The Evolution of Motion Estimation and Real-time 3D Reconstruction

Lukas Koestler & Simon Weber

Nov. 23, 2021

Chair of Computer Vision & Artificial Intelligence, Technical University of Munich

This lecture should introduce you to the fundamentals of Motion Estimation and Real-time 3D Reconstruction. These are

- Problem Description
- Terminology and System Design
- Mathematical Notation and Concepts

You can cite this lecture within your presentation and assume that students know its content.

Contents

1. Problem Description

Motivation

We humans can estimate the motion as well as the geometry from a monocular video alone.[1]

Developing algorithms for real-time motion estimation and 3D reconstruction can endow computers with this fundamental skill.

[1]You can convince yourself of this fact by watching a video on a screen. In the physical world humans also use inertial data from the vestibular system (ear), stereo data from both eyes, etc. for motion estimation.

Brief History

The notion of perspective projection has its roots among the ancient Greeks (Euclid of Alexandria, ca. 400 B.C.). (Cremers 2019)

The first work on the problem of multiple view geometry was that of Erwin Kruppa (Kruppa 1913) who showed that two views of five points are sufficient to determine the motion between the two views and the 3D locations of the points.

Visual SLAM/Odometry dates back to at least “Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover” (Moravec 1980).

2. Terminology and System Design

Sensors

Different algorithms can be distinguished based on the sensor-data used:

- Camera: Visual-SLAM (V-SLAM)
- Camera + Inertial Measurement Unit (IMU): Visual-Inertial SLAM (VI-SLAM)
- Camera + Depth Camera (RGB-D Camera): RGB-D SLAM
- Type of Camera: Monocular or Stereo SLAM

Combination of the above and further sensors, e.g. LiDAR, are possible.

Output

Different algorithms can be distinguished based on the output they provide. The nomenclature is quite ambiguous:

- Odometry/Tracking: Provides the motion of the camera, without global correction. If the camera moves within a constrained environment (room) the trajectory will drift away over time.
- SLAM: Provides the motion of the camera, with global correction, as well as the geometry (e.g. a point cloud). The correction w.r.t. the map prohibits the position to drift over time in a constrained environment.
- 3D Reconstruction: Provides motion and geometry and the focus lies on the reconstructed geometry.

Keyframes

Typical cameras operate at 20–30 frames per second (FPS), which results in ca. 30 milliseconds (ms) per frame. The information in two temporally adjacent frames is highly redundant.

Therefore, many algorithms operate on a sparser set of frames called keyframes, e.g. every 10th frame. The exact keyframe management is a crucial part of every SLAM system.

Every frame can still be used for short-term camera tracking. The frontend processes every frame and selects keyframes and the backend optimizes the keyframe poses.

Filtering & Optimization

Typically, a SLAM algorithm estimates the trajectory (keyframe poses) as-well-as landmarks, e.g. the 3D locations of feature points.

- Filtering: The algorithm has a current state of the poses & landmarks and probabilistically updates the state with the current information by using e.g. the Kalman Filter (Kalman 1960).
- Bundle Adjustment: Optimizing the poses and landmark positions simultaneously using e.g. the Gauss-Newton algorithm.
- Pose Graph Optimization: Optimizing only the poses but not the landmark positions.

Global Scale

For monocular cameras the global scale of the reconstructed scene in unobservable, i.e. even a perfect algorithm using perfect data cannot produce a metrically scaled trajectory and 3D reconstruction.[2]

The scale becomes observable by using additional sensors, e.g. an IMU or a depth camera. The scale can also be obtained by using prior information about the world, e.g. that a room is generally 2.5 meters high – this prior information can be encoded in a neural network.

[2]You can convince yourself of this fact by considering a camera that is moving in a house, or a puppet house. If the puppet house were perfect, there is no way of knowing in which of the two the camera is.

Feature Points

Many algorithms abstract the image into a set of feature points with descriptors, which are used to match the feature points.

This approach has the advantage that using matched features, the problem of SLAM is ”purely geometrical”, however, erroneous matches can be hard to deal with after the matching.

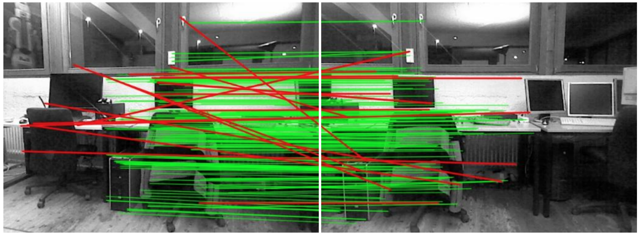

Figure 1: Correct (green) and erroneous (red) feature matches.

3. Mathematical Notation and Concepts

Acknowledgement

This section is inspired by the practical course Vision-based Navigation that was initially developed by Vladyslav Usenko and which we can highly recommend! Ressources:

- The video lectures for Vision-based Navigation[3]
- The Multiple View Geometry (MVG) lecture[4]
- The tutorial on SE(3) by Blanco-Claraco[5]
- GTSAM has a blog-style introduction on Lie groups[6]

[3] Demmel et al. 2020.
[4] Cremers 2019.
[5] Blanco-Claraco 2021.
[6] Mattamala 2021.

3. Mathematical Notation and Concepts

3.1 Rigid-body Motion

Representations of Rotation

The pose of the camera is comprised of the position and the rotation. The position can be easily represented by a vector in $\mathbb{R}^3$, but representing the rotation is more challenging.

There are at least the following representations:

- SO(3) matrices
  $\rightarrow$ Potentially the most used representation in SLAM
- Quaternions
  $\rightarrow$ Good representation that is used in SLAM
- Euler Angles
  $\rightarrow$ Hard to combine rotations & gimbal lock

The Orthogonal Group O(n)

A matrix $A \in \mathbb{R}^{3\times 3}$ is called orthogonal if $A^{-1} = A^\top$.

The group

$$\mathrm{O}(n) = \{A \in \mathbb{R}^{3\times 3} \mid A^{-1} = A^\top\}$$

is called the orthogonal group.

The special orthogonal group

$$\mathrm{SO}(n) = \{A \in \mathrm{O}(n) \mid \det(A) = 1\}$$

is a subgroup of the orthogonal group and is the group of 3-dimensional rotation matrices.

Representation of Rigid-body Motion

A rigid body motion $g_t(x) : \mathbb{R}^3 \to \mathbb{R}^3$ that consists of a rotation $R \in \mathrm{SO}(3)$ and translation $T \in \mathbb{R}^3$ can be written as

$$g_t(x) = Rx + T\,.$$

We can use homogeneous coordinates to simplify the notation

$$g_t(x) = \begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ 1 \end{pmatrix}\,.$$

The special Euclidean group

$$\mathrm{SE}(3) = \left\{ \begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix} \mid R \in \mathrm{SO}(3),\ T \in \mathbb{R}^3 \right\}$$

represents the rigid-body motions in 3D.

3. Mathematical Notation and Concepts

3.2 Manifold Optimization

Optimization over Rotations

Generally, the SLAM problem might involve an optimization problem of the form

$$\min_{R \in SO(3)} f(R)\,,$$

where $f : \mathbb{R}^{3\times3} \rightarrow \mathbb{R}$ is a non-linear function.

Naively, one could consider

$$\min_{R \in \mathbb{R}^{3\times3}} f(R)\,, \quad \text{s.t.} \quad R^{-1} = R^{\top} \quad \text{and} \quad det(R) = 1\,.$$

However, the problem has dimension 9 instead of 3 and is a constrained problem, which is generally harder to solve.

Manifold Optimization to the rescue!

Manifolds

Informally, a manifold is a space that is not Euclidean, but can locally be represented by an Euclidean space (= tangent space).

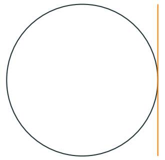

Figure 2: The circle in $\mathbb{R}^2$ with its tangent space at the point $(1, 0)^\top$.

We can locally consider the optimization problem on the tangent space to obtain an update and then map back to the manifold. (Absil et al. 2009)

The Lie Group SO(3)

We are specifically interested in the group of rotation matrices SO(3). Because SO(3) is also a manifold it is called a Lie group.

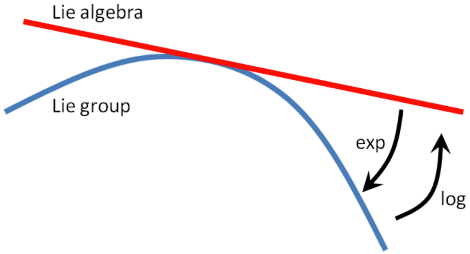

Figure 3: The mapping from the Lie algebra (=tangent space) to the Lie group is called the exponential map. Its inverse is the logarithm.

The Lie Algebra so(3)

The Lie algebra so(3) is the tangent space at the identity of the group SO(3) and it is comprised of the skew-symmetric matrices

$$\mathrm{so}(3) = \{\widehat{w} \in \mathbb{R}^{3\times 3} | \widehat{w}^\top = -\widehat{w}\} .$$

For each skew-symmetric matrix $\widehat{w} \in \mathbb{R}^{3\times 3}$ there exists $w \in \mathbb{R}^3$:

$$\widehat{w}\, y = w \times y \;\; \forall y \in \mathbb{R}^3 , \qquad \widehat{w} = \begin{pmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{pmatrix} .$$

The Exponential Map

The exponential map $\exp : \mathrm{so}(3) \to \mathrm{SO}(3)$ is given by

$$R = \exp(\widehat{w}) = \sum_{n=0}^{\infty} \frac{\widehat{w}^n}{n!} \,.$$

Here, $\exp$ is the matrix exponential which is implemented in many linear algebra libraries. For $\exp : \mathrm{so}(3) \to \mathrm{SO}(3)$ there exists the Rodrigues' Formula:

$$\exp(\widehat{w}) = I + \frac{\sin(|w|)}{|w|}\,\widehat{w} + \frac{1-\cos(|w|)}{|w|^2}\,\widehat{w}^2 \,.$$

The Logarithm

The logarithm $\mathbf{log} : \mathrm{SO}(3) \to \mathrm{so}(3)$ is given by $\widehat{w} = \mathbf{log}(R)$ with

$$|w| = \cos^{-1}\left(\frac{\operatorname{trace}(R) - 1}{2}\right), w = \frac{|w|}{2\sin(|w|)}\begin{pmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{pmatrix},$$

for $R \neq I$ and $\widehat{w} = 0$ for $R = I$.

The above statement says: Any rotation $R \in \mathrm{SO}(3)$ can be represented by rotating by an angle $|w|$ around an axis $w/|w|$ as defined above. This representation is called angle-axis representation and $w \in \mathbb{R}^3$ a rotation vector.

Optimization over SO(3) using so(3)

Let R^k be the current iterate for the optimization problem

$$\min_{R \in \mathrm{SO}(3)} f(R)\,.$$

We consider the increment $\widehat{w} \in \mathrm{so}(3)$ s.t. $R^{k+1} = \exp(\widehat{w})R^k$ and its coordinate representation $w \in \mathbb{R}^3$. For gradient descent, we compute

$$\Delta w = -\eta \, \frac{\partial f(\exp(\widehat{w})R^k)}{\partial w}\,.$$

In $w \in \mathbb{R}^3$ we have an unconstrained optimization problem with a minimal number of parameters!

For more details, e.g. how to compute the gradient, please consider Cremers 2019 or Demmel et al. 2020.

3. Mathematical Notation and Concepts

3.3 Camera Models

Pinhole Camera

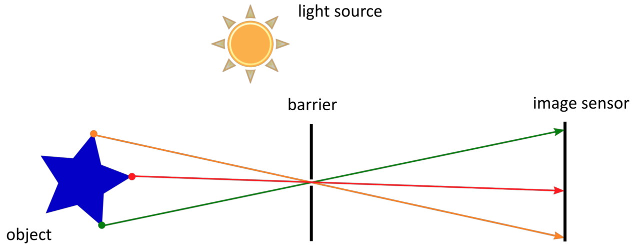

Figure 4: The pinhole camera is parametrized by the focal lengths f_x, f_y and the optical center c_x, c_y.

A 3D point $(x, y, z)^\top$ is projected to the pixel coordinates $(u, v)^\top$

$$u = f_x \frac{x}{z} + c_x \,, \quad v = f_y \frac{y}{z} + c_y \,.$$

Distortion

The pinhole camera model is often, especially for wide-angle lenses, not sufficient to model the physics of the camera. Often images are undistorted beforehand.

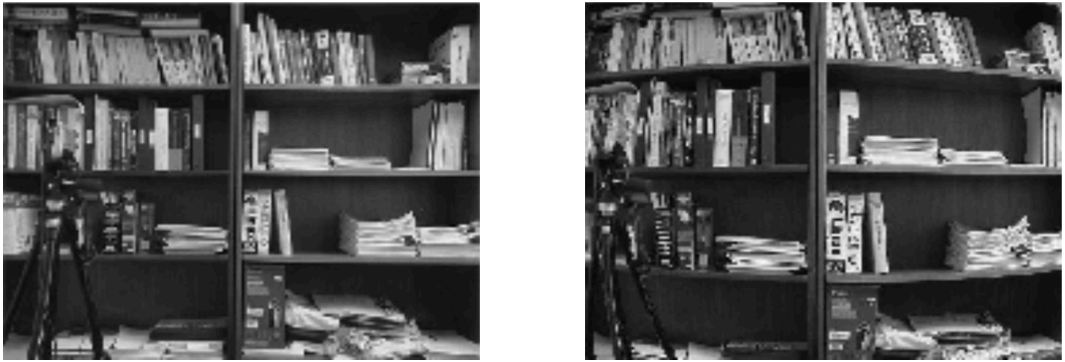

Figure 5: A bookshelf w/o distortion (left) and w/ distortion (right).

3. Mathematical Notation and Concepts

3.4 Optimization

Energy & Residuals

Many SLAM problems can be formulated as a least squares optimization problem

$$\min_{x \in X} E(x) = \min_{x \in X} \frac{1}{2} \sum_k r_k(x)^\top r_k(x) \,,$$

with energy $E : X \to \mathbb{R}$ and residuals $r_k : X \to \mathbb{R}^{o_k}$.

Kinds of residuals:

- Projective: $(u(x), v(x))^\top - (u_{\text{measured}}, v_{\text{measured}})^\top$
- Priors on motion, camera parameters, ...
- Based on depth measurements
- ...

Least Squares

By using manifold optimization, the problem can usually be transformed to

$$\min_{x \in \mathbb{R}^d} \frac{1}{2} \sum_k r_k(x)^\top r_k(x) \,,$$

a non-linear, un-constrained optimization problem.

In simple cases, one can solve

$$\frac{\partial E(x)}{\partial x} = 0$$

analytically. Otherwise, we use iterative methods starting at x^0

- At iteration n find an increment $\Delta x^n = \arg\min_{\Delta x} E(x^n + \Delta x)$
- If the (change in) error is small enough stop
- If not, set $x^{n+1} = x^n + \Delta x^n$ and continue

Gradient Descent

Use a first order Taylor expansion of the energy $E(x + \Delta x) = E(x) + G(x)\Delta x + O(\|\Delta x\|^2)$ and the update step

$$\Delta x = -\eta G(x)\,.$$

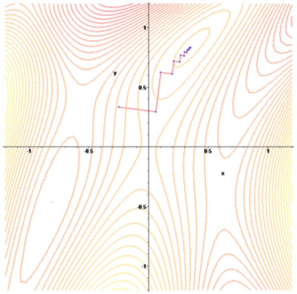

Figure 6: Slow convergence due to "zig-zag" pattern.

Second Order Methods

Use a second order Taylor expansion of the energy

$$E(x+\Delta x) = \underbrace{E(x) + G(x)\Delta x + \frac{1}{2}\Delta x^\top H(x)\Delta x}_{\tilde{E}(\Delta x;x)} + O(\|\Delta x\|^3)\,.$$

Setting $\partial_{\Delta x}\tilde{E}(\Delta x; x) = 0$ gives the update step

$$\Delta x = -H^{-1}(x)G(x)\,.$$

This is called Newton's method. The convergence is faster than for first order methods, but computing $H(x)$ might be hard. Can we avoid computing $H(x)$ and keep the convergence speed?

Gauss-Newton Method

Key idea: Utilize the least-squares structure of $E(x)$ by applying the Taylor expansion to the residuals

$$
\begin{aligned}
E(x+\Delta x) &= \frac{1}{2} r(x+\Delta x)^\top r(x+\Delta x) \\
&= \frac{1}{2}\left(r(x) + J(x)\Delta x + O(\|\Delta x\|^2)\right)^\top \left(r(x) + J(x)\Delta x + O(\|\Delta x\|^2)\right) \\
&= E(x) + r(x)^\top J(x)\Delta x + \frac{1}{2}\Delta x^\top J(x)^\top J(x)\Delta x + O(\|\Delta x\|^2)
\end{aligned}
$$

We obtain the update step

$$\Delta x = -(J(x)^\top J(x))^{-1} J(x)^\top r(x)\,.$$

The Hessian $H(x)$ is approximated by $J(x)^\top J(x)$, which is only positive semi-definite and thus the inversion might be undefined.

Levenberg-Marquardt Method

Key idea: The Taylor approximation is only valid in a trust region. Evaluate the approximation by

$$\rho = \frac{\text{real descent}}{\text{predicted descent}} = \frac{r(x + \Delta x) - r(x)}{J(x)\Delta x}\,.$$

Increase (ρ large) or decrease (ρ small) the region. Use regularization to obtain an increment "within" the region

$$E(x + \Delta x) = \frac{1}{2} r(x + \Delta x)^\top r(x + \Delta x) + \lambda \|\Delta x\|^2\,.$$

Choose λ based on ρ. For $\lambda = 0$ LM becomes GN, for $\lambda \to \infty$ LM becomes gradient descent.

Optimization Summary

- GN and LM are used most often in SLAM
- For all iterative methods the initialization is important
- More methods (semi-definite relaxation, dog-leg, ...) exist and might be advantageous for specific problems
- Sometimes custom implementations might be more efficient than libraries (GTSAM, Ceres, g2o, ...)
- SLAM-specific tricks exist, e.g. null-space handling, and are sometimes not well-documented in the literature
- Insight into the SLAM problem, e.g. residual definition, can make the resulting optimization problem much easier

Figure References

- Figure 1: Demmel et al. 2020
- Figure 2: Own
- Figure 3: Cremers 2019
- Figure 4: Demmel et al. 2020
- Figure 5: Cremers 2019
- Figure 6: Demmel et al. 2020

Hints

- If one of the web pages mentioned should not be available anymore you can usually find it saved in the wayback machine *`https://web.archive.org/`*.

4. References

Absil, P.-A., R. Mahony, and R. Sepulchre (2009). *Optimization algorithms on matrix manifolds*. (Cit. on p. 22).

Blanco-Claraco, J. L. (2021). “A tutorial on SE(3) transformation parameterizations and on-manifold optimization”. In: URL: https://arxiv.org/abs/2103.15980 (cit. on p. 15).

Cremers, D. (2019). *Computer Vision II: Multiple View Geometry*. URL: https://vision.in.tum.de/teaching/online/mvg (cit. on pp. 6, 15, 27, 39).

References ii

Demmel, N. and D. Schubert (2020). *Vision-based Navigation*. URL: *https://vision.in.tum.de/teaching/ws2020/visnav_ws2020* (cit. on pp. 15, 27, 39).

Kalman, R. E. (1960). “A new approach to linear filtering and prediction problems”. In: (cit. on p. 11).

Kruppa, E. (1913). *Zur Ermittlung eines Objektes aus zwei Perspektiven mit innerer Orientierung*. (Cit. on p. 6).

Mattamala, M. (2021). *Reducing the uncertainty about the uncertainties*. *https://gtsam.org/2021/02/23/uncertainties-part1.html* (cit. on p. 15).

Moravec, H. P. (1980). “Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover”. PhD thesis (cit. on p. 6).