Measurements

6.1> Introduction

⇒ Measurement models describe the formation process by which sensor measurements are generated in the physical world.

⇒ Probablistic robotics explicitly models the noise in sensor measurements.

⇒ The Measurement Model is defined as a conditional probability distribution $P(z_t | x_t, m)$, where x_t is the robot pose, z_t is measurement at time t and m is the map of the environment.

⇒ In practice it is often impossible to model a sensor accurately, primarily for two reasons:

① Developing an accurate sensor model can be extremely time-consuming.

② An accurate model may require state variables that we might not know (such as the surface material)

→ When devising a probabilistic model, care has to be taken to capture the different types of uncertainties that may affect a sensor measurement.

⇒ Many sensors generate more than one numerical measurement value when queried.

↳ We will denote the number of such measurements values within a measurement z_t by K:

$$z_t = \{z_t^1, \dots z_t^{K_t}\}$$

{ We will use z_t^k to refer to an individual measurement (e.g. one range value) }

So,

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

Scanned by CamScanner

6.2> Map

⇒ A Map of the environment is a list of objects in the environment and their locations.

$$\boxed{m = \{m_1, m_2, \ldots\ldots m_N\}}$$

{Here N is total number of objects in the environment}

⇒ Maps are usually indexed in one of two ways, knows as feature-based and location-based.

① Feature-based map

→ n is the feature index.

→ The value of m_n contains properties of a feature & Cartesian location.

② Location-based map

→ Index n corresponds to a specific location.

→ In planar maps, it is common to denote a map element by $m_{x,y}$ instead of m_n, to make explicit that $m_{x,y}$ is the property of a specific world coordinate (x,y).

⇒ Feature representation makes it easier to adjust the position of an object e.g. as a result of additional sensing.

↳ For this reason, feature-based maps are popular in the robotic mapping field, where maps are constructed from sensor data.

⇒ A classical map representation is known as Occupancy grid map.

→ {Location based map}

↳ They assign to each x-y coordinate a binary occupancy value which specifies whether or not a location is occupied with an object.

Scanned by CamScanner

6.3> Beam Models of Range finders

{Among most popular Sensor in Robotics}

6.3.1> The basic measurement Algorithm

⇒ (i) Our model incorporate four types of measurement error, all of which are essential to making this model work.

1. Small measurement noise
2. Error due to unexpected object
3. Error due to failure to detect objects.
4. Random unexplained noise.

⇒ The desired model $p(z_t|x_t, m)$ is therefore a mixture of four densities, each of which corresponds to a particular type of error:

1. Correct range with local measurement Noise

⇒ Let us use z_t^{k*} to denote the "true" range of the object measured by z_t^k.

⇒ In location-based map, the range z_t^{k*} can be determined using ray casting. In feature based maps.

⇒ This noise is usually modeled by a narrow Gaussian with mean z_t^{k*} and standard deviation σ_{hit}.

⇒ We will denote the Gaussian by P_{hit}.

⇒ In practice, the values measured by the range sensor are limited to the interval $[0; z_{max}]$, where z_{max} denotes the maximum sensor range.

$$P_{hit}(z_t^k | x_t, m) = \begin{cases} \eta \cdot N(z_t^k; z_t^{k*}, \sigma_{hit}^2) & \forall\ 0 \le z_t^k \le z_{max} \\ 0 & \text{otherwise} \end{cases}$$

$$N(z_t^k; z_t^{k*}, \sigma_{hit}^2) = \frac{1}{\sqrt{2\pi\sigma_{hit}^2}} e^{-\frac{1}{2}\frac{(z_t^k - z_t^{k*})^2}{\sigma_{hit}^2}}$$

$$\eta = \left(\int_0^{z_{max}} N(z_t^k; z_t^{k*}, \sigma_{hit}^2)\, dz_t^k \right)^{-1}$$

2. Unexpected object

⇒ Environments of mobile robots are dynamic, whereas maps m are static.

⇒ So objects not contained in the map can cause range finders to produce surprisingly short ranges.

⇒ One way to deal with such objects is to treat them as part of the state vector and estimate their location.

↳ Another much simpler approach is to treat them as sensor noise.

⇒ Treated as sensor noise, unmodeled objects have the property that they cause ranges to be shorter than z_t^{k*}, not longer.

⇒ Mathematically, the probability of range measurements in such situations is described by an <u>exponential distribution</u>.

⇒ The parameter of this distribution, λ_{short}, is an intrinsic parameter of the measurement model.

⇒ According to the definition of an exponential distribution we obtain the following equation for $P_{short}(z_t^k|x_t,m)$:

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

$$\eta = \frac{1}{1 - e^{-\lambda_{short} z_t^{k*}}}$$

Scanned by CamScanner

3. Failures

⇒ A typical result of sensor failures are max-range measurements: the sensor returns its maximum allowable value Z_{max}.

⇒ Such event is quite frequent.

⇒ We will model this case with a point-mass distribution centered as Z_{max}

$$P_{max}(Z_t^K | x_t, m) = I(z = Z_{max}) = \begin{cases} 1 & \forall\ z = Z_{max} \\ 0 & \text{otherwise} \end{cases}$$

4. Random measurements

⇒ Finally, range finder occasionally produce entirely unexplained measurements.

⇒ To keep things simple such measurements will be modeled using a uniform distribution spread over the entire sensor measurement range $[0; Z_{max}]$:

$$P_{rand}(Z_t^K | x_t, m) = \begin{cases} \frac{1}{Z_{max}} & \text{if } 0 \leq Z_t^K \leq Z_{max} \\ 0 & \text{otherwise} \end{cases}$$

⇒ These four different distributions are now mixed by a weighted average, defined by the parameters Z_{hit}, Z_{short}, Z_{max} and Z_{rand}.

where, $Z_{hit} + Z_{short} + Z_{max} + Z_{rand} = 1$

$$P(Z_t^K \mid x_t, m) = \begin{pmatrix} Z_{hit} \\ Z_{short} \\ Z_{max} \\ Z_{rand} \end{pmatrix}^T \begin{pmatrix} P_{hit}(Z_t^K \mid x_t, m) \\ P_{short}(Z_t^K \mid x_t, m) \\ P_{max}(Z_t^K \mid x_t, m) \\ P_{rand}(Z_t^K \mid x_t, m) \end{pmatrix}$$

Algorithm beam_range_finder_model (z_t, x_t, m):

$q = 1$

for $k = 1$ to K do

Compute z_t^{k*} for the measurement z_t^k using ray casting.

$$P = Z_{hit} \cdot P_{hit}(Z_t^k \mid x_t, m) + Z_{short} \cdot P_{short}(Z_t^k \mid x_t, m) + Z_{max} \cdot P_{max}(Z_t^k \mid x_t, m) + Z_{rand} \cdot P_{rand}(Z_t^k \mid x_t, m)$$

$q = q \cdot p$

return q

$$\boxed{P(Z_t \mid x_t, m)}$$

6.3.2 ⇒

$\Theta = ($

⇒ C

⇒ A

⇒ A

⇒ A

6.3.2> Adjusting the Intrinsic Model parameter

$$\Theta = (z_{hit}, z_{short}, z_{max}, z_{rand}, \sigma_{hit}, \lambda_{short})$$

⇒ One way is to determine the intrinsic Parameters is to rely on data.

⇒ A perfectly acceptable way to set the intrinsic parameter Θ is by hand.

↳ Simply eyeball the resulting density until it agrees with your experience.

⇒ Another, more principled way, is to learn those parameters from actual data.

⇒ Algorithms that maximize the likelihood of data are known as maximum likelihood estimators or ML estimator in short.

Scanned by CamScanner

Algorithm learn intrinsic parameter (Z, X, m)

repeat until converge criterion satisfied

for all z_i in Z do

$$\eta = \left[P_{hit}(z_i \mid x_i, m) + P_{short}(z_i \mid x_i, m) + P_{max}(z_i \mid x_i, m) + P_{rand}(z_i \mid x_i, m)\right]^{-1}$$

calculate z_i^*

$$e_{i,hit} = \eta \, P_{hit}(z_i \mid x_i, m)$$

$$e_{i,short} = \eta \, P_{short}(z_i \mid x_i, m)$$

$$e_{i,max} = \eta \, P_{max}(z_i \mid x_i, m)$$

$$e_{i,rand} = \eta \, P_{rand}(z_i \mid x_i, m)$$

$$z_{hit} = |Z|^{-1} \sum_i e_{i,hit}$$

$$z_{short} = |Z|^{-1} \sum_i e_{i,short}$$

$$z_{max} = |Z|^{-1} \sum_i e_{i,max}$$

$$z_{rand} = |Z|^{-1} \sum_i e_{i,rand}$$

$$\sigma_{hit} = \sqrt{\frac{1}{\sum_i e_{i,hit}} \sum_i e_{i,hit} (z_i - z_i^*)^2}$$

$$\lambda_{short} = \frac{\sum_i e_{i,short}}{\sum_i e_{i,short} \, z_i}$$

return $\Theta = \{z_{hit}, z_{short}, z_{max}, z_{rand}, \sigma_{hit}, \lambda_{short}\}$

6.3.4> Practical Consideration

⇒ In practice, computing the densities of all sensor readings can be quite time consuming.

(laser range scanners often returns 100's of values per scan, at a rate of several scan per second.)

⇒ One typical approach to solve this problem is to incorporate only a small subset of all measurements. (equally spaced)

⇒ The main drain of computing time for beam-based model is the ray casting operation.

↳ This can be substantially reduced by precomputing the ray casting algorithm and storing the result in memory.

6.4 Likelyhood field for Range finder

6.4.1 Basic Algorithm

⇒ The beam based sensor model, suffers two major drawbacks:

- **Lak of Smoothness**

⇒ In cluttered environments with many obstacles, the distribution $p(z_t^k | x_t, m)$ can be very unsmooth in x_t.

⇒ This lak of smoothness has two problematic consequences:

1) Any approximate belief representation runs danger to miss the correct state, as nearby states might have drastically different posterior likelihoods.

2) Hill climbing methods for finding the most likely state are prone to local minima, due to the large number of local maxima in such unsmooth models.

- **Computational Complexity**

⇒ Pre-Computing the ranges over a discrete grid in pose space is also computationally expensive and requires substantial memory.

⇒ Otherwise ray casting is computationally expensive.

⇒ No[illegible]
the[illegible]

⇒ The
of
sp[illegible]

To
z_t^k
m[illegible]
Hi[illegible]
, c[illegible]

⇒ Lo[illegible]
at[illegible]

⇒ Now <u>likelihood field model</u>, which over comes these limitations.

- → This model lacks a reasonable physical explanation.
- → It is an "ad hoc" alogorithm that does not necessarily compute a Conditional Probability relative to any meaningful generative model of the physics of sensor.
- → However the approach works well in Practice.
  - → Results much smooth posteriors even in cluttered space.
  - → Computation is typically more efficient.

⇒ The Key idea is to first project the end points of a sensor scan z_t into the global coordinate space of the map.

→ To project an individual sensor measurement z_t^k into the global coordinate frame of the map m, we need to have global position the point from where sensor beam z_k originated, and where it points.

⇒ Let $x_t = (x, y, \theta)^T$ denote a robot pose at time t.

⇒ Let us denote the relative location of the sensor in the robot's fixed, local coordinate system by $(x_{k,sens}, y_{k,sens})^T$.

↳ An the angular orientation of the sensor beam relative to the robot's heading direction by $\theta_{k,sens}$.

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

⇒ If the range sensor takes on its maximum value, that is, $z_t^k = z_{max}$, these coordinates have no meaning in the physical world.

↳ The likelihood field measurement model simply discards max-range readings.

⇒ Similar to the beam model, we assume three types of sources of noise and uncertainty.

1. <u>Measurement Noise</u>

⇒ Noise arising from the measurement process is modeled using Gaussians.

⇒ In x-y space, this involves finding the nearest obstacle in the map.

⇒ Let 'dist' denote the Euclidean distance between the measurement coordinates $(x_{z_t^k} \; y_{z_t^k})^T$ and the nearest object in the map m.

⇒ Then the probability of a sensor measurement is given by a zero-centered Gaussian modeling the sensor noise:

$$P_{hit}(Z_t^K | x_t, m) = \epsilon_{\sigma_{hit}^2}(dist^2)$$

$\boxed{\mathcal{E}_b(a) = \frac{1}{\sqrt{2\pi b}} e^{-\frac{a^2}{2b}}}$ Normal distribution with zero mean and variance b

⇒ The density P_{hit} is now obtained by intersecting the likelihood field by the (and normalizing) sensor axis.

2. Failures

⇒ We assume that max-range readings has a distinct large likehood.

↳ As before, this is modeled by a point-mass distribution P_{max}.

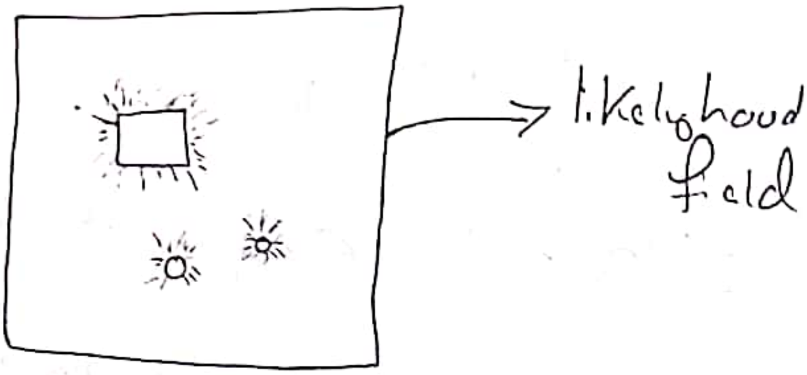
3. Random measurements

↳ finally, a uniform distribution P_{rand} is used to model random noise in perception.

⇒ Just as the beam-based sensor model, the desired probability $p(Z_t^K | x_t, m)$ integrates all three distributions.

$$Z_{hit} \cdot P_{hit} + Z_{rand} \cdot P_{rand} + Z_{max} \cdot P_{max}$$

⇒ The parameters can be adjusted by hand or learned using the ML estimator.



Algorithm likelyhood_field_range_finder_model(z_t, x_t, m):

$q = 1$

for all k do

  if $z_t^k \neq z_{max}$

  $$x_{z_t^k} = x + x_{k,sens}\cos\theta - y_{k,sens}\sin\theta + z_t^k\cos(\theta + \theta_{k,sens})$$

  $$y_{z_t^k} = y + y_{k,sens}\cos\theta + x_{k,sens}\sin\theta + z_t^k\sin(\theta + \theta_{k,sens})$$

  $$dist^2 = \min_{x',y'}\left\{(x_{z_t^k} - x')^2 + (y_{z_t^k} - y')^2 \,\middle|\, \langle x', y'\rangle \text{ occupied in } m\right\}$$

  $$q = q \cdot \left(z_{hit} \cdot prob(dist^2, \sigma_{hit}^2) + \frac{z_{random}}{z_{max}}\right)$$

return q

⇒ The most costly operation in algorithm likelihood field range finder model is the search for the nearest neighbor in the map (line 7).

↳ To speed up this search it is advantageous to pre-compute the likelihood field.

⇒ Of course, if a discrete grid is used, the result of the lookup is only approximate, in that it might return the wrong obstacle coordinates.

↳ However, the effect on the probability $P(z_t^k | x_t, m)$ is typically small even for moderately course grids.

6.4.2> <u>Extensions</u>

⇒ The likelyhood-field range finder model has three key disadvantages:

① It does not explicitly model people and other dynamics, that might cause short readings.

② It treats sensor as if they can see through wall.

③ Approach does not take map uncertainty into account.

↳ It cannot handle unexplored area.

Scanned by CamScanner

⇒ The basic algorithm likelyhood field range finder model can be ~~extracted~~ extended to diminish the effect of these limitations.

↳ For example, one might sort map occupancy values into three categories:

occupied, free and unknown

instead of just first two.

↳ When a sensor measurement z_t^k falls into the category unknown, its probability $P(z_t^k | x_t, m)$ is assumed to be the constant value $\frac{1}{z_{max}}$.

⇒ Likelihood fields over the visible space can also be defined for the most recent scan, which in fact defines a local map.

⇒ Leading implementations of the likelihood field technique rely on the extended symmetric algorithm.

6.5 Correlation-based Sensor Models

⇒ There exists a number of range sensor modes in the literature that measures correlations between a measurement and the map.

↳ A common technique is known as map matching.

Ability to transform scans into occupancy map

⇒ Typically, map making compiles small number of consecutive scans into local maps, denoted by m_{local}.

⇒ The sensor measurement model compares the local map m_{local} to the global map m, such that the more similar m and m_{local}, the larger $p(m_{local} | x_t, m)$

⇒ If the robot is at location x_t, we denote the grid cell in the local map by $m_{x,y,local}(x_t)$ the grid cell that corresponds to $(x,y)^T$ in global coordinates. in the same reference

⇒ Once both maps are in the same reference, they can be compared using the map correlation function, which is defined as follows:

$$\rho_{m, m_{local}, x_t} = \frac{\sum_{x,y} (m_{x,y} - \bar{m}) \cdot (m_{x,y,local}(x_t) - \bar{m})}{\sqrt{\sum_{x,y} (m_{x,y} - \bar{m})^2 \sum_{x,y} (m_{x,y,local}(x_t) - \bar{m})^2}}$$

⇒ Where,

$$\overline{m} = \frac{1}{2N} \sum_{x,y} (m_{x,y} + m_{x,y,local})$$

N = Number of elements in the overlap between the local and global map.

⇒ The correlation f_{m, m_{local}, x_t} Scales between ± 1.

$$P(m_{local} \mid x_t, m) = \max\{ f_{m, m_{local}, x_t}, 0 \}$$

⇒ Correlation based Sensor model is easy to compute.

- It does not yield smooth probabilities in the pose parameter x_t.
- One way is obtain smoothness is to convolve the map m with a Gaussian smoothness kernel and run map matching on this smoothed map.

⇒ A key advantage of map matching over likelihood fields is that it <u>explicitly considers the free-space in the scoring of two maps.</u>

6.6) Feature-based Sensor Models

6.6.1) Feature Extraction:

⇒ The Sensor models discussed thus far are all based on raw sensor measurements.

↳ An alternative approach is to extract features from the measurements.

⇒ If we denote the feature extractor as a function f, the features extracted from a range measurement are given by $f(z_t)$.

⇒ Most feature extractors extract a small number of features from high-dimensional sensor measurements.

⇒ A Key advantage of this approach is the enormous reduction of computational complexity.

⇒ For range sensors, it is common to identify lines, corners or local minima in range scans, which corresponds to walls, corners, or object such as tree trunks.

⇒ When camera are used for navigation, the processing of camera image falls into the realm of computer vision.

6.6.2) Landmark Measurements

⇒ In many robotics applications, features correspond to distinct objects in the physical world.

↳ In robotics it is common to call these physical objects landmarks, to indicate that they are being used for robot navigation.

⇒ The most common model for processing landmarks assumes that the sensor can measure the range and the bearing of the landmark relative to the robot's local coordinate frame.

⇒ The feature extractor may generate a signature.

↳ Signature may be a numerical value or multi-dimensional vector.

⇒ If we denote the range by r, the bearing ϕ, and the signature by s, the feature vector is given by a collection of triplets.

$$f(z_t) = \{ f_t^1, f_t^2, \dots \} = \left\{ \begin{pmatrix} r_t^1 \\ \phi_t^1 \\ s_t^1 \end{pmatrix}, \begin{pmatrix} r_t^2 \\ \phi_t^2 \\ s_t^2 \end{pmatrix} \dots \right\}$$

⇒ The number of features identified at each time step is variable.

⇒ Many probabilistic robotic algorithms assume conditional independence between features that is,

$$P(f(z_t) \mid x_t, m) = \prod_i P(r_t^i, \phi_t^i, s_t^i \mid x_t, m)$$

⇒ Let us devise a sensor model for features.

↳ landmark measurement models are usually defined only for feature based map.

⇒ We will model noise in landmark perception by independent Gaussian noise on the range, bearing, and the signature.

$$\begin{pmatrix} r_t^i \\ \phi_t^i \\ s_t^i \end{pmatrix} = \begin{pmatrix} \sqrt{(m_{j,x}-x)^2 + (m_{j,y}-y)^2} \\ \operatorname{atan2}(m_{j,y}-y,\, m_{j,x}-x) - \theta \\ s_j \end{pmatrix} + \begin{pmatrix} \varepsilon_{\sigma_r^2} \\ \varepsilon_{\sigma_\phi^2} \\ \varepsilon_{\sigma_s^2} \end{pmatrix}$$

⇒ where the i^{th} feature at time t corresponds to the j^{th} landmark in the map.

⇒ $\varepsilon_{\sigma_r^2}$, $\varepsilon_{\sigma_\phi^2}$ and $\varepsilon_{\sigma_s^2}$ are zero-mean gaussian error variables with variances σ_r^2, σ_ϕ^2 and σ_s^2 respectively.

5.6.3> Sensor model with Known Correspondence

⇒ To implement the measurement model, we need to define a variable that establishes correspondence between the feature f_t^i and the landmark m_j in the map.

→ We will denote this by c_t^i with $c_t^i \in \{1, \dots, N+1\}$; N is the number of landmarks in the map m.

→ If $c_t^i = j \leq N$, then the ith feature observed at time t corresponds to the jth landmark in the map.

→ If $c_t^i = N+1$, here feature observation does not correspond to any feature in the map m.

Algorithm landmark_model_Known correspondence (f_t^i, c_t^i, x_t, m):

$j = c_t^i$

$\hat{r} = \sqrt{(m_{j,x} - x)^2 + (m_{j,y} - y)^2}$

$\hat{\phi} = \mathrm{atan2}(m_{j,y} - y,\ m_{j,x} - x)$

$q = \mathrm{prob}(r_t^i - \hat{r}, \sigma_r^2) \cdot \mathrm{prob}(\phi_t^i - \hat{\phi}, \sigma_\phi^2) \cdot \mathrm{prob}(s_t^i - s_j, \sigma_s^2)$

return q

5.6.4 Sampling Poses

⇒ Sometime it is desirable to sample robot pose x_t that correspond to a measurement f_t^i with feature identity c_t^i.

Algorithm sample_landmark_model_known_correspondance(f_t^i, c_t^i, m):

$j = c_t^i$

$\hat{\gamma} = \text{rand}(0, 2\pi)$

$\hat{r} = r_t^i + \text{sample}(\sigma_r^2)$

$\hat{\phi} = \phi_t^i + \text{sample}(\sigma_\phi^2)$

$x = m_{j,x} + \hat{r} \cos \hat{\gamma}$

$y = m_{j,y} + \hat{r} \sin \hat{\gamma}$

$\theta = \hat{\gamma} - \pi - \hat{\phi}$

return $(x, y, \theta)^T$

⇒ With the advent of fast computers, features have gradually lost importance in the field of robotics.