Chapter 8: Direct Approaches to Visual SLAM

Konrad Koniarski

2016/07/26

1 Direct Methods

1.1 Classical Approaches to Multiple View Reconstruction

In the past chapters we have studied **classical approaches to multiple view reconstruction**. These methods tackel the problem of structure and motion estimation (or visual SLAM) in several steps:

- A set of **feature points** is extracted from the images - ideally points such **corners** which can be reliably identified in subsequent images as well.
- One determines a **correspondence of these points across the various images**. This can be done either through local tracking (using optical flow approaches) or by random sampling of possible partners based on a feature description (SIFT, SURF, etc.) associated with each point.
- The **camera motion is estimated** based on a set of corresponding points. In many approaches this is done by a series of algorithms such as the **eight-point algorithm** or the five-point algorithm followed by **bundle adjustment**.
- For a given camera motion one can then compute a **dense reconstruction** using photometric stereo approaches.

1.2 Shortcomings of Classical Approaches

Such classical approaches are **indirect** in the sense that they do not compute structure and motion directly from the images but rather from a spare set of precomputed feature points. Despite a number of successes, they have several drawbacks:

- From the point of view of statistical inference, they are **suboptimal**: in the selection of feature points much potentially valuable information constrained in the colors of each image is discarded.
- They invariably **lack robustness**: Errors in the point correspondence may have devastating effects on the estimated camera motion. Since one often selects very few point pairs only (8 points for the eigth-point algorithm,5 points for the five point algorithm), any incorrect correspondence will lead to an incorrect motion estimate.
- They do not address the **highly coupled problems of motion estimation and dense structure estimation**. They merely do so far a sparse set of points. As a consequence, improvements in the estimated dense geometry will not be used to improve the camera motion estimates.

1.3 Toward Direct Approaches to Multiview Reconstruction

In the last few years, researchers have been promoting **direct approaches to multi-view reconstruction**. Rather than extracting a sparse set of feature points to determine the camera motion, **direct methods aim at estimating camera motion and dense or semi-dense scene geometry directly from the input images**. This has several advantages:

- Direct methods tend to be **more robust** to noise and other nuisances because they **exploit all available input information**.
- Direct methods provide a **semi-dense geometric reconstruction** of the scene which goes well beyond the sparse point cloud generated by the eight-point algorithm or bundle adjustment. Depending on the application, a separate dense reconstruction step may no longer be necessary.
- Direct methods are **typically faster** because the feature-point extraction and correspondence finding is omitted: They can provide fairly accurate camera motion and scene structure in real-time on a CPU.

1.4 Direct Methods for Multi-view Reconstruction

In the following, we will briefly review several recent work on direct methods for multiple-view reconstruction:

- the methods for **Stuhmer, Gumhold, Cremers, DAGM 2010** which computes dense geometry from a handheld camera in real-time. For given camera motions, the dense reconstruction problem is computed directly from the images.
- the methods of **Steinbrucker, Sturm, Cremers, 2011** and **Kerl, Sturm, Cremers, 2013** which directly compute the camera mation of an RGB-D camera without feature extraction.
- the methods of **Newcombe, Lovegrove, Davison, ICCV 2011** which directly determines dense geometry and caemra motion from the images.
- the method of **Engel, Sturm, Cremers ICCV 2013** and **Engel, Schops, Cremers ECCV 2014** which directly computes camera motion and semi-dense geometry for a handheld (monocular) camera.

2 Realtime Dense Geometry

2.1 Real Dense Geometry from a Handheld Camera

Let $g_i \in SE(3)$ be the rigid body motion from the first camera to the i-th camera, and let $I_i : \Omega \to \mathbb{R}$ be the i-th image. A **dense depth map** $h : \Omega \to \mathbb{R}$ can be computed by solving the optimization problem:

$$\min_h \sum_{i=2}^{n} \int_\Omega |I_1(x) - I_i(\pi g_i(hx))| dx + \lambda \int_\Omega |\nabla h| dx \tag{1}$$

where x is represented in homogeneous coordinates and hx is the corresponding 3D point.

Like in optical flow estimation, the unknown depth map should be such that for all pixels $x \in \Omega$, the transformation into the other images I_i should give rise to the same color as in the reference image I_1.

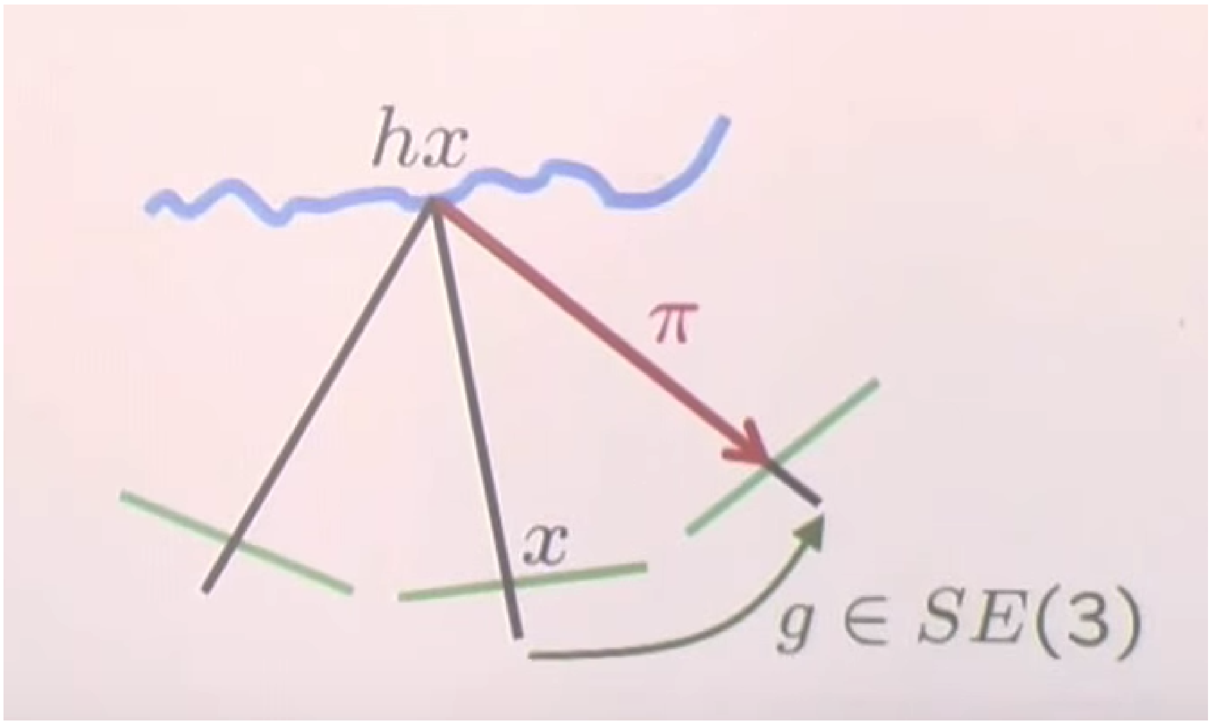
This cost function can be minimized at framerate by **sparse-to-fine linearization** solved in parallel on a GPU.

3 Dense RGB-D Tracking

3.1 Dense RGB-D Tracking

The approach of **Stuhmer et al. (2010)** relies on a sparse feature-point based camera trackier (PTAM) and computes dense geometry directly on the image. **Steinbrucker, Sturm, Cremers (2011)** propose a complementary approach to directly compute the camera motion from RGB-D images. The idea is to compute the rigid body motion g_ξ which optimally aligns to subsequent color images I_1 and I_2:

$$\min_{\xi \in \mathfrak{se}(3)} \int_\Omega |I_1(x) - I_2(\pi g_\xi(hx))|^2 dx \tag{2}$$



3.2 Dense RGB-D Tracking

The above non-convex problem can be approximated as a convex problem by linearizing the residuum around an initial guess ξ_0:

$$E(\xi) \approx \int_\Omega |I_1(x) - I_2(\pi g_{\xi_0}(hx)) - \nabla I_2^T (\frac{d\pi}{dg_\xi})(\frac{dg_\xi}{d\xi})\xi|^2 dx \tag{3}$$

This is a convex quadratic cost function which gives rise to a linear optimality condition:

$$\frac{dE(\xi)}{d\xi} = A\xi + b = 0 \tag{4}$$

To account for large motions of the camera, this problem is solved in a coarse-to-fine manner. The linearization of the residuum is identical with a Gauss-Newton approach. It corresponds to an approximation of the Hessian by a positive definite matrix.

3.3 Dense RGB-D Tracking

In the small-baseline setting, this image aligning approach provides more accurate camera motion than the commonly used generalized Iterated Closest Points (GICP) approach.

3.4 Combining Photometric and Geometric Consistency

Kerl, Sturm, Cremers, IROS 2013 propose an extension of the RGB-D camera tracker which combines **color consistency** and **geometric consistency** of subsequent RGB-D images.

Assuming that the vector $r_i = (r_{ci}, r_{zi}) \in \mathbb{R}^2$ containing the color and geometric discrepancy for pixel i follows a **bivariate t-distribution**, the maximum likelihood pose estimate can be computed as:

$$\min_{\xi \in \mathbb{R}^6} \sum_i w_i r_j^T \Sigma^{-1} r_i \tag{5}$$

with weights w_i based on the student t-distribution:

$$s_i = \frac{\nu + 1}{\nu + r_i^T \Sigma^{-1} r_i} \tag{6}$$

This nonlinear weighted least squares problem can be solved in an iteratively reweighted least squares manner by alternating a **Gauss-newton style optimization** with a **re-estimation of the weights w_i and the matrix Σ**.

4 Loop Closure and Global Consistency

4.1 Loop Closure and Global Consistency

When tracking a camera over a long period of time, **errors tend to accumulate**. While a single room may still be mapped more or less accurately, mapping a larger environment will lead to increasing distorions: Corridors and walls will no longer be straight but slightly curved.

A remedy is to introduce **loop closure**, a technique popularized in laser-based SLAM systems. The key idea is to estimate the relative camera motion $\hat{\xi_{ij}}$ for any camera pair i and j in a certain neighborhood. Subsequently, one can determine a **globally consistent camera trajectory** $\xi = \xi_{i i=1..T}$ by solving the **nonlinear least squares problem**

$$\min_{\xi} \sum_{i \sim j} (\hat{\xi}_{ij} - \xi_i \circ \xi_j^{-1})^T \Sigma_{ij} (\hat{\xi}_{ij} - \xi_i \circ \xi_j^{-1}) \tag{7}$$

where Σ_{ij} denotes the uncertainty of measurement $\hat{\xi}_{ij}$. This problem can be solved using, for example, a **Levenberg-Marquardt algorithm**.

5 Dense Tracking and Mapping

5.1 Dense Tracking and Mapping

Newcombe, Lovegrove and Davison (ICCV 2011) propose an algorithm which computes both the geometry of the scene and the camera motion from a direct and dense algorithm.

They compute the **inverse depth** $u = 1/h$ by minimizing a cost function of the form

$$\min_u \sum_{i=2}^{n} \int_{\Sigma} |I_1(x) - I_i(\pi g_i(\frac{x}{u}))| dx + \lambda \int_{\Omega} \rho(x) |\nabla u| dx \tag{8}$$

for fixed camera motions g_i. The function ρ introduces an **edge-dependency weighting** assigning small weights in locations where the input images exhibit strong gradients:

$$\rho(x) = exp(-|\nabla I_\sigma(x)|^\alpha) \tag{9}$$

The **camera tracking** is then performed **with respect to the textured reconstruction** in a manner similar to **Steinbrucker et al. (2011)**. The method is initialized using feature point based stereo.

6 Large Scale Direct Monocular SLAM

6.1 Large-Scale Direct Monocular SLAM

A method for real-time direct monocular SLAM is proposed in **Engel, Sturm, Cremers, ICCV 2013** and **Engel, Schops, Cremers, ECCV 2014**. It combines several contributions which make it well-studied for robust large-scale monocular SLAM.

- Rather then tracking and putting into correspondence a sparse set of feature points, the method estimates a **semi-dense depth map** which associates a inverse depth with each pixel that exhibits sufficient gray value variation.
- To account for noise and **uncertainty** each inverse depth value is associated with an uncertainty which is **propagated and updated over time** like in a Kalman filter.
- Since monocular SLAM is inveriably defined up to scale only, we explicitly facilitate scaling of the reconstruction by modeling the camera motion using the **Lie group of 3D similarity transformations Sim(3)**.
- Global consistency is assured by **loop closuring on Sim(3)**.

6.2 Tracking by Direct $\mathfrak{sim}(3)$ Image Alignment

Since reconstructions from a monocular camera are only defined up to scale, **Engel, Schops, Cremers, ECCV 2014** account for rescaling of the environment by representing the camera motion as an element in the **Lie group of 3D similarity transformations Sim(3)** which is defined as:

$$Sim(3) = \{\begin{pmatrix} sR & T \\ 0 & 1 \end{pmatrix} \text{ with } R \in SO(3), T \in \mathbb{R}^3, s \in \mathbb{R}_+\} \tag{10}$$

One can minimize a **nonlinear least squares problem**

$$\min_{\xi \in \mathfrak{sim}(3)} \sum_i w_i r_i^2(\xi) \tag{11}$$

where r_i denotes the color residuum across different images and w_i a weighting as suggested in Kerl et al. IROS 2013.

The above cost function can then be optimized by a **weighted Gauss-Newton algorithm on the Lie group Sim(3)**:

$$\xi^{(t+1)} = \Delta_\xi \circ \xi^{(t)}, \text{ with } \Delta_\xi = (J^T W J)^{-1} J^T W r, j = \frac{\partial r}{\partial \xi} \tag{12}$$