Localization: Part 8
A Brief Introduction to SLAM

Computer Science 4766/6912

Department of Computer Science
Memorial University of Newfoundland

July 26, 2017
So far we have considered localization based on a map that is provided to the robot.

Autonomous Mapping

Maps may not be so easily attainable: furniture, people, and other objects are typically not included in a building’s blueprints. Maps exist for most outdoor environments—yet they are usually incomplete and at an inadequate scale. Any human-supplied map will be based on features that humans find convenient; a robot may require different features. Thus, autonomous robot mapping is of great interest. This problem is often referred to as Simultaneous Localization And Mapping (SLAM). SLAM is challenging because the two aspects of the problem (localization and mapping) are interdependent.
So far we have considered localization based on a map that is provided to the robot.

Maps may not be so easily attainable:

- Furniture, people, and other objects are typically not included in a building’s blueprints.
- Maps exist for most outdoor environments—yet they are usually incomplete and at an inadequate scale.
- Any human-supplied map will be based on features that humans find convenient; a robot may require different features.

Thus, autonomous robot mapping is of great interest. This problem is often referred to as Simultaneous Localization And Mapping (SLAM). SLAM is challenging because the two aspects of the problem (localization and mapping) are interdependent.
Autonomous Mapping

- So far we have considered localization based on a map that is provided to the robot.
- Maps may not be so easily attainable:
  - Furniture, people, and other objects are typically not included in a building’s blueprints.

SLAM is challenging because the two aspects of the problem (localization and mapping) are interdependent.
So far we have considered localization based on a map that is provided to the robot.

Maps may not be so easily attainable:
- Furniture, people, and other objects are typically not included in a building’s blueprints.
- Maps exist for most outdoor environments—yet they are usually incomplete and at an inadequate scale.

This problem is often referred to as Simultaneous Localization And Mapping (SLAM).

SLAM is challenging because the two aspects of the problem (localization and mapping) are interdependent.
Autonomous Mapping

So far we have considered localization based on a map that is provided to the robot.

Maps may not be so easily attainable:
- Furniture, people, and other objects are typically not included in a building’s blueprints.
- Maps exist for most outdoor environments—yet they are usually incomplete and at an inadequate scale.
- Any human-supplied map will be based on features that humans find convenient.

Thus, autonomous robot mapping is of great interest. This problem is often referred to as Simultaneous Localization And Mapping (SLAM). SLAM is challenging because the two aspects of the problem (localization and mapping) are interdependent.
Autonomous Mapping

- So far we have considered localization based on a map that is provided to the robot.
- Maps may not be so easily attainable:
  - Furniture, people, and other objects are typically not included in a building’s blueprints.
  - Maps exist for most outdoor environments—yet they are usually incomplete and at an inadequate scale.
  - Any human-supplied map will be based on features that humans find convenient.

This problem is often referred to as Simultaneous Localization And Mapping (SLAM). SLAM is challenging because the two aspects of the problem (localization and mapping) are interdependent.
So far we have considered localization based on a map that is provided to the robot
Maps may not be so easily attainable:
  - Furniture, people, and other objects are typically not included in a building’s blueprints
  - Maps exist for most outdoor environments—yet they are usually incomplete and at an inadequate scale
  - Any human-supplied map will be based on features that humans find convenient; A robot may require different features

This problem is often referred to as Simultaneous Localization And Mapping (SLAM), SLAM is challenging because the two aspects of the problem (localization and mapping) are interdependent
So far we have considered localization based on a map that is provided to the robot.

Maps may not be so easily attainable:

- Furniture, people, and other objects are typically not included in a building’s blueprints.
- Maps exist for most outdoor environments—yet they are usually incomplete and at an inadequate scale.
- Any human-supplied map will be based on features that humans find convenient; A robot may require different features.

Thus, autonomous robot mapping is of great interest.
So far we have considered localization based on a map that is provided to the robot.

Maps may not be so easily attainable:

- Furniture, people, and other objects are typically not included in a building’s blueprints.
- Maps exist for most outdoor environments—yet they are usually incomplete and at an inadequate scale.
- Any human-supplied map will be based on features that humans find convenient; A robot may require different features.

Thus, autonomous robot mapping is of great interest.

This problem is often referred to as Simultaneous Localization And Mapping (SLAM),
So far we have considered localization based on a map that is provided to the robot.

Maps may not be so easily attainable:
- Furniture, people, and other objects are typically not included in a building’s blueprints.
- Maps exist for most outdoor environments—yet they are usually incomplete and at an inadequate scale.
- Any human-supplied map will be based on features that humans find convenient; A robot may require different features.

Thus, autonomous robot mapping is of great interest.

This problem is often referred to as Simultaneous Localization And Mapping (SLAM),
- SLAM is challenging because the two aspects of the problem (localization and mapping) are interdependent.
Online vs. Full SLAM

There are two different classes of SLAM problems

- **Online SLAM**: Estimates variables only up to the current time $t$.
- **Full SLAM**: Estimates the full path of the robot.

Full SLAM may be more accurate as it can take advantage of recent data to correct previous pose estimates. But it is also more costly, computationally.
Online vs. Full SLAM

There are two different classes of SLAM problems. The first is *online SLAM* which involves estimating the following:

\[
p(x_t, m | z_{1:t}, u_{1:t})
\]

This is exactly the \( \text{bel}(x_t) \) function from the chapter on localization, except that we are also estimating the map \( m \). This is called online SLAM because it estimates variables only up to the current time \( t \). Full SLAM estimates the full path of the robot:

\[
p(x_{1:t}, m | z_{1:t}, u_{1:t})
\]

Full SLAM may be more accurate as it can take advantage of recent data to correct previous pose estimates. But it is also more costly, computationally.
Online vs. Full SLAM

There are two different classes of SLAM problems. The first is online SLAM which involves estimating the following:

\[ p(x_t, m|z_{1:t}, u_{1:t}) \]
There are two different classes of SLAM problems. The first is *online SLAM* which involves estimating the following:

\[ p(x_t, m|z_{1:t}, u_{1:t}) \]

This is exactly the \( bel(x_t) \) function from the chapter on localization, except that we are also estimating the map \( m \).
Online vs. Full SLAM

There are two different classes of SLAM problems. The first is *online SLAM* which involves estimating the following:

\[ p(x_t, m | z_{1:t}, u_{1:t}) \]

This is exactly the \( bel(x_t) \) function from the chapter on localization, except that we are also estimating the map \( m \). This is called online SLAM because it estimates variables only up to the current time \( t \).
Online vs. Full SLAM

There are two different classes of SLAM problems. The first is *online SLAM* which involves estimating the following:

\[ p(x_t, m|z_{1:t}, u_{1:t}) \]

This is exactly the \( \text{bel}(x_t) \) function from the chapter on localization, except that we are also estimating the map \( m \). This is called online SLAM because it estimates variables only up to the current time \( t \). Full SLAM estimates the full path of the robot:
There are two different classes of SLAM problems. The first is *online SLAM* which involves estimating the following:

\[ p(x_t, m|z_{1:t}, u_{1:t}) \]

This is exactly the \( \text{bel}(x_t) \) function from the chapter on localization, except that we are also estimating the map \( m \). This is called online SLAM because it estimates variables only up to the current time \( t \). Full SLAM estimates the full path of the robot:

\[ p(x_{1:t}, m|z_{1:t}, u_{1:t}) \]
Online vs. Full SLAM

There are two different classes of SLAM problems. The first is online SLAM which involves estimating the following:

\[ p(x_t, m|z_{1:t}, u_{1:t}) \]

This is exactly the \( \text{bel}(x_t) \) function from the chapter on localization, except that we are also estimating the map \( m \). This is called online SLAM because it estimates variables only up to the current time \( t \). Full SLAM estimates the full path of the robot:

\[ p(x_{1:t}, m|z_{1:t}, u_{1:t}) \]

Full SLAM may be more accurate as it can take advantage of recent data to correct previous pose estimates. But it is also more costly, computationally.
Research in SLAM faces many challenges:

- Computational complexity
Research in SLAM faces many challenges:

- Computational complexity
- The **data association problem**
- Loop closure: This is the problem of recognizing a feature that was previously added to the map. It may require a costly global update step.
Research in SLAM faces many challenges:

- **Computational complexity**
- **The data association problem**
  - The map consists of features, which have some signature $s_i$ (e.g. colour or appearance). Meanwhile, the current sensor data contains features, but which ones corresponds with features in the map?
Research in SLAM faces many challenges:

- **Computational complexity**
- **The **data association problem**
  - The map consists of features, which have some signature $s_i$ (e.g. colour or appearance). Meanwhile, the current sensor data contains features, but which ones correspond with features in the map?
  - a.k.a **the correspondence problem**
Research in SLAM faces many challenges:

- **Computational complexity**
- **The data association problem**
  - The map consists of features, which have some signature \( s_i \) (e.g. colour or appearance). Meanwhile, the current sensor data contains features, but which ones corresponds with features in the map?
  - a.k.a **the correspondence problem**
- **Loop closure:**
Research in SLAM faces many challenges:

- **Computational complexity**
- **The data association problem**
  - The map consists of features, which have some signature $s_i$ (e.g. colour or appearance). Meanwhile, the current sensor data contains features, but which ones corresponds with features in the map?  
  - a.k.a **the correspondence problem**
- **Loop closure**:
  - This is the problem of recognizing a feature that was previously added to the map. It may require a costly global update step.
EKF SLAM was the earliest SLAM algorithm.
EKF SLAM was the earliest SLAM algorithm. It is based on EKF localization.
EKF SLAM was the earliest SLAM algorithm. It is based on EKF localization.

- Like EKF localization, it employs features (a.k.a point landmarks) which must be robustly recognizable.
EKF SLAM was the earliest SLAM algorithm. It is based on EKF localization.

- Like EKF localization, it employs features (a.k.a point landmarks) which must be robustly recognizable.
- The state vector is augmented to include the robot pose stacked on top of vectors describing all of the features in the map.
EKF SLAM was the earliest SLAM algorithm. It is based on EKF localization.

- Like EKF localization, it employs features (a.k.a point landmarks) which must be robustly recognizable
- The state vector is augmented to include the robot pose stacked on top of vectors describing all of the features in the map

\[ x_t = [x, y, \theta, m_1^x, m_1^y, s_1, m_2^x, m_2^y, s_2, \ldots] \]

\((m_i^x, m_i^y)\) are the coordinates of the features. \(s_i\) is the feature’s signature. e.g. colour or appearance

With \(n\) features, the size of this vector is \(3n + 3\).

Updating this large vector and its associated covariance matrix (size \((3n + 3)^2\)) can be very expensive.
EKF SLAM was the earliest SLAM algorithm. It is based on EKF localization.

- Like EKF localization, it employs features (a.k.a point landmarks) which must be robustly recognizable.
- The state vector is augmented to include the robot pose stacked on top of vectors describing all of the features in the map,

\[ x_t = [x, y, \theta, m_{1,x}, m_{1,y}, s_1, m_{2,x}, m_{2,y}, s_2 \ldots]^T \]

- \((m_{i,x}, m_{i,y})\) are the coordinates of the features
EKF SLAM was the earliest SLAM algorithm. It is based on EKF localization.

- Like EKF localization, it employs features (a.k.a point landmarks) which must be robustly recognizable.
- The state vector is augmented to include the robot pose stacked on top of vectors describing all of the features in the map,

\[ x_t = [x, y, \theta, m_{1,x}, m_{1,y}, s_1, m_{2,x}, m_{2,y}, s_2 \ldots]^T \]

- \((m_{i,x}, m_{i,y})\) are the coordinates of the features
- \(s_i\) is the feature’s signature. e.g. colour or appearance
EKF SLAM was the earliest SLAM algorithm. It is based on EKF localization.

- Like EKF localization, it employs features (a.k.a point landmarks) which must be robustly recognizable.
- The state vector is augmented to include the robot pose stacked on top of vectors describing all of the features in the map,

\[ x_t = [x, y, \theta, m_{1,x}, m_{1,y}, s_1, m_{2,x}, m_{2,y}, s_2 \ldots ]^T \]

- \((m_{i,x}, m_{i,y})\) are the coordinates of the features
- \(s_i\) is the feature’s signature. e.g. colour or appearance
EKF SLAM was the earliest SLAM algorithm. It is based on EKF localization.

- Like EKF localization, it employs features (a.k.a point landmarks) which must be robustly recognizable.
- The state vector is augmented to include the robot pose stacked on top of vectors describing all of the features in the map,

\[ x_t = [x, y, \theta, m_{1,x}, m_{1,y}, s_1, m_{2,x}, m_{2,y}, s_2, \ldots]^T \]

- \((m_{i,x}, m_{i,y})\) are the coordinates of the features
- \(s_i\) is the feature’s signature. e.g. colour or appearance

With \(n\) features, the size of this vector is \(3n + 3\).
EKF SLAM was the earliest SLAM algorithm. It is based on EKF localization.

- Like EKF localization, it employs features (a.k.a point landmarks) which must be robustly recognizable.
- The state vector is augmented to include the robot pose stacked on top of vectors describing all of the features in the map, 

\[ x_t = [x, y, \theta, m_{1,x}, m_{1,y}, s_1, m_{2,x}, m_{2,y}, s_2 \ldots]^T \]

- \((m_{i,x}, m_{i,y})\) are the coordinates of the features
- \(s_i\) is the feature’s signature. e.g. colour or appearance

With \(n\) features, the size of this vector is \(3n + 3\). Updating this large vector and its associated covariance matrix (size \((3n + 3)^2\)) can be very expensive.
Algorithm EKF SLAM known correspondences($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, c_t$):

1. $F_x = \begin{pmatrix}
1 & 0 & 0 & \cdots & 0 \\
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1
\end{pmatrix}$

2. $\bar{\mu}_t = \mu_{t-1} + F_x^T \begin{pmatrix}
-\frac{\nu_1}{\omega_t} \sin(\mu_{t-1,0} + \omega_t \Delta t) + \frac{\nu_1}{\omega_t} \sin(\mu_{t-1,0} + \omega_t \Delta t) \\
\frac{\nu_2}{\omega_t} \cos(\mu_{t-1,0} - \frac{\nu_2}{\omega_t} \cos(\mu_{t-1,0} + \omega_t \Delta t) \\
\frac{\nu_2}{\omega_t} \cos(\mu_{t-1,0} + \omega_t \Delta t)
\end{pmatrix}$

3. $G_t = I + F_x^T F_x$

4. $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + F_x^T R_t F_x$

5. $Q_t = \begin{pmatrix}
\sigma_r & 0 & 0 \\
0 & \sigma_\phi & 0 \\
0 & 0 & \sigma_s
\end{pmatrix}$

6. for all observed features $z_t^j = (r_t^j, \phi_t^j, s_t^j)^T$ do

7. if landmark $j$ never seen before

8. $\bar{\mu}_{j,x} = \mu_{t,x} + r_t^j \cos(\phi_t^j + \omega_t \Delta t)$

9. $\bar{\mu}_{j,y} = \mu_{t,y} + r_t^j \sin(\phi_t^j + \omega_t \Delta t)$

10. $\bar{\mu}_{j,z} = \mu_{t,z} + r_t^j$

11. endif

12. $\delta = \begin{pmatrix}
\delta_x \\
\delta_y
\end{pmatrix} = \begin{pmatrix}
\bar{\mu}_{j,x} - \bar{\mu}_{t,x} \\
\bar{\mu}_{j,y} - \bar{\mu}_{t,y}
\end{pmatrix}$

13. $q = \delta^T \delta$

14. $\tilde{z}_t^j = \begin{pmatrix}
\text{atan2}(\delta_y, \delta_x) - \bar{\mu}_{t,0} \\
\bar{\mu}_{j,x} \\
\bar{\mu}_{j,y}
\end{pmatrix}$

15. $F_{x,j} = \begin{pmatrix}
1 & 0 & 0 & \cdots & 0 \\
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1
\end{pmatrix}$

16. $H_t^j = \frac{1}{q} \begin{pmatrix}
\sqrt{q} \delta_x & -\sqrt{q} \delta_y & -\sqrt{q} \delta_z & \sqrt{q} \delta_y & 0 \\
\delta_y & \delta_x & -1 & \delta_y & -\delta_x & 0 \\
0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}$

17. $K_t^j = \bar{\Sigma}_t H_t^j (H_t^j \bar{\Sigma}_t H_t^{jT} + Q_t)^{-1}$

18. endfor

19. $\mu_t = \mu_t + \sum_j K_t^j (z_t^j - \tilde{z}_t^j)$

20. $\Sigma_t = (I - \sum_j K_t^j H_t^j) \bar{\Sigma}_t$

21. return $\mu_t, \Sigma_t$
Figure 10.3  EKF applied to the online SLAM problem. The robot’s path is a dotted line, and its estimates of its own position are shaded ellipses. Eight distinguishable landmarks of unknown location are shown as small dots, and their location estimates are shown as white ellipses. In (a)–(c) the robot’s positional uncertainty is increasing, as is its uncertainty about the landmarks it encounters. In (d) the robot senses the first landmark again, and the uncertainty of all landmarks decreases, as does the uncertainty of its current pose. Image courtesy of Michael Montemerlo, Stanford University.
In the localization section we were concerned with estimating
\[ p(x_t|z_{1:t}, u_{1:t}) \]
where the state space is often three-dimensional.
In the localization section we were concerned with estimating \( p(x_t|z_{1:t}, u_{1:t}) \) where the state space is often three-dimensional.

In online SLAM we estimate both \( x_t \) and \( m \); For feature-based approaches with three dimensions for all \( n \) features the size of the state space grows to \( 3n + 3 \).
In the localization section we were concerned with estimating \( p(x_t|z_{1:t}, u_{1:t}) \) where the state space is often three-dimensional.

In online SLAM we estimate both \( x_t \) and \( m \); For feature-based approaches with three dimensions for all \( n \) features the size of the state space grows to \( 3n + 3 \).

Another quantity that should ideally be incorporated into the state space is the set of correspondences between sensor and map-based features:

\[
p(x_t, m, c_t|z_{1:t}, u_{1:t})
\]
In the localization section we were concerned with estimating $p(x_t|z_{1:t}, u_{1:t})$ where the state space is often three-dimensional.

In online SLAM we estimate both $x_t$ and $m$; For feature-based approaches with three dimensions for all $n$ features the size of the state space grows to $3n + 3$.

Another quantity that should ideally be incorporated into the state space is the set of correspondences between sensor and map-based features:

$$p(x_t, m, c_t|z_{1:t}, u_{1:t})$$
In the localization section we were concerned with estimating 
\( p(x_t|z_{1:t}, u_{1:t}) \) where the state space is often three-dimensional.

In online SLAM we estimate both \( x_t \) and \( m \); For feature-based approaches with three dimensions for all \( n \) features the size of the state space grows to \( 3n + 3 \).

Another quantity that should ideally be incorporated into the state space is the set of correspondences between sensor and map-based features:

\[
p(x_t, m, c_t|z_{1:t}, u_{1:t})
\]

(Often \( c_t \) is not estimated and the ‘best guess’ is used instead.)

In practice, SLAM algorithms rely on approximations which have recently allowed real-time performance in some applications [see video at https://youtu.be/z_NJxbkQnBU]