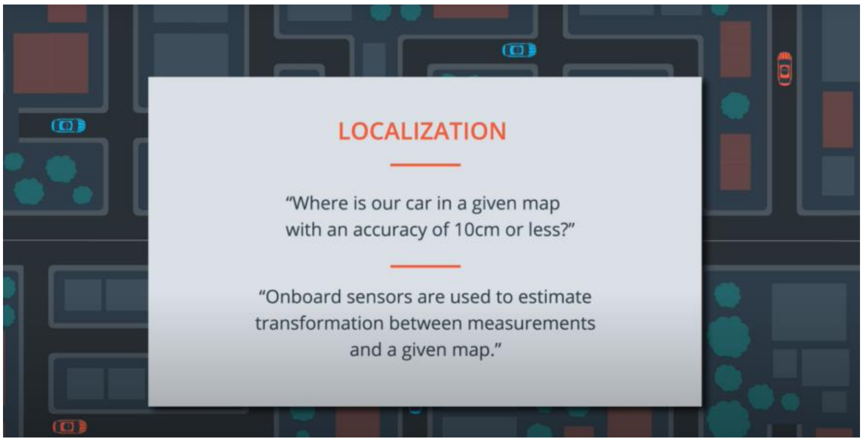
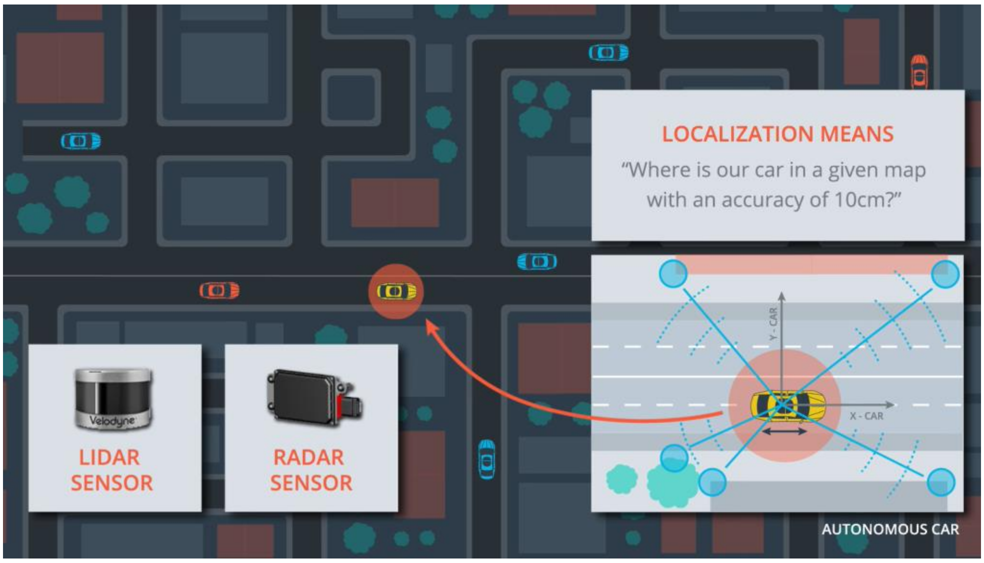
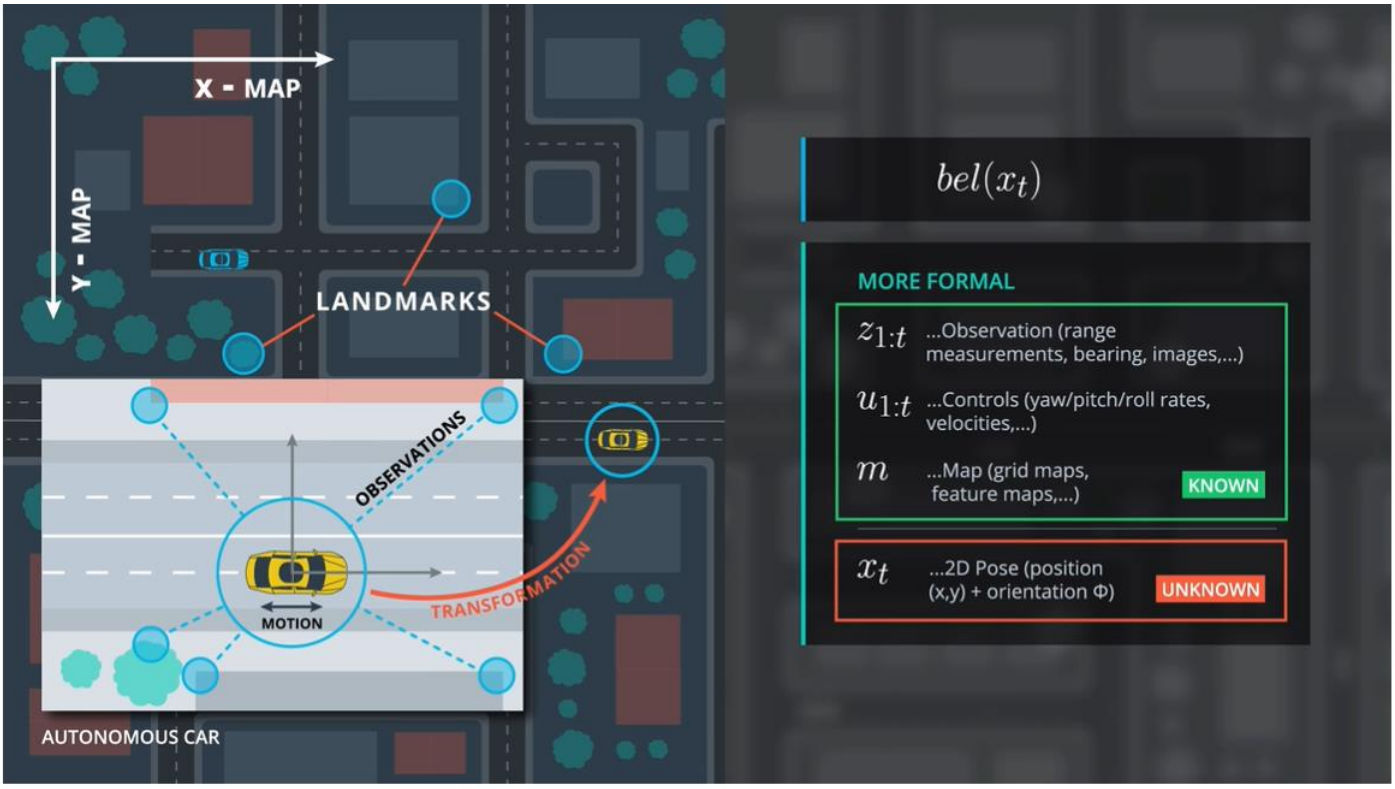




SLAM, or Simultaneous Localization and Mapping, that does not need a good map prior to beginning

Markov Localization or Bayes Filter for Localization is a generalized filter for localization and all other localization approaches are realizations of this approach, as we'll discuss later on. By learning how to derive and implement (coding exercises) this filter we develop intuition and methods that will help us solve any vehicle localization task, including implementation of a particle filter. We don't know exactly where our vehicle is at any given time, but can approximate it's location. As such, we generally think of our vehicle location as a probability distribution, each time we move, our distribution becomes more diffuse (wider). We pass our variables (map data, observation data, and control data) into the filter to concentrate (narrow) this distribution, at each time step. Each state prior to applying the filter represents our prior and the narrowed distribution represents our Bayes' posterior.



Formal Definition of Variables

$z_{1:t}$ represents the observation vector from time 0 to t (range measurements, bearing, images, etc.).

$u_{1:t}$ represents the control vector from time 0 to t (yaw/pitch/roll rates and velocities).

m represents the map (grid maps, feature maps, landmarks)

x_t represents the pose (position (x,y) + orientation θ)

bel(xt) → belief

Given the map, the control elements of the car, and the observations, what is the definition of the posterior distribution for the state x at time t?

(A)

$$bel(x_t) = p(x_t | z_t, m, u_t)$$

(B)

$$bel(x_t) = p(x_t | z_{1:t}, u_{1:t})$$

(C)

$$bel(x_t) = p(x_t, m_t | z_{1:t}, u_{1:t})$$

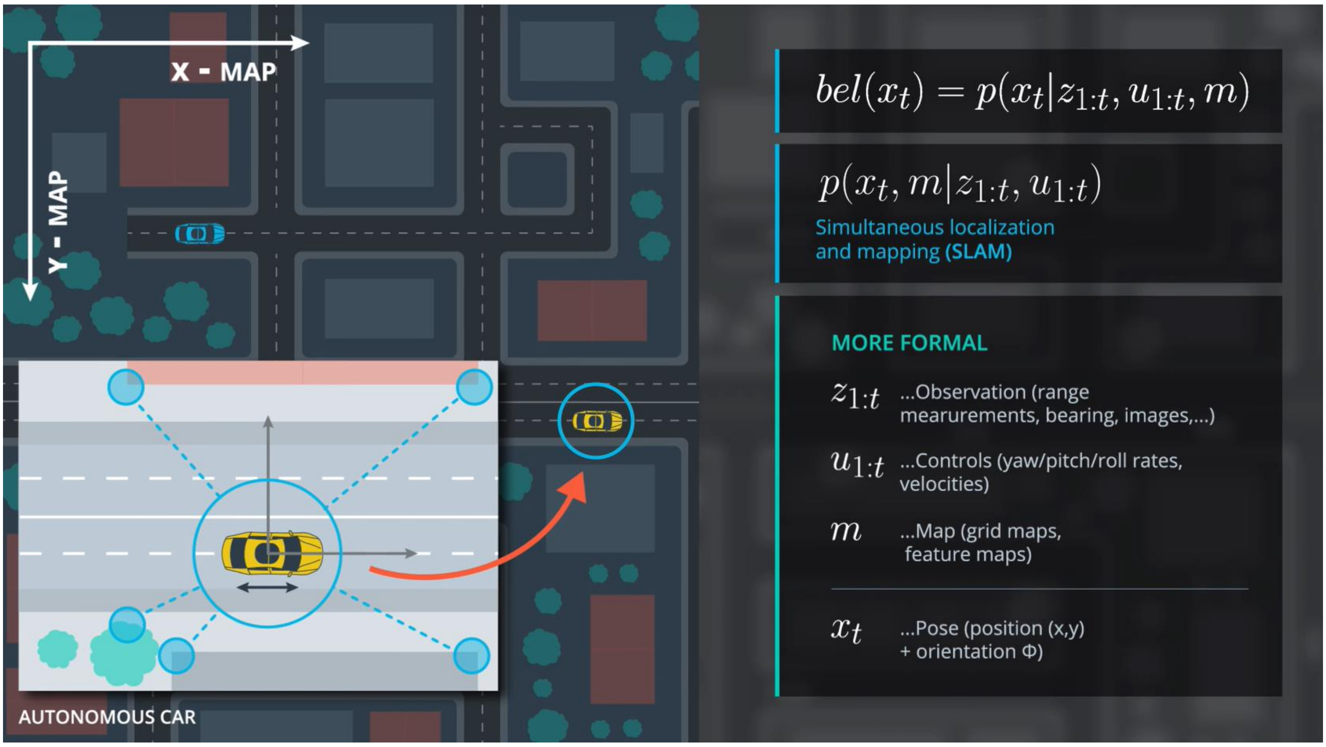
(D)

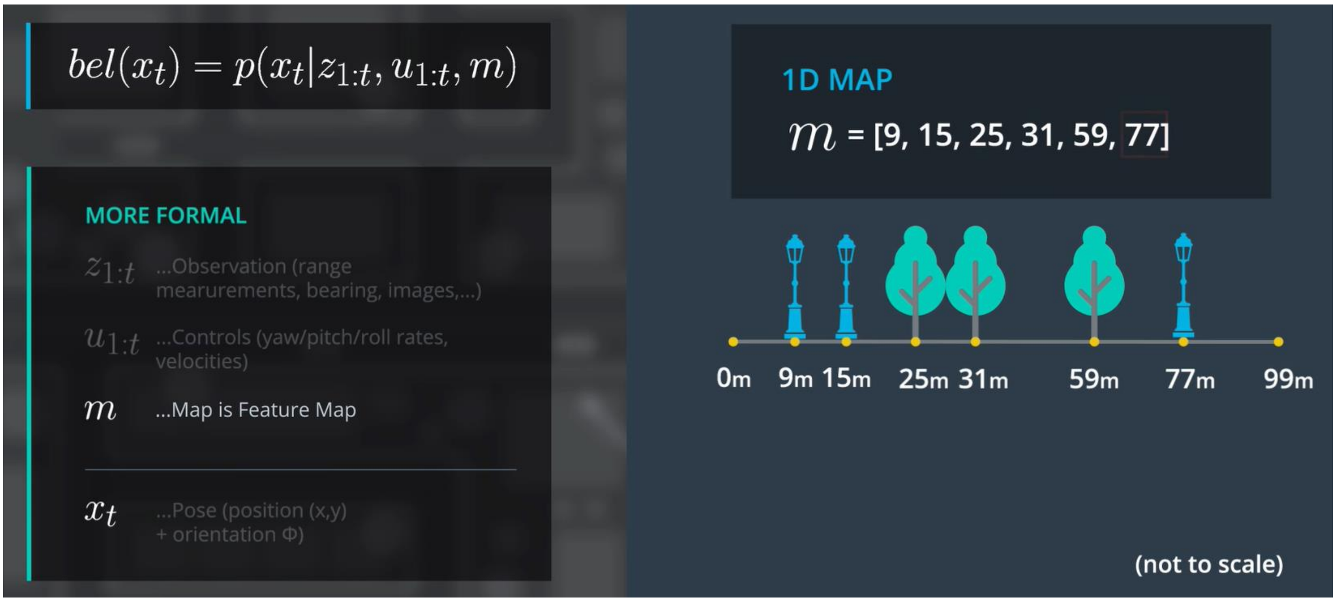
$$bel(x_t) = p(x_t | z_{1:t}, u_{1:t}, m)$$

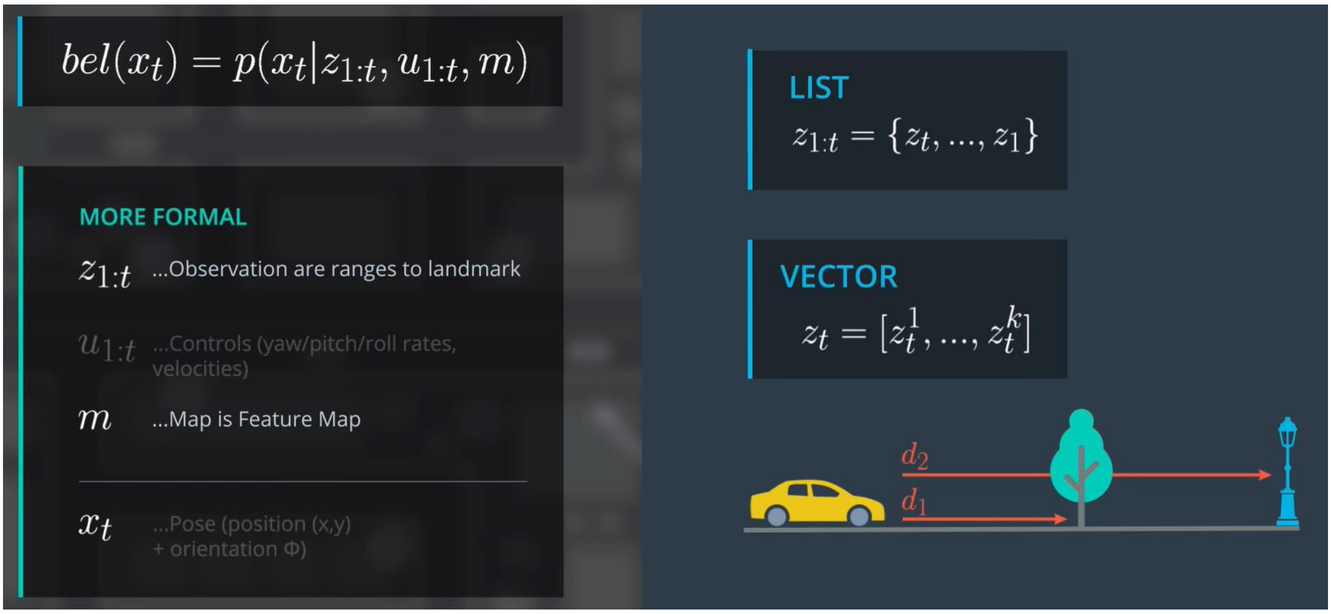
QUIZ QUESTION

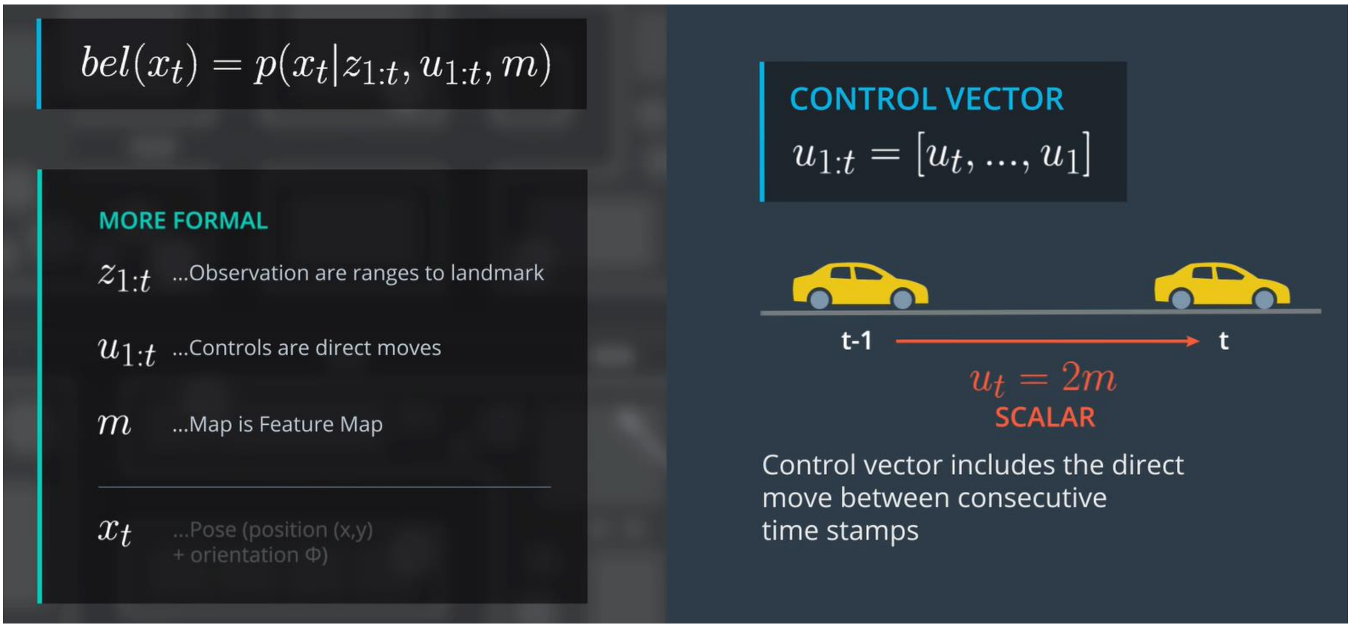
Given the map, the control elements of the car, and the observations, what is the definition of the posterior distribution for the state x at time t?

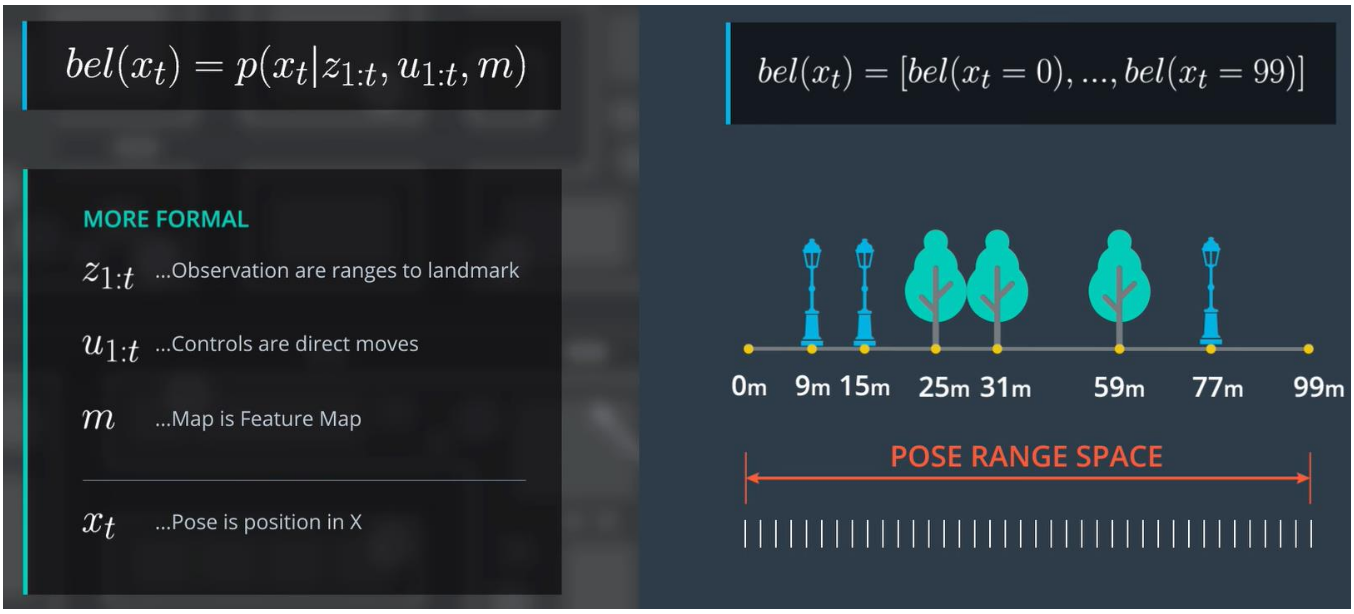
- (A)
- (B)
- (C)
- ✓ (D)




$bel(x_t) = p(x_t|z_{1:t}, u_{1:t}, m)$
MORE FORMAL
$z_{1:t}$ ...Observation (range mearurements, bearing, images,...)
$u_{1:t}$ ...Controls (yaw/pitch/roll rates, velocities)
m ...Map is Feature Map
x_t ...Pose (position (x,y) + orientation Φ)
1D MAP
m = [9, 15, 25, 31, 59, 77]
0m 9m 15m 25m 31m 59m 77m 99m
(not to scale)


$bel(x_t) = p(x_t|z_{1:t}, u_{1:t}, m)$
MORE FORMAL
$z_{1:t}$ ...Observation are ranges to landmark
$u_{1:t}$ ...Controls (yaw/pitch/roll rates, velocities)
m ...Map is Feature Map
x_t ...Pose (position (x,y) + orientation Φ)
LIST
$z_{1:t} = \{z_t, ..., z_1\}$
VECTOR
$z_t = [z_t^1, ..., z_t^k]$
d_2
d_1


$bel(x_t) = p(x_t|z_{1:t}, u_{1:t}, m)$
MORE FORMAL
$z_{1:t}$ ...Observation are ranges to landmark
$u_{1:t}$ ...Controls are direct moves
m ...Map is Feature Map
x_t ...Pose (position (x,y) + orientation Φ)
CONTROL VECTOR
$u_{1:t} = [u_t, ..., u_1]$
t-1
t
$u_t = 2m$
SCALAR
Control vector includes the direct move between consecutive time stamps



Recall that Bayes' Rule enables us to determine the conditional probability of a state given evidence P(a|b) by relating it to the conditional probability of the evidence given the state P(b|a) in the form of:

$$P(a) * P(b|a) = P(b) * P(a|b)$$

which can be rearranged to:

$$P(a|b) = \frac{P(b|a)\, P(a)}{P(b)}$$

In other words the probability of state a, given evidence b, is the probability of evidence b, given state a, multiplied by the probability of state a, normalized by the total probability of b over all states.

Bayes' Rule Applied

Let's say we have two bags of marbles, bag 1 and bag 2, filled with two types of marbles, red and blue. Bag 1 contains 10 blue marbles and 30 red marbles, whereas bag 2 contains 20 of each color marble.

If a friend were to choose a bag at random and then a marble at random, from that bag, how can we determine the probability that that marble came from a specific bag? You guessed it - Bayes' Rule!

In this scenario, our friend produces a red marble, in that case, what is the probability that the marble came from bag 1? Rewriting this in terms of Bayes' Rule, our solution becomes:

$$P(Bag1|Red) = \frac{P(Red|Bag1)\, P(Bag1)}{P(Red)}$$

QUESTION 1 OF 4

What is the prior probability of choosing bag 1? This is the term P(Bag1) .

- 0.75
- ✓ 0.50

QUESTION 2 OF 4

What is the probability of choosing a red marble from bag 1? This is our likelihood term P(Red|Bag1) .

- 0.50
- 0.30
- 0.25
- ✓ 0.75

QUESTION 3 OF 4

What is the total probability of choosing a red marble? This is our normalization term (denominator) P(Red) .

- 0.50
- 1.25
- ✓ 0.625
- 0.75

QUESTION 4 OF 4

Now, putting everything together, using the formula for Bayes' Rule, what is our posterior probability of the red marble originating from bag 1? This is our posterior term P(Bag1 | Red) ?

- 0.815
- 0.625
- 0.375
- ✓ 0.60

the lesson continues, but the generalized form Bayes' Filter for Localization is shown below. You may recognize this as being similar to a Kalman filter. In fact, many localization filters, including the Kalman filter are special cases of Bayes' Filter.

Remember the general form for Bayes' Rule:

$$P(a|b) = \frac{P(b|a)\,P(a)}{P(b)}$$

With respect to localization, these terms are:

1. $P(location|observation)$: This is P(a | b), the **normalized** probability of a position given an observation (posterior).
2. $P(observation|location)$: This is P(b | a), the probability of an observation given a position (likelihood)
3. $P(location)$: This is P(a), the prior probability of a position
4. $P(observation)$: This is P(b), the total probability of an observation

Initialize Belief State

To help develop an intuition for this filter and prepare for later coding exercises, let's walk through the process of initializing our prior belief state. That is, what values should our initial belief state take for each possible position? Let's say we have a 1D map extending from 0 to 25 meters. We have landmarks at x = 5.0, 10.0, and 20.0 meters, with position standard deviation of 1.0 meter. If we know that our car's initial position is at one of these three landmarks, how should we define our initial belief state?

Since we know that we are parked next to a landmark, we can set our probability of being next to a landmark as 1.0. Accounting for a position precision of +/- 1.0 meters, this places our car at an initial position in the range **[4, 6]** (5 +/- 1), **[9, 11]** (10 +/- 1), or **[19, 21]** (20 +/- 1). All other positions, not within 1.0 meter of a landmark, are initialized to 0. We normalize these values to a total probability of 1.0 by dividing by the total number of positions that are potentially occupied. In this case, that is 9 positions, 3 for each landmark (the landmark position and one position on either side). This gives us a value of 1.11E-01 for positions +/- 1 from our landmarks (1.0/9). So, our initial belief state is:

```
{0, 0, 0, 1.11E-01, 1.11E-01, 1.11E-01, 0, 0, 1.11E-01, 1.11E-01, 1.11E-01, 0, 0, 0,
0, 0, 0, 0, 1.11E-01, 1.11E-01, 1.11E-01, 0, 0, 0, 0}
```

```
// TODO: Complete the initialize_priors function
vector<float> initialize_priors(int map_size, vector<float> landmark_positions,
                                float position_stdev) {

  // initialize priors assuming vehicle at landmark +/- 1.0 meters position stdev
  float prior = 1 / (landmark_positions.size() * (1 + 2 * position_stdev));
  // set all priors to 0.0
  vector<float> priors(map_size, 0.0);

  // TODO: YOUR CODE HERE
  for(int i = 0; i < priors.size(); i++)
      for(auto j : landmark_positions)
        if(i >= j - position_stdev && i <= j + position_stdev)
            priors[i] = prior;

  return priors;
}
```

```
0
0
0.111111
0.111111
0.111111
0
0
0.111111
0.111111
0.111111
0
0
0
0
```

Quiz

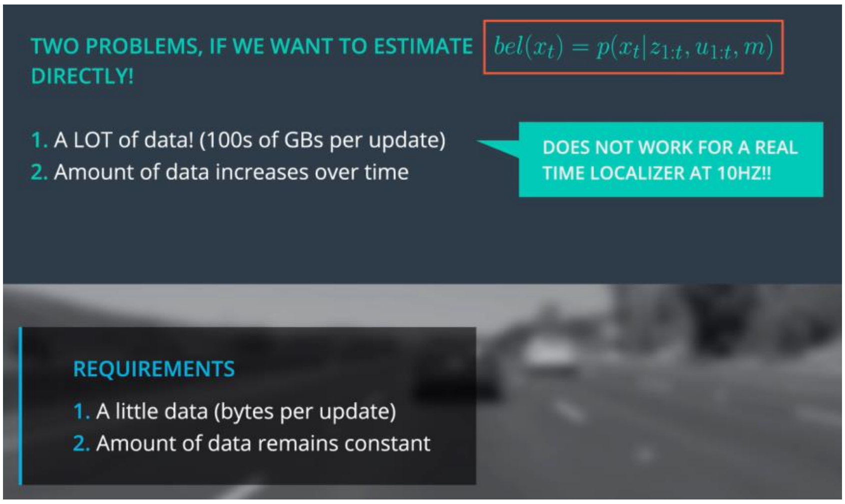
About how much data is in $z_{1:t}$?

QUIZ QUESTION

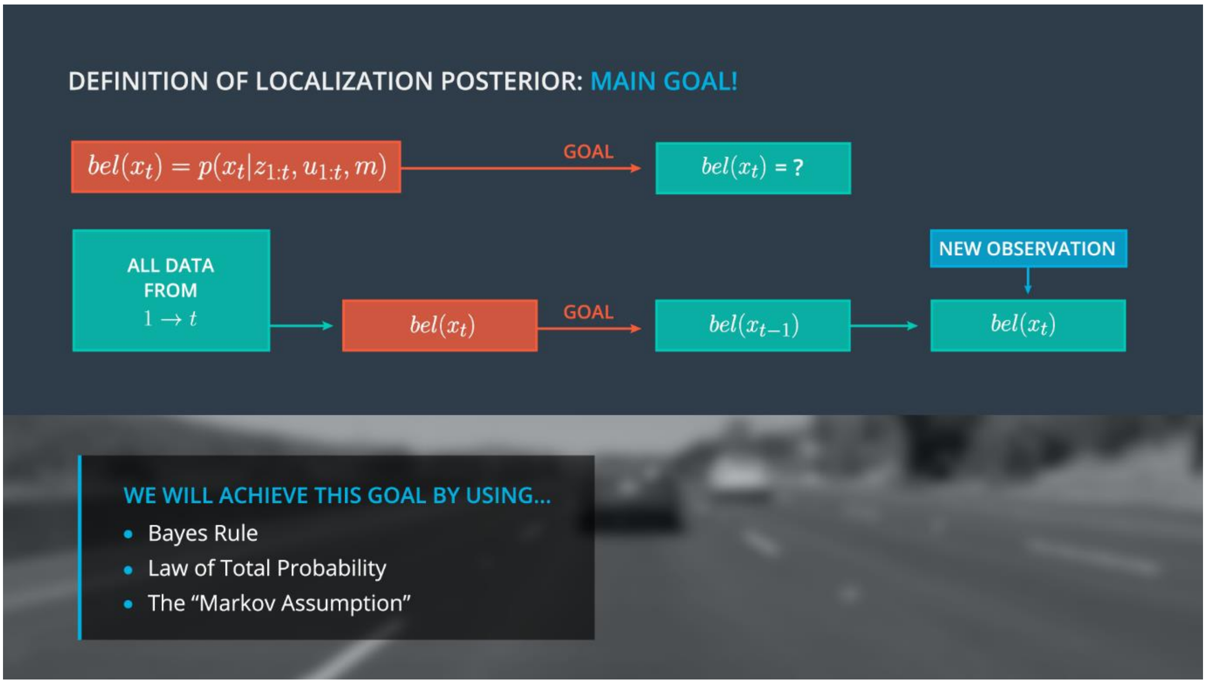
Remember:

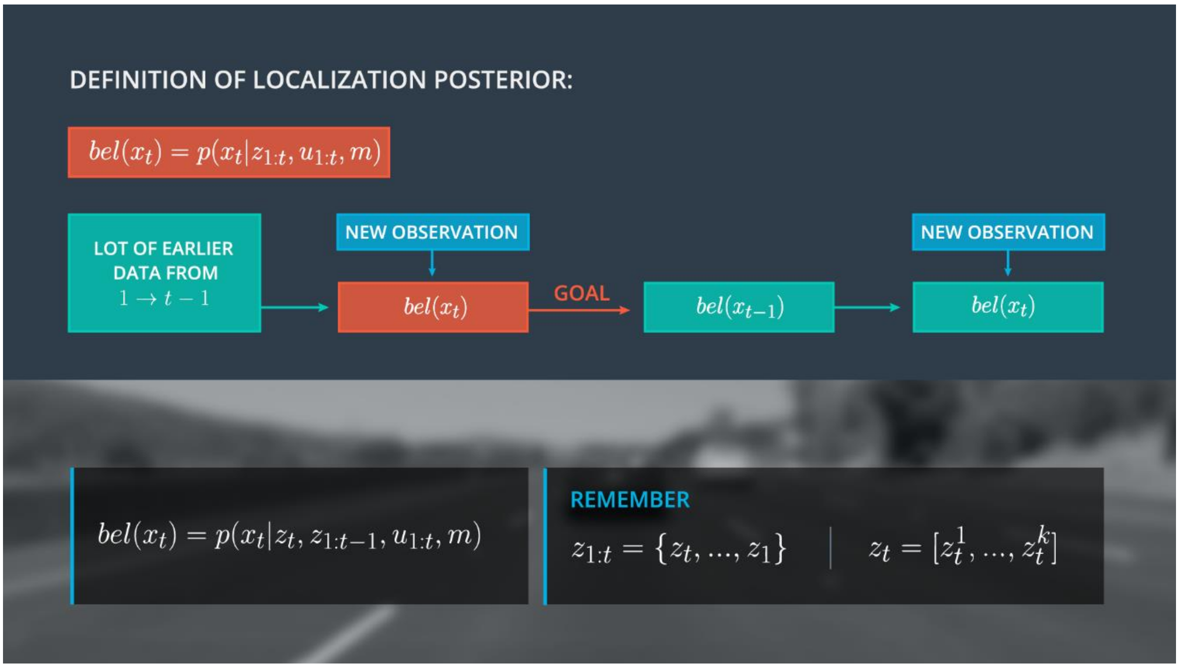
- The car has driven for 6 hours
- LIDAR refreshes 10 times per seconds (10 Hertz)
- LIDAR sends 100,000 data points per observation
- Each of the 100,000 observations contains 5 pieces of data
- Each piece of data requires 4 bytes

- ○ 4 kB
- ○ 430 kB
- ○ 4 MB
- ○ 430 MB
- ○ 4 GB
- ✓ 430 GB


TWO PROBLEMS, IF WE WANT TO ESTIMATE $bel(x_t) = p(x_t|z_{1:t}, u_{1:t}, m)$ DIRECTLY!
1. A LOT of data! (100s of GBs per update)
2. Amount of data increases over time
DOES NOT WORK FOR A REAL TIME LOCALIZER AT 10HZ!!
REQUIREMENTS
1. A little data (bytes per update)
2. Amount of data remains constant


$bel(x_t) = p(x_t|z_t, z_{1:t-1}, u_{1:t}, m)$
BAYES RULE
$p(x_t|z_t, \boxed{z_{1:t-1}, u_{1:t}, m})$ = ??
REMEMBER
LIKELIHOOD
PRIOR
$p(a|b) = \frac{p(b|a) \times p(a)}{p(b)}$
POSTERIOR
NORMALIZING CONSTANT


DEFINITION OF LOCALIZATION POSTERIOR: MAIN GOAL!
$bel(x_t) = p(x_t|z_{1:t}, u_{1:t}, m)$
GOAL
$bel(x_t)$ = ?
ALL DATA FROM $1 \rightarrow t$
$bel(x_t)$
GOAL
$bel(x_{t-1})$
NEW OBSERVATION
$bel(x_t)$
WE WILL ACHIEVE THIS GOAL BY USING...
Bayes Rule
Law of Total Probability
The "Markov Assumption"



Bayes Rule

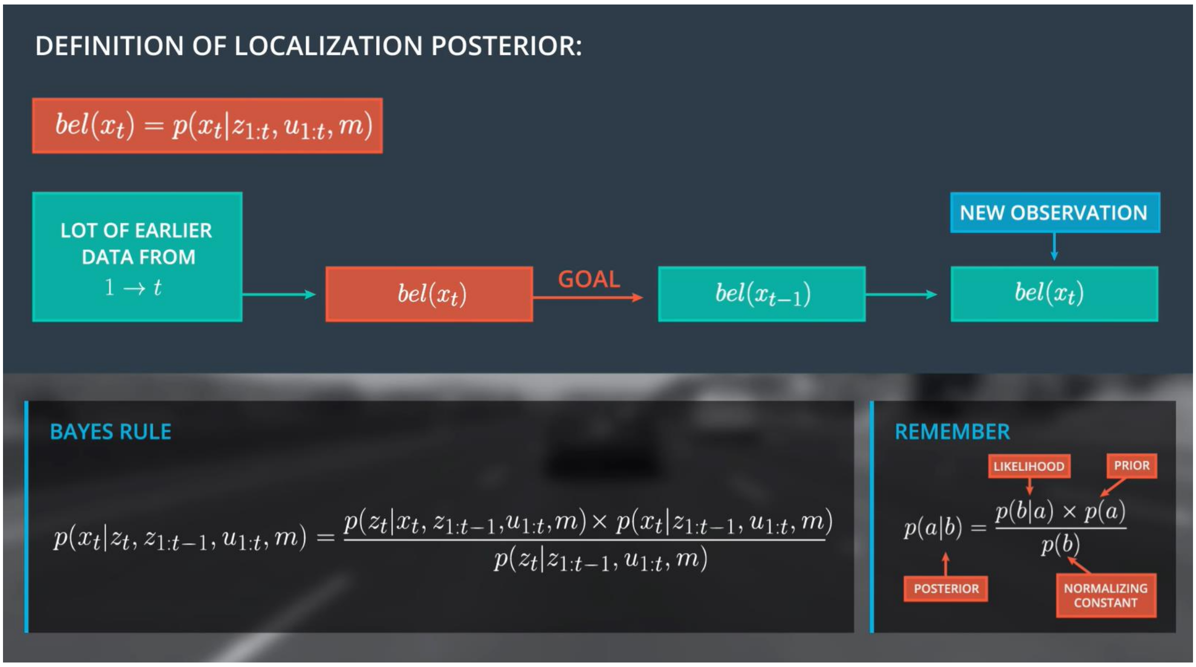
$$P(a \mid b) = \frac{P(b \mid a)\, P(a)}{P(b)}$$

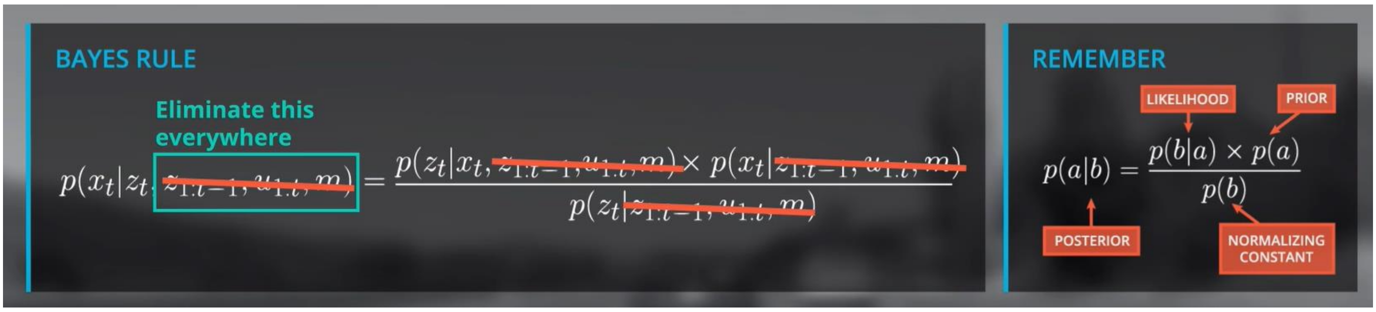
Quiz

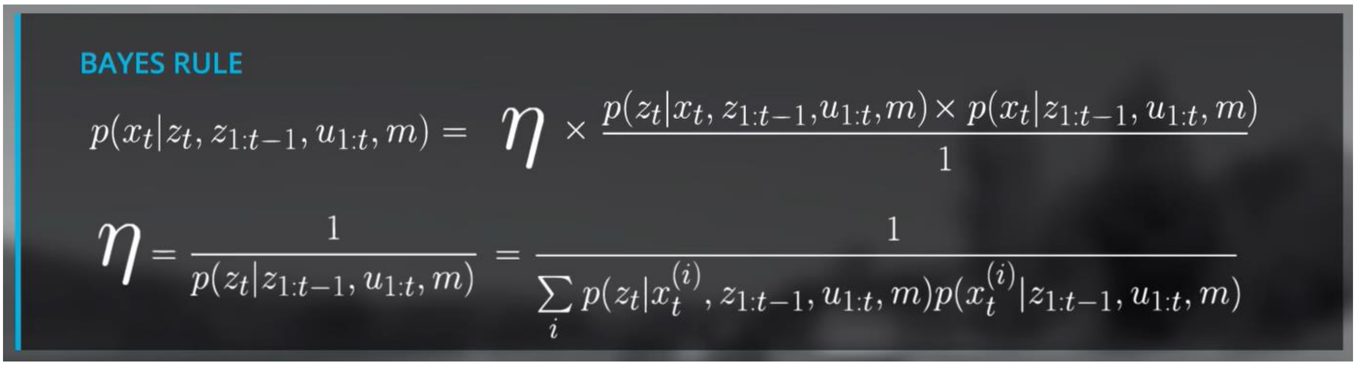
Please apply Bayes Rule to determine the right side of Bayes rule, where the posterior, $P(a|b)$, is $p(x_t|z_t, z_{1:t-1}, u_{1:t}, m)$

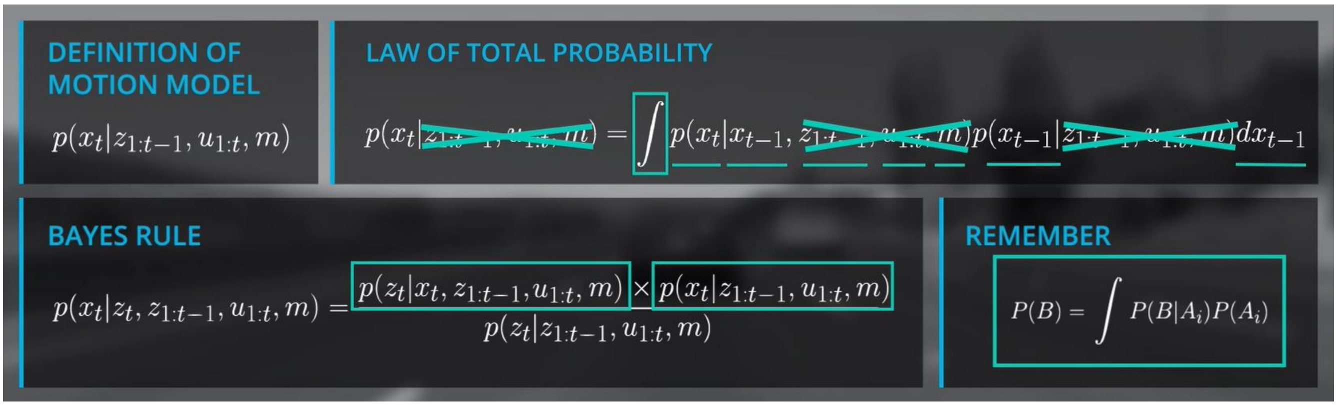
(B)

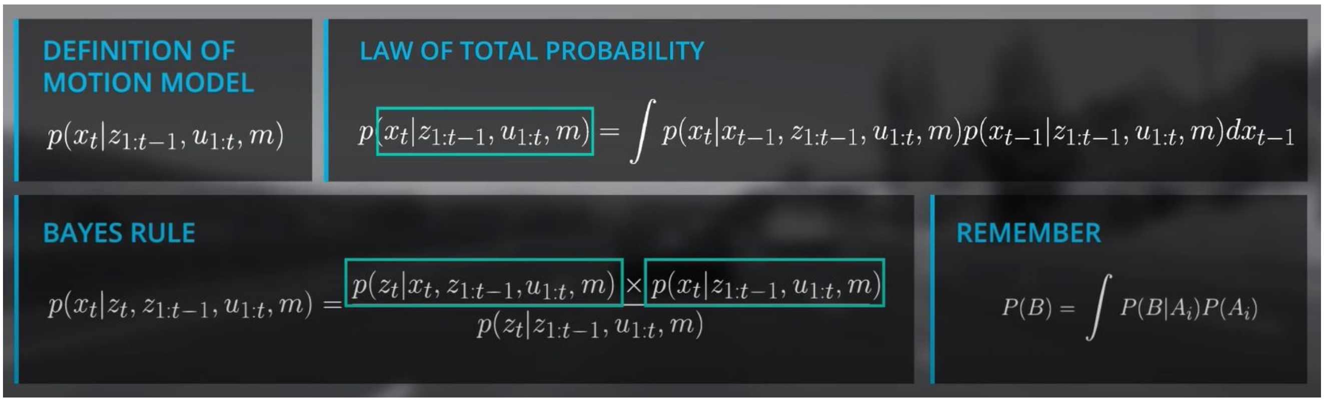
$$\frac{p(z_t|x_t, z_{1:t-1}, u_{1:t}, m)p(x_t|z_{1:t-1}, u_{1:t}, m)}{p(z_t|z_{1:t-1}, u_{1:t}, m)}$$





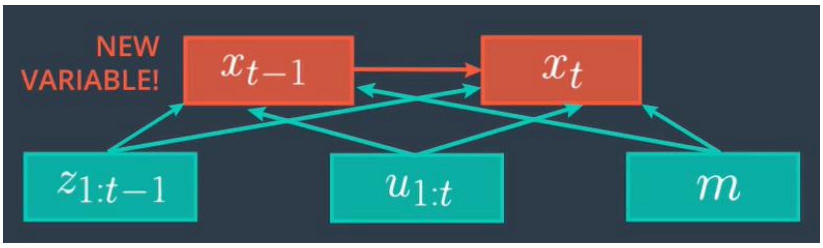






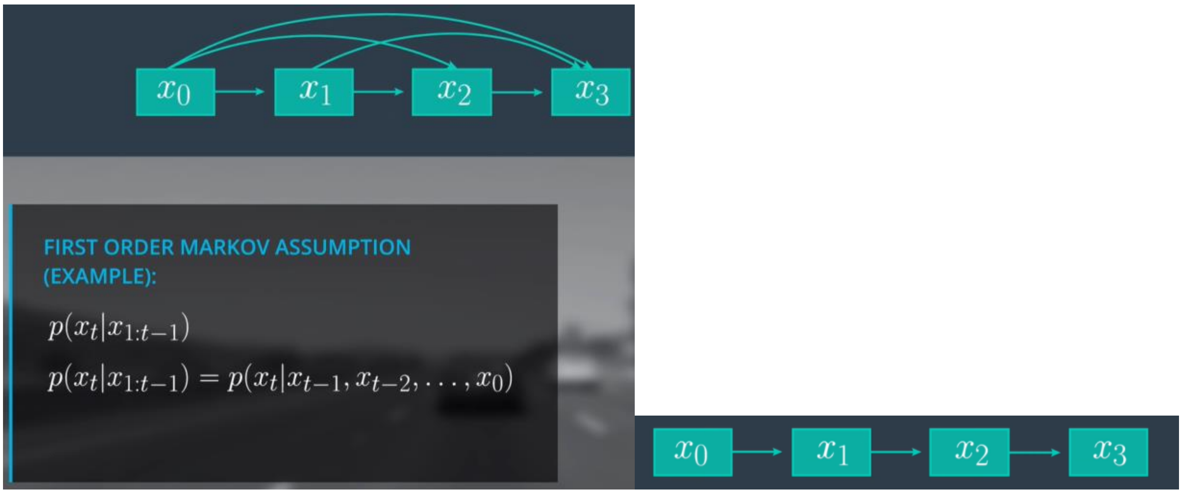
Law of Total Probability

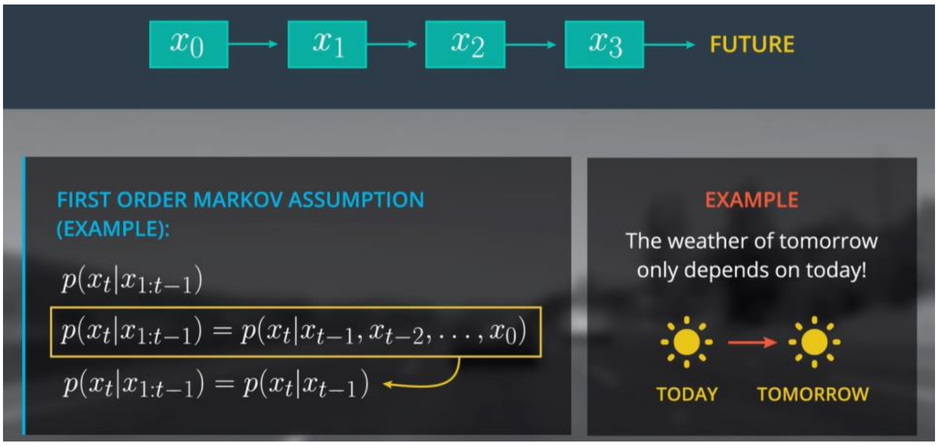
$$P(B) = \sum_{i-1}^{\infty} P(B|A_i)P(A_i)$$

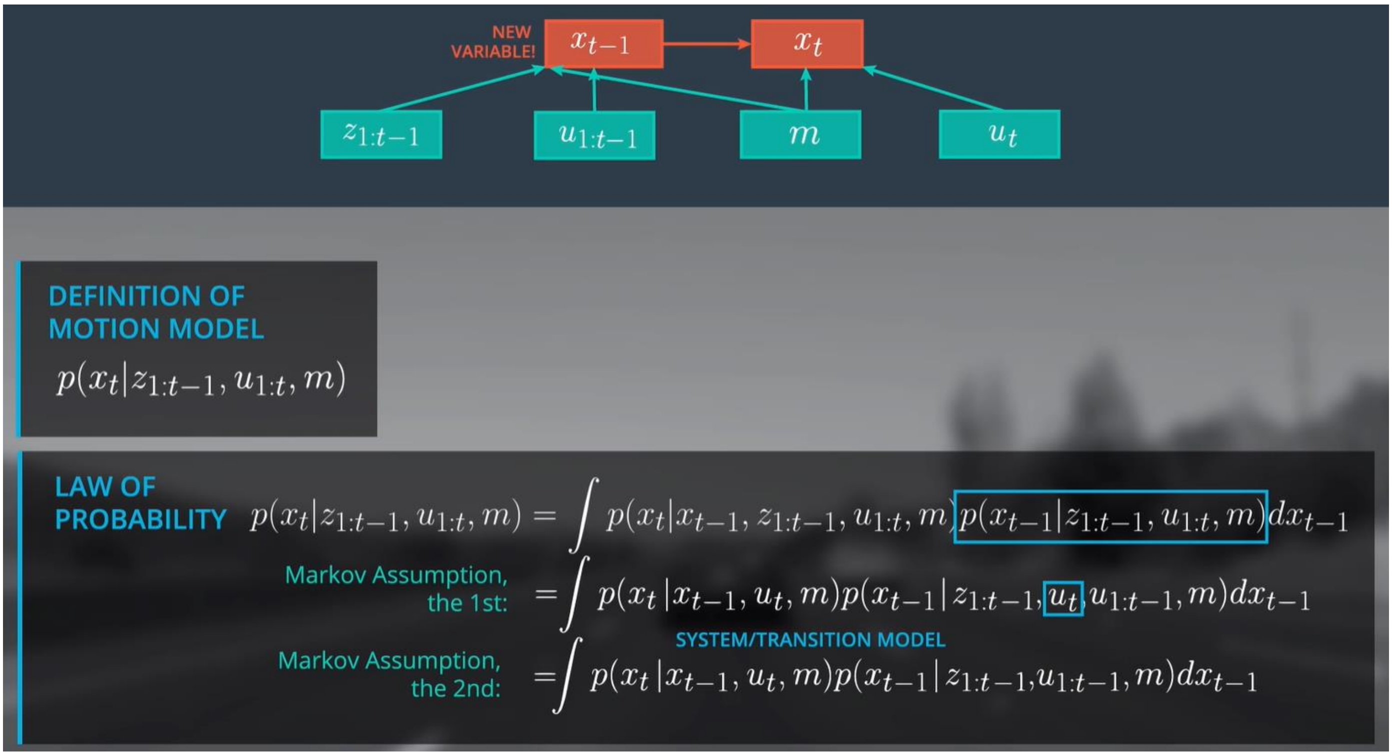


(a) Since we (hypothetically) know in which state the system is at time step t-1, the past observations $z_{1:t-1}$ and controls $u_{1:t-1}$ would not provide us additional information to estimate the posterior for x_t, because they were already used to estimate x_{t-1}. This means, we can simplify $p(x_t|x_{t-1}, z_{1:t-1}, u_{1:t}, m)$ to $p(x_t|x_{t-1}, u_t, m)$.

(b) Since u_t is "in the future" with reference to x_{t-1}, u_t does not tell us much about x_{t-1}. This means the term $p(x_{t-1}|z_{1:t-1}, u_{1:t}, m)$ can be simplified to $p(x_{t-1}|z_{1:t-1}, u_{1:t-1}, m)$.







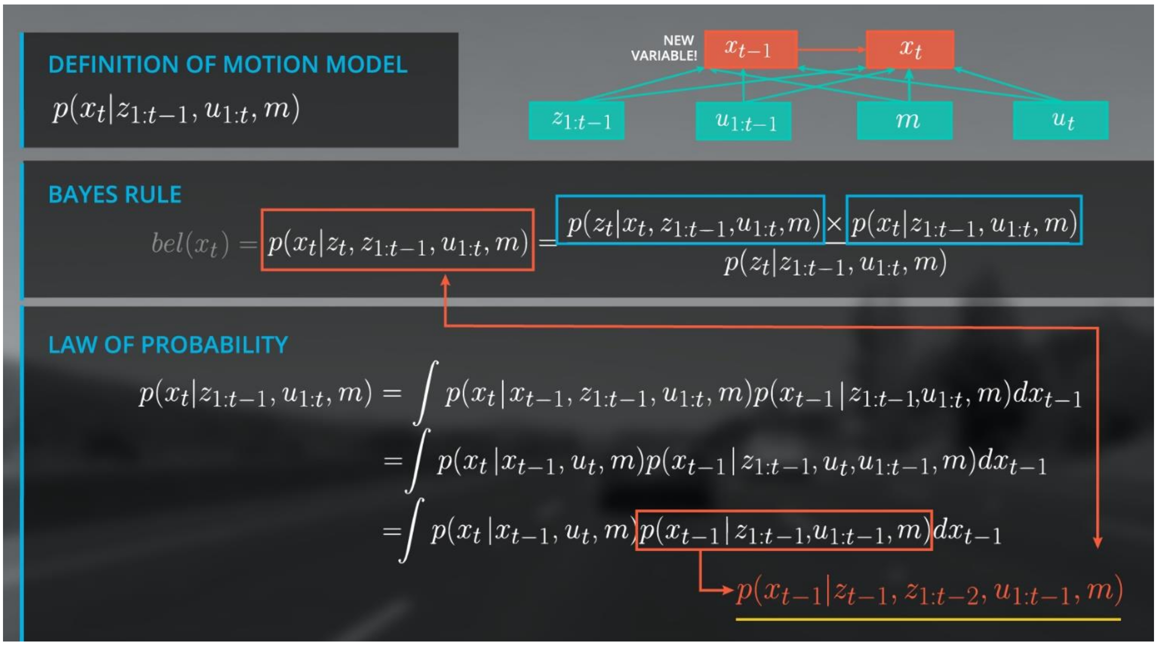
Markov Assumption

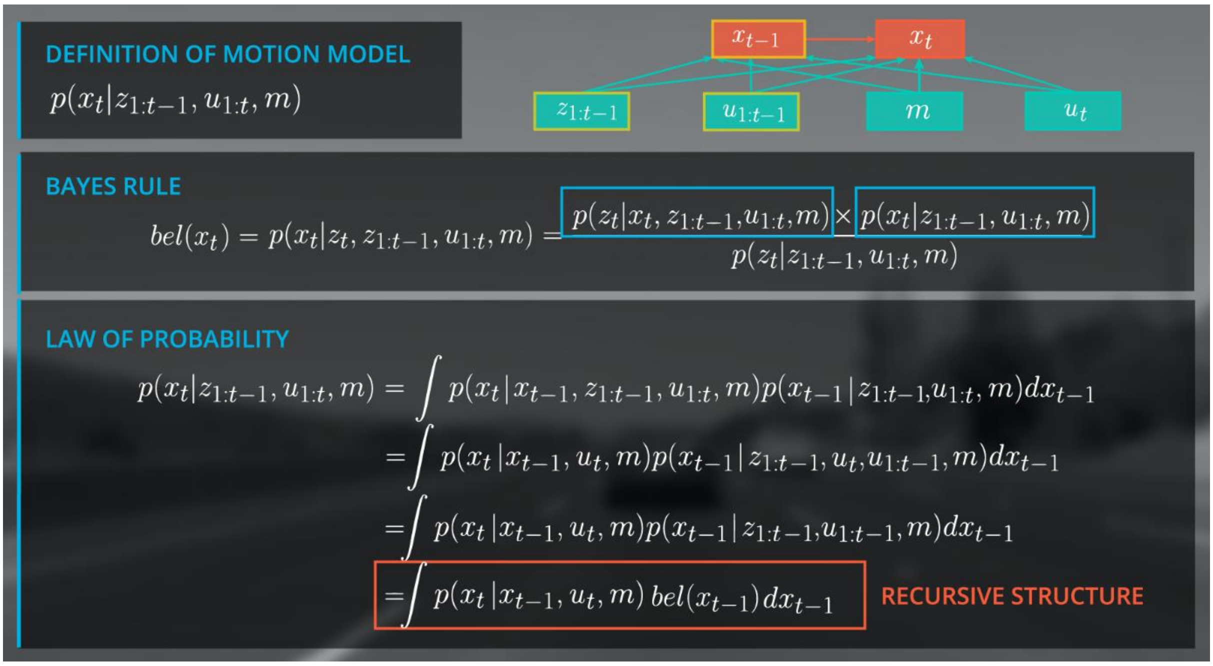
A Markov process is one in which the conditional probability distribution of future states (ie the next state) is dependent only upon the current state and not on other preceding states. This can be expressed mathematically as:

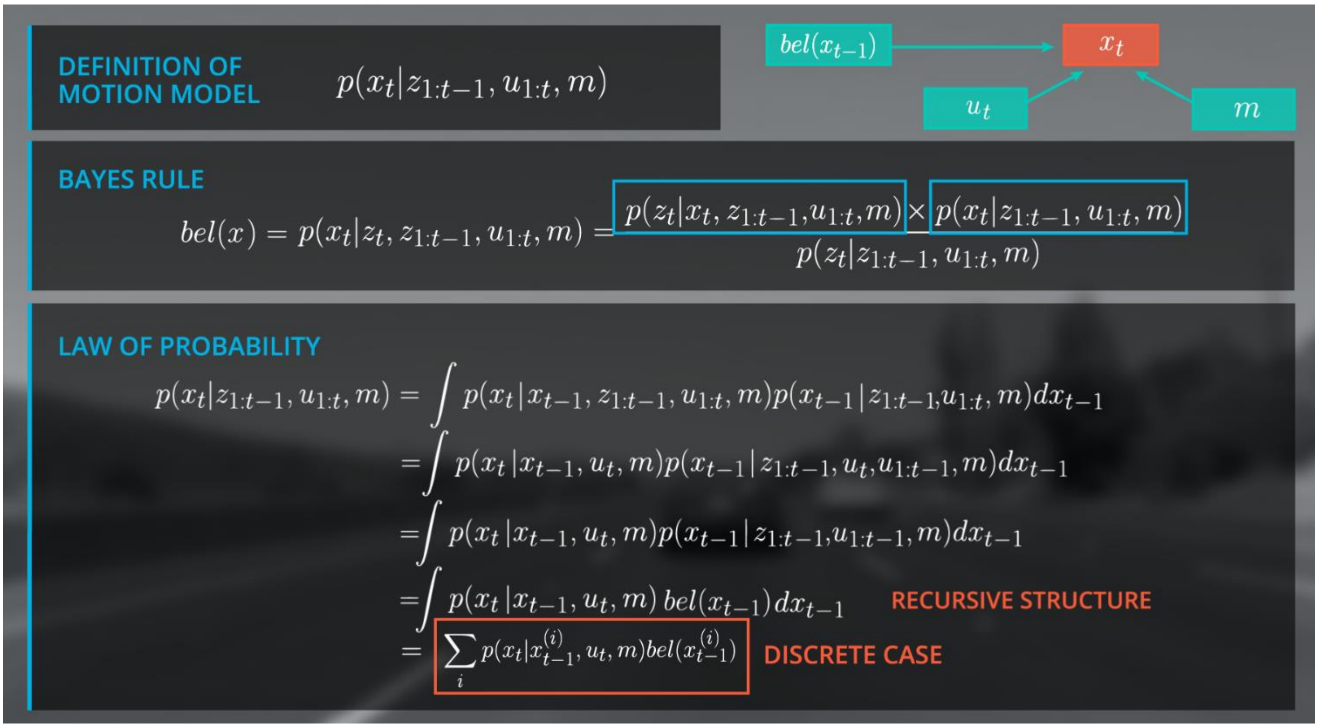
$$P(x_t|x_{1-t}, ..., x_{t-i}, ..., x_0) = P(x_t|x_{t-1})$$

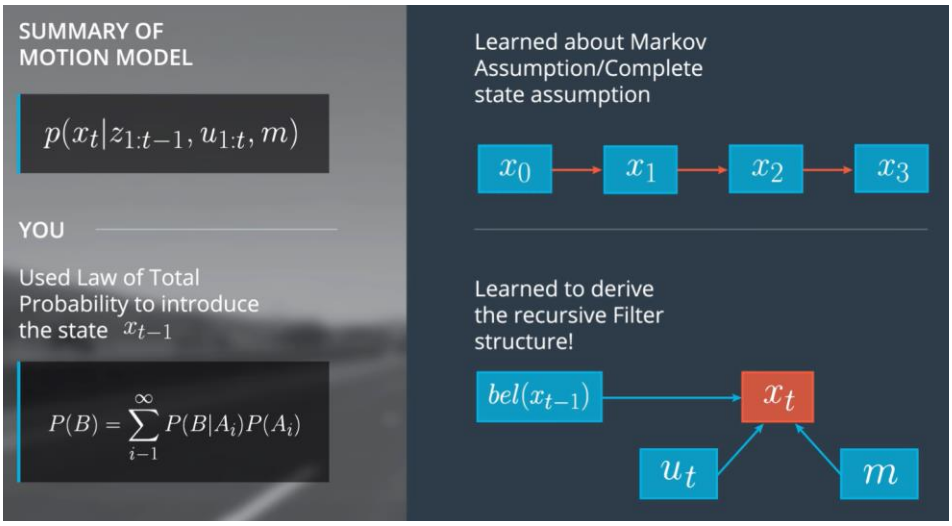
It is important to note that the current state may contain all information from preceding states. That is the case discussed in this lesson.

After applying the Markov Assumption, the term $p(x_{t-1}|z_{1:t-1}, u_{1:t-1}, m)$ describes exactly the belief at x_{t-1} ! This means we achieved a recursive structure!

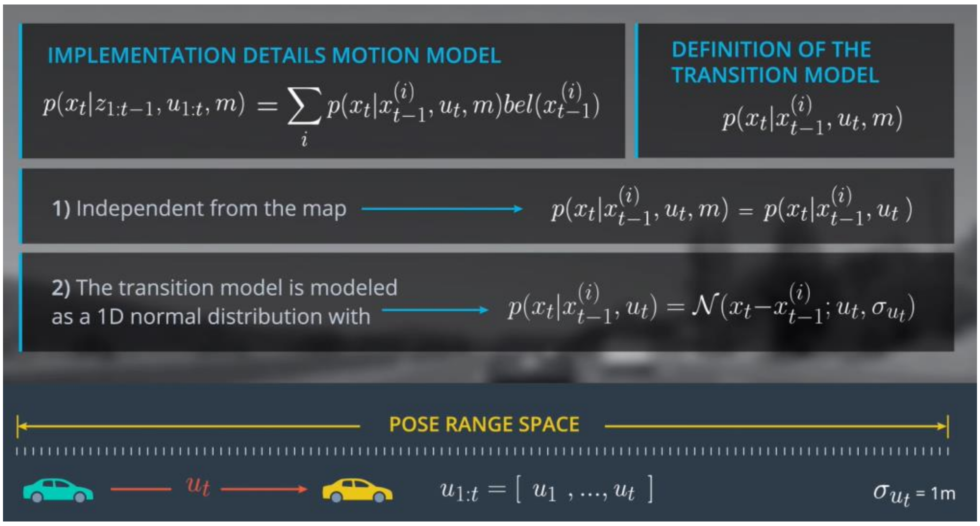


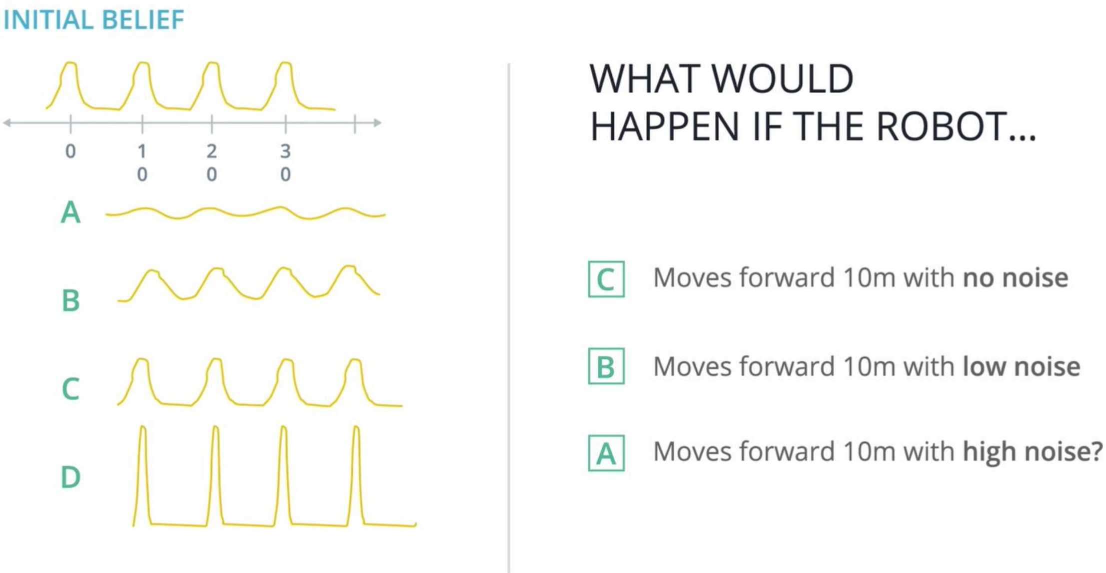




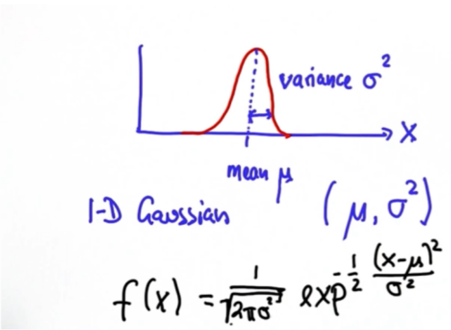


Finally, we replace the integral by a sum over all x_i because we have a discrete localization scenario in this case, to get the same formula in Sebastian's lesson for localization. The process of predicting x_t with a previous beliefs (x_{t-1}) and the transition model is technically a convolution. If you take a look to the formula again, it is essential that the belief at $x_t = 0$ is initialized with a meaningful assumption. It depends on the localization scenario how you set the belief or in other words, how you initialize your filter. For example, you can use GPS to get a coarse estimate of your location.





To implement these models in code, we need a function to which we can pass model parameters/values and return a probability. Fortunately, we can use a normalized probability density function (PDF). Let's revisit Sebastian's discussion of this topic.



QUESTION 1 OF 2

Given pseudo position x and a control parameter of 1 (move 1 unit each time step), which pre-pseudo position maximizes our probability?

- x-4
- x-2
- x-3
- ✓ x-1

Our value will always be maximized when our parameter and value are equal. In this case our control value is 1 (move 1 unit per time step), generally speaking we will see our maximum probability at x - control_parameter.

'i'th Motion Model Probability:

$$p(x_t|x_{t-1}^{(i)}, u_t, m) * bel(x_{t-1}^{(i)})$$

Discretized Motional Model Calculation

Given a transition probability of 3.99E-1 and a belief state $bel(x_{t-1})$ of 5.56E-2, what is the position probability returned by the motion model? Write the answer in scientific notation with an accuracy of two decimal places, for example 3.14E-15.

2.22E-2

RESET

- **Discretized Motion Model:**

$$\sum_i p(x_t|x_{t-1}^{(i)}, u_t, m) bel(x_{t-1}^{(i)})$$

- **Transition Model:**

$$p(x_t|x_{t-1}^{(i)}, u_t, m)$$

- **'i'th Motion Model Probability:**

$$p(x_t|x_{t-1}^{(i)} u_t, m) * bel(x_{t-1}^{(i)})$$

pseudo_position (x)	pre-pseudo_position	delta position	P(transition)	$bel(x_{t-1})$	P(position)
7	1	6	1.49E-06	5.56E-02	8.27E-08
7	2	5	1.34E-04	5.56E-02	7.44E-06
7	3	4	4.43E-03	5.56E-02	2.46E-04
7	4	?	5.40E-02	0.00E+00	0.00E+00
7	5	2	?	0.00E+00	0.00E+00
7	6	1	3.99E-01	0.00E+00	0.00E+00
7	7	0	2.42E-01	?	1.66E-03
7	8	-1	5.40E-02	1.79E-03	?

Delta Position

What is difference in position for an x of 7 and a pre-pseudo position of 4?

3

RESET

Show Solution

Transition Probability

Use `normpdf` (bottom of page) to determine the transition probability for x = 7 and a pre-pseudo_position of 5, and a control parameter of 1, and a standard deviation of 1. The transition probability can be determined through `normpdf(delta_position, control_parameter, position_stdev)`. The answer must be in scientific notation with two decimal place accuracy, for example 3.14E-15.

2.41E-1

RESET

Show Solution

Determine the belief state

In practice we only set our initial belief state, but making the following calculation is helpful in building intuition. What is the belief state $bel(x_{t-1})$ for the penultimate row of our table above? Write the answer in scientific notation with an accuracy of two decimal places, for example 3.14E-15.

6.86E-3

RESET

Position Probability

What is the discretized position probability for x = 7 and a pre-pseudo_position of 8, given the belief state in the table above? Write the answer in scientific notation with an accuracy of two decimal places, for example 3.14E-15.

9.67E-5

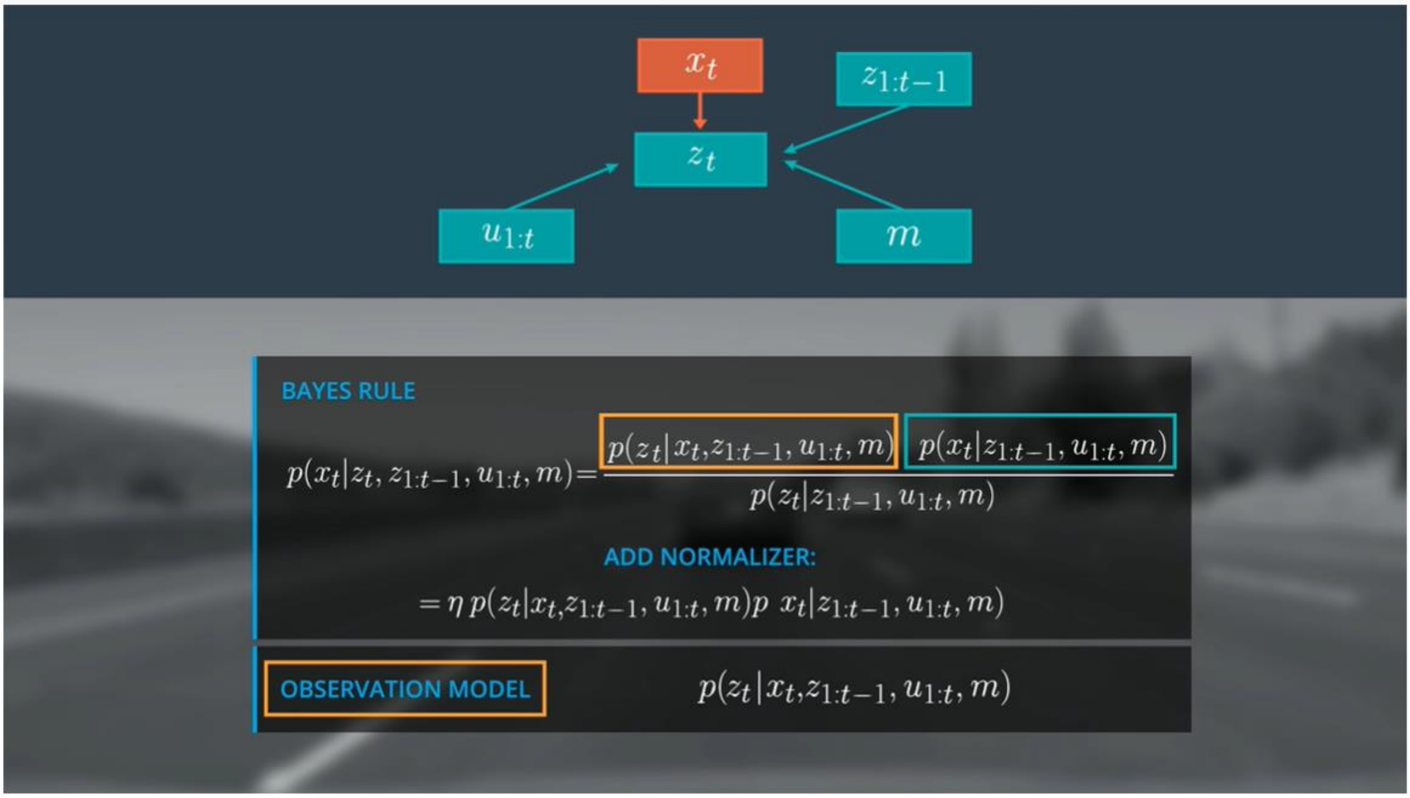
RESET

Show Solution

We have completed our table of discretized calculation for each ith positon probability value. To determine the final probability returned by the motion model, we must sum the probabilities.

Aggregating Discretized P(position)

Given the table above, what is the final probability returned by our motion model. Enter the answer in scientific notation with an accuracy of two decimal places, for example 3.14E-15.



Observation Model

$$p(z_t|x_t, z_{1:t-1}, u_{1:t}, m)$$

Motion Model

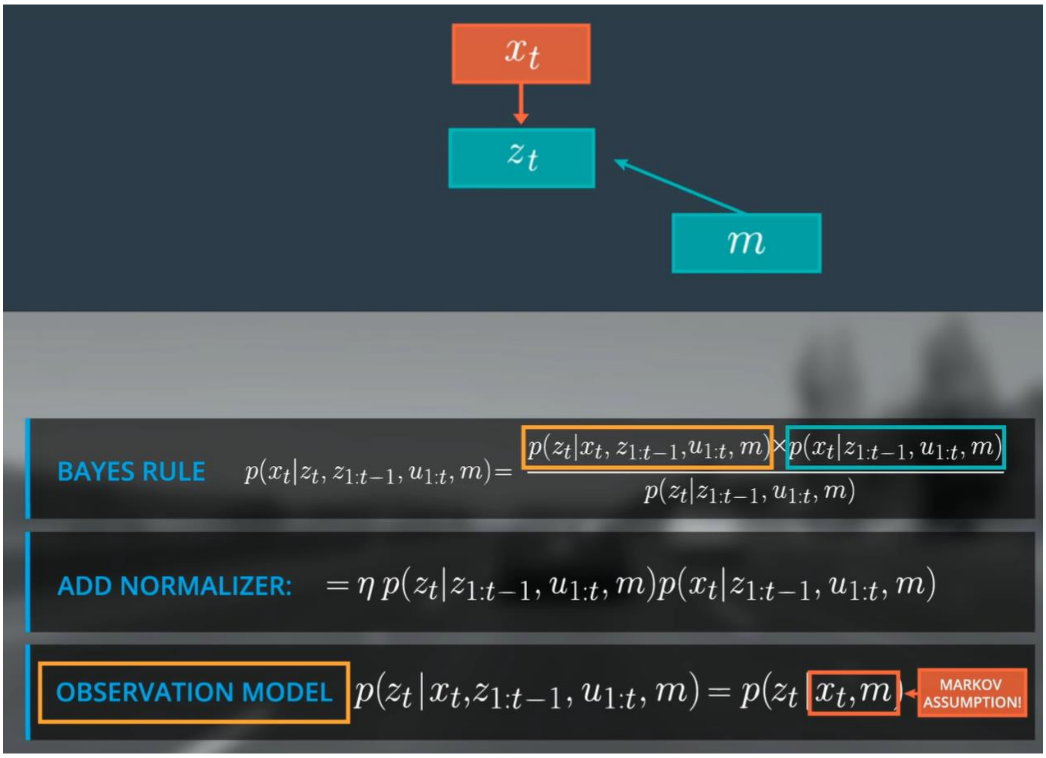
$$p(x_t|z_{1:t-1}, u_{1:t}, m)$$

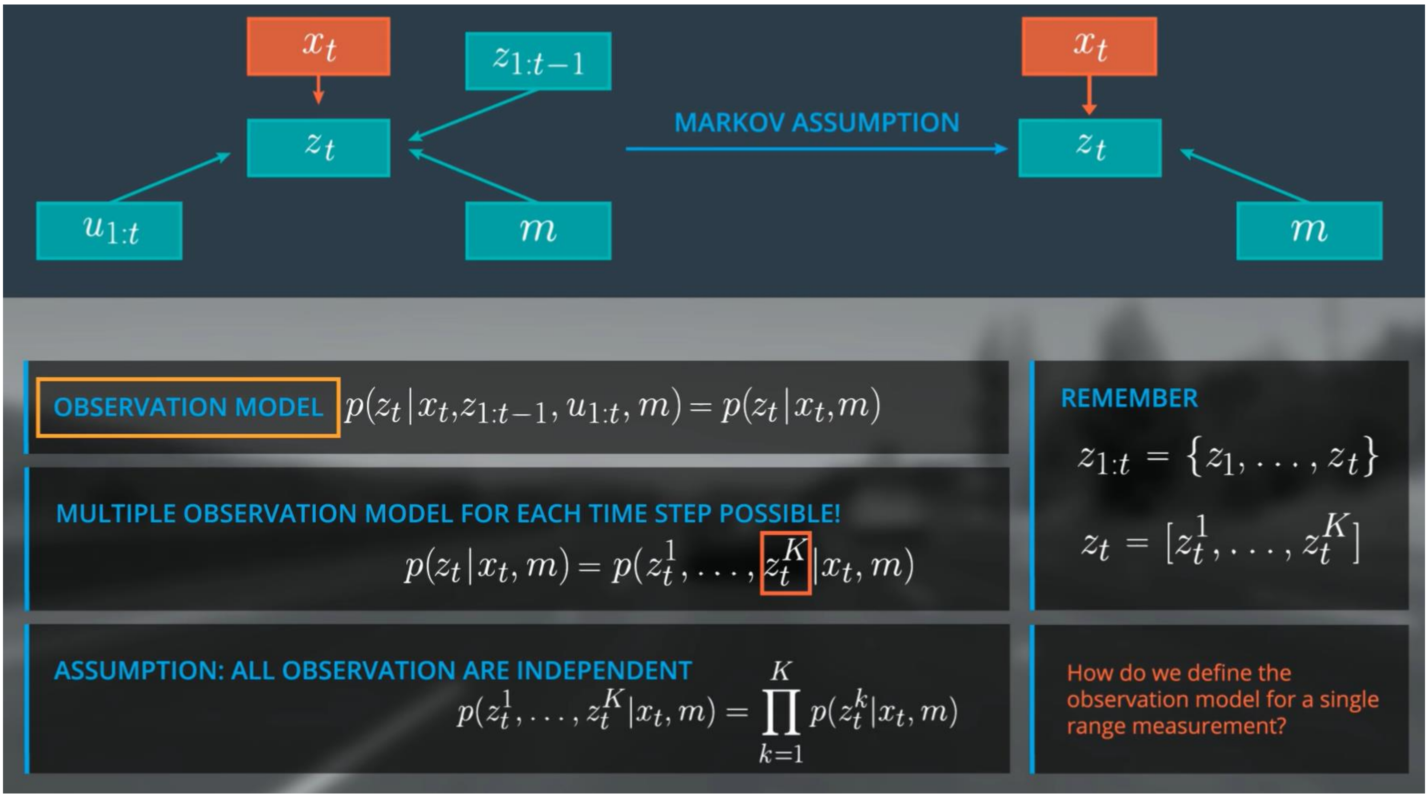
Quiz: Simplifying the Observation Model

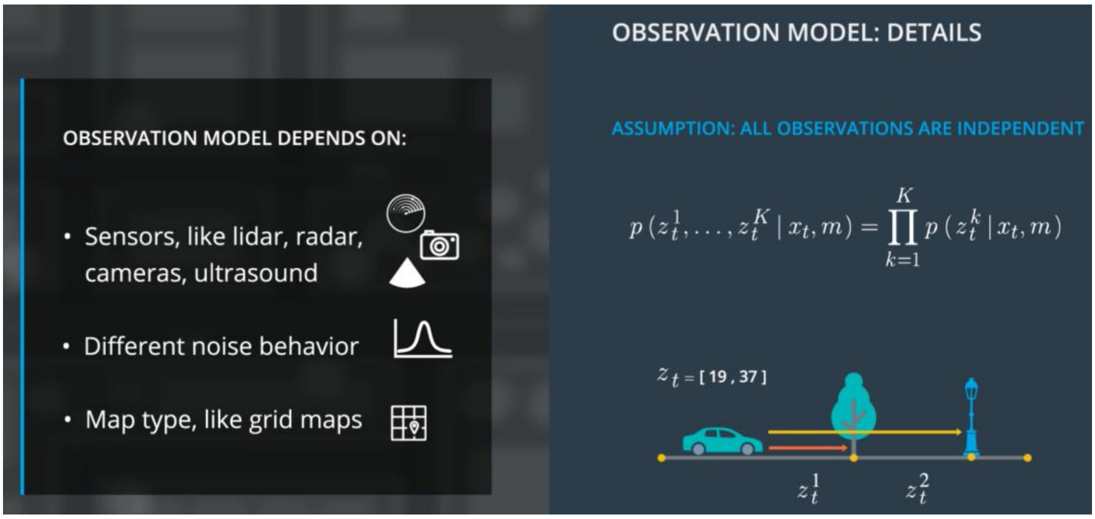
QUIZ QUESTION

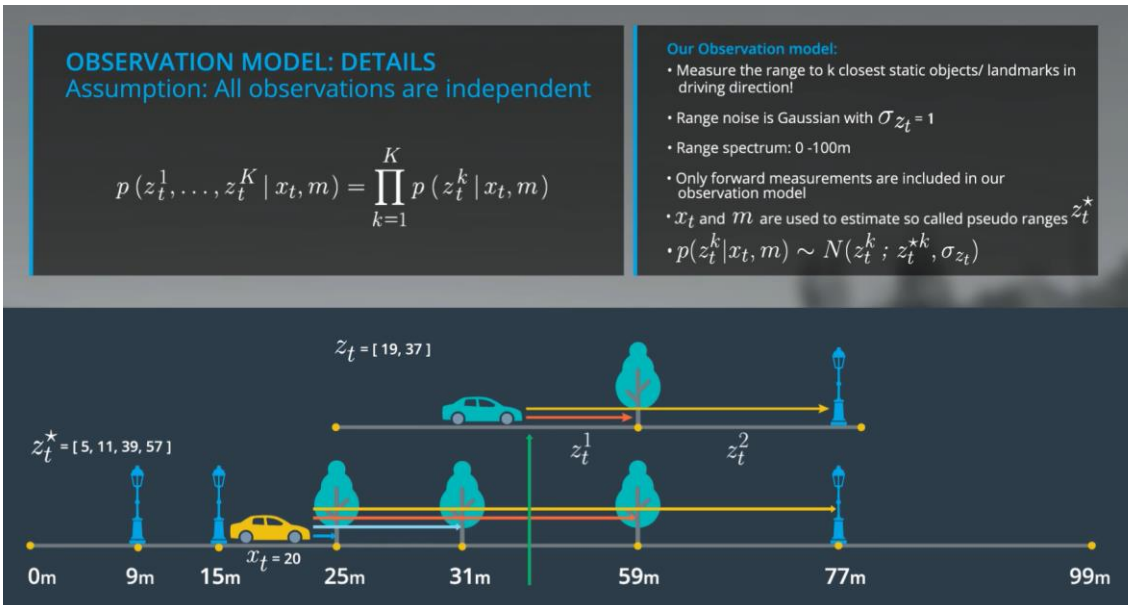
What "trick" can we use there, which helps us to manipulate/simplify the observation model:

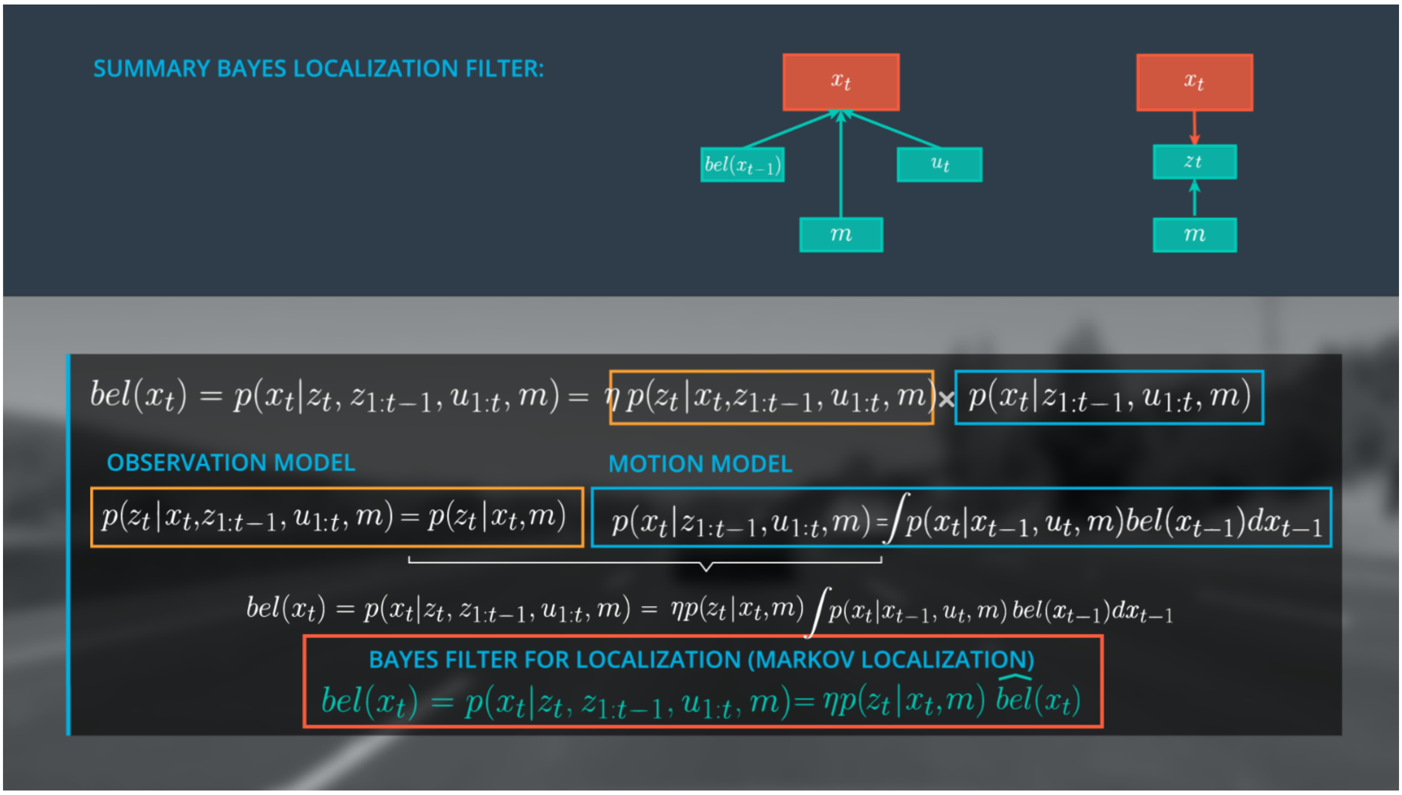
- Using well-known Bayes Rule again!
- Using the law of total Probability!
- ✓ Using the Markov Assumption (Completeness of the State Assumption)

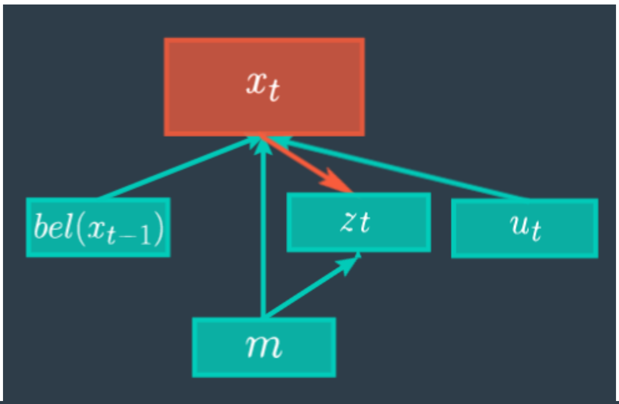


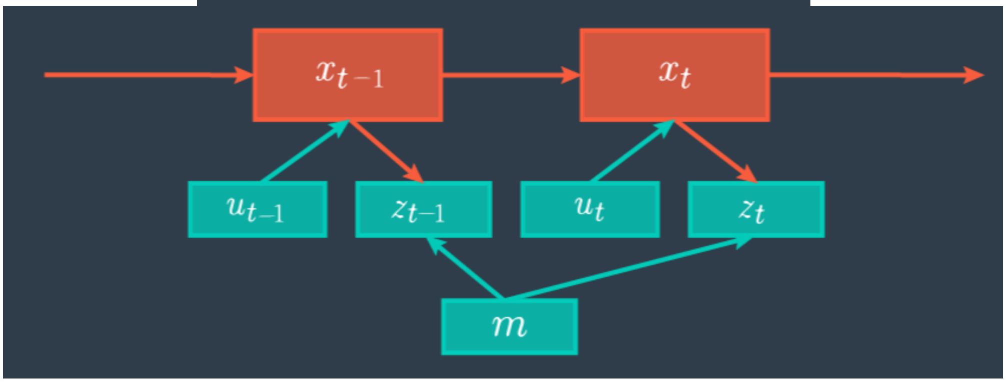


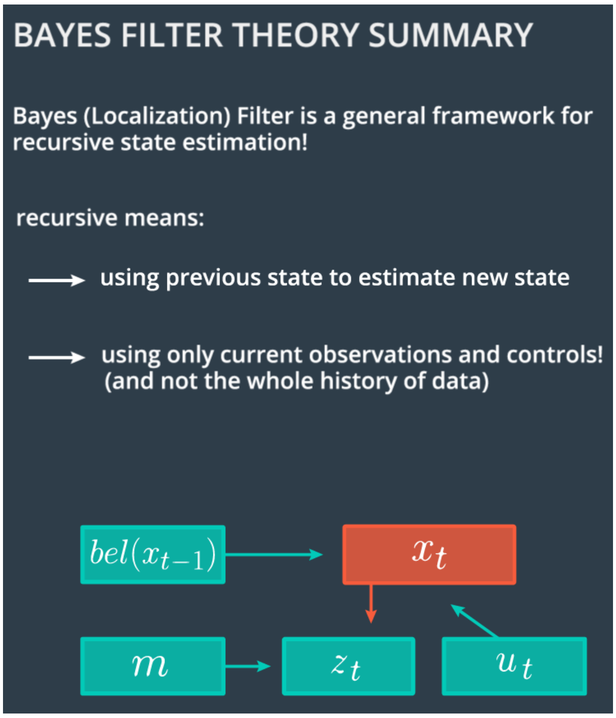

OBSERVATION MODEL DEPENDS ON:
• Sensors, like lidar, radar, cameras, ultrasound
• Different noise behavior
• Map type, like grid maps
OBSERVATION MODEL: DETAILS
ASSUMPTION: ALL OBSERVATIONS ARE INDEPENDENT
$p(z_t^1, \dots, z_t^K \mid x_t, m) = \prod_{k=1}^{K} p(z_t^k \mid x_t, m)$
z_t = [19 , 37]
z_t^1
z_t^2

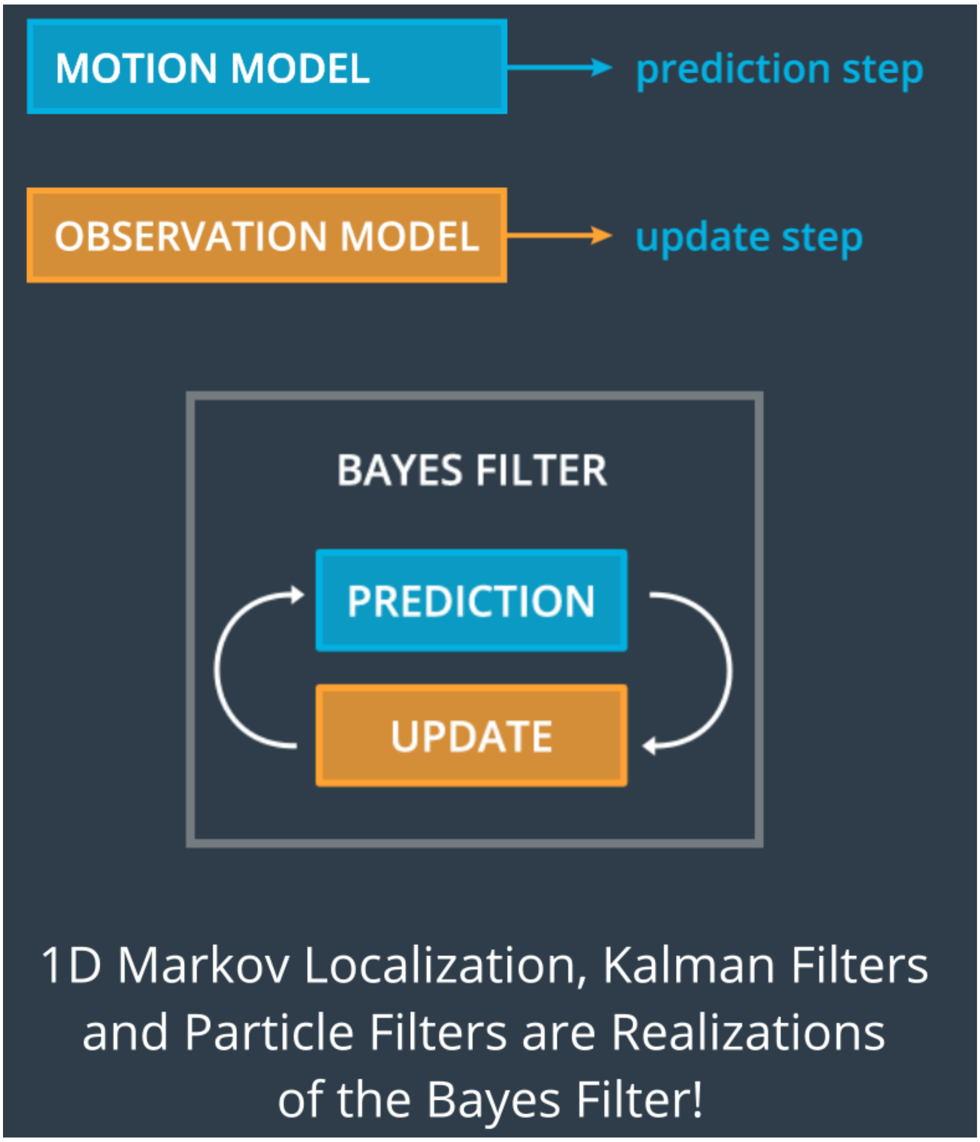

OBSERVATION MODEL: DETAILS
Assumption: All observations are independent
$p(z_t^1, \dots, z_t^K \mid x_t, m) = \prod_{k=1}^{K} p(z_t^k \mid x_t, m)$
Our Observation model:
• Measure the range to k closest static objects/ landmarks in driving direction!
• Range noise is Gaussian with $\sigma_{z_t} = 1$
• Range spectrum: 0 -100m
• Only forward measurements are included in our observation model
• x_t and m are used to estimate so called pseudo ranges $z_t^\star$
• $p(z_t^k|x_t, m) \sim N(z_t^k \,;\, z_t^{\star k}, \sigma_{z_t})$
z_t = [19, 37]
$z_t^\star$ = [5, 11, 39, 57]
z_t^1
z_t^2
x_t = 20
0m
9m
15m
25m
31m
59m
77m
99m


SUMMARY BAYES LOCALIZATION FILTER:
x_t
$bel(x_{t-1})$
u_t
m
x_t
z_t
m
$bel(x_t) = p(x_t|z_t, z_{1:t-1}, u_{1:t}, m) = \eta\, p(z_t|x_t, z_{1:t-1}, u_{1:t}, m) \times p(x_t|z_{1:t-1}, u_{1:t}, m)$
OBSERVATION MODEL
MOTION MODEL
$p(z_t|x_t, z_{1:t-1}, u_{1:t}, m) = p(z_t|x_t, m)$
$p(x_t|z_{1:t-1}, u_{1:t}, m) = \int p(x_t|x_{t-1}, u_t, m) bel(x_{t-1}) dx_{t-1}$
$bel(x_t) = p(x_t|z_t, z_{1:t-1}, u_{1:t}, m) = \eta p(z_t|x_t, m) \int p(x_t|x_{t-1}, u_t, m)\, bel(x_{t-1}) dx_{t-1}$
BAYES FILTER FOR LOCALIZATION (MARKOV LOCALIZATION)
$bel(x_t) = p(x_t|z_t, z_{1:t-1}, u_{1:t}, m) = \eta p(z_t|x_t, m)\, \widehat{bel}(x_t)$


x_t
$bel(x_{t-1})$
z_t
u_t
m


x_{t-1}
x_t
u_{t-1}
z_{t-1}
u_t
z_t
m


BAYES FILTER THEORY SUMMARY
Bayes (Localization) Filter is a general framework for recursive state estimation!
recursive means:
using previous state to estimate new state
using only current observations and controls! (and not the whole history of data)
$bel(x_{t-1})$
x_t
m
z_t
u_t



$$\underbrace{\dot{\theta}}_{\text{Yaw rate}} = 0$$

$$\underbrace{\boxed{x_f}}_{\text{Final x position}} = \underbrace{x_0}_{\text{Initial x position}} + \underbrace{v}_{\text{Velocity}}\underbrace{(dt)}_{\text{Time elapsed}}\underbrace{(cos(\theta_0))}_{\text{X-component of velocity}}$$

$$\underbrace{\boxed{y_f}}_{\text{Final y position}} = \underbrace{y_0}_{\text{Initial y position}} + \underbrace{v}_{\text{Velocity}}\underbrace{(dt)}_{\text{Time elapsed}}\underbrace{(sin(\theta_0))}_{\text{Y-component of velocity}}$$

$$\underbrace{\boxed{\theta_f}}_{\text{Final yaw}} = \underbrace{\theta_0}_{\text{Initial yaw}}$$

$$\underbrace{\dot{\theta}}_{\text{Yaw rate}} \neq 0$$

$$\underbrace{x_f}_{\text{Final x position}} = \underbrace{x_0}_{\text{Initial x position}} + \frac{\overbrace{v}^{\text{Velocity}}}{\underbrace{\dot{\theta}}_{\text{Yaw rate}}}[sin(\underbrace{\theta_0}_{\text{Initial yaw}} + \underbrace{\dot{\theta}}_{\text{Yaw rate}}\underbrace{(dt)}_{\text{Time elapsed}}) - sin(\underbrace{\theta_0}_{\text{Initial yaw}})]$$

$$\underbrace{y_f}_{\text{Final y position}} = \underbrace{y_0}_{\text{Initial y position}} + \frac{\overbrace{v}^{\text{Velocity}}}{\underbrace{\dot{\theta}}_{\text{Yaw rate}}}[cos(\underbrace{\theta_0}_{\text{Initial yaw}}) - cos(\underbrace{\theta_0}_{\text{Initial yaw}} + \underbrace{\dot{\theta}}_{\text{Yaw rate}}\underbrace{(dt)}_{\text{Time elapsed}})]$$

$$\underbrace{\theta_f}_{\text{Final yaw}} = \underbrace{\theta_0}_{\text{Initial yaw}} + \underbrace{\dot{\theta}}_{\text{Yaw rate}}\underbrace{(dt)}_{\text{Time elapsed}}$$

Localization	Sensor Fusion
Things are in vehicle coordinates OR map coordinates. The entire objective of localization is to find the transformation between vehicle coordinates and map coordinates. In other words, we're trying to find the position of the car in the map!	Everything is in vehicle coordinates where the x axis points in the direction of the car's heading and the y axis points to the left of the car.
The position of the car is described in map coordinates.	The car is always assumed to be at the origin of the vehicle coordinate system.
The sensor measurements are usually described in vehicle coordinates. Vehicle coordinates have the x-axis in the direction of the car's heading, the y-axis pointing orthogonal to the left of the car, and the z-axis pointing upwards.	Sensor measurements are in vehicle coordinates.
Map landmarks are in map coordinates.	There's no map involved!

QUIZ QUESTION

Will you need to calculate all three rotations to localize the car effectively?

- ○ Yes, definitely. All three rotations are almost always significant enough to make a significant impact on the position of the car.
- ✓ It really depends on the road (i.e. the curvature and steepness)
- ○ No, a flat world assumption is good enough.

In really hilly places like San Francisco or Pittsburgh, the pitch of the car is pretty important. For most other roads though, the extra effort put into the extra calculations are usually not worth the miniscule improvement in accuracy.

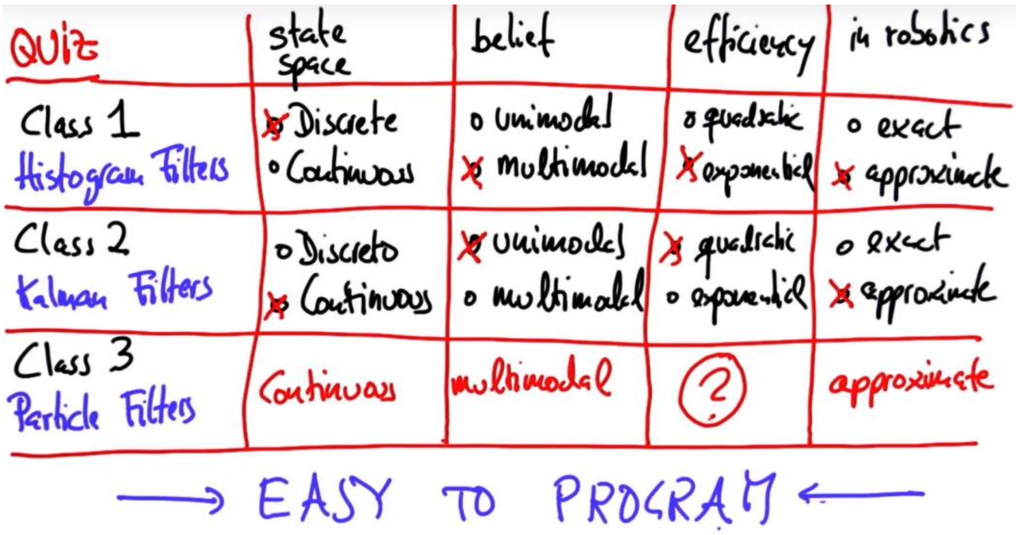
$$\underbrace{x_f}_{\text{Final x position}} = \underbrace{x_0}_{\text{Initial x position}} + \underbrace{\#turns}_{\text{Odometry}}\underbrace{(cos(\theta))}_{\text{x - component}}\underbrace{(c_{wheel})}_{\text{Circumference of the wheel}}$$

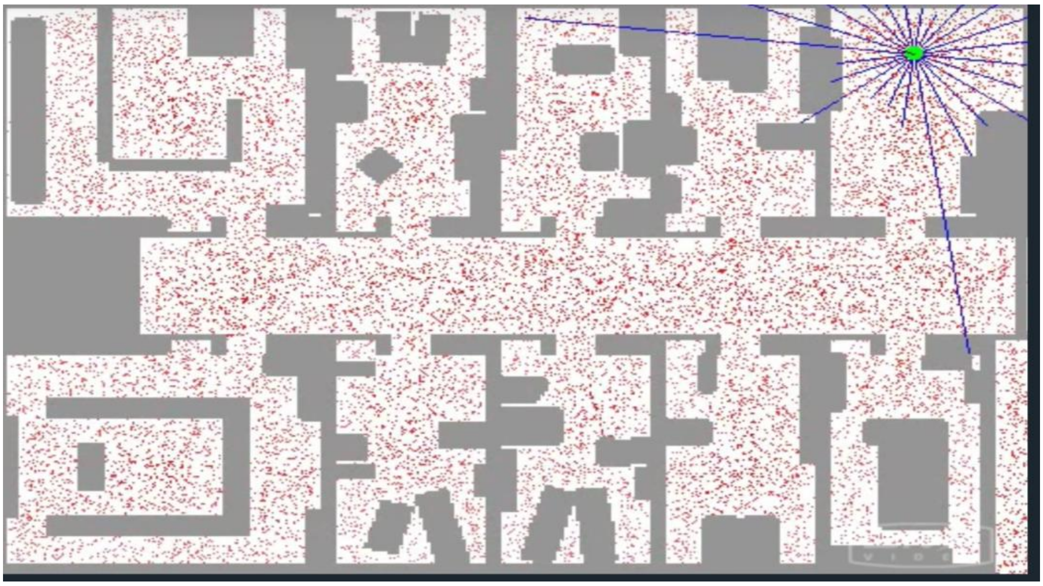
$$\underbrace{y_f}_{\text{Final y position}} = \underbrace{y_0}_{\text{Initial y position}} + \underbrace{\#turns}_{\text{Odometry}}\underbrace{(sin(\theta))}_{\text{y - component}}\underbrace{(c_{wheel})}_{\text{Circumference of the wheel}}$$

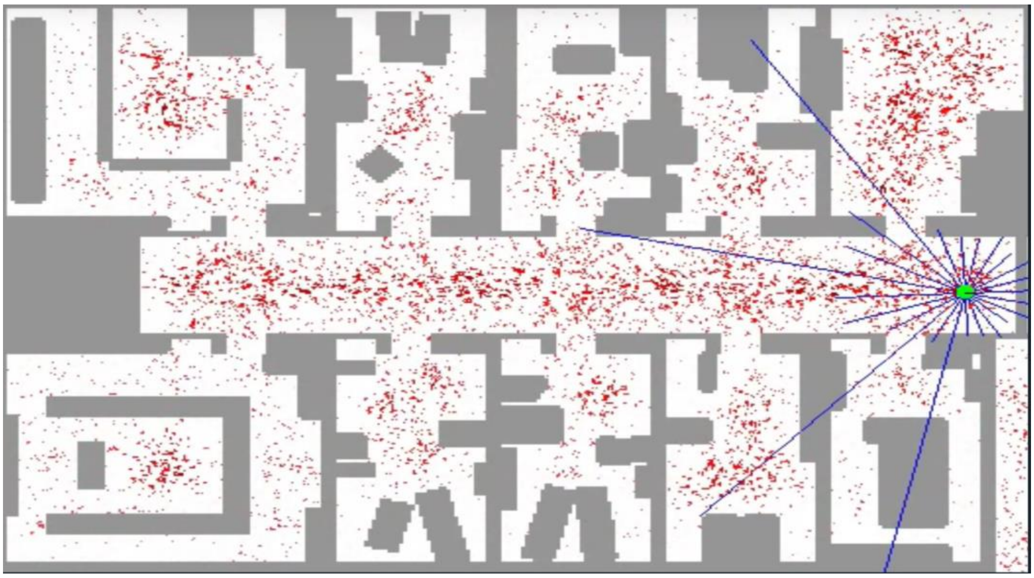
QUIZ QUESTION

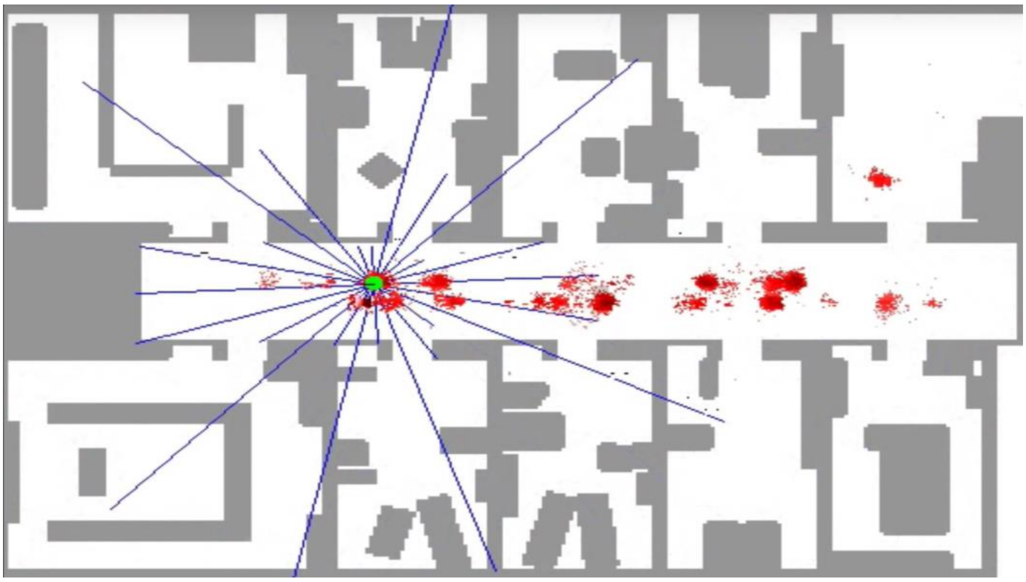
When do you think using odometry from a wheel encoder will create large errors in your position estimates?

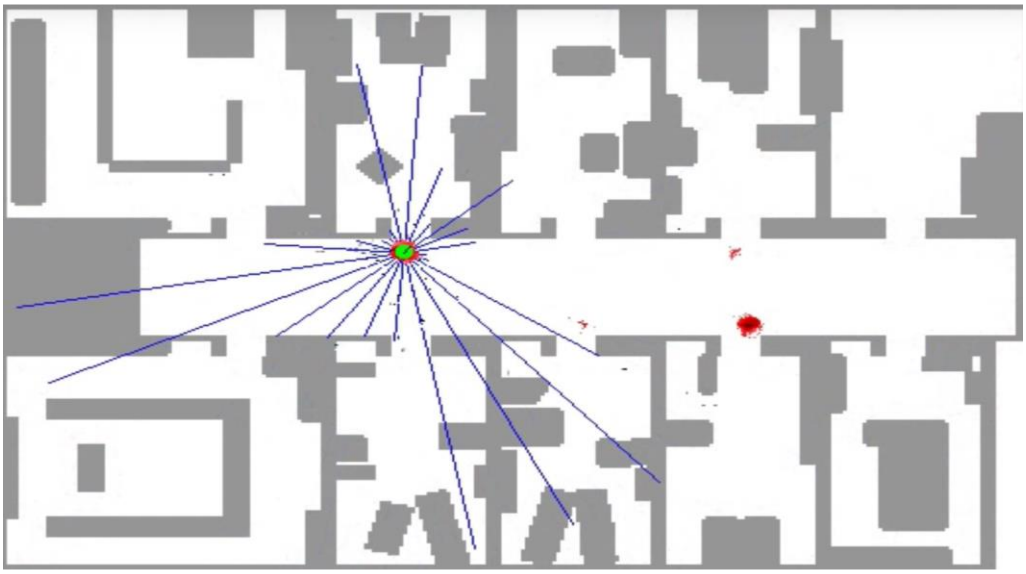
- [x] On a slick, wet road
- [ ] On a dry, paved road
- [x] On a road with lots of bumps
- [ ] On a road with lots of turns

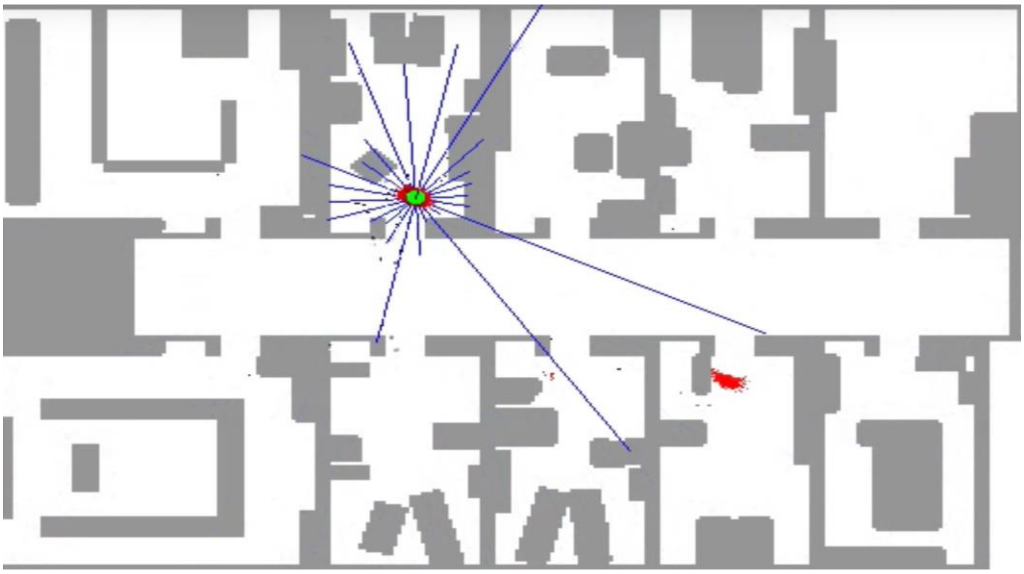


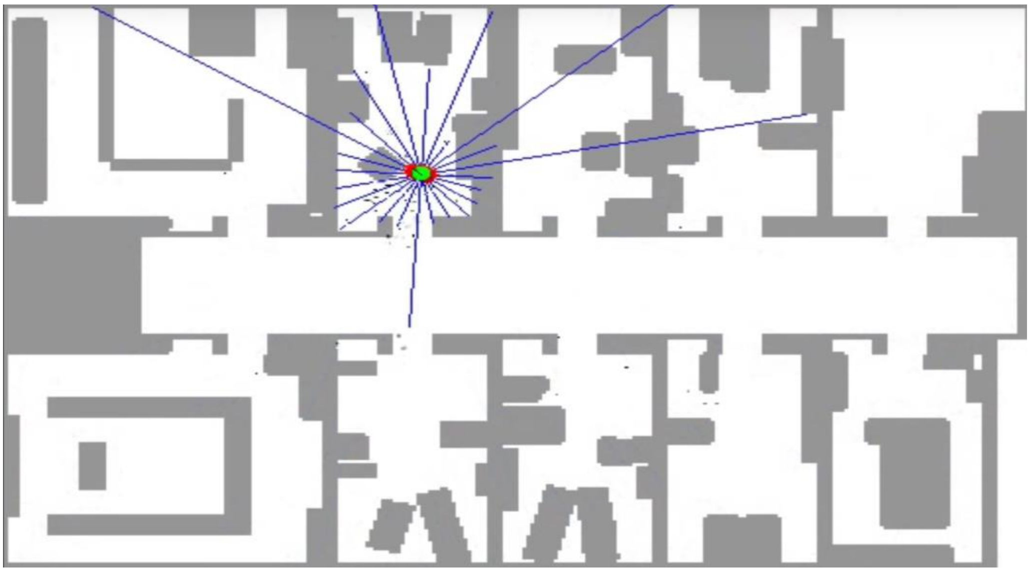


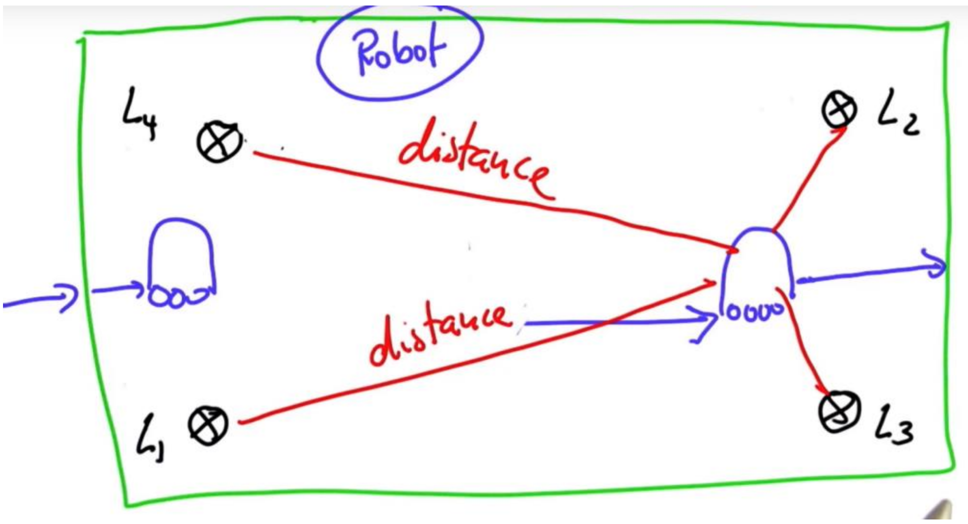


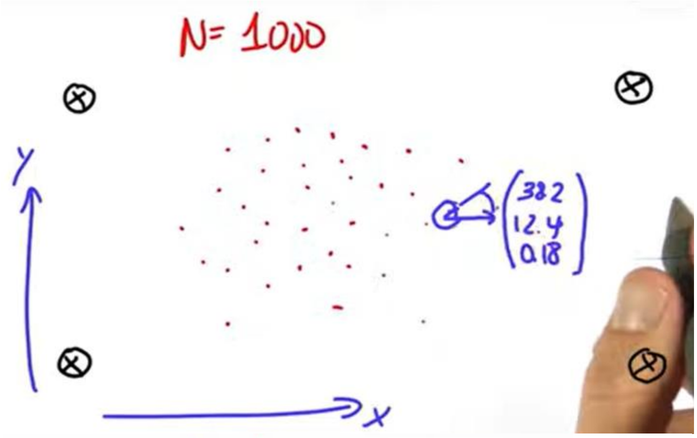


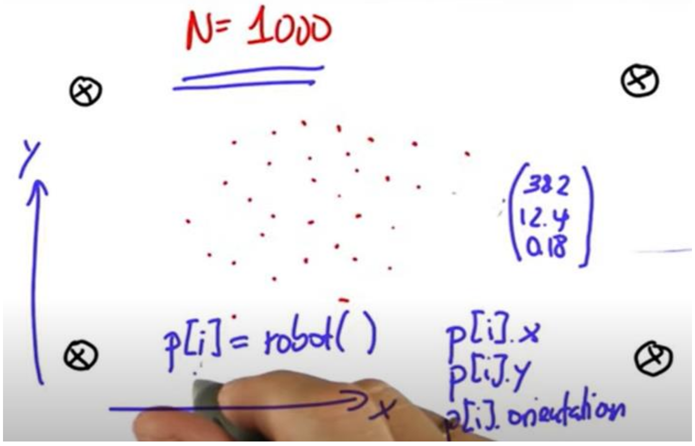


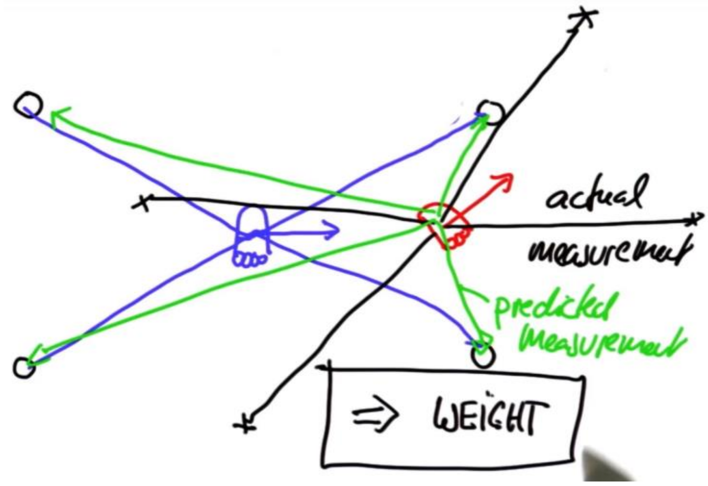


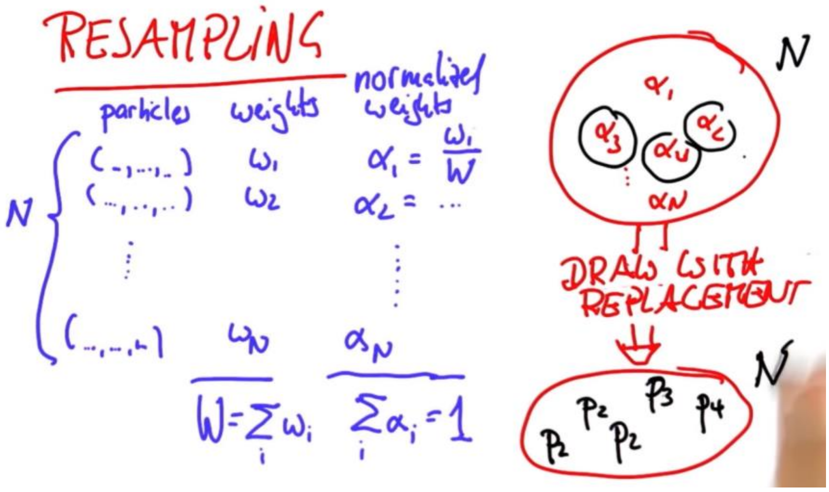


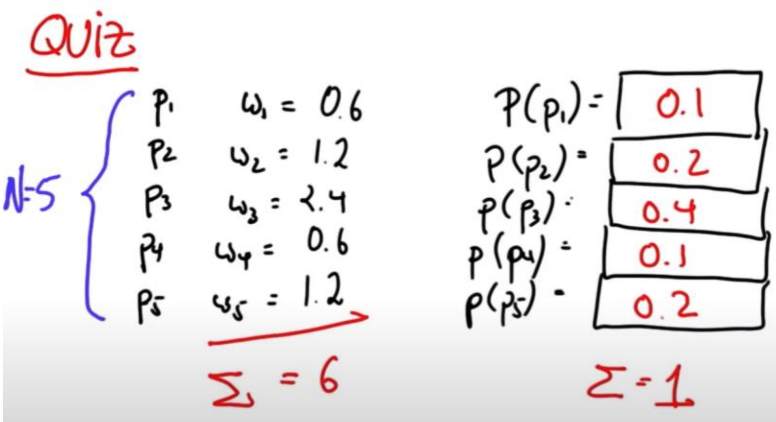

Robot
L_4
L_2
distance
distance
L_1
L_3

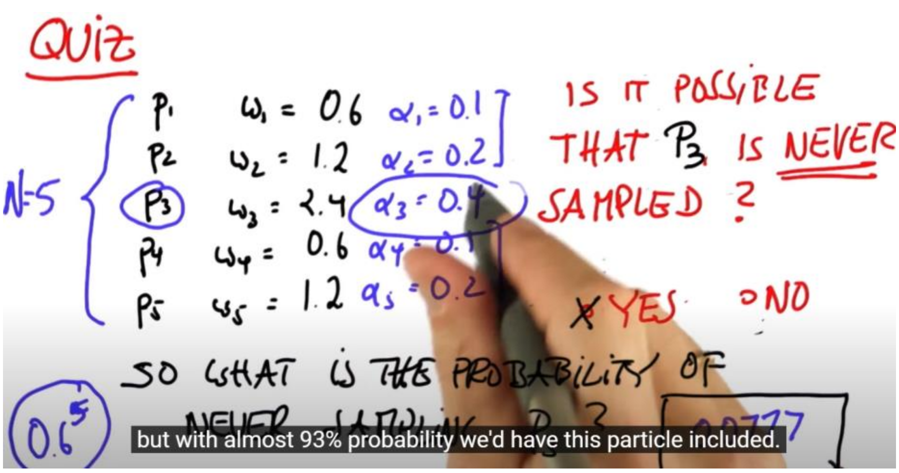

N= 1000
y
x
382
12.4
0.18

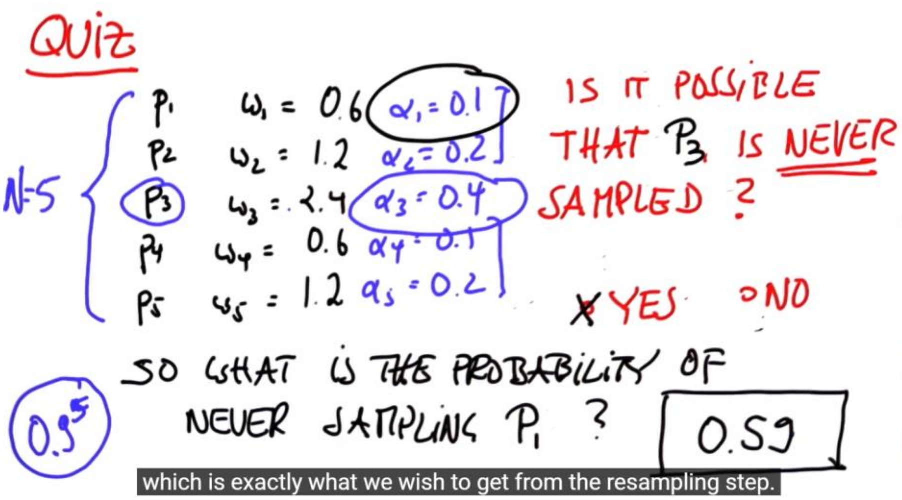

N= 1000
y
x
382
12.4
0.18
p[i] = robot( )
p[i].x
p[i].y
p[i].orientation

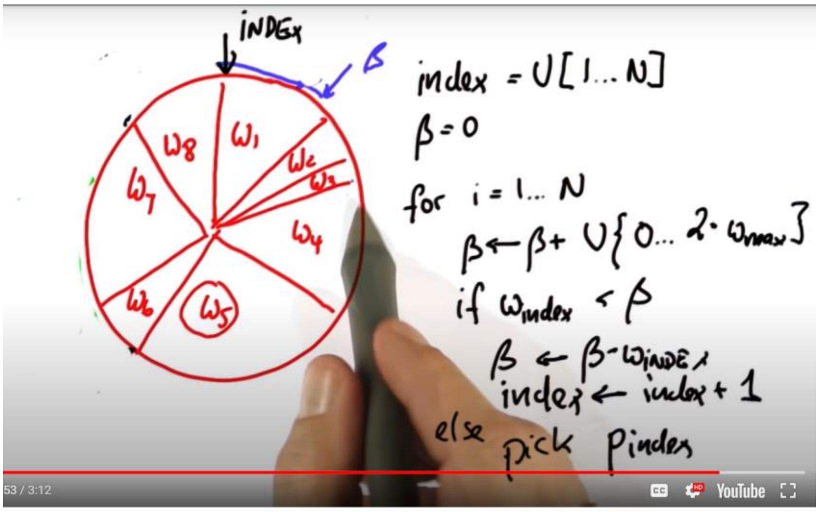

actual
measurement
predicted
measurement
⇒ WEIGHT


RESAMPLING
particles
weights
normalized
weights
N
ω_1
$\alpha_1 = \frac{\omega_1}{W}$
ω_2
$\alpha_2 = \dots$
ω_N
α_N
$W = \sum_i \omega_i$
$\sum_i \alpha_i = 1$
N
α_1
α_3
α_4
α_2
α_N
DRAW WITH
REPLACEMENT
N
p_2
p_2
p_3
p_4
p_2


QUIZ
N=5
p_1 $w_1 = 0.6$
p_2 $w_2 = 1.2$
p_3 $w_3 = 2.4$
p_4 $w_4 = 0.6$
p_5 $w_5 = 1.2$
$\Sigma = 6$
$P(p_1) = 0.1$
$P(p_2) = 0.2$
$P(p_3) = 0.4$
$P(p_4) = 0.1$
$P(p_5) = 0.2$
$\Sigma = 1$


QUIZ
N=5
p_1 $w_1 = 0.6$ $\alpha_1 = 0.1$
p_2 $w_2 = 1.2$ $\alpha_2 = 0.2$
p_3 $w_3 = 2.4$ $\alpha_3 = 0.4$
p_4 $w_4 = 0.6$
p_5 $w_5 = 1.2$ $\alpha_5 = 0.2$
IS IT POSSIBLE THAT P_3 IS NEVER SAMPLED ?
X YES o NO
SO WHAT IS THE PROBABILITY OF
0.6^5
but with almost 93% probability we'd have this particle included.

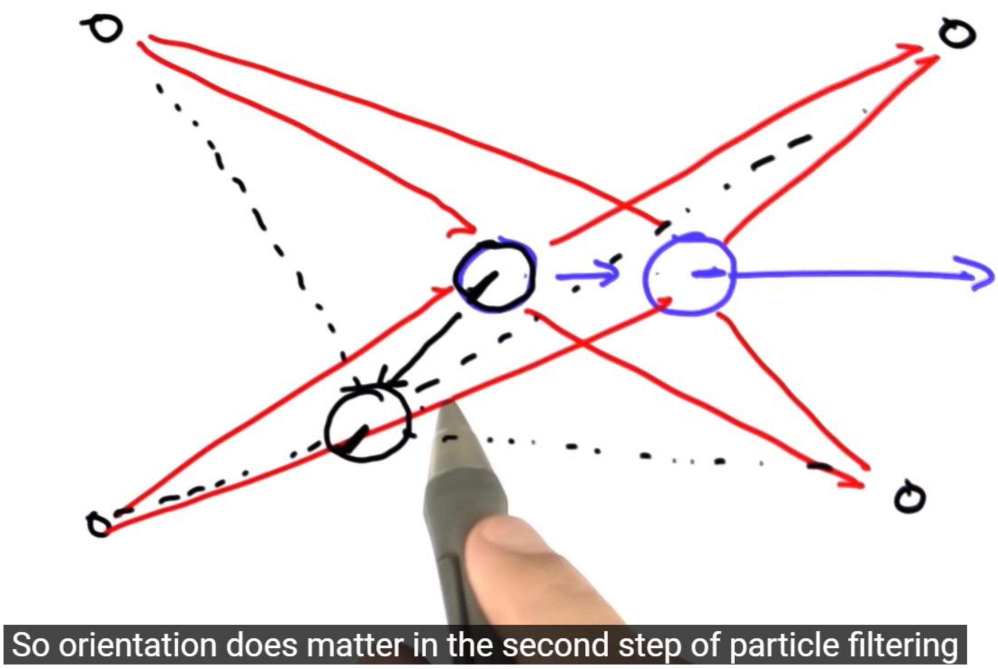

QUIZ
N=5
p_1 $w_1 = 0.6$ $\alpha_1 = 0.1$
p_2 $w_2 = 1.2$ $\alpha_2 = 0.2$
p_3 $w_3 = 2.4$ $\alpha_3 = 0.4$
p_4 $w_4 = 0.6$ $\alpha_4 = 0.1$
p_5 $w_5 = 1.2$ $\alpha_5 = 0.2$
IS IT POSSIBLE THAT P_3 IS NEVER SAMPLED ?
X YES o NO
SO WHAT IS THE PROBABILITY OF NEVER SAMPLING P_1 ?
0.59
0.9^5
which is exactly what we wish to get from the resampling step.



The pseudocode in the video should be like this (instead of an if-else block):

```
while w[index] < beta:
    beta = beta - w[index]
    index = index + 1

select p[index]
```



MEASUREMENT UPDATES

$$\underline{P(X|Z)} \propto P(Z|X)\,P(X)$$

resampling ↑ importance weights ↑ particles

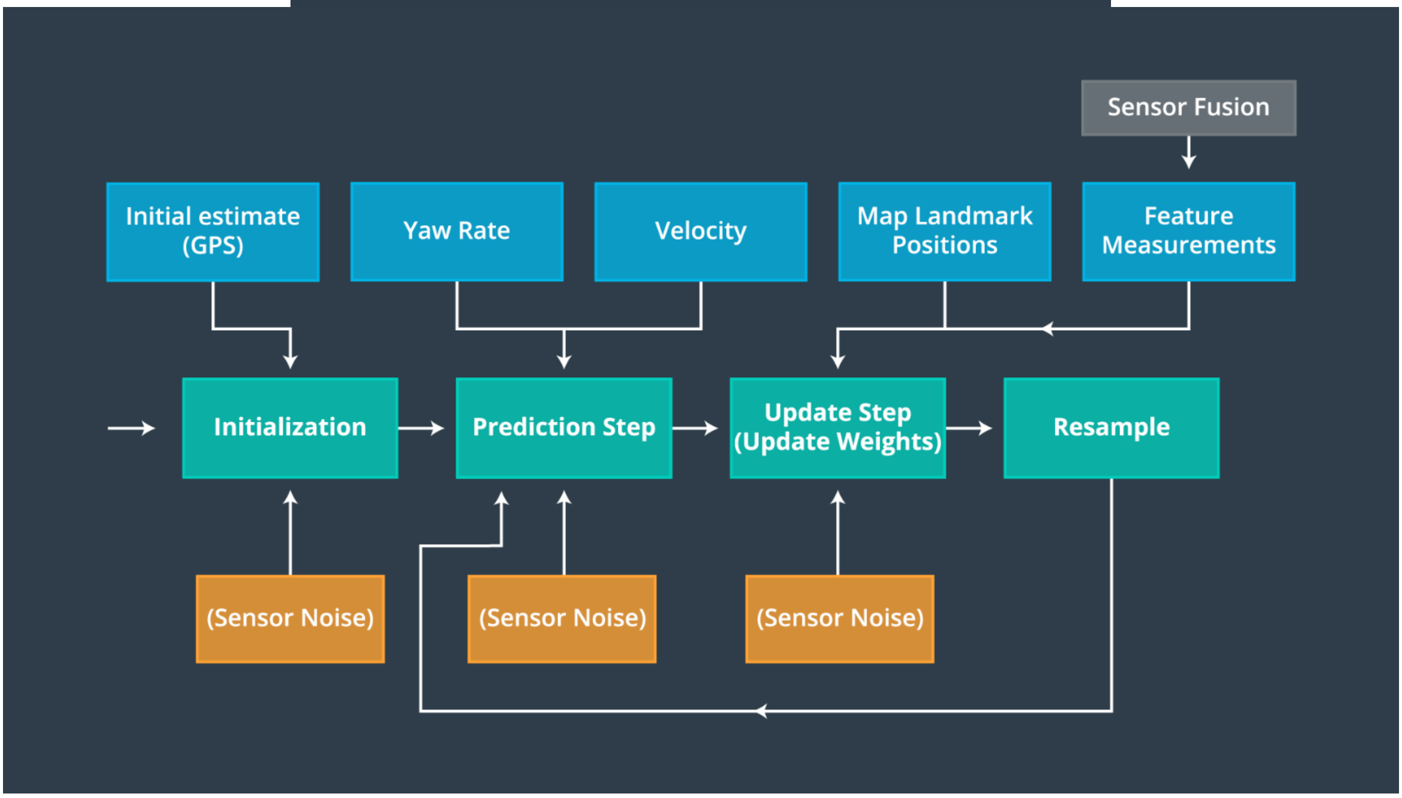
MOTION UPDATES

$$\underline{P(X')} = \sum \underbrace{P(X'|X)}_{\text{sample}} P(X)$$

sampled ↑ particles

Particle Filter $(\mathcal{X}_{t-1}, u_t, z_t)$

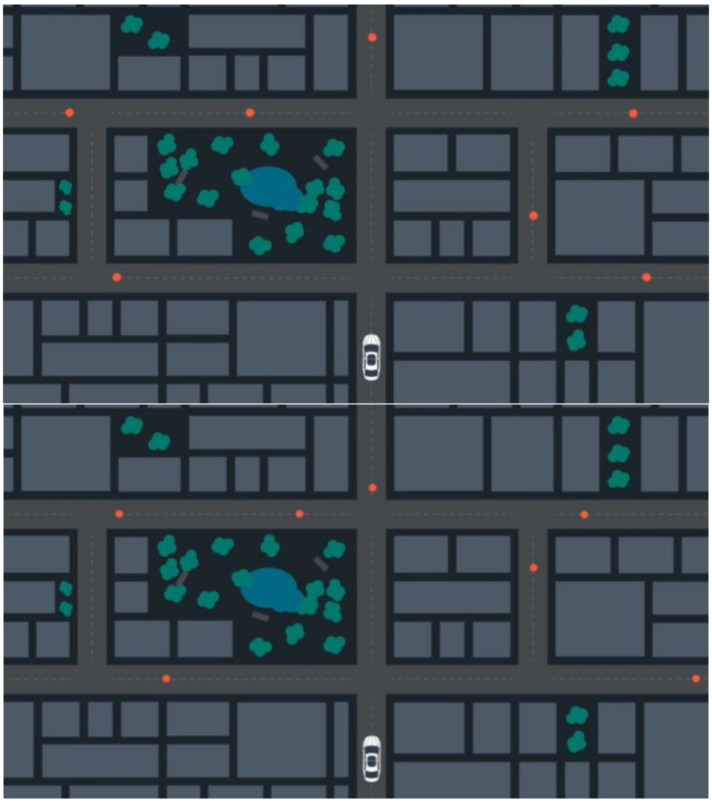
1. $\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
2. for $m = 1$ to M do
3. sample $x_t^{[m]} \sim p(x_t | u_t, x_{t-1}^{[m]})$
4. $w_t^{[m]} = p(z_t | x_t^{[m]})$
5. $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \left\langle x_t^{[m]}, w_t^{[m]} \right\rangle$
6. endfor
7. for $m = 1$ to M do
8. draw i with probability α $w_t^{[i]}$
9. add $x_t^{[i]}$ to $\mathcal{X}_t$
10. endfor
11. return $\mathcal{X}_t$



Initialization

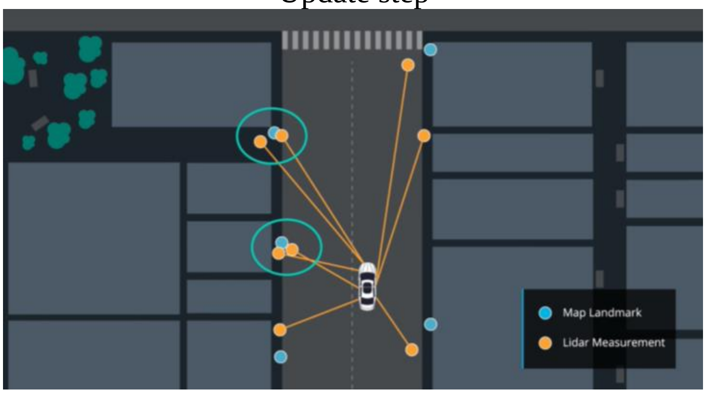
The most practical way to initialize our particles and generate real time output, is to make an initial estimate using GPS input. As with all sensor based operations, this step is impacted by noise.

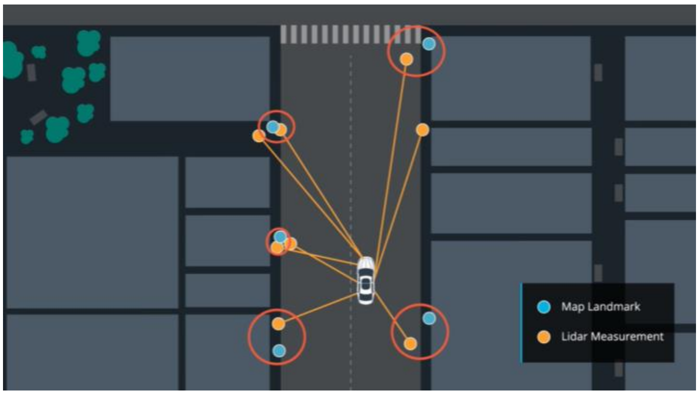
Prediction



Here we will use what we learned in the motion models lesson to predict where the vehicle will be at the next time step, by updating based on yaw rate and velocity, while accounting for Gaussian sensor noise.

Update step



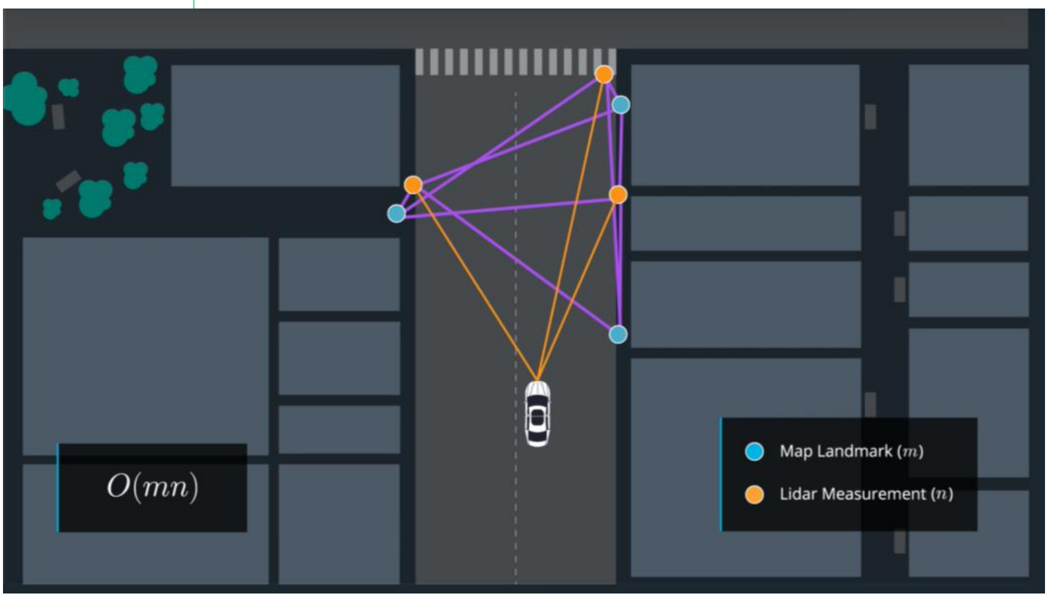


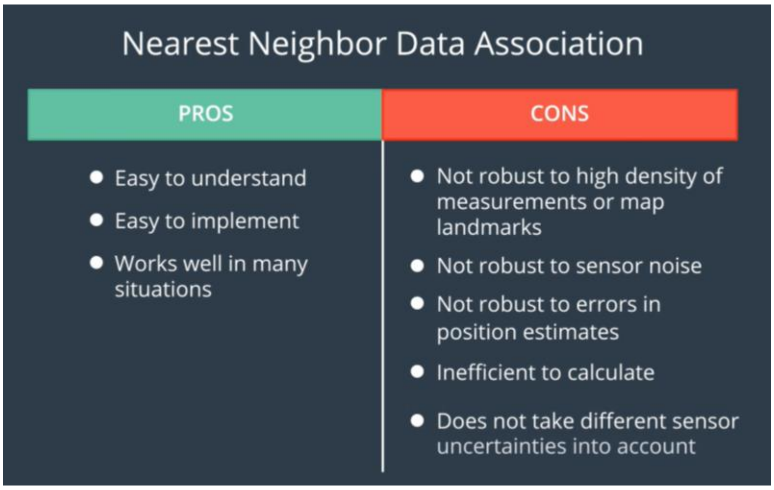
Nearest Neighbor Effectiveness Quiz

QUIZ QUESTION

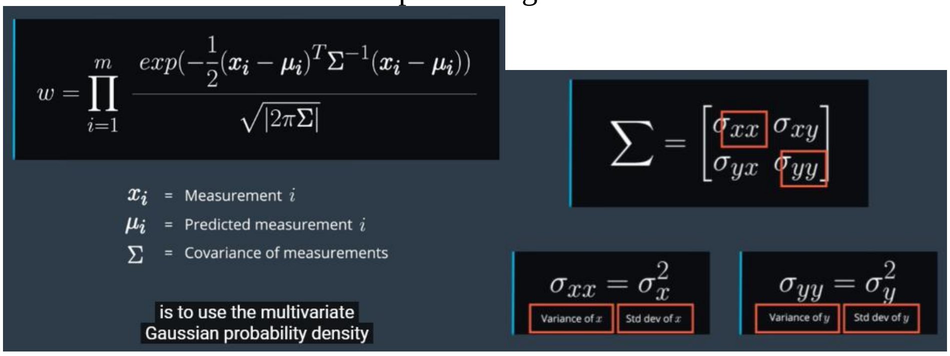
Which of the following characteristics of the input data would help nearest neighbor data association become more effective?

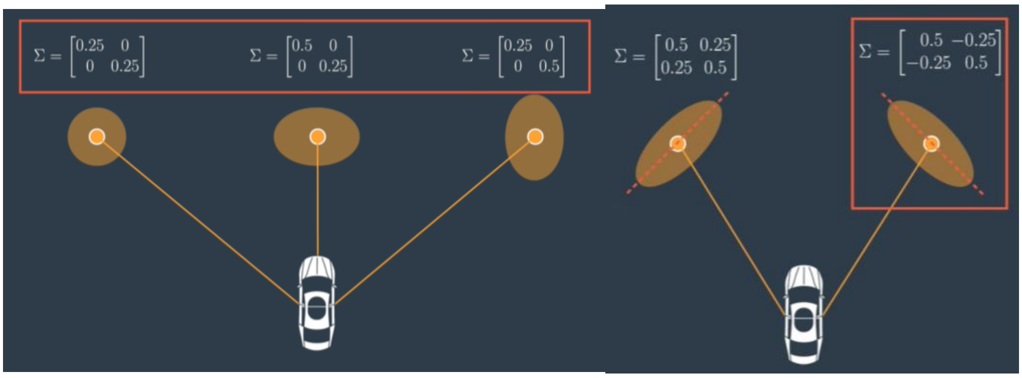
- [ ] Higher density of landmarks or measurements
- [x] High signal-to-noise ratio for sensors
- [x] A very accurate motion model

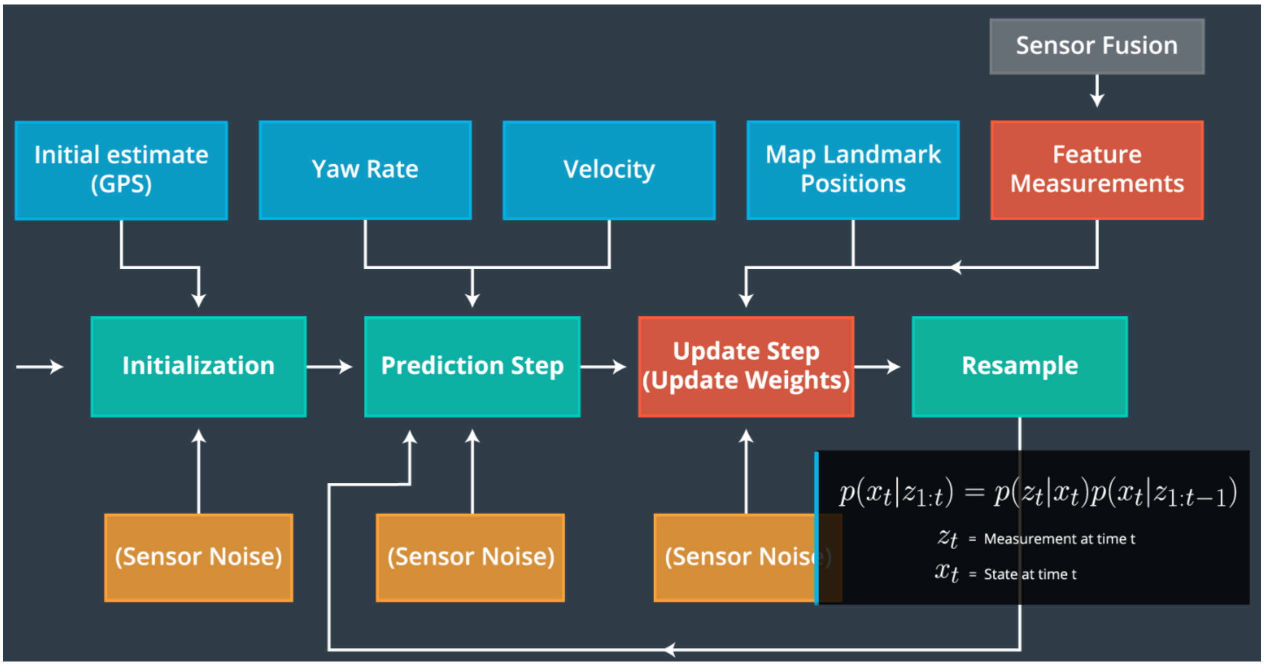




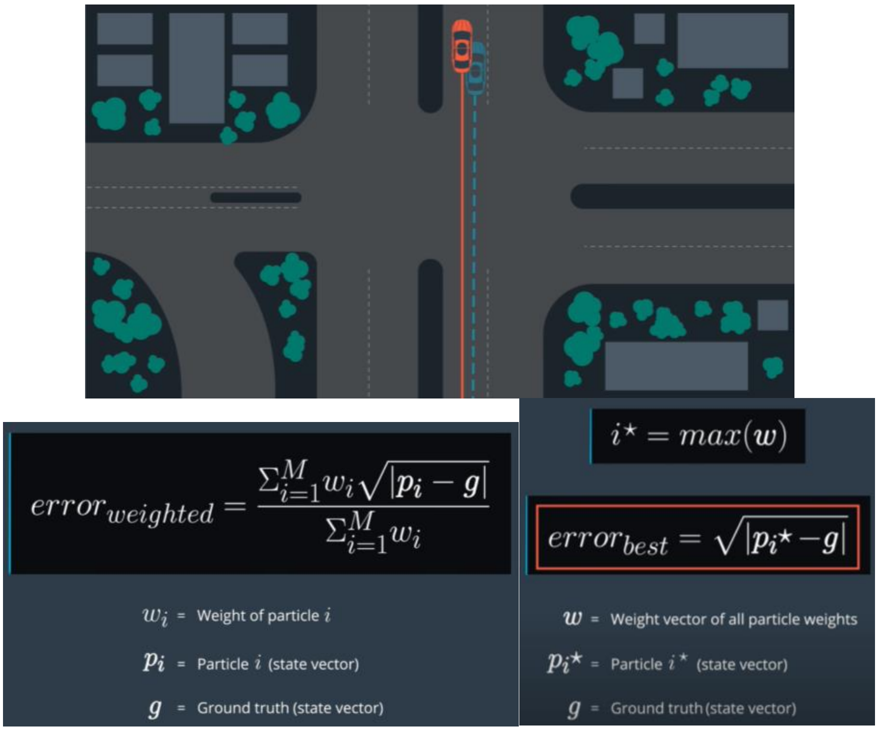
Update weights







Now that we have incorporated velocity and yaw rate measurement inputs into our filter, we must update particle weights based on LIDAR and RADAR readings of landmarks.



Note that for a vector $\mathbf{v} = (\mathrm{x,y})$, $|\mathbf{v}|$ is used here to denote the vector length or magnitude = $\sqrt{x^2 + y^2}$.

For a difference between two vectors, $\mathbf{v} = (x_v, y_v)$ and $\mathbf{w} = (x_w, y_w)$, the magnitude $|\mathbf{v} - \mathbf{w}|$ is also the Euclidean distance between these two vectors = $\sqrt{(x_v - x_w)^2 + (y_v - y_w)^2}$

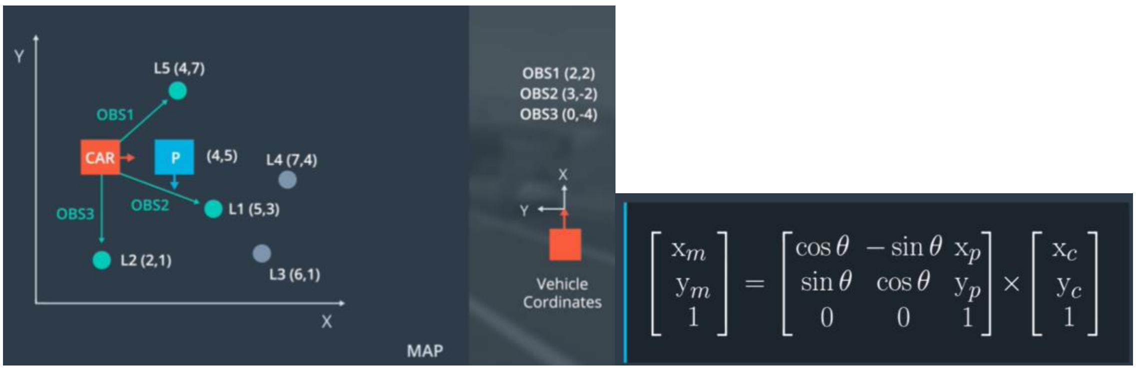
In this case $\sqrt{|p_i - g|}$ is meant to note the root squared error of a particular particle, resulting in the general form described in the quiz:

$$\sqrt{(x_p - x_g)^2 + (y_p - y_g)^2}$$

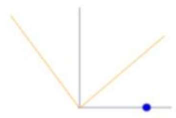
where:

- **Position RMSE** = $\sqrt{(x_p - x_g)^2 + (y_p - y_g)^2}$
- **Theta RMSE** = $\sqrt{(\theta_p - \theta_g)^2}$

Homogeneous transformation = Rotation + Traslation
From car coordinate system to map coordinate system



Particle (blue dot) in Map Frame (grey)



Particle (blue dot) in Vehicle Frame (orange)

Calculating the Particle's Final Weight

Now we that we have done the measurement transformations and associations, we have all the pieces we need to calculate the particle's final weight. The particles final weight will be calculated as the product of each measurement's Multivariate-Gaussian probability density.

The Multivariate-Gaussian probability density has two dimensions, x and y. The mean of the Multivariate-Gaussian is the measurement's associated landmark position and the Multivariate-Gaussian's standard deviation is described by our initial uncertainty in the x and y ranges. The Multivariate-Gaussian is evaluated at the point of the transformed measurement's position. The formula for the Multivariate-Gaussian can be seen below.

$$P(x,y) = \frac{1}{2\pi\sigma_x\sigma_y} e^{-\left(\frac{(x-\mu_x)^2}{2\sigma_x^2} + \frac{(y-\mu_y)^2}{2\sigma_y^2}\right)}$$