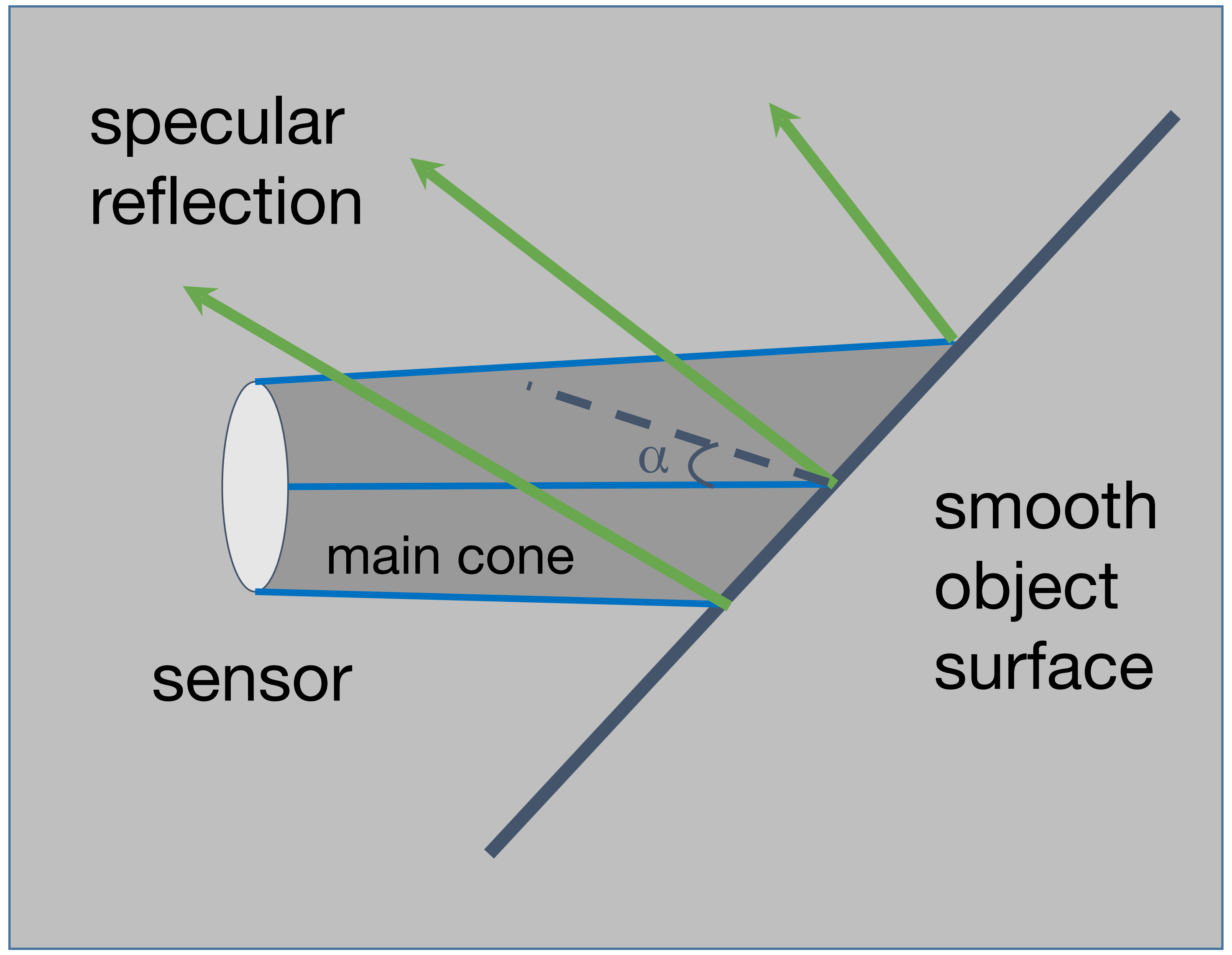




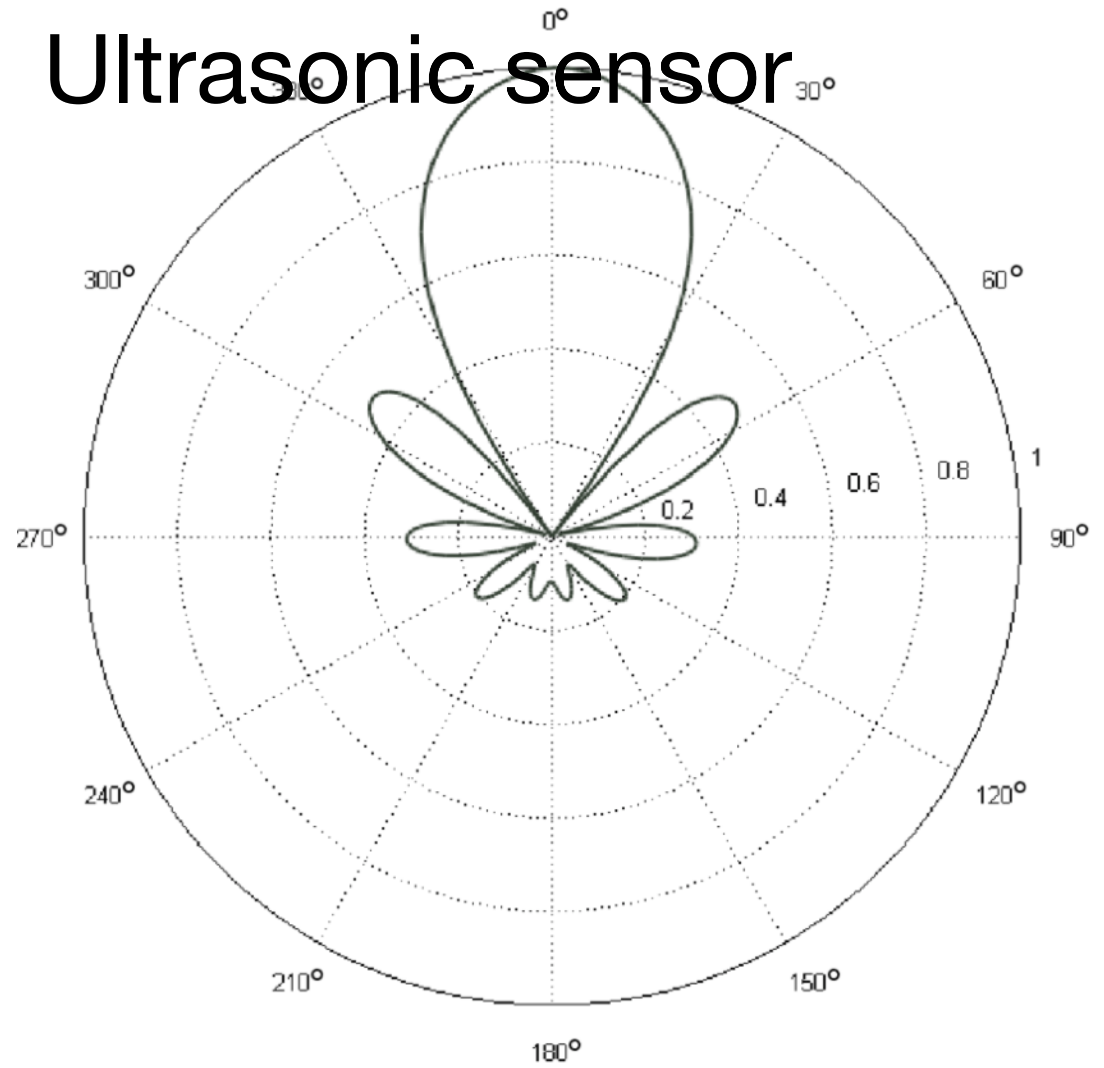


Range Sensor Inaccuracies “noise”

- **Readings > true distance**
  - Surface material
  - Angle between surface normal and direction of sensor cone
  - Width of the sensor cone of measurement
  - Sensitivity of the sensor cone
- **Readings < true distance**
  - Crosstalk between different sensors
  - Unmodeled objects in the proximity of the robot, such as people



Ultrasonic sensor

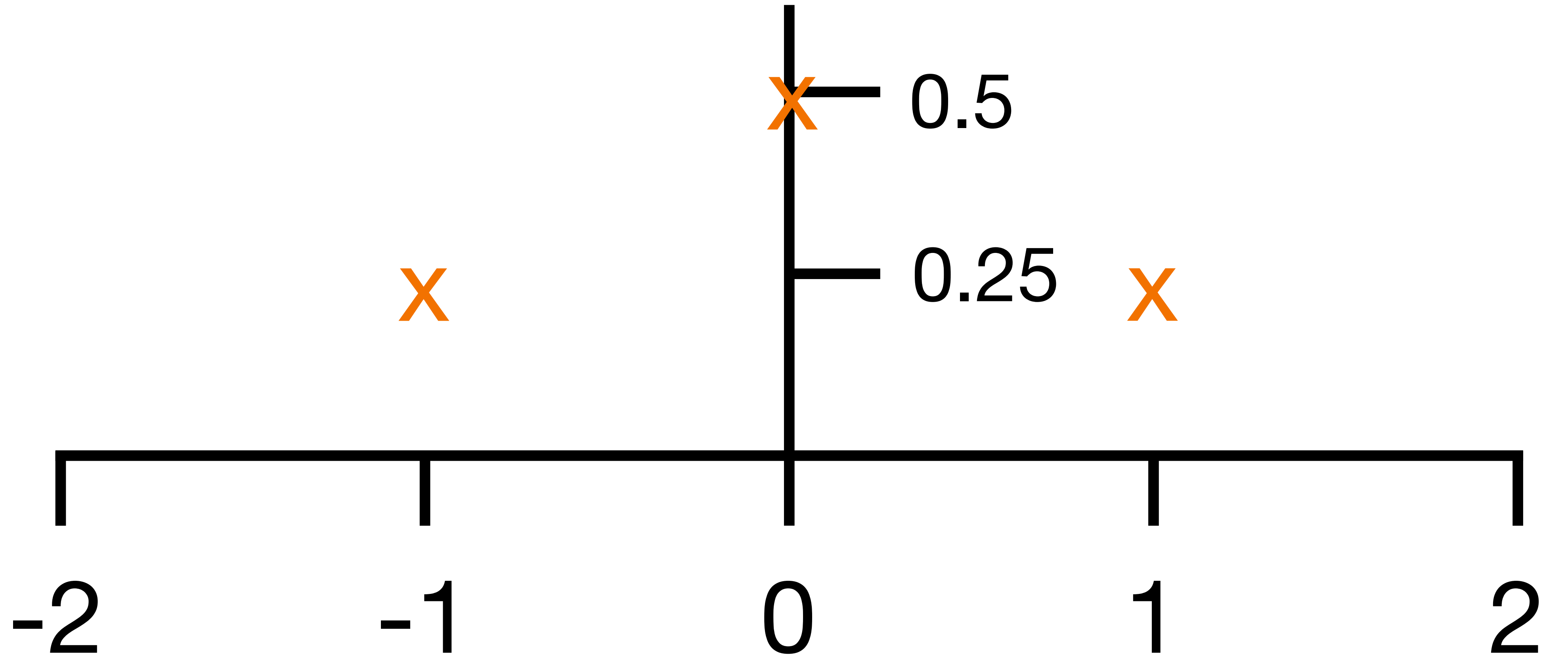


Probabilistic Sensor Model

- Perfect sensor models…
  - $z = f(x)$
  - … practically impossible
  - ….computationally intractable
- Practical sensor models…
  - $p(z \mid x)$
- Three common sensor models
  - Beam model
  - Likelihood model
  - Feature-based model

Until now our sensor models have been simple

- $p(z = \text{correct})$
- $p(z \mid x)$ for a small state space



Beam Model

Beam model of range finders

- Let there be K **individual measurement values** within a measurement z_t

$$z_t = \{z_t^1, z_t^2, \ldots, z_t^K\}$$

- Individual measurements are independent given the robot state

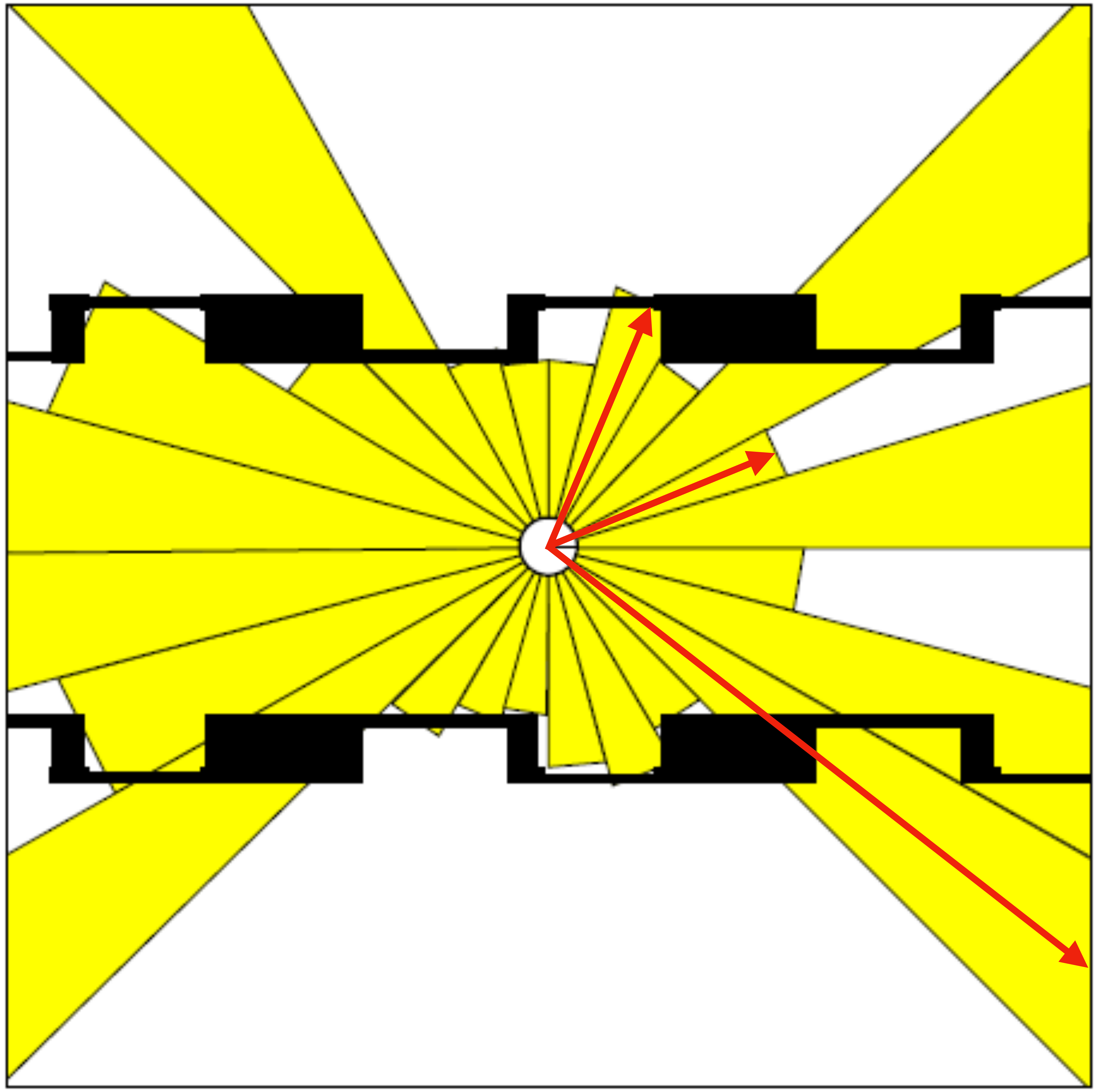
$$p(z_t, x_t, m) = \prod_{k=1}^{K} p(z_t^k \mid x_t, m)$$

Sensor measurements are caused by real world objects

- Can you think of violations to that assumption?
  - People, errors in the map model m, approximations in the posterior, etc.
  - But it makes computation much more tractable

Range measurements

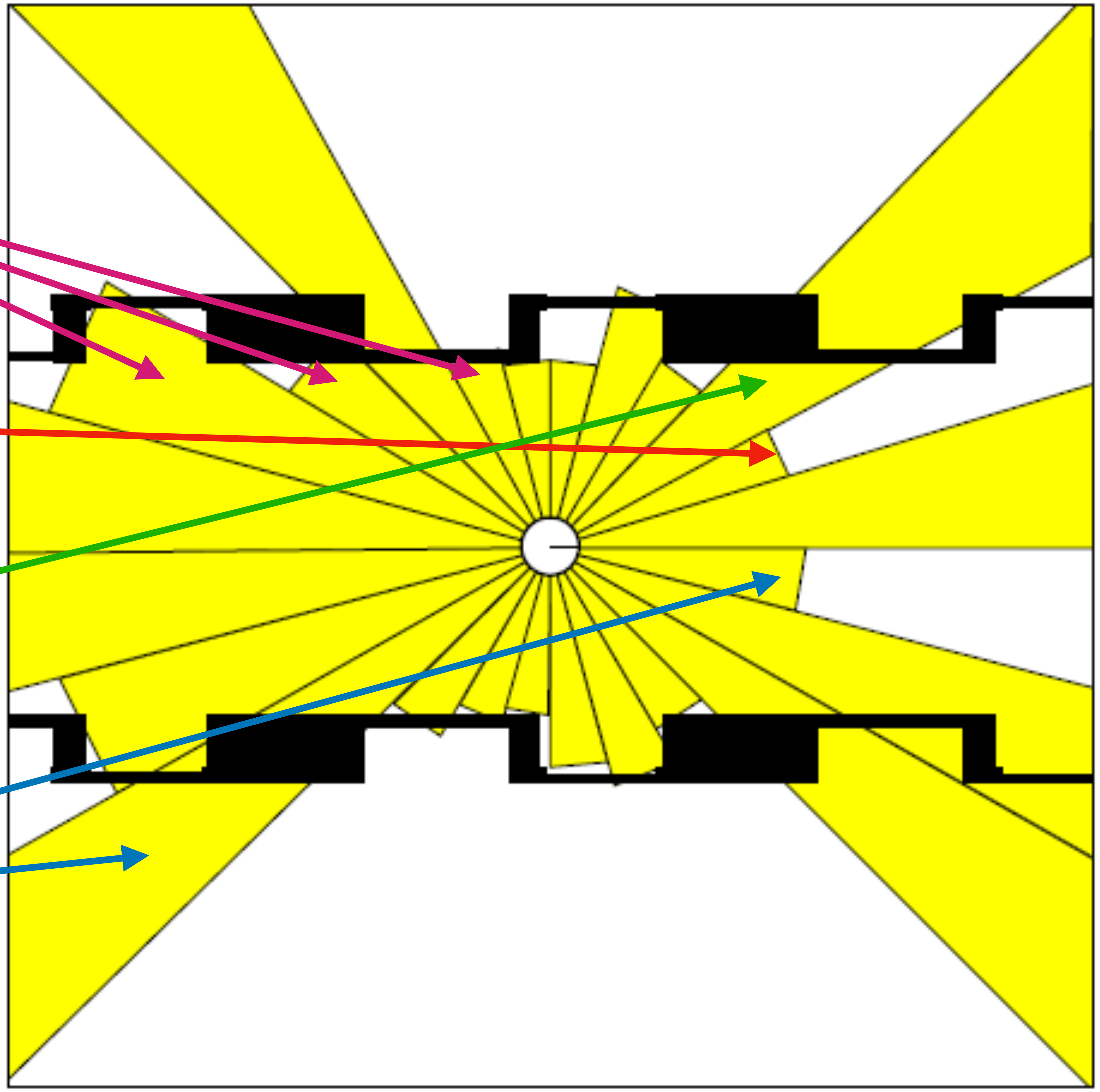
Typical measurement errors



Range measurements

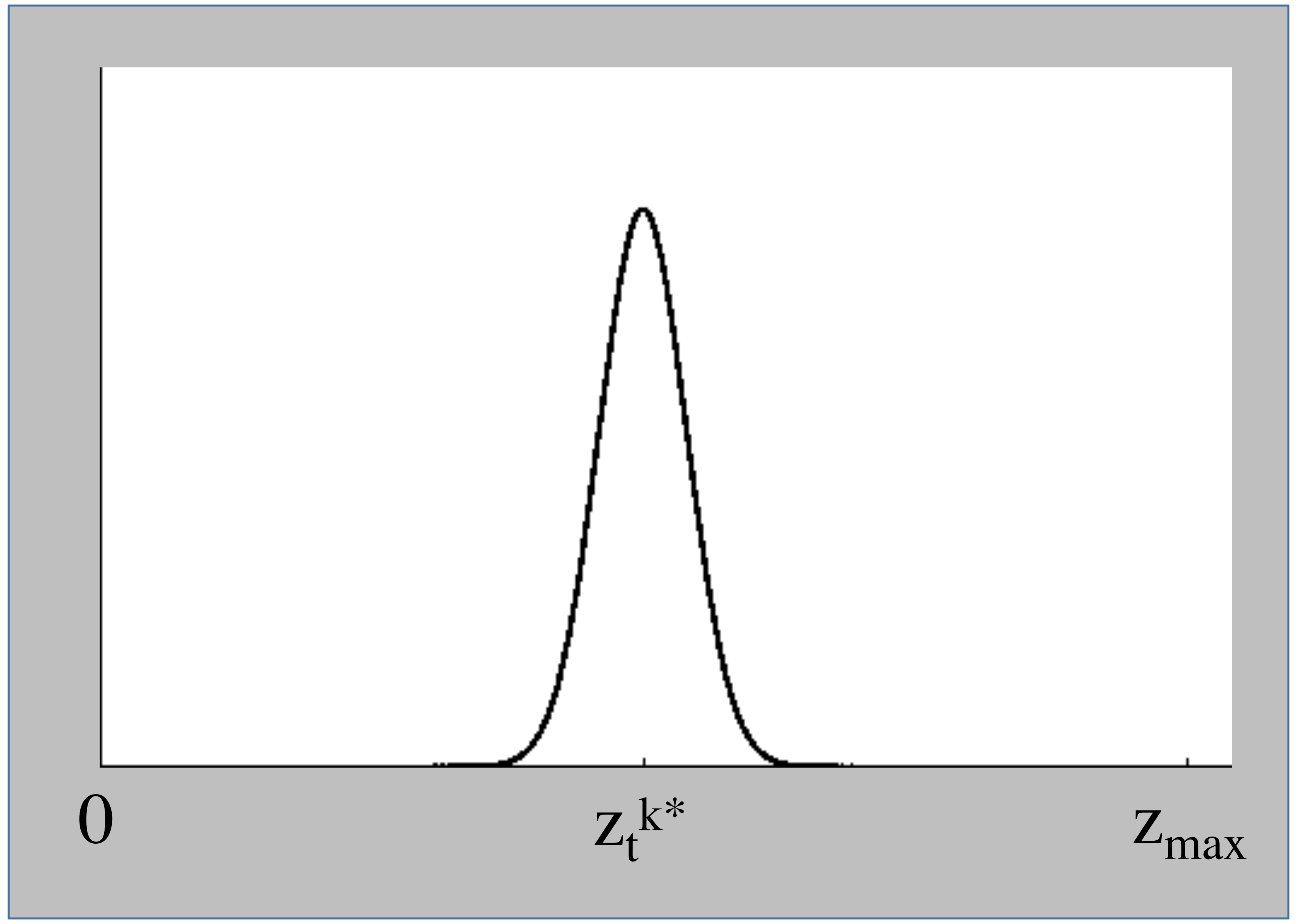
Typical measurement errors

- **Correct range measurements**
  - Beams reflected by obstacles
- **Unexpected objects**
  - Beams reflected by persons
  - Crosstalk
- **Failures**
- **Random Measurements**



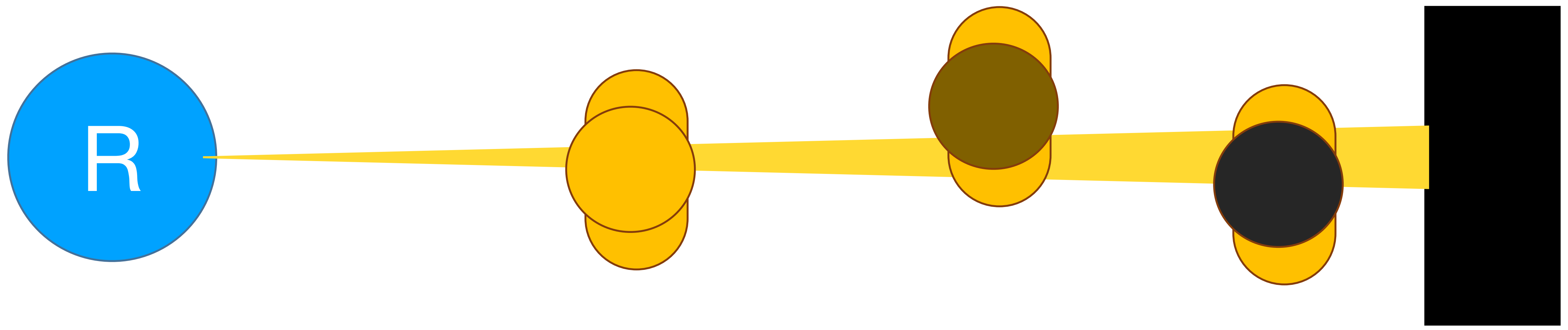
Correct range measurements

- Reading: z_t^k
- True value: z_t^{k*}
  - In a location-based map, z_t^{k*} is usually estimated by **ray casting**
- Measurement noise
  - Narrow **Gaussian** p_{hit} with mean z_t^{k*} and standard deviation σ_{hit}

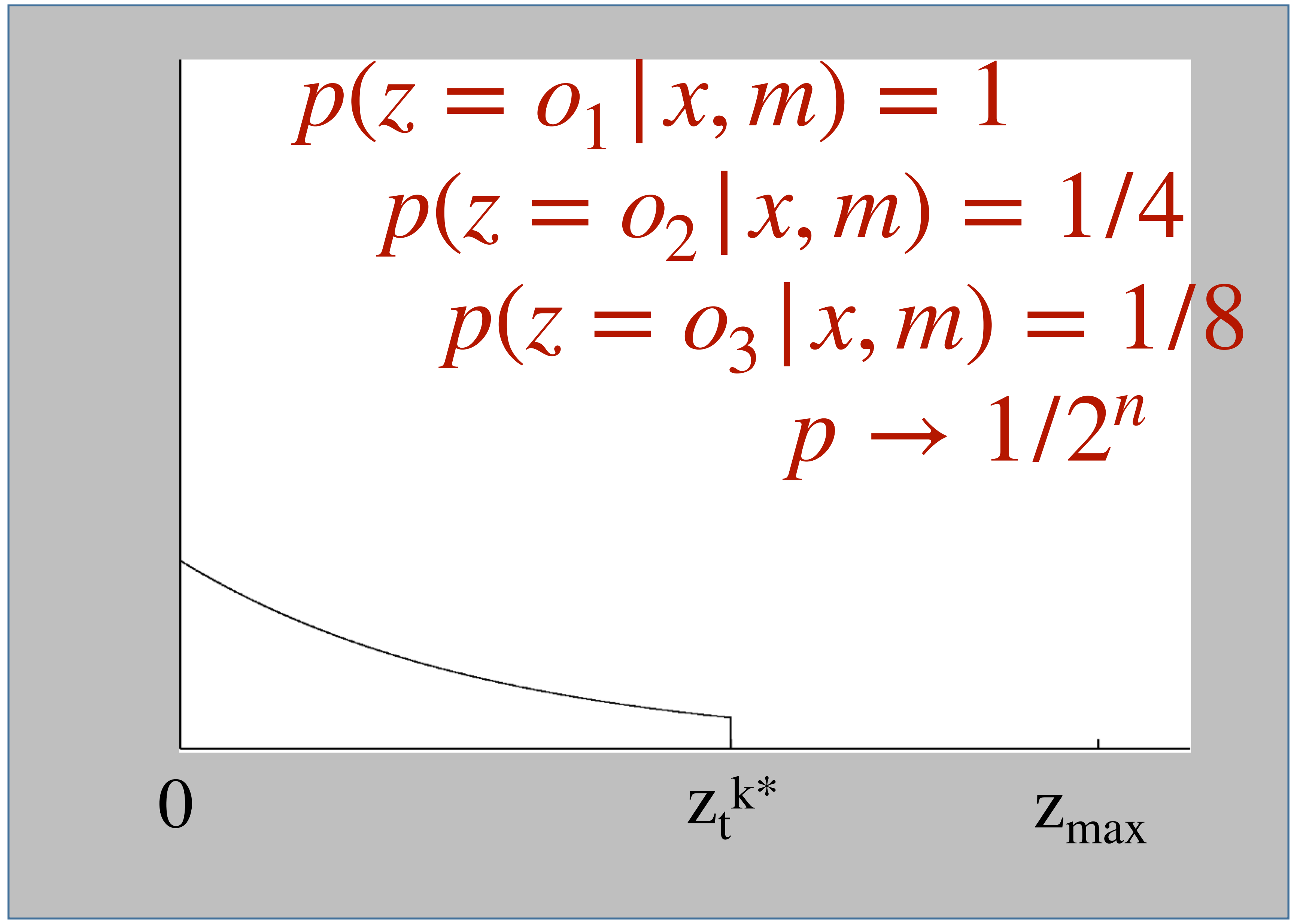


$$p_{hit}(z_t^k \mid x_t, m) = \begin{cases} \eta f(z_t^k, z_t^{k*}, \sigma_{hit}) & \text{if } 0 \leq z_t^k \leq z_{max} \\ 0 & \text{otherwise} \end{cases}$$

Unexpected objects



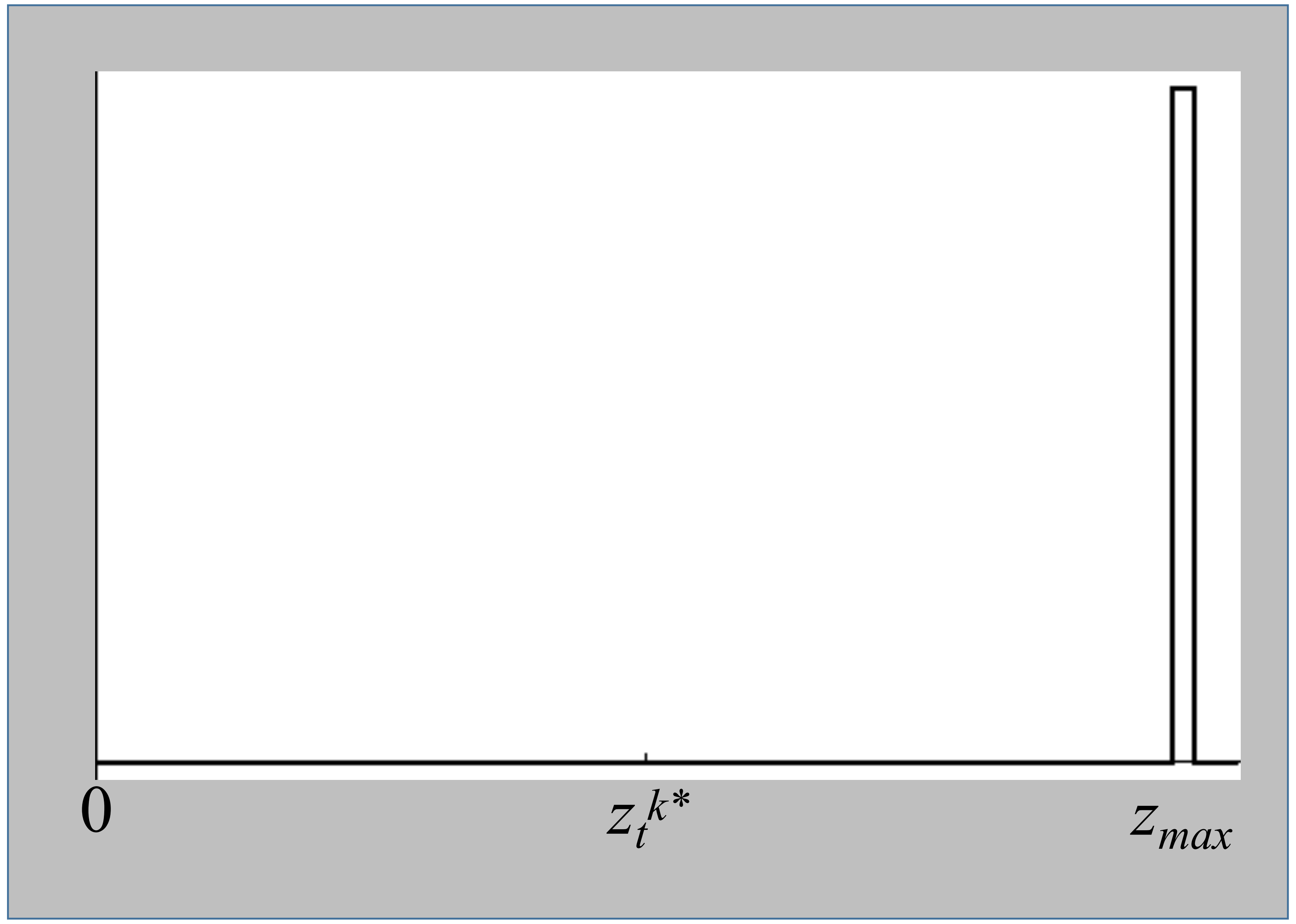
- Real world is dynamic
- **Objects not contained in the map** can cause shorter readings
  - Treat them as part of the state vector and estimate their location
  - Treat them as sensor noise
- The likelihood of sensing unexpected objects decreases with range
- Model as **exponential distribution** p_{short}



$$p_{short}(z_t^k | x_t, m) = \begin{cases} \eta \lambda_{short} e^{-\lambda_{short} z_t^k} & \text{if } 0 \leq z_t^k \leq z_t^{k*} \\ 0 & \text{otherwise} \end{cases}$$

Failures

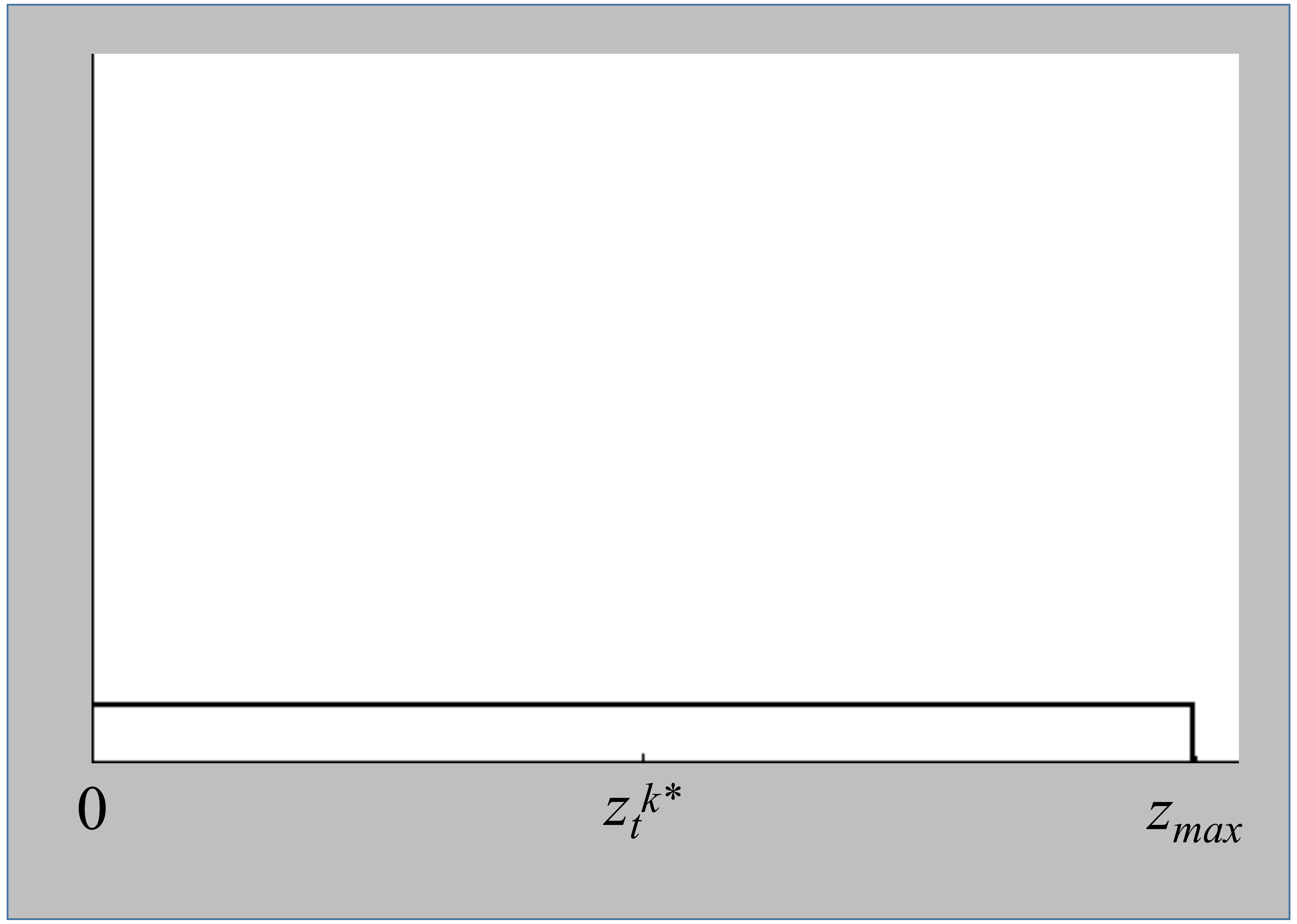
- **Obstacles might be missed altogether**
- The result is a max-range measurement z_{max}
- Model as a **point-mass distribution** p_{max}



$$p_{max}(z_t^k \mid x_t, m) = I(z = z_{max}) = \begin{cases} 1 & \text{if } z = z_{max} \\ 0 & \text{otherwise} \end{cases}$$

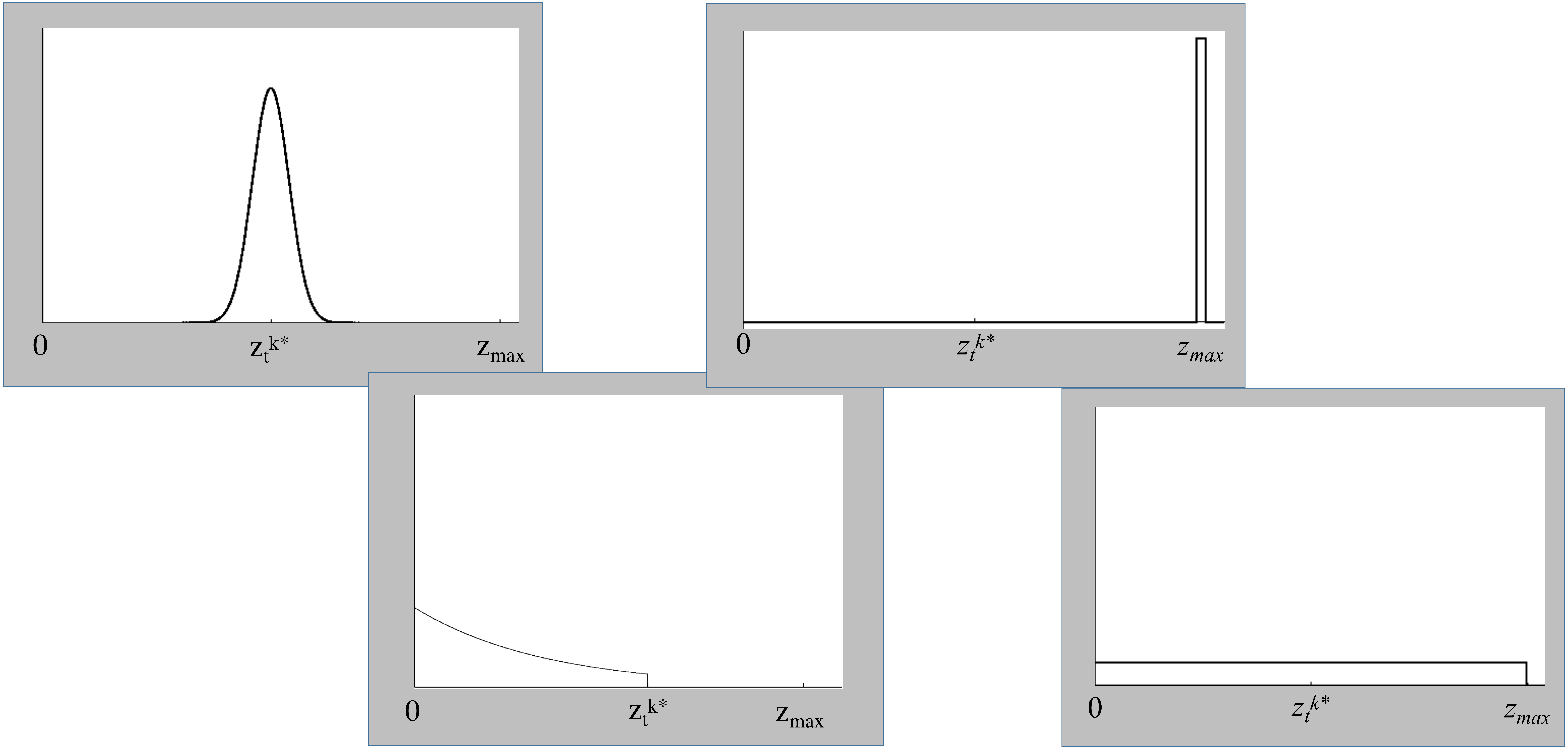
Random measurements

- Range finders can occasionally produce **entirely inexplicable measurements**
- Modelled as a **uniform distribution** p_{rand} over the measurement range



$$p_{rand}(z_t^k \mid x_t, m) = \begin{cases} \frac{1}{z_{max}} & \text{if } 0 \le z_t^k \le z_{max} \\ 0 & \text{otherwise} \end{cases}$$

Beam Model

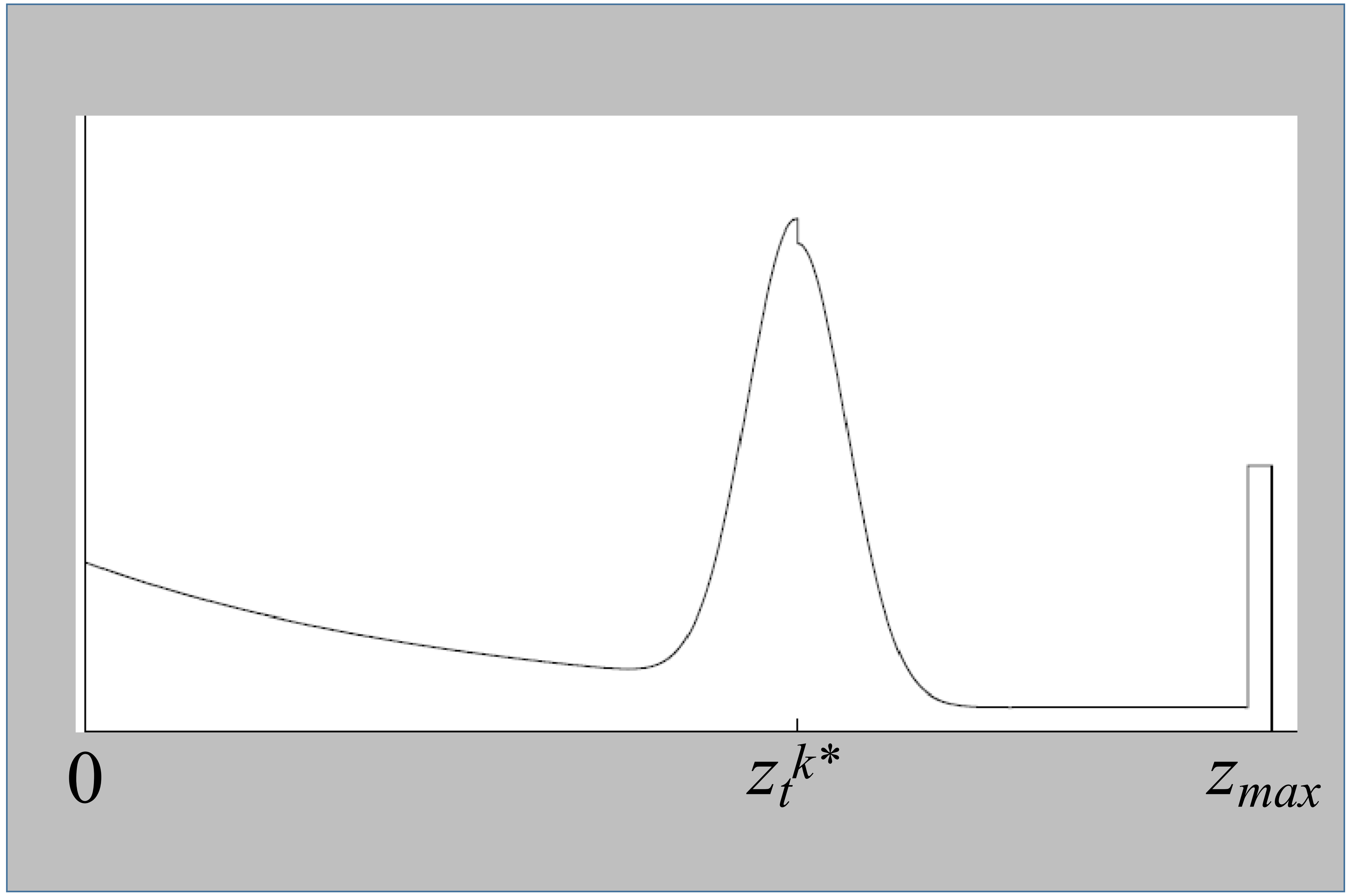


Beam range model as a mixture density

- The four different distributions are mixed by a weighted average

$$p(z_t^k \mid x_t, m) = \begin{pmatrix} \alpha_{hit} \\ \alpha_{short} \\ \alpha_{max} \\ \alpha_{rand} \end{pmatrix} \cdot \begin{pmatrix} p_{hit}(z_t^k \mid x_t, m) \\ p_{short}(z_t^k \mid x_t, m) \\ p_{max}(z_t^k \mid x_t, m) \\ p_{rand}(z_t^k \mid x_t, m) \end{pmatrix}$$

$$\alpha_{hit} + \alpha_{short} + \alpha_{max} + \alpha_{rand} = 1$$



Algorithm for beam model

Algorithm beam_range_finder_model (z_t, x_t, m) :

$q = 1$

for $k = 1$ to K do

compute z_t^{k*} for z_t^k using ray casting

$$p = \alpha_{hit} \cdot p_{hit}(z_t^k \,|\, x_t, m) + \alpha_{short} \cdot p_{short}(z_t^k \,|\, x_t, m)$$
$$+\alpha_{max} \cdot p_{max}(z_t^k \,|\, x_t, m) + \alpha_{rand} \cdot p_{rand}(z_t^k \,|\, x_t, m)$$

$q = q \cdot p$

return q

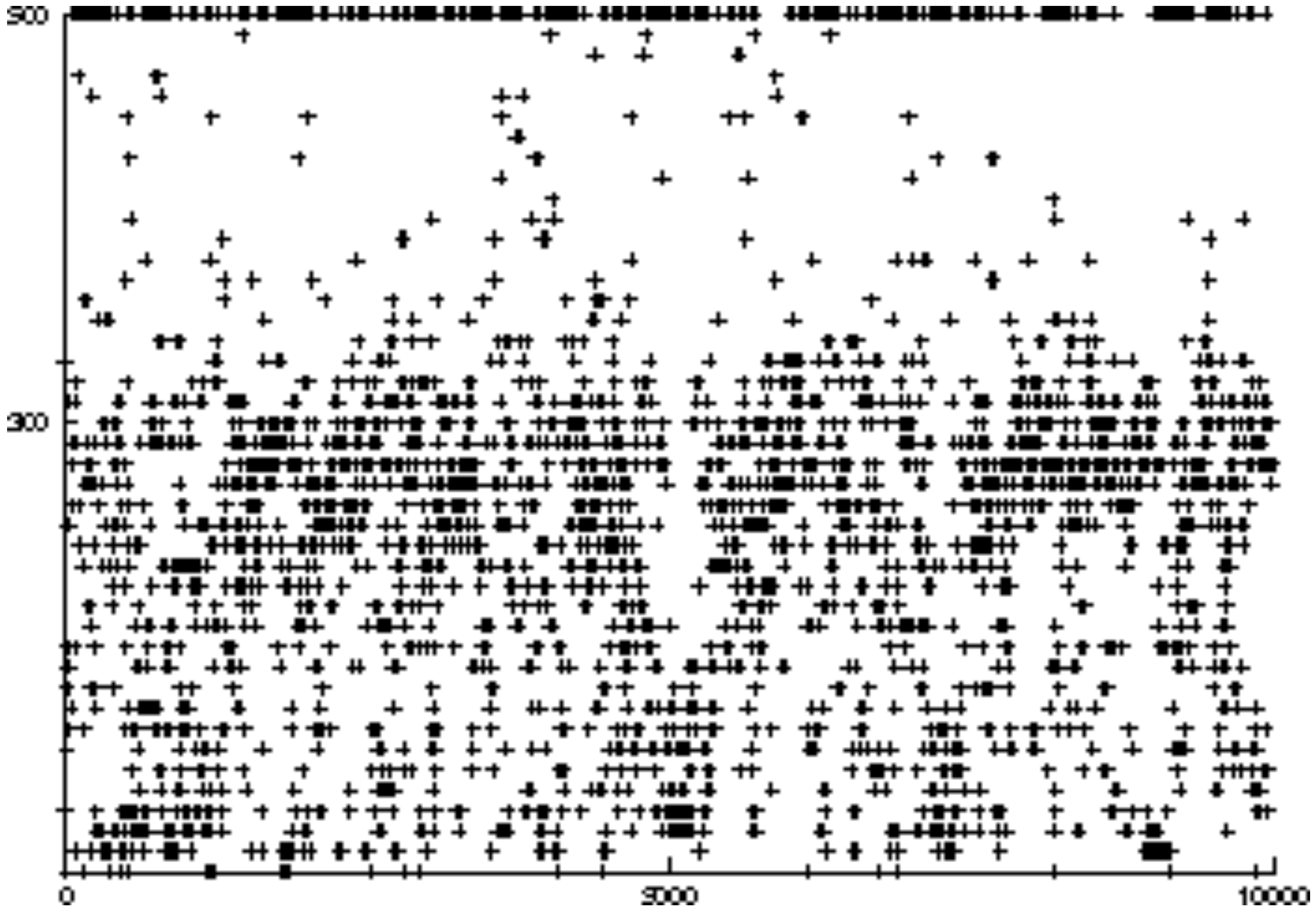
Beam Range Model

Parameters

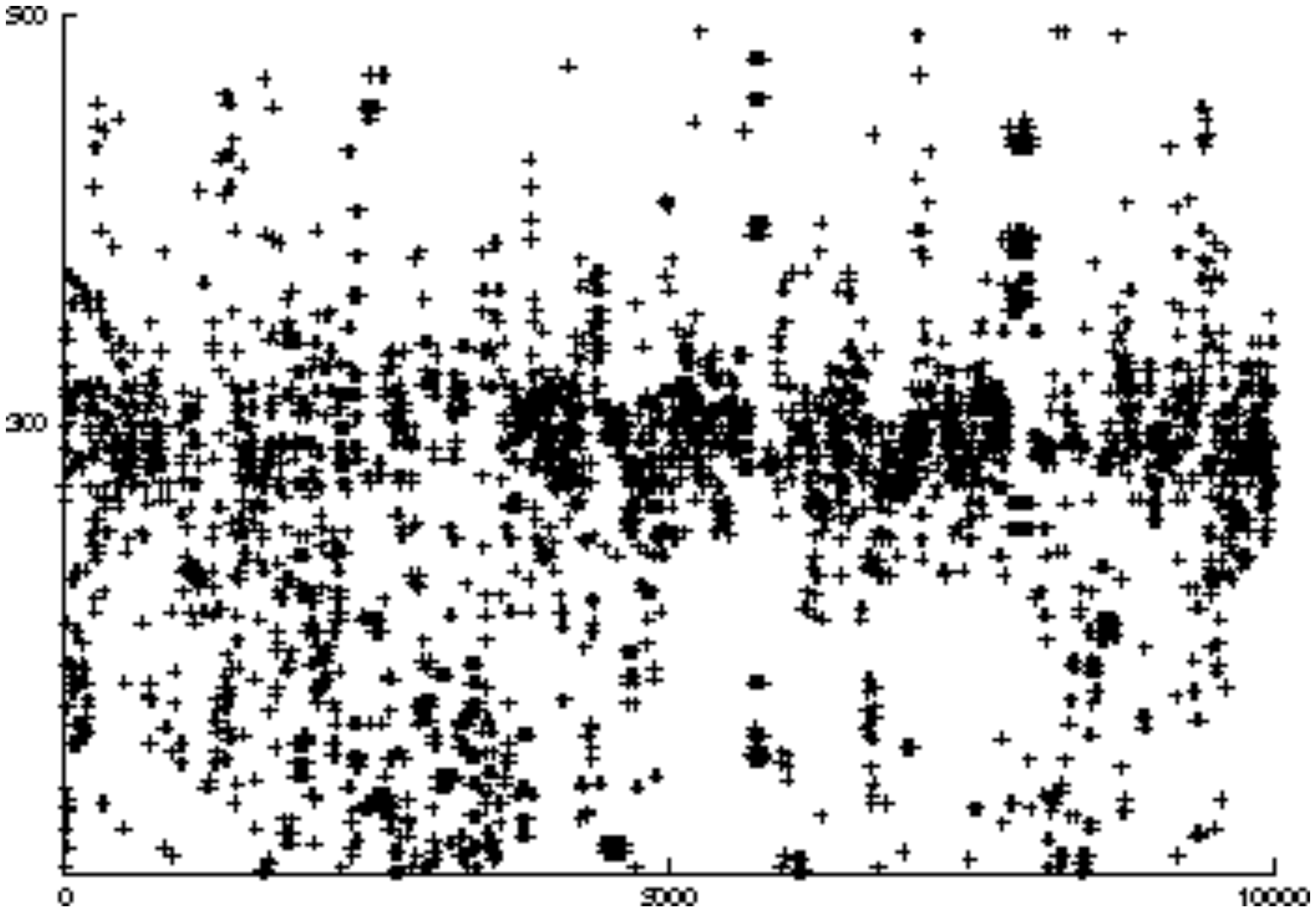
- Intrinsic parameters Θ of the beam range model
  - $\alpha_{hit}, \alpha_{short}, \alpha_{max}, \alpha_{rand}, \lambda_{short}$
  - Affect the likelihood of any sensor measurement
- Estimation methods
  - Guesstimate the resulting density
  - Learn parameters using a Maximum Likelihood Estimator
  - Hill climbing, gradient descent, genetic algorithms, etc.

Raw sensor data

Sonar sensor data

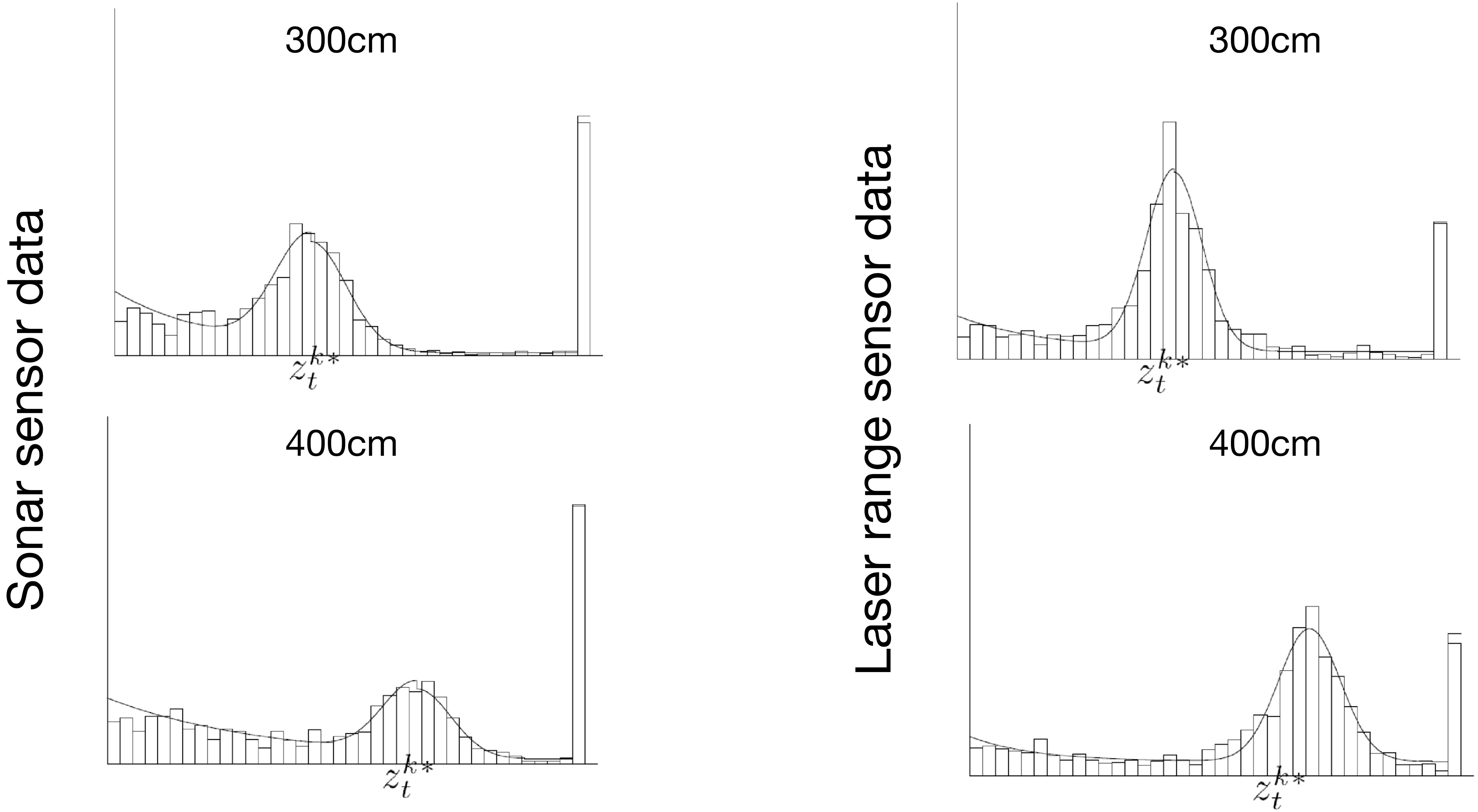


Laser range sensor



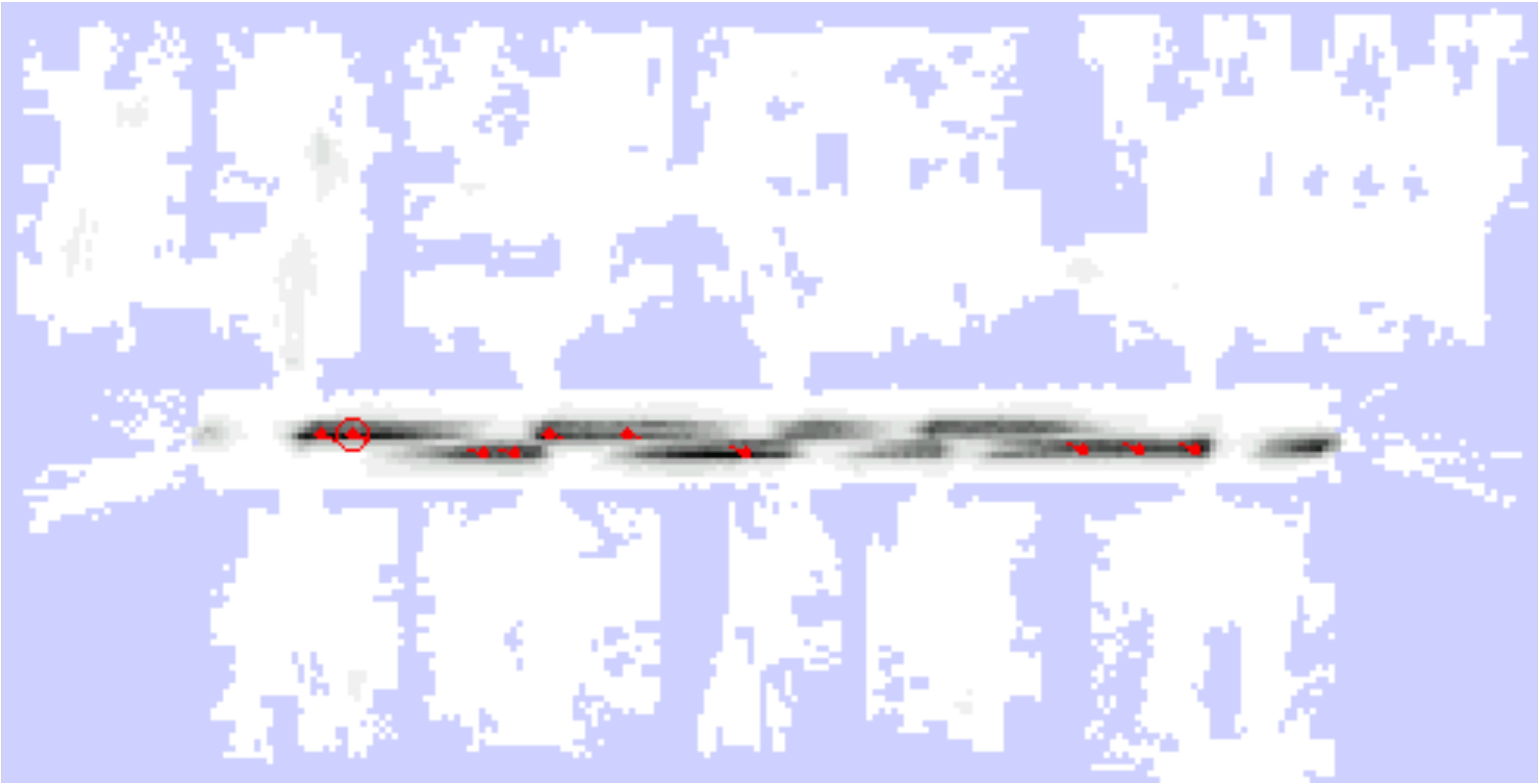
True range is 300 cm and maximum range is 500 cm

Approximation results with MLE



Beam model in action

Laser scan projected into a partial map m



Likelihood $p(z_t \mid x_t, m)$ for all positions x_t projected into the map. The darker a position, the larger $p(z_t \mid x_t, m)$

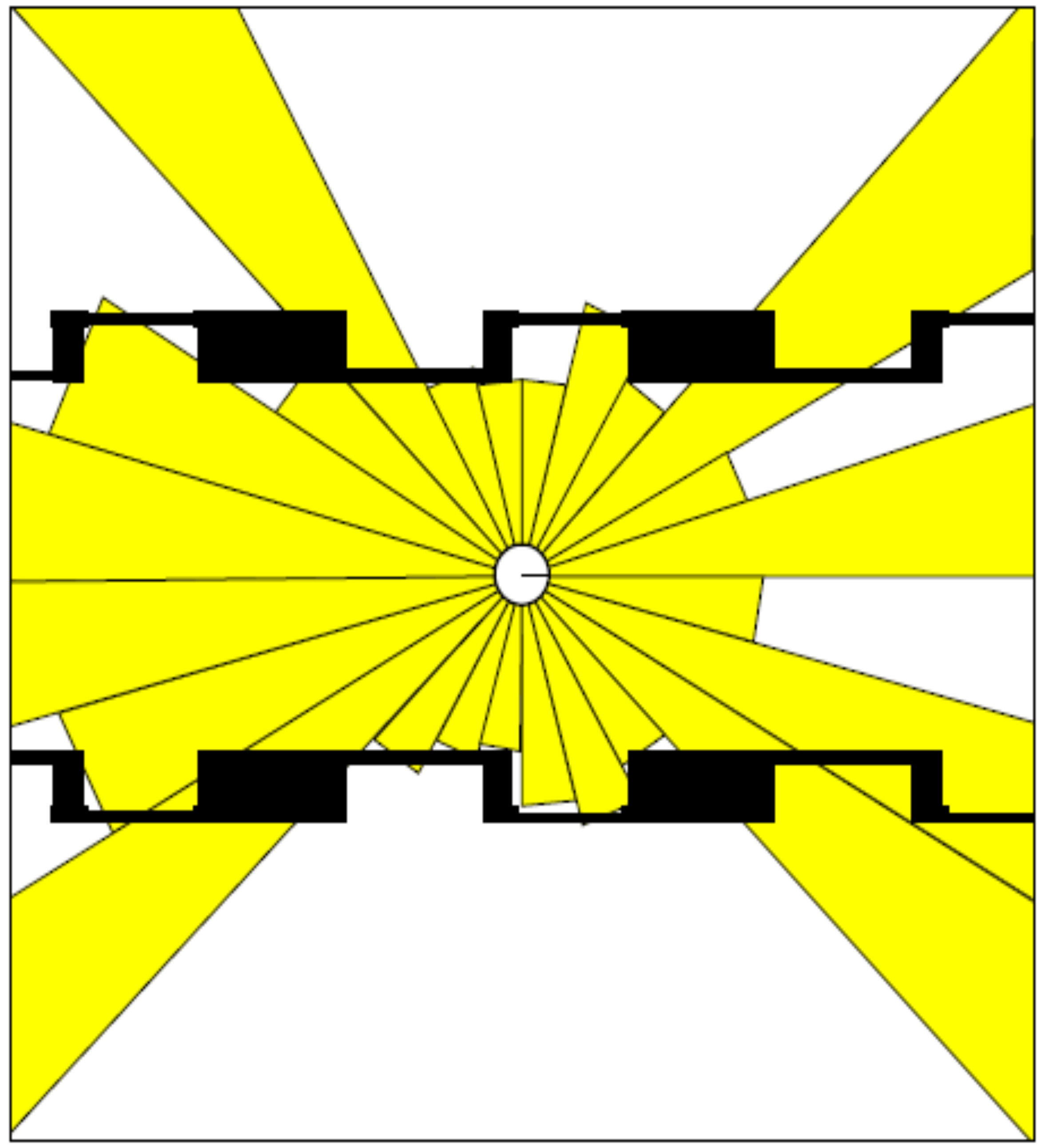
Summary of beam model

- **Overconfident**
  - Assumes independence between individual measurements
- Models **physical causes** for measurements
- Implementation involves **learning parameters** based on real data
- **Limitations**
  - Different models are needed for every possible scenario (e.g., hit angles for intensity sensors)
  - Raytracing is computationally expensive (but can be pre-processed)
  - Not smooth for small obstacles, at edges, or in cluttered environments

Likelihood fields

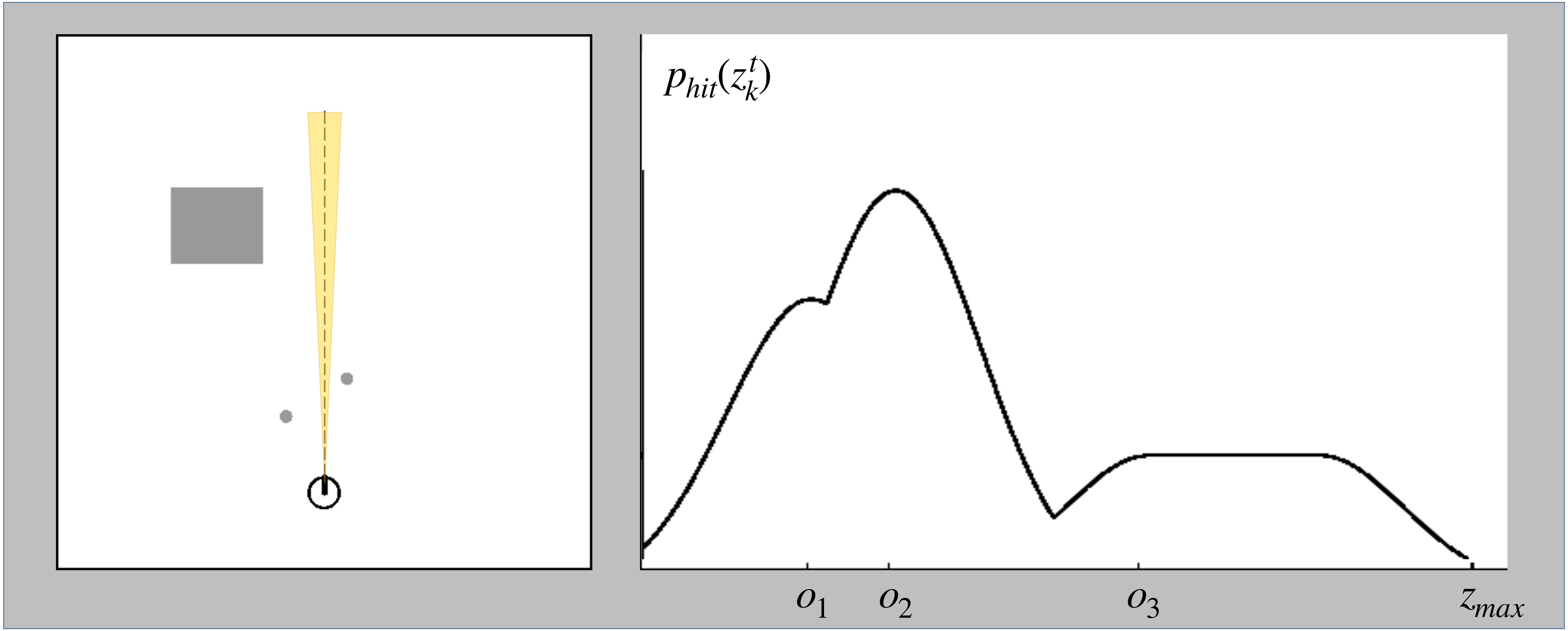
Likelihood fields of range finders

- Instead of following along the beam, just check the end point
- Project sensor scan z_t into the map and compute the closest end point
- Probability function is a mixture of
  - A Gaussian distribution with mean at the distance closest to the obstacle
  - A uniform distribution for random measurements
  - A point-mass distribution for max range measurements



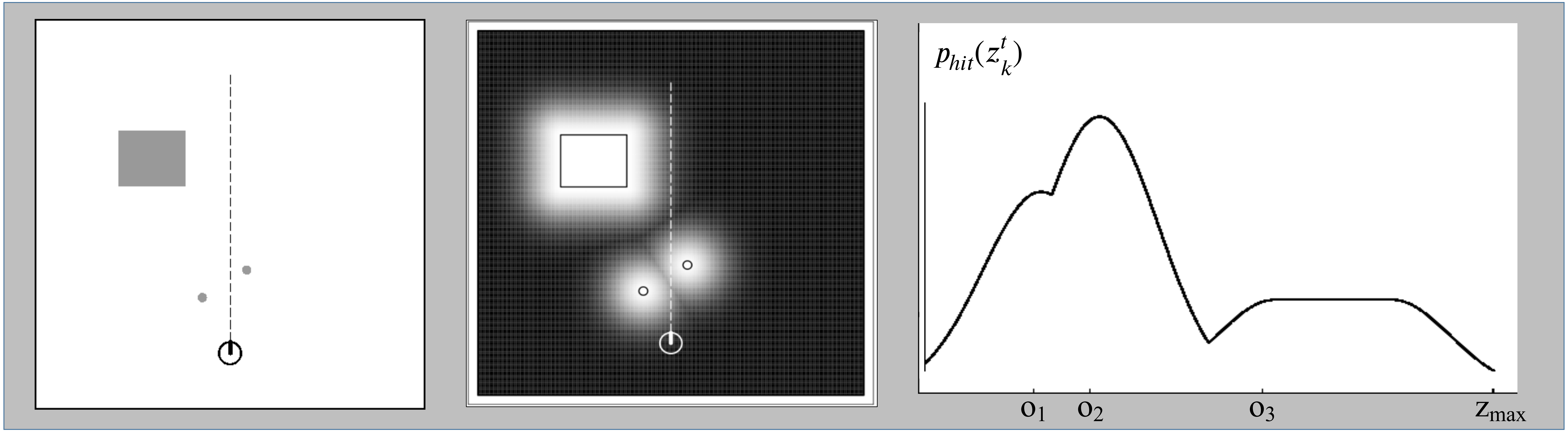
Measurement noise

- Modelled using Gaussians



Measurement noise

- Modelled using Gaussians
- In xy space, this involves finding the nearest obstacle in the map
- The probability of a sensor measurement is given by a Gaussian that depends on the Euclidean distance between measurement coordinates and nearest object in the map m



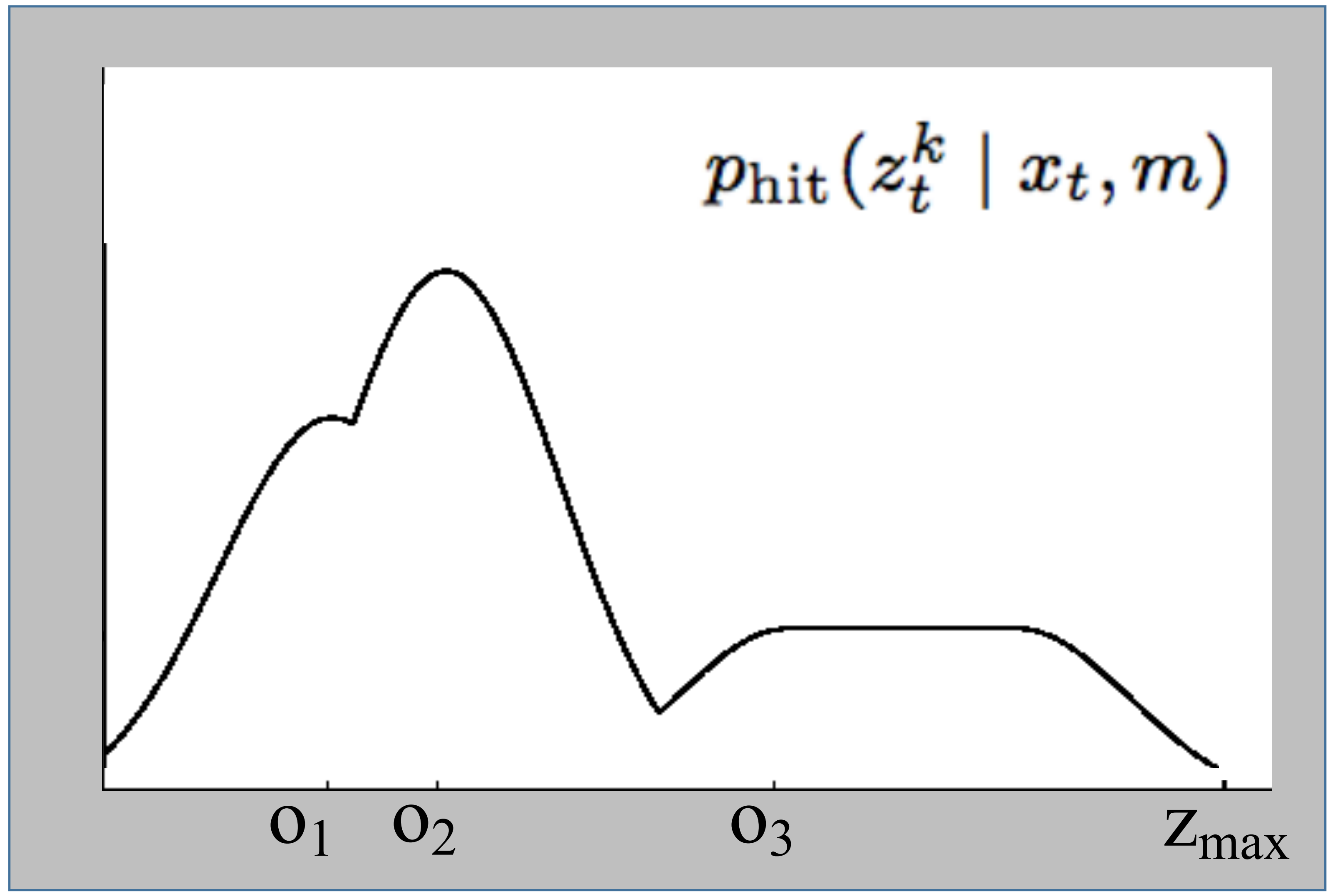
Likelihood fields of range finders

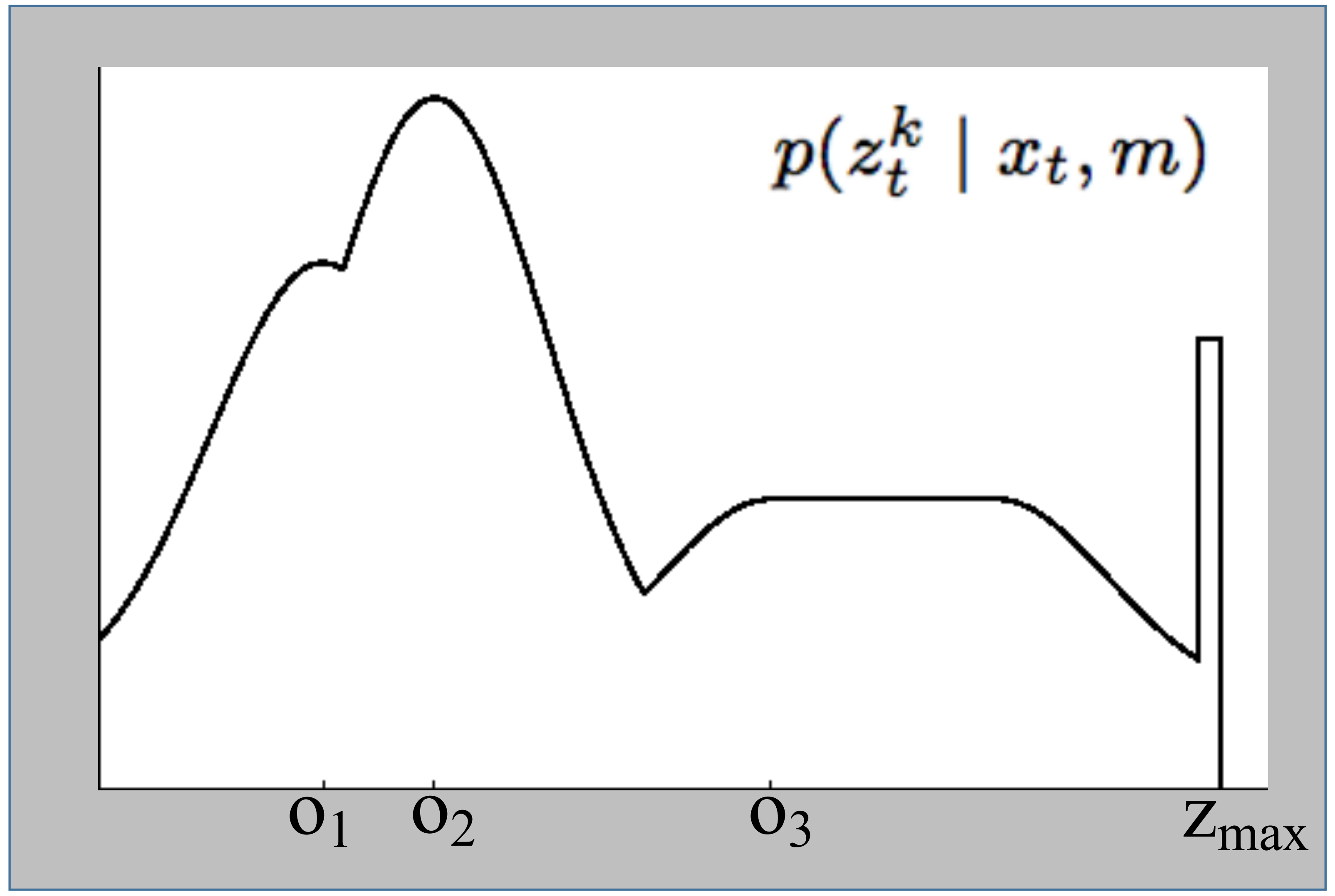
- Robot pose in the world frame: $x_t = (x, y, \theta)^T$
- Sensor measurement in the robot frame: $(x_{k,sens}, y_{k,sens}, \theta_{k,sens})$
- z_t^k hit/"end" points in the world frame

$$\begin{pmatrix} x_{z_t^k} \\ y_{z_t^k} \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & cos(\theta) \end{pmatrix} \begin{pmatrix} x_{z^{k,sens}} \\ y_{z^{k,sens}} \end{pmatrix} + z_t^k \begin{pmatrix} \cos(\theta + \theta^{k,sens}) \\ \sin(\theta + \theta^{k,sens}) \end{pmatrix}$$

Likelihood fields for range finders

- Assume independence between individual measurements
- **Three types of sources of noise and uncertainty**
  - Measurement noise
  - Failures
    - Max range readings are modeled by a point-mass distribution
  - Unexplained random measurements
    - Uniform distribution





Algorithm for likelihood fields

1. **Algorithm likelihood_field_range_finder_model** (z_t, x_t, m) :
2. $q = 1$
3. **for** $k = 1$ to K do
4. $x_{z_t^k} = x + x_{z_k,sens}\cos(\theta) - y_{z_k,sens}\sin(\theta) + z_k^t\cos(\theta + \theta_{k,sens})$
5. $y_{z_t^k} = y + y_{z_k,sens}\cos(\theta) + x_{z_k,sens}\sin(\theta) + z_k^t\sin(\theta + \theta_{k,sens})$
6. $dist = \min_{x',y'}\{\sqrt{(x_{z_k^t} - x')^2 + (y_{z_k^t} - y')^2} \mid \langle x', y' \rangle \text{ occupied in } m\}$
7. $q = q \cdot \left(z_{hit} \cdot f(dist;\; 0,\; \sigma_{hit}) + \frac{z_{rand}}{z_{max}} \right)$
8. **return** q

Transform sensor reading to world frame

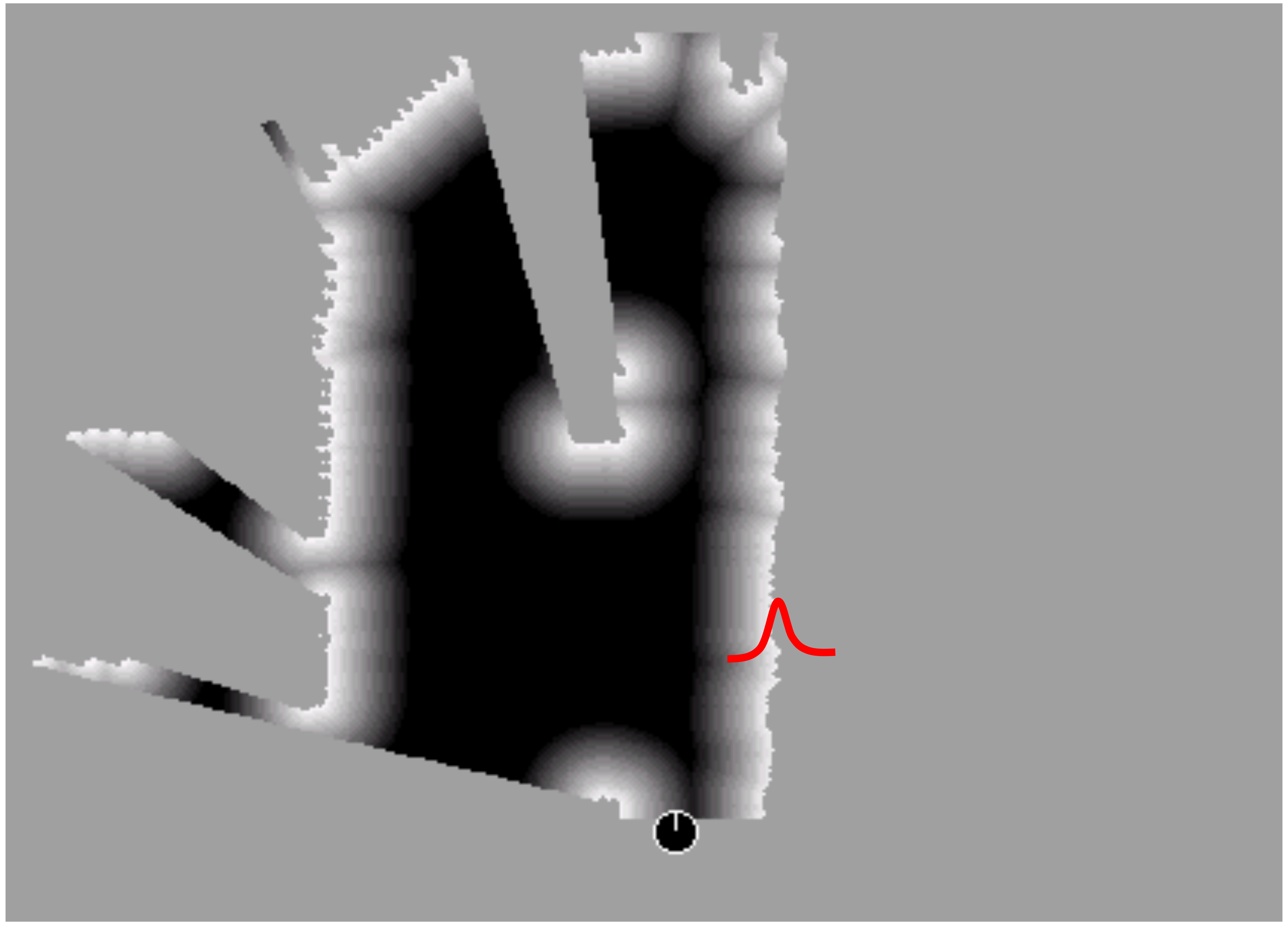
Find distance to closest object

Compute likelihood

Likelihood field from sensor data

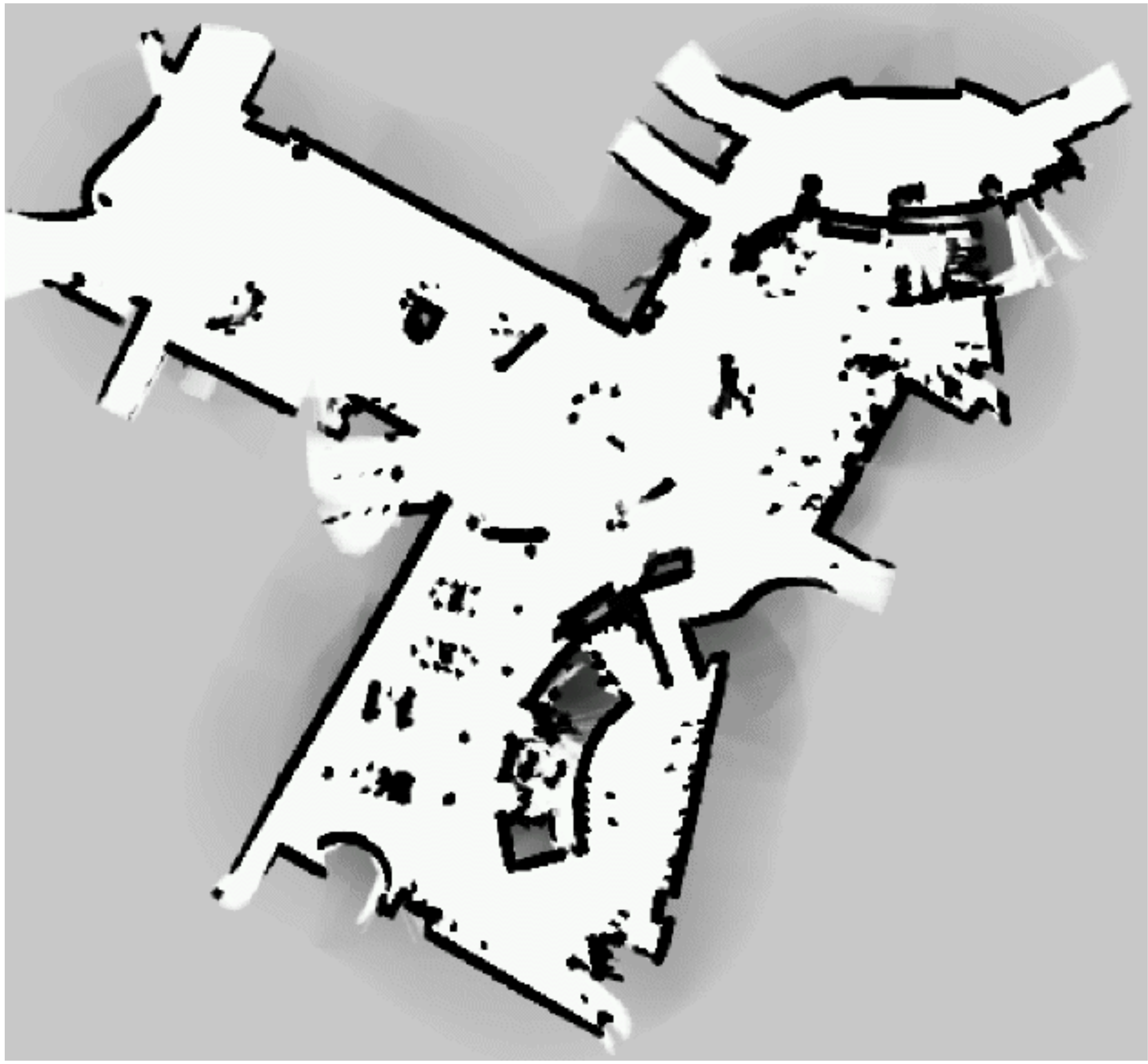
Sensor data projected into map

Corresponding likelihood function

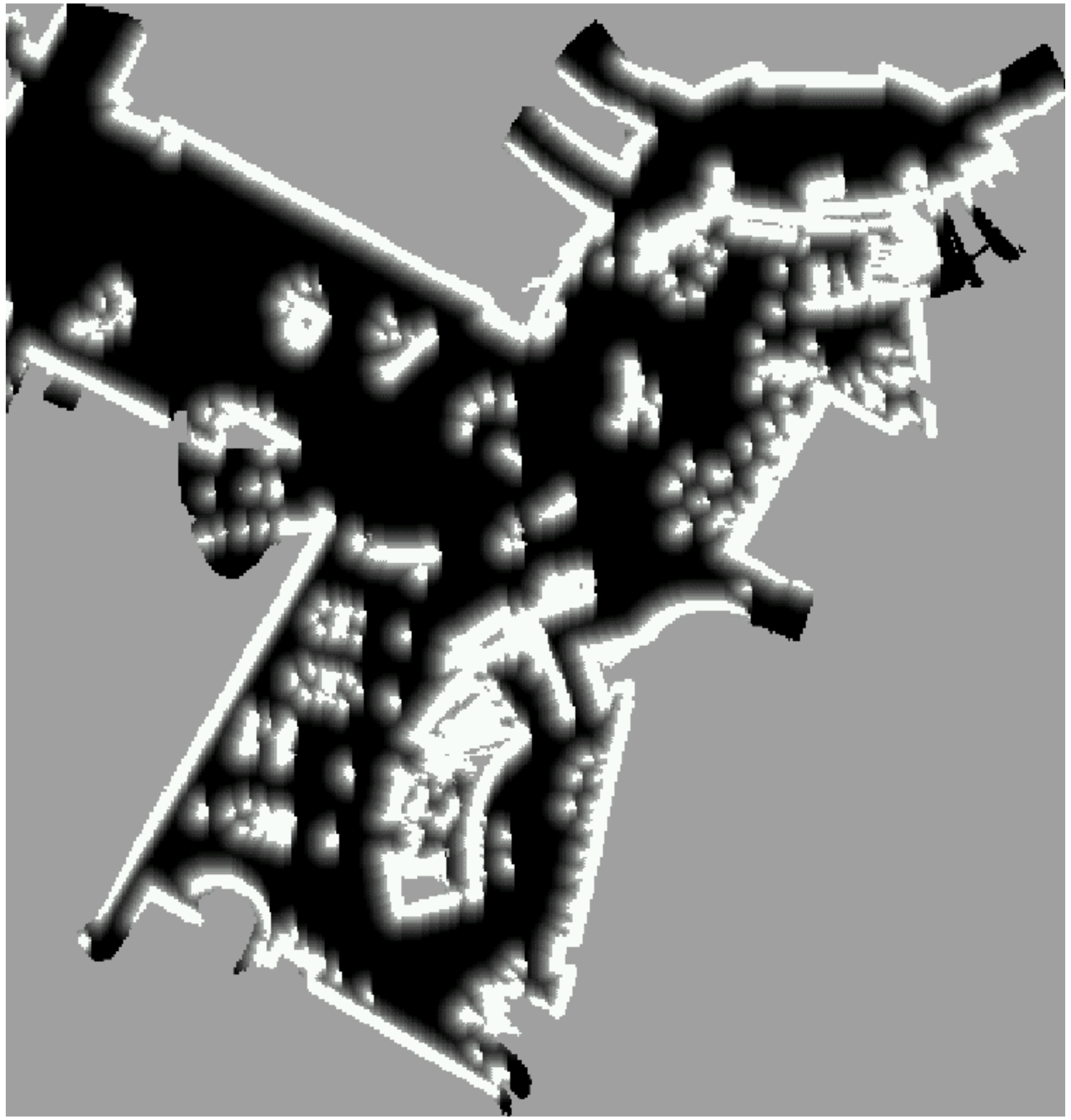


San Jose Tech Museum

Occupancy grid map



Likelihood field



Summary of likelihood fields

- **Advantages**
  - Highly efficient (computation in 2D instead of 3D)
  - Smooth w.r.t. small changes in robot position
- **Limitations**
  - Does not model people and other dynamics that might cause short readings
  - Ignores physical properties of beams

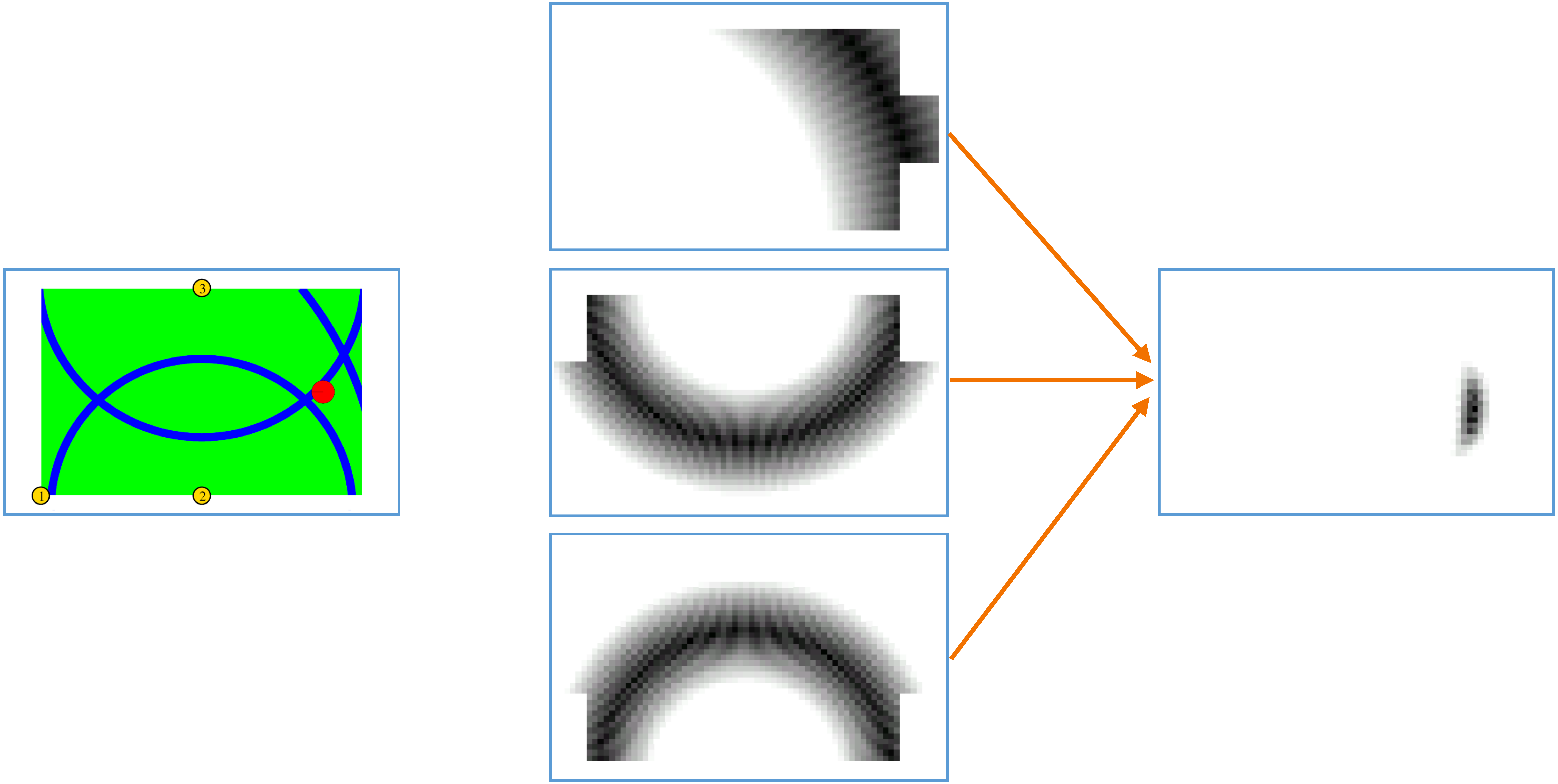
Feature-based models

Feature-based models

- **Extract features from dense raw measurements**
  - For range sensors: lines and corners
  - Often from cameras (edges, corners, distinct patterns, etc.)
- Feature extraction methods
- Features correspond to distinct physical objects in the real world and are often referred to as **landmarks**
  - Sensors output the range and/or bearing of the landmark w.r.t. the robot frame
  - Trilateration
  - Triangulation
  - Interference in the feature space can be more efficient

Trilateration using range measurements

Trilateration using range measurements



Summary of sensor model

- Robustness comes from explicitly modeling sensor uncertainty
- Measurement likelihood is given by “probabilistically comparing” the actual with the expected measurement
- Often, good models can be found by:
  - Determining a parametric model of noise-free measurements
  - Analyzing sources of noise
  - Adding adequate noise to parameters (mixed density functions)
  - Learning (and verifying) parameters by fitting model to data
- **It is extremely important to be aware of the underlying assumptions!**

References

1. Thrun, Sebastian, Wolfram Burgard, and Dieter Fox. Probabilistic robotics. MIT press, 2005.
2. http://ais.informatik.uni-freiburg.de/teaching/ss10/robotics/slides/07-sensor-models.pdf
3. http://www.cs.cmu.edu/~16831-f14/notes/F12/16831_lecture03_mtaylormshomin.pdf
4. Gaussian Distribution: https://www.asc.ohio-state.edu/gan.1/teaching/spring04/Chapter3.pdf
5. Gaussian Distribution: http://www2.stat.duke.edu/~rcs46/modern_bayes17/lecturesModernBayes17/lecture-3/03-normal-distribution.pdf