Visual Odometry Using Essential Matrix Estimation

Felix Wempe

Software Practical

January 18, 2016

Section 1

Motivation

Problem

- Monocular odometry.
- Estimate the position and the orientation of a car.
- On the KITTI dataset.

Section 2

Approach

Approach

- Estimate the Essential matrix from one frame to the next.
- Use geometry of the Essential Manifold:

$$\mathcal{E} := \{\Omega R | \Omega \in \mathfrak{so}_3, R \in \mathrm{SO}_3, \|\Omega\|^2 = 2\}$$

- Define a Newton type algorithm on the manifold.

HELMKE, UWE: Essential Matrix Estimation Using Gauss-Newton Iterations on a Manifold. In: *International Journal of Computer Vision 74(2)* (2007)

Section 3

Essential Manifold

Essential Matrix

- Essential matrix relates corresponding points in two images.

$$m_1^\top E m_2 = 0$$

- Epipolar constraint.
- $E = \Omega R$, with $\Omega \in \mathfrak{so}_3$ and $R \in \mathrm{SO}_3$

$$\Omega = \begin{pmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{pmatrix}$$

Essential Manifold

- For every Essential Matrix E holds (with $\mu \in \mathbb{R}$) μE is also an Essential Matrix.
- Only consider normalised Essential Matrices:

$$\|E\| = \sqrt{2}$$

$$\mathcal{E} := \left\{ \Omega R | \Omega \in \mathfrak{so}_3, R \in \mathrm{SO}_3, \|\Omega\|^2 = 2 \right\}$$

- $\mathcal{E}$ is a five-dimensional smooth manifold.
- Normalised Essential Matrices have singular values $\{1, 1, 0\}$.

- Using the SVD we find another representation:

$$\mathcal{E} = \{UE_0V^\top | U, V \in \mathrm{SO}_3\}$$

- Using this we get the tangent space at E:

$$T_E\mathcal{E} = \{U(\Omega E_0 - E_0\Psi)V^\top | \Omega, \Psi \in \mathfrak{so}_3\}$$

Local Parametrization

- For computations on the manifold we need a local parametrization.
- A local parametrization $\mu : \mathbb{R}^5 \to \mathcal{E}$ at $E \in \mathcal{E}$ satisfies:
  - $\mu_{(U,V)}(0) = E$
  - $\mu_{(U,V)}$ is a local diffeomorphism around 0.
  - There exists a map $L : \mathcal{S} \to \mathrm{GL}_5$ such that

$$\mu_{(U\Gamma,V\Gamma)}(x) = \mu_{(U,V)}(L(\Gamma)x)$$

Exponential Projection

We use the exponential projection:

$$\mu_{(U,V)} : \mathbb{R}^5 \to \mathcal{E}$$

$$\mu_{(U,V)}(x) = U e^{\Omega_1(x)} E_0 e^{-\Omega_2(x)} V^\top$$

$$\Omega_1(x) = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 & -\frac{x_3}{\sqrt{2}} & x_2 \\ \frac{x_3}{\sqrt{2}} & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{pmatrix}, \Omega_2(x) = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 & \frac{x_3}{\sqrt{2}} & x_5 \\ -\frac{x_3}{\sqrt{2}} & 0 & -x_4 \\ -x_5 & x_4 & 0 \end{pmatrix}$$

Section 4

Algorithm

Algorithm

- Minimize a cost function $f : \mathcal{E} \to \mathbb{R}$ given matched points.
- Using a Newton-type algorithm.
  - Newton directions: $-H_{f\circ\mu}(0)^{-1}\nabla(f \circ \mu)(0)$
  - Gauss-Newton directions: $-\hat{H}_{f\circ\mu}(0)^{-1}\nabla(f \circ \mu)(0)$.

Algorithm 1 Essential Matrix Estimation

1: Determine the initial Estimate $E_1 = U_1 E_0 V_1^\top$.
2: Set $k = 1$ and $\epsilon > 0$ to prescribed accuracy.
3: Given U_k, V_k compute the gradient ∇_k and Hessian H_k.
4: **if** $H_k \succ 0$ **then**
5:     Go in Newton direction.
6: **else**
7:     Go in Gauss-Newton direction.
8: **end if**
9: Project back on the manifold. $\rightarrow U_{k+1}, V_{k+1}$
10: **if** $\|\nabla_{k+1}\| < \epsilon$ **then**
11:     Terminate.
12: **else**
13:     Set $k = k + 1$ and go to Line 3.
14: **end if**

Section 5

Cost Function

Cost Function

- The chosen cost function is quadratic on the error measure.
- Added by a weighted smoothing factor.
- The matched points m_1^i, m_2^i we write as ($M^i = \hat{m}_2^i \hat{m}_1^{i\top}$):

$$\overline{M} := \begin{pmatrix} \mathrm{vec}^\top(M^{1\top}) \\ \vdots \\ \mathrm{vec}^\top(M^{n\top}) \end{pmatrix}$$

$$\mathcal{M} := \frac{1}{n}(\overline{M}^\top \overline{M}) \geq 0$$

The cost function $f : \mathcal{E} \to \mathbb{R}$ is defined as:

$$f(E) = \frac{1}{2n} \sum_{i=1}^{n} \left(\hat{m}_1^{i\top} E \hat{m}_2^i \right)^2 + \lambda \frac{1}{2} \|E - \hat{E}\|_{\mathrm{F}}^2$$

We may rewrite the quadratic part:

$$\frac{1}{2n} \sum_{i=1}^{n} \left(\hat{m}_1^{i\top} E \hat{m}_2^i \right)^2 = \frac{1}{2} \|\mathrm{vec}(E)\|_{\mathcal{M}}^2$$

We get:

$$\nabla(f \circ \mu_{(U,V)})(0) = J^\top \mathcal{M}\mathrm{vec}(E) + \lambda J^\top (E - \hat{E})$$

$$H_{f\circ\mu_{(U,V)}}(0) = \hat{H}_{f\circ\mu_{(U,V)}}(0) + \tilde{H}_{f\circ\mu_{(U,V)}}(0)$$

for

$$\hat{H}_{f\circ\mu_{(U,V)}}(0) = J^\top \mathcal{M} J + \lambda J^\top J \geq 0$$

Section 6

RANSAC

RANSAC

- Wrong point matches have strong influence on the quadratic cost function.
- To get rid of these outliers RANSAC is used.

Algorithm 2 RANSAC

return consensus set
loop
  Choose randomly 10 matched point pairs.
  Compute Essential matrix for these 10 points.
  for All matched point pairs **do**
    Check if $m_1^\top E m_2 < \epsilon$. (test set)
  end for
  if $\#\{\text{test set}\} > 100$ **then**
    Add test set to the consensus set.
  end if
end loop

Section 7

Evaluation

Evaluation

- Test data from "The KITTI Vision Benchmark Suite".
- Comparison of the different weights.
  - Computation times.
  - Average errors.
  - Plot of the paths.
- Comparison of the RANSAC to one implemented in Matlab.

- We Estimated U, $V \in \mathrm{SO}_3$, such that: $E = UE_0V^\top$.
- Want to get the Rotation and Translation: $E = \Omega R$.

$$\Omega = U \begin{pmatrix} 0 & \varepsilon & 0 \\ -\varepsilon & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} U^\top$$

$$R = U \begin{pmatrix} 0 & -\varepsilon & 0 \\ \varepsilon & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} V^\top$$

Error Measures

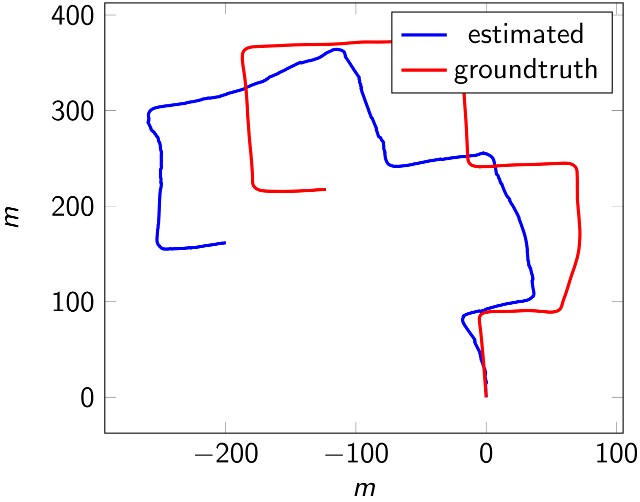
- Distance function for rotation matrices:

$$d(R, \tilde{R}) = \| \log(R^\top \tilde{R}) \|_F$$

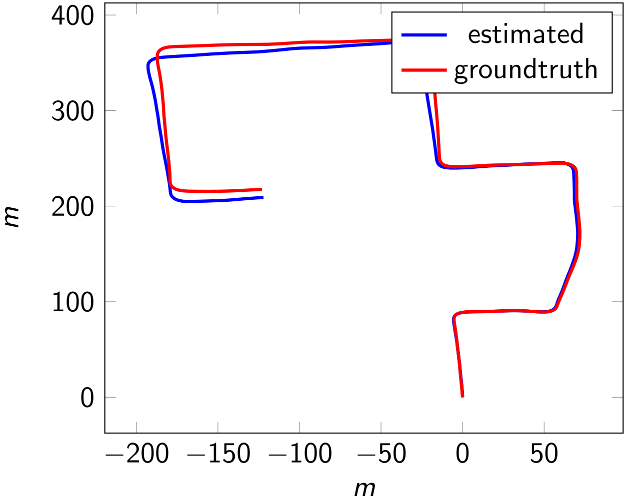
- Distance for the angle of the translation:

$$d(t, \tilde{t}) = \arccos\left(\frac{\langle t, \tilde{t} \rangle}{\|t\| \|\tilde{t}\|}\right)$$

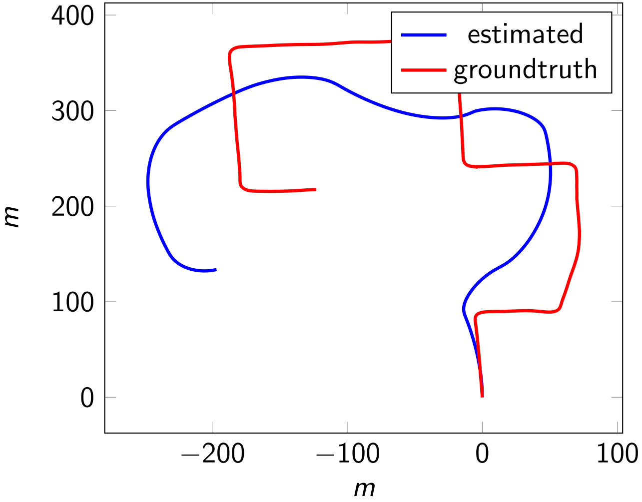
Estimation without RANSAC, $\lambda = 0$



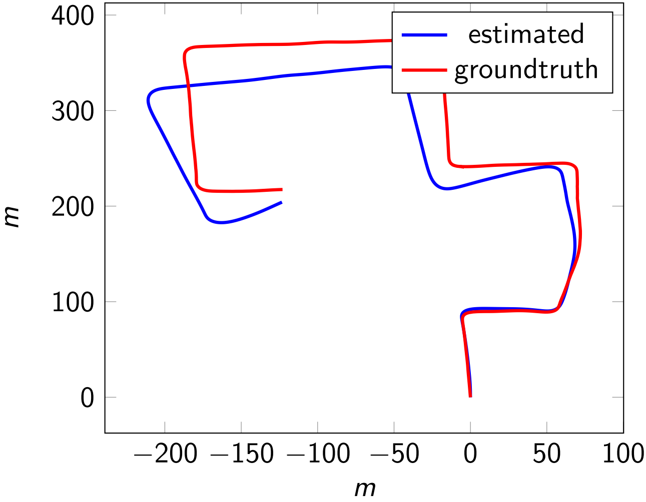
Estimation with $\lambda = 0$



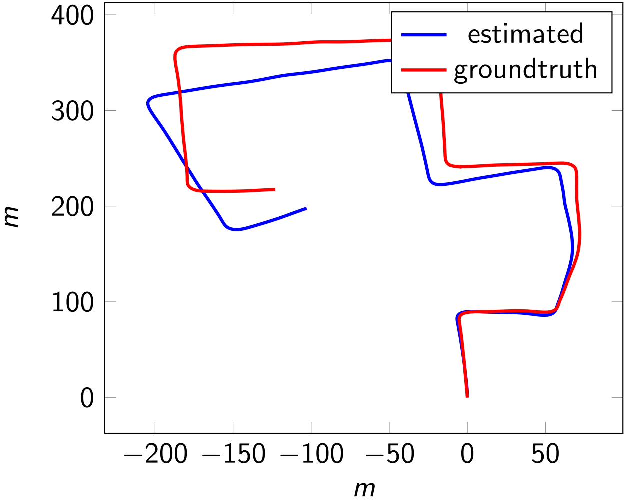
Estimation with $\lambda = 2$



Estimation with $\lambda = 0.2$



Estimation with $\lambda = 0.02$



Average Computation Times

(in seconds)

Sequence	$\lambda = 0$	$\lambda = 2$	$\lambda = 0.2$	$\lambda = 0.02$
00	0,0154	0,0263	0,0222	0,0194
01	0,0194	0,0302	0,024	0,0216
02	0,0144	0,0296	0,0208	0,0203

Average Iterations used

Sequence	$\lambda = 0$	$\lambda = 2$	$\lambda = 0.2$	$\lambda = 0.02$
00	10,48	14,32	12,03	10,06
01	11,35	16	13,27	10,99
02	9,8	14,29	11,27	9,86

Average Translation Error

(in degree)

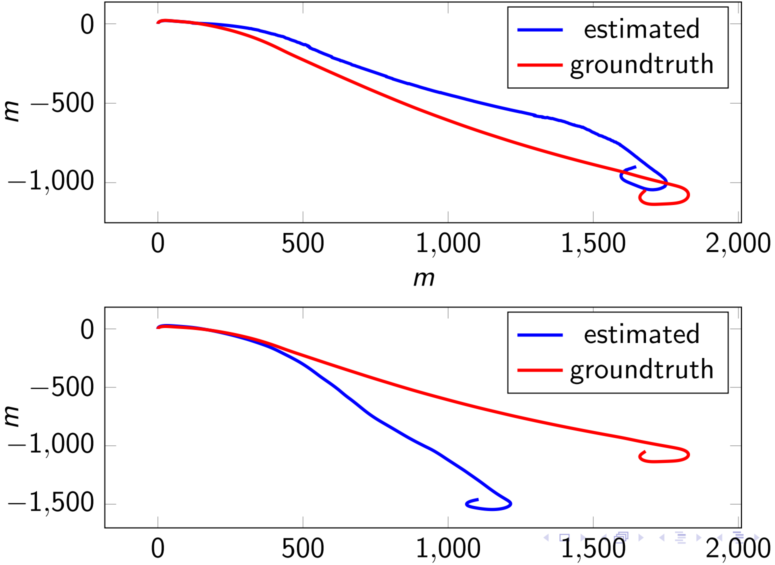
Sequence	$\lambda = 0$	$\lambda = 2$	$\lambda = 0.2$	$\lambda = 0.02$
00	1,806	1,829	1,66	1,669
01	3,697	1,582	1,523	2,439
02	1,338	1,086	1,428	1,443

Average Rotation Error

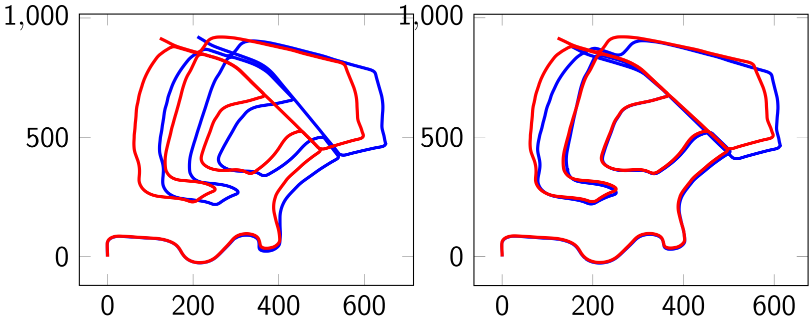
(in degree)

Sequence	$\lambda = 0$	$\lambda = 2$	$\lambda = 0.2$	$\lambda = 0.02$
00	2,152	1,484	1,974	2,149
01	1,436	1,289	1,231	1,354
02	1,892	1,299	1,702	1,838

Estimation with $\lambda = 0$ and $\lambda = 0.2$



Comparison RANSAC with $\lambda = 0$



Comparison RANSAC

Sequence 02	Own	Matlab
average time	0,3903s	0,7392s
Rotation error	1,845	1,851
Translation error	1,312	1,306

Section 8

Conclusion

- Using the geometry of the essential manifold gives a fast algorithm.
- RANSAC is necessary, since outliers have a strong influence on the estimation.
- Smoothing leads to better results in the average errors.