MonoSLAM: Real-Time Single Camera SLAM

Andrew J. Davison, Ian D. Reid, Member, IEEE, Nicholas D. Molton, and Olivier Stasse, Member, IEEE

Abstract—We present a real-time algorithm which can recover the 3D trajectory of a monocular camera, moving rapidly through a previously unknown scene. Our system, which we dub MonoSLAM, is the first successful application of the SLAM methodology from mobile robotics to the “pure vision” domain of a single uncontrolled camera, achieving real time but drift-free performance inaccessible to Structure from Motion approaches. The core of the approach is the online creation of a sparse but persistent map of natural landmarks within a probabilistic framework. Our key novel contributions include an active approach to mapping and measurement, the use of a general motion model for smooth camera movement, and solutions for monocular feature initialization and feature orientation estimation. Together, these add up to an extremely efficient and robust algorithm which runs at 30 Hz with standard PC and camera hardware. This work extends the range of robotic systems in which SLAM can be usefully applied, but also opens up new areas. We present applications of MonoSLAM to real-time 3D localization and mapping for a high-performance full-size humanoid robot and live augmented reality with a hand-held camera.

Index Terms—Autonomous vehicles, 3D/stereo scene analysis, tracking.
Simultaneous Localization and Mapping

Given a single camera feed, estimate the 3D position of the camera and the 3D positions of all landmark points in the world.
Real-Time Camera Tracking in Unknown Scenes
General Filtering Equations

\[ P(x_t|z_{1:t}) \propto P(z_t|x_t) \int_{x_{t-1}} P(x_t|x_{t-1}) P(x_{t-1}|z_{1:t-1}) dx_{t-1} \]

**Prediction:**

\[ P(x_t|z_{1:t-1}) = \int_{x_{t-1}} P(x_t|x_{t-1}) P(x_{t-1}|z_{1:t-1}) dx_{t-1} \]

**Update:**

\[ P(x_t|z_{1:t}) = P(z_t|x_t) P(x_t|z_{1:t-1}) \]
General Filtering Equations

\[ P(\mathbf{x}_t | z_{1:t}) \propto P(z_t | \mathbf{x}_t) \int_{\mathbf{x}_{t-1}} P(\mathbf{x}_t | \mathbf{x}_{t-1}) P(\mathbf{x}_{t-1} | z_{1:t-1}) d\mathbf{x}_{t-1} \]

\text{Prediction:}

\[ P(\mathbf{x}_t | z_{1:t-1}) = \int_{\mathbf{x}_{t-1}} P(\mathbf{x}_t | \mathbf{x}_{t-1}) P(\mathbf{x}_{t-1} | z_{1:t-1}) d\mathbf{x}_{t-1} \]

\text{Update:}

\[ P(\mathbf{x}_t | z_{1:t}) = P(z_t | \mathbf{x}_t) P(\mathbf{x}_t | z_{1:t-1}) \]
What is the camera (robot) state?

\[ \mathbf{x}_c = \begin{bmatrix} \mathbf{r} \\ \mathbf{q} \\ \mathbf{v} \\ \mathbf{\omega} \end{bmatrix} \]

- position
- rotation (quaternion)
- velocity
- angular velocity

13 total
What is the camera (robot) state?

\[ x_c = \begin{bmatrix}
  r \\
  q \\
  v \\
  \omega
\end{bmatrix} \]

- Position: 3 dimensions
- Rotation (quaternion): 4 dimensions
- Velocity: 3 dimensions
- Angular velocity: 3 dimensions

13 total dimensions.
What is the world (robot+environment) state?

\[ x = \begin{bmatrix} x_c \\ y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \]

- state of the camera
- location of feature 1
- location of feature 2
- location of feature N

13+3N total
What is the world (robot+environment) state?

\[ x = \begin{bmatrix} x_c \\ y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \]

- state of the camera: 13
- location of feature 1: 3
- location of feature 2: 3
- location of feature N: 3

Total: 13 + 3N
What is the covariance (uncertainty) of the world state?

\[
\Sigma = \begin{bmatrix}
\sum x_c x_c & \sum x_c y_1 & \cdots & \sum x_c y_N \\
\sum y_1 x_c & \sum y_1 y_1 & \cdots & \sum y_1 y_N \\
\vdots & \vdots & \ddots & \vdots \\
\sum y_N x_c & \sum y_N y_1 & \cdots & \sum y_N y_N
\end{bmatrix}
\]

What are the dimensions?

\((13+3N) \times (13+3N)\)
General Filtering Equations

\[ P(x_t | z_{1:t}) \propto P(z_t | x_t) \int_{x_{t-1}} P(x_t | x_{t-1})P(x_{t-1} | z_{1:t-1})dx_{t-1} \]

**What are the observations?**

**Prediction:**
\[ P(x_t | z_{1:t-1}) = \int_{x_{t-1}} P(x_t | x_{t-1})P(x_{t-1} | z_{1:t-1})dx_{t-1} \]

**Update:**
\[ P(x_t | z_{1:t}) = P(z_t | x_t)P(x_t | z_{1:t-1}) \]
General Filtering Equations

\[
P(x_t|z_{1:t}) \propto P(z_t|x_t) \int_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}|z_{1:t-1})dx_{t-1}
\]

**Prediction:**

\[
P(x_t|z_{1:t-1}) = \int_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}|z_{1:t-1})dx_{t-1}
\]

**Update:**

\[
P(x_t|z_{1:t}) = P(z_t|x_t)P(x_t|z_{1:t-1})
\]

*What are the observations?*
Observations are...

detected visual features of landmark points.
General Filtering Equations

\[ P(x_t|z_{1:t}) \propto P(z_t|x_t) \int_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}|z_{1:t-1})dx_{t-1} \]

Prediction:

\[ P(x_t|z_{1:t-1}) = \int_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}|z_{1:t-1})dx_{t-1} \]

Update:

\[ P(x_t|z_{1:t}) = P(z_t|x_t)P(x_t|z_{1:t-1}) \]
General Filtering Equations

\[ P(x_t | z_{1:t}) \propto P(z_t | x_t) \int_{x_{t-1}} P(x_t | x_{t-1}) P(x_{t-1} | z_{1:t-1}) dx_{t-1} \]

**Prediction:**

\[ P(x_t | z_{1:t-1}) = \int_{x_{t-1}} P(x_t | x_{t-1}) P(x_{t-1} | z_{1:t-1}) dx_{t-1} \]

What does the prediction step look like?

**Update:**

\[ P(x_t | z_{1:t}) = P(z_t | x_t) P(x_t | z_{1:t-1}) \]
What is the motion model? \[ P(x_t|x_{t-1}) \]

What is the form of the belief? \[ P(x_t|z_{1:t-1}) \]
What is the motion model? \( P(x_t|x_{t-1}) \)

**Landmarks:**
constant position (identity matrix)

**Camera:**
constant velocity (not identity matrix!)

What is the form of the belief? \( P(x_t|z_{1:t-1}) \)
What is the motion model? \[ P(x_t | x_{t-1}) \]

**Landmarks:**
constant position
(identity matrix)

**Camera:**
constant velocity
(not identity matrix!)

What is the form of the belief? \[ P(x_t | z_{1:t-1}) \]

**Gaussian!**
(everything is parametrized by a mean and Gaussian)
Constant Velocity Motion Model

\[
\mathbf{r}_t = \mathbf{r}_{t-1} + \mathbf{v}_{t-1} \Delta t
\]

(position)

\[
\mathbf{q}_t = \mathbf{q}_{t-1} \times [\mathbf{q}(\mathbf{\omega}) \Delta t]
\]

(rotation (quaternion))

\[
\mathbf{v}_t = \mathbf{v}_{t-1}
\]

(velocity)

\[
\mathbf{\omega}_t = \mathbf{\omega}_{t-1}
\]

(angular velocity)
Gaussian noise uncertainty (only on velocity)

\[ v_t = v_{t-1} + V \]
\[ \omega_t = \omega_{t-1} + \Omega \]

\[ V \sim \mathcal{N}(0, \begin{bmatrix} \sigma_v & 0 & 0 \\ 0 & \sigma_v & 0 \\ 0 & 0 & \sigma_v \end{bmatrix}) \]

\[ \Omega \sim \mathcal{N}(0, \begin{bmatrix} \sigma_w & 0 & 0 \\ 0 & \sigma_w & 0 \\ 0 & 0 & \sigma_w \end{bmatrix}) \]
Prediction (mean of camera state):

\[ P(x_t|z_{1:t-1}) = \int_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}|z_{1:t-1})dx_{t-1} \]

\[ f_t = \begin{bmatrix} r_t \\ q_t \\ v_t \\ \omega_t \end{bmatrix} = \begin{bmatrix} r_{t-1} + v_{t-1}\Delta t \\ q_{t-1} + q(\omega)_{t-1}\Delta t \\ v_{t-1} \\ \omega_{t-1} \end{bmatrix} \]
Prediction (covariance of camera state): 

\[ P(x_t|z_{1:t-1}) = \int_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}|z_{1:t-1})dx_{t-1} \]

\[ \Sigma_{xx} = \left( \frac{\partial f_t}{\partial x} \right) \Sigma_{xx} \left( \frac{\partial f_t}{\partial x} \right)^\top + Q_t \]

Where does this motion model approximation come from?
What are the dimensions?
Skipping over many details…
Prediction (covariance of camera state):

\[ P(x_t | z_{1:t-1}) = \int_{x_{t-1}} P(x_t | x_{t-1}) P(x_{t-1} | z_{1:t-1}) dx_{t-1} \]

\[ \Sigma_{xx} = \begin{bmatrix} \frac{\partial f_t}{\partial x} \end{bmatrix} \Sigma_{xx} \begin{bmatrix} \frac{\partial f_t}{\partial x} \end{bmatrix}^\top + Q_t \]

Bit of a pain to compute this term…
We just covered the **prediction** step for the camera state

\[
P(x_t|z_{1:t-1}) = \int_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}|z_{1:t-1})dx_{t-1}
\]

\[
f_t = \begin{bmatrix} r_t \\ q_t \\ v_t \\ \omega_t \end{bmatrix} = \begin{bmatrix} r_{t-1} + v_{t-1} \\ q_{t-1} + q(\omega)_{t-1} \\ v_{t-1} \\ \omega_{t-1} \end{bmatrix}
\]

\[
\Sigma_{xx} = \frac{\partial f_t}{\partial x} \Sigma_{xx} \frac{\partial f_t}{\partial x}^\top + Q_t
\]

Now we need to do the **update** step!
General Filtering Equations

\[ P(x_t|z_{1:t}) \propto P(z_t|x_t) \int_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}|z_{1:t-1})dx_{t-1} \]

**Prediction:**

\[ P(x_t|z_{1:t-1}) = \int_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}|z_{1:t-1})dx_{t-1} \]

**Update:**

\[ P(x_t|z_{1:t}) = P(z_t|x_t)P(x_t|z_{1:t-1}) \]
Belief state \[ P(\mathbf{x}_t | z_{1:t}) = P(z_t | x_t) P(x_t | z_{1:t-1}) \]

State observation

Predicted State

What are the observations?

2D projections of 3D landmarks
Recall, the state includes the 3D location of landmarks

\[ x = \begin{bmatrix} x_c \\ y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \]

What is the projection from 3D point to 2D image point?
Observation Model

\[ P(z_t|x_t) \]

If you know the 3D location of a landmark, what is the 2D projection?

Non-linear observation model

\[ h \sim P_y y \]

2D Image Point
Camera matrix
3D World Point

\[ P = K [R|T] \]

What do we know about \( P \)?

How do we make the observation model linear?
\[ H = \frac{\partial h}{\partial x} \]

(2n x 13)

n: number of visible points

I will spare you the pain of deriving the partial derivative...
\[ P(x_t | z_{1:t}) = P(z_t | x_t) P(x_t | z_{1:t-1}) \]

**Update step (mean):**

\[ x_t = x_t + K_t (z_t - h(y; x_t)) \]

Updated state  Predicted state  Matched 2D features  2D projection of 3D point

**Update step (covariance):**

\[ \Sigma_t = (I - K_t H_t) \Sigma_t \]

Covariance (updated)  Identity  Kalman gain  Jacobian  Covariance (predicted)
Kintinuous: Spatially Extended Kinect Fusion

Thomas Whelan, John McDonald
National University of Ireland Maynooth, Ireland

Michael Kaess, Maurice Fallon, Hordur Johannsson, John J. Leonard
Computer Science and Artificial Intelligence Laboratory, MIT, USA