Chapter 4
Estimating Point Correspondence

Multiple View Geometry
Summer 2018

Prof. Daniel Cremers
Chair for Computer Vision and Artificial Intelligence
Departments of Informatics & Mathematics
Technische Universität München

Overview

From Photometry to Geometry

In the last sections, we discussed how points and lines are transformed from 3D world coordinates to 2D image and pixel coordinates.

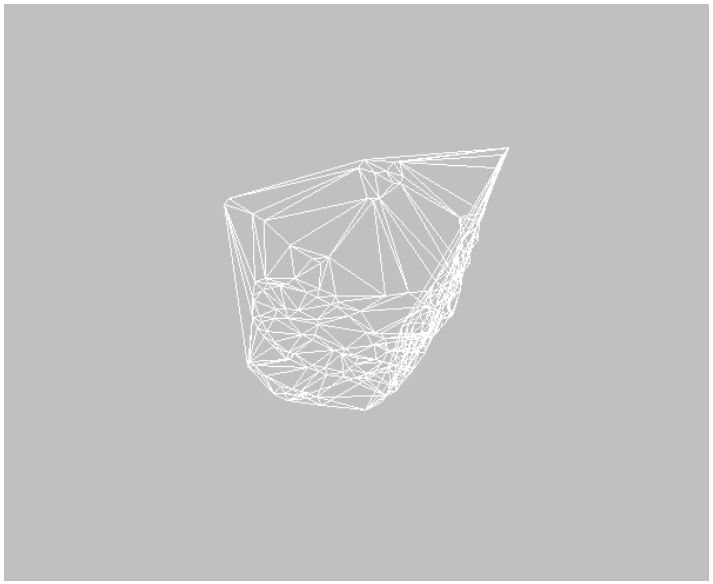
In practice, we do not actually observe points or lines, but rather brightness or color values at the individual pixels. In order to transfer from this photometric representation to a geometric representation of the scene, one can identify points with characteristic image features and try to associate these points with corresponding points in the other frames.

The matching of corresponding points will allow us to infer 3D structure. Nevertheless, one should keep in mind that this approach is suboptimal: By selecting a small number of feature points from each image, we throw away a large amount of potentially useful information contained in each image. Yet, retaining all image information is computationally challenging. The selection and matching of a small number of feature points, on the other hand, allows tracking of 3D objects from a moving camera in real time - even with limited processing power.

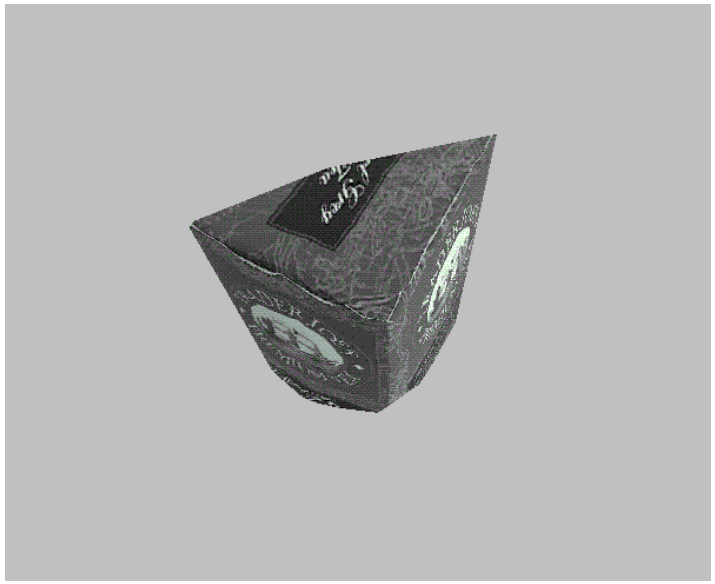
Example of Tracking

Input frame 1

Input frame 2



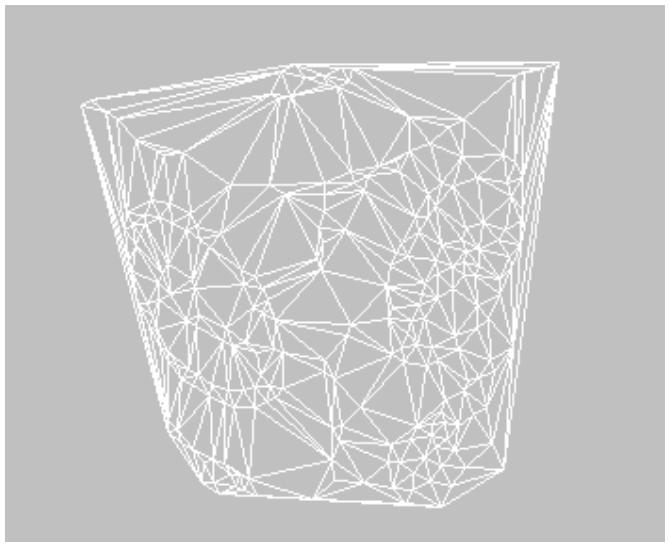
Wire frame reconstruction



with texture map

Example of Tracking

Tracked input sequence



Textured reconstruction

Identifying Corresponding Points

Input frame 1

Input frame 2

To identify corresponding points in two or more images is one of the biggest challenges in computer vision. Which of the points identified in the left image corresponds to which point in the right one?

Non-rigid Deformation

In what follows we will assume that objects move rigidly.

However, in general, objects may also deform non-rigidly.
Moreover, there may be partial occlusions:

Image 1

Image 2

Registration

Cremers, Guetter, Xu, *CVPR '06*

Small Deformation versus Wide Baseline

In point matching one distinguishes two cases:

- Small deformation: The deformation from one frame to the other is assumed to be (infinitesimally) small. In this case the displacement from one frame to the other can be estimated by classical optic flow estimation, for example using the methods of Lucas/Kanade or Horn/Schunck. In particular, these methods allow to model dense deformation fields (giving a displacement for every pixel in the image). But one can also track the displacement of a few feature points which is typically faster.

- Wide baseline stereo: In this case the displacement is assumed to be large. A dense matching of all points to all is in general computationally infeasible. Therefore, one typically selects a small number of feature points in each of the images and develops efficient methods to find an appropriate pairing of points.

Small Deformation

The transformation of all points of a rigidly moving object is given by:

$$x_2 = h(x_1) = \frac{1}{\lambda_2(\boldsymbol{X})} \left(R\lambda_1(\boldsymbol{X})\, x_1 + T \right).$$

Locally this motion can be approximated in several ways.

- Translational model:

$$h(x) = x + b.$$

- Affine model:

$$h(x) = Ax + b.$$

The 2D affine model can also be written as:

$$h(x) = x + u(x)$$

with

$$u(x) = S(x)p = \begin{pmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{pmatrix} (p_1, p_2, p_3, p_4, p_5, p_6)^\top.$$

Optic Flow Estimation

The optic flow refers to the apparent 2D motion field observable between consecutive images of a video. It is different from the motion of objects in the scene, in the extreme case of motion along the camera axis, for example, there is no optic flow, while on the other hand camera rotation generates an optic flow field even for entirely static scenes.

In 1981, two seminal works on optic flow estimation were published, namely the works of Lucas & Kanade, and of Horn & Schunck. Both methods have become very influential with thousands of citations. They are complementary in the sense that the Lucas-Kanade method generates sparse flow vectors under the assumption of constant motion in a local neighborhood, whereas the Horn-Schunck method generates a dense flow field under the assumption of spatially smooth motion. Despite more than 30 years of research, the estimation of optic flow fields is still a highly active research direction.

Due to its simplicity, we will review the Lucas-Kanade method.

The Lucas-Kanade Method

- Brightness Constancy Assumption: Let $x(t)$ denote a moving point at time t, and $I(x,t)$ a video sequence, then:

$$I(x(t),t) = \text{const.} \quad \forall t,$$

i.e. the brightness of point $x(t)$ is constant. Therefore the total time derivative must be zero:

$$\frac{d}{dt} I(x(t),t) = \nabla I^\top \left(\frac{dx}{dt} \right) + \frac{\partial I}{\partial t} = 0.$$

This constraint is often called the (differential) optical flow constraint. The desired local flow vector (velocity) is given by $v = \frac{dx}{dt}$.

- Constant motion in a neighborhood: Since the above equation cannot be solved for v, one assumes that v is constant over a neighborhood $W(x)$ of the point x:

$$\nabla I(x',t)^\top v + \frac{\partial I}{\partial t}(x',t) = 0 \quad \forall x' \in W(x).$$

The Lucas-Kanade Method

The brightness is typically not exactly constant and the velocity is typically not exactly the same for the local neighborhood.

Lucas and Kanade (1981) therefore compute the best velocity vector v for the point x by minimizing the least squares error

$$E(v) = \int_{W(x)} \left| \nabla I(x', t)^\top v + I_t(x', t) \right|^2 dx'.$$

Expanding the terms and setting the derivative to zero one obtains:

$$\frac{dE}{dv} = 2Mv + 2q = 0,$$

with

$$M = \int_{W(x)} \nabla I \, \nabla I^\top dx', \quad \text{and} \quad q = \int_{W(x)} I_t \, \nabla I \, dx'.$$

If M is invertible, i.e. $\det(M) \neq 0$, then the solution is

$$v = -M^{-1} q.$$

Estimating Local Displacements

- Translational motion: Lucas & Kanade '81:

$$E(b) = \int_{W(x)} \left| \nabla I^\top b + I_t \right|^2 dx' \quad \rightarrow \ \min .$$

$$\frac{dE}{db} = 0 \quad \Rightarrow \quad b = \cdots$$

- Affine motion:

$$E(p) = \int_{W(x)} \left| \nabla I(x')^\top S(x')\, p + I_t(x') \right|^2 dx'$$

$$\frac{dE}{dp} = 0 \quad \Rightarrow \quad p = \cdots$$

When can Small Motion be Estimated?

In the formalism of Lucas and Kanade, one cannot always estimate a translational motion. This problem is often referred to as the aperture problem. It arises for example, if the region in the window $W(x)$ around the point x has entirely constant intensity (for example a white wall), because then $\nabla I(x) = 0$ and $I_t(x) = 0$ for all points in the window.
In order for the solution of b to be unique the structure tensor

$$M(x) = \int_{W(x)} \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix} dx'.$$

needs to be invertible. That means that we must have $\det M \neq 0$.
If the structure tensor is not invertible but not zero, then one can estimate the normal motion, i.e. the motion in direction of the image gradient.
For those points with $\det M(x) \neq 0$, we can compute a motion vector $b(x)$. This leads to the following simple feature tracker.

A Simple Feature Tracking Algorithm

Feature tracking over a sequence of images can now be done as follows:

- For a given time instance t, compute at each point $x \in \Omega$ the structure tensor

$$M(x) = \int_{W(x)} \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix} dx'.$$

- Mark all points $x \in \Omega$ for which the determinant of M is larger than a threshold $\theta > 0$:

$$\det M(x) \geq \theta.$$

- For all these points the local velocity is given by:

$$b(x,t) = -M(x)^{-1} \begin{pmatrix} \int I_x I_t dx' \\ \int I_y I_t dx' \end{pmatrix}.$$

- Repeat the above steps for the points $x + b$ at time $t + 1$.

Robust Feature Point Extraction

Even $\det M(x) \neq 0$ does not guarantee robust estimates of velocity — the inverse of $M(x)$ may not be very stable if, for example, the determinant of M is very small. Thus locations with $\det M \neq 0$ are not always reliable features for tracking.

One of the classical feature detectors was developed by Moravec '80, Förstner '84, '87 and Harris & Stephens '88.

It is based on the structure tensor

$$M(x) \equiv G_\sigma * \nabla I \nabla I^\top = \int G_\sigma(x - x') \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}(x')\, dx',$$

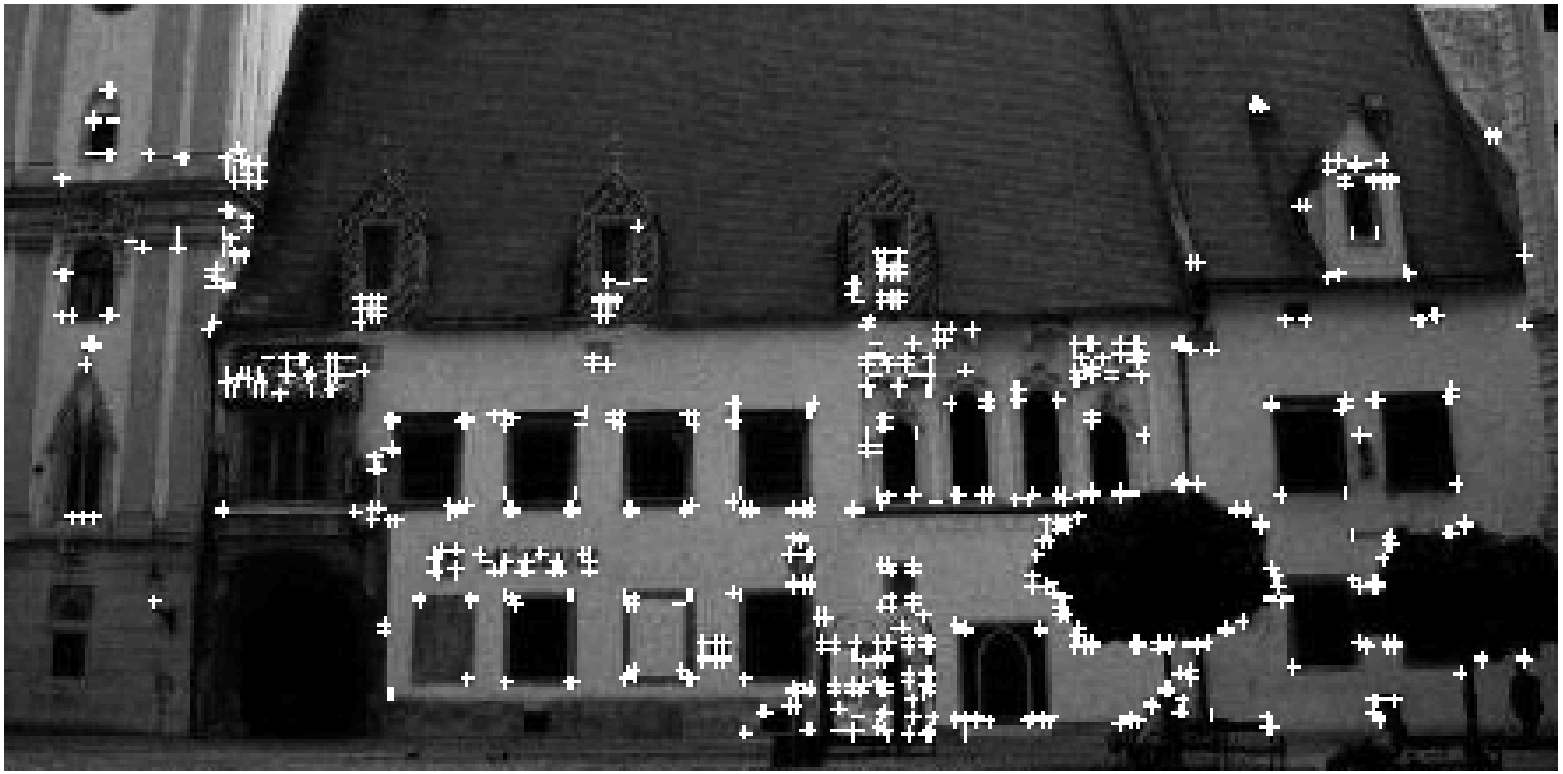
where rather than simple summing over the window $W(x)$ we perform a summation weighted by a Gaussian G of width σ.

Harris and Stephens propose the following expression:

$$C(x) = \det(M) - \kappa \operatorname{trace}^2(M),$$

and select points for which $C(x) > \theta$ with a threshold $\theta > 0$.

Response of Förstner Detector



Wide Baseline Matching

Corresponding points and regions may look very different in different views. Determining correspondence is a challenge.

In the case of wide baseline matching, large parts of the image plane will not match at all because they are not visible in the other image. In other words, while a given point may have many potential matches, quite possibly it does not have a corresponding point in the other image.

Extensions to Larger Baseline

One of the limitations of tracking features frame by frame is that small errors in the motion accumulate over time and the window gradually moves away from the point that was originally tracked. This is known as drift.
A remedy is to match a given point back to the first frame. This generally implies larger displacements between frames.

Two aspects matter when extending the above simple feature tracking method to somewhat larger displacements:

- Since the motion of the window between frames is (in general) no longer translational, one needs to generalize the motion model for the window $W(x)$, for example by using an affine motion model.
- Since the illumination will change over time (especially when comparing more distant frames), one can replace the sum-of-squared-differences by the normalized cross correlation which is more robust to illumination changes.

Normalized Cross Correlation

The normalized cross correlation is defined as:

$$NCC(h) = \frac{\int_{W(x)} \left(I_1(x') - \bar{I}_1\right)\left(I_2\left(h(x')\right) - \bar{I}_2\right) dx'}{\sqrt{\int_{W(x)} \left(I_1(x') - \bar{I}_1\right)^2 dx' \int_{W(x)} \left(I_2\left(h(x')\right) - \bar{I}_2\right)^2 dx'}},$$

where $\bar{I}_1$ and $\bar{I}_2$ are the average intensity over the window $W(x)$. By subtracting this average intensity, the measure becomes invariant to additive intensity changes $I \to I + \gamma$.

Dividing by the intensity variances of each window makes the measure invariant to multiplicative changes $I \to \gamma I$.

If we stack the normalized intensity values of respective windows into one vector, $v_i \equiv \mathrm{vec}(I_i - \bar{I}_i)$, then the normalized cross correlation is the cosine of the angle between them:

$$NCC(h) = \cos \angle(v_1, v_2).$$

Special Case: Optimal Affine Transformation

The normalized cross correlation can be used to determine the optimal affine transformation between two given patches. Since the affine transformation is given by:

$$h(x) = A\,x + d,$$

we need to maximize the cross correlation with respect to the 2×2-matrix A and the displacement d:

$$\hat{A}, \hat{d} = \arg\max_{A,d} NCC(A, d),$$

where

$$NCC(A, d) = \frac{\int_{W(x)} \Big(I_1(x') - \bar{I}_1\Big)\Big(I_2\big(Ax' + d\big)\big) - \bar{I}_2\Big)\,dx'}{\sqrt{\int_{W(x)} \Big(I_1(x') - \bar{I}_1\Big)^2 dx' \int_{W(x)} \Big(I_2\big(Ax' + d\big)\big) - \bar{I}_2\Big)^2 dx'}},$$

Efficiently finding appropriate optima, however, is a challenge.