

Computer Vision Group Prof. Daniel Cremers



Practical Course: Vision-based Navigation Summer Term 2015

Lecture 2: Visual Motion Estimation – Overview and Sparse Methods

Dr. Jörg Stückler

What we will cover today

- Introduction to visual motion estimation approaches
 - Visual odometry (VO) vs. visual SLAM
 - Overview on VO approaches for monocular, stereo, RGB-D cameras
 - The notions of sparse, dense, and direct
- Sparse, keypoint-based visual odometry

Visual Motion Estimation a.k.a. Visual Odometry

Robust Odometry Estimation for RGB-D Cameras

Christian Kerl, Jürgen Sturm, Daniel Cremers



Computer Vision and Pattern Recognition Group Department of Computer Science Technical University of Munich



Visual Motion Estimation a.k.a. Visual Odometry

Semi-Dense Visual Odometry for AR on a Smartphone

Thomas Schöps, Jakob Engel, Daniel Cremers ISMAR 2014, Munich



Computer Vision Group Department of Computer Science Technical University of Munich



Visual Motion Estimation a.k.a. Visual Odometry

SVO: Fast Semi-Direct Monocular Visual Odometry

Christian Forster, Matia Pizzoli, Davide Scaramuzza







The Term "Visual Odometry"

- Odometry:
 - Greek: "hodos" path, "metron" measurement
 - Motion or position estimation from measurements or controls
 - Typical example: wheel encoders
- Visual Odometry (VO):
 - 1980-2004: Dominant research by NASA JPL for Mars exploration rovers (Spirit and Opportunity in 2004)
 - David Nister's "Visual Odometry" paper from 2004 about keypoint-based methods for monocular and stereo cameras





Visual Odometry

- VO is often used to complement other motion sensors
 - GPS
 - Inertial Measurement Units (IMUs)
 - Wheel odometry
 - etc.
- Important in GPS-denied environments (indoors, underwater, etc.)
- Relation to Visual Simultaneous Localization and Mapping (SLAM):
 - Local (VO) vs. global (VSLAM) consistency
 - VO: 3D reconstruction only at local scale (if at all)
 - VO: Real-time requirements

Sensors for Visual Odometry

- Monocular:
 - Pros: Low-power, light-weight, low-cost, simple to calibrate and use
 - Cons: requires motion parallax and textured scenes, scale not observable
- Stereo:
 - Pros: depth without motion, less power than active structured light
 - Cons: requires textured scenes, accuracy depends on baseline, requires extrinsic calibration of the cameras, synchronization of the cameras
- Active RGB-D sensors:
 - Pros: also work in untextured scenes, similar to stereo processing
 - Cons: active sensing consumes power, blackbox depth estimation







Keypoints, Direct, Sparse, Dense



- Sparse: use a small set of selected pixels (keypoints)
- Dense: use all (valid) pixels

Sparse Keypoint-based Visual Odometry



Extract and match keypoints



Determine relative camera pose (R, t) from keypoint matches

Keypoint Extraction

- Detection repeatability
 - We want to find the (accurate) image of the same 3D point from different view-points



- Descriptor distinctiveness
 - We want a descriptor that achieves (in the ideal case) a unique and correct association of corresponding keypoints



Keypoint Detectors and Descriptors

- Keypoint detection and description in images has been extensively studied
- Nowadays there is plenty of fast and repeatable detectors available, e.g.,
 - Harris corner variants
 - FAST corner variants (e.g. ORB detector)
 - DoG blob variants (SIFT, SURF)
 - Learning-based keypoints
- Many detectors come with a suitable descriptor, e.g.,
 - ORB (binary pixel comparisons locally around keypoint)
 - SIFT/SURF (grayscale gradient patterns locally around keypoint)

Monocular Keypoint-based Motion Estimation

- In the monocular case, we do not have depth available at keypoints
- If we knew the relative pose of the cameras and the 3D position of each keypoint match, we could directly compute to which pixels the keypoints should project in each camera image
- To find the unknown pose and 3D positions, we could formulate an optimization problem that minimizes the reprojection error of all keypoints

$$E(R, t, x_1, \dots, x_N) = \frac{1}{N} \sum_{i} \left\| z_{1,i} - \pi(x_i) \right\|_2^2 + \left\| z_{2,i} - \pi(Rx_i + t) \right\|_2^2$$

 Reprojection error: difference between measured and expected pixel position of a keypoint





Motion from Epipolar Geometry





• The rays from each camera to the keypoint and the baseline t are coplanar!

$$\bar{x}_1^T(t \times R\bar{x}_2) = 0 \iff \bar{x}_1^T[t]_{\times}R\bar{x}_2 = 0$$

- The essential matrix $E = [t]_{\times}R$ captures the relative camera pose
- Each keypoint match provides an "epipolar constraint"
- 8 matches suffice to determine E (8-point algorithm)
- In the uncalibrated case, the camera calibration needs to be subsumed into the so-called fundamental matrix $F = K^{-T}EK^{-1}$

8-Point Algorithm (Longuet-Higgins, 1981)

- Find approximation to essential matrix:
 - Construct matrix $A = (a_1, a_2, \dots, a_N)^T$ with $a_i = \bar{x}_{1,i} \times \bar{x}_{2,i}$.
 - Apply a singular value decomposition (SVD) on $A = USV^T$ and unstack the 9th column vector of V into \tilde{E}
 - Project the approximate \tilde{E} into the (normalized) essential space: Determine the SVD of $\tilde{E} = U \operatorname{diag}(\sigma_1, \sigma_2, \sigma_3) V^T$ and replace the singular values $\sigma_1, \sigma_2, \sigma_3$ with 1,1,0 to find $E = U \operatorname{diag}(1,1,0) V^T$
 - Determine one of the following 4 possible solutions that intersect the points in front of both cameras:

$$R = U R_Z^T \left(\pm \frac{\pi}{2} \right) V^T$$

with $R_Z^T \left(\pm \frac{\pi}{2} \right) = \begin{pmatrix} 0 & \pm 1 & 0 \\ \mp 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$
 $[t]_{\times} = U R_Z \left(\pm \frac{\pi}{2} \right) \operatorname{diag}(1,1,0) U^T$

3D Keypoint-based Motion Estimation

- In the stereo case, rotation and translation between the left and right image are known
- We can first match keypoints between the left and right image, and triangulate their 3D positions
- To estimate motion between two stereo image pairs, we could use the 8point algorithm as well on the keypoints in the left images and recover scale from the triangulated stereo depth
- Alternatively, least-squares optimization of the reprojection error is now simpler, since we know the 3D positions of the keypoints

$$E(R,t) = \frac{1}{N} \sum_{i} \left\| z_{1,i} - \pi (R^T x_i - R^T t) \right\|_2^2 + \left\| z_{2,i} - \pi (R x_i + t) \right\|_2^2$$

Triangulation

- Given: n cameras $\{M_j = K_j(R_j \mathbf{t}_j)\}$ Point correspondence $\mathbf{x}_0, \mathbf{x}_1$
- Wanted: Corresponding 3D point p



Triangulation

• Where do we expect to see $\mathbf{p} = (X \ Y \ Z \ W)^{\top}$?

$$\hat{x} = \frac{m_{11}X + m_{12}Y + m_{13}Z + m_{14}W}{m_{31}X + m_{32}Y + m_{33}Z + m_{34}W} \qquad \qquad \hat{y} = \frac{m_{21}X + m_{22}Y + m_{23}Z + m_{24}W}{m_{31}X + m_{32}Y + m_{33}Z + m_{34}W}$$

Minimize the residuals

$$\mathbf{p}^* = \arg\min_{\mathbf{p}} \sum_j d(\mathbf{x}_j, \hat{\mathbf{x}}_j)^2$$

Triangulation

Multiply with denominator gives

$$0 = (x_j m_{31} - m_{11})X + (x_j m_{32} - m_{12})Y + (x_j m_{33} - m_{13})Z + (x_j m_{34} - m_{14})W$$

$$0 = (y_j m_{31} - m_{21})X + (y_j m_{32} - m_{22})Y + (y_j m_{33} - m_{23})Z + (y_j m_{34} - m_{24})W$$

Solve for $\mathbf{p} = (X \ Y \ Z \ W)^{\top}$ using:

- Linear least squares with W=1
- Linear least squares using SVD
- Non-linear least squares of the residuals (most accurate)

Robust Keypoint Matching



- Keypoint detectors and descriptors are not perfect
- Pose estimation can be very sensitive to wrong correspondences (especially when using the 8-point algorithm)
- What can we do?
- Idea: try out different combinations of 8 matches until we find a good fit for most of the overall keypoints
- Random Sample Consensus (RANSAC) algorithm

Vision-based Navigation

Robust Estimation

Example: Fit a line to 2D data containing outliers



- Input data is a mixture of
 - Inliers (perturbed by Gaussian noise)
 - Outliers (unknown distribution)
- Let's fit a line using least squares...

Robust Estimation

Example: Fit a line to 2D data containing outliers



- Input data is a mixture of
 - Inliers (perturbed by Gaussian noise)
 - Outliers (unknown distribution)
- Least squares fit gives poor results!

RANdom SAmple Consensus (RANSAC) [Fischler and Bolles, 1981]

Goal: Robustly fit a model to a data set S which contains outliers **Algorithm:**

- 1. Randomly select a (minimal) subset
- 2. Instantiate the model from it
- **3.** Using this model, classify all data points as inliers or outliers
- **4.** Repeat 1-3 for *N* iterations
- 5. Select the largest inlier set, and re-estimate the model from all points in this set

Example

• Step 1: Sample a random subset





• Step 2: Fit a model to this subset



Example

 Step 3: Classify points as inliers and outliers (e.g., using a threshold distance)





Step 4: Repeat steps 1-3 for N iterations



Example

Step 4: Repeat steps 1-3 for N iterations





 Step 5: Select the best model (most inliers), then re-fit model using all inliers



How Many Iterations Do We Need?

- For a probability of success p, we need

$$N = \frac{\log(1-p)}{\log(1-(1-\epsilon)^s)}$$

iterations

for subset size s and outlier ratio $\ \epsilon$

• E.g., for p=0.99:

	Required points s	Outlier ratio ε						
		10 %	20 %	30 %	40 %	50 %	60 %	70 %
Line	2	3	5	7	11	17	27	49
Plane	3	4	7	11	19	35	70	169
Essential matrix	8	9	26	78	272	1177	7025	70188

Summary on RANSAC

- Efficient algorithm to estimate a model from noisy and outliercontaminated data
- RANSAC is used today very widely
- Often used in feature matching / visual motion estimation
- Many improvements/variants (e.g., PROSAC, MLESAC, …)

Lessons Learned Today

- Overview on visual odometry and SLAM
- How to estimate motion from keypoints from monocular images using the 8-point algorithm
- How to use the 8-point algorithm for stereo and RGB-D
- How to triangulate keypoint matches given the camera pose
- How to separate inliers from outliers using RANSAC

Questions ?