Epipolar Geometry and its application for the construction of state-of-the-art sensors.

Karel Zimmermann

Czech Technical University in Prague
Faculty of Electrical Engineering, Department of Cybernetics
Center for Machine Perception
http://cmp.felk.cvut.cz/~zimmerk, zimmerk@fel.cvut.cz
Motivation

- You are given two images of an object captured by two cameras P and Q from different view-points.
Motivation

- Given pair of corresponding pixels \((u, v)\) (i.e. pixels corresponding to the same unknown 3D point \(X\) on the object), you can easily compute \(X\).
Motivation

Given pair of corresponding pixels \((u, v)\) (i.e. pixels corresponding to the same unknown 3D point \(X\) on the object), you can easily compute \(X\).
Motivation

- The only problem is, that you do not have the correspondence $(u, v)$ and naïve matching of pixel neighbourhoods does not work.
Motivation

- This lesson is about
  - how to get 3D points from images captured by known cameras and
  - how to use this knowledge to built state-of-the-art depth sensors.
Outline

- Epipolar geometry
  - Epipolar line, essential and fundamental matrix
  - $L_2$ estimation of the essential matrix
- Depth sensors: Stereo, Kinect and RealSense
- Depth from a single camera and the robust estimation of the essential matrix (RANSAC).
Projection of the 3D point to a single camera

- You are given $3 \times 4$ camera matrix $P = \begin{bmatrix} p_1^\top \\ p_2^\top \\ p_3^\top \end{bmatrix}$
- 3D point with homogeneous coordinates $X$ projects on pixel $u$
Projection of the 3D point to a single camera

- You are given a $3 \times 4$ camera matrix $P = \begin{bmatrix} p_1^\top \\ p_2^\top \\ p_3^\top \end{bmatrix}$

- 3D point with homogeneous coordinates $X$ projects on pixel $u$

  $$u_1 = \frac{p_1^\top X}{p_3^\top X}, \quad u_2 = \frac{p_2^\top X}{p_3^\top X}$$
Projection of the 3D point to a single camera

- What if $u$ is known? Which $X$ correspond to $u$?
What if \( \mathbf{u} \) is known? Which \( \mathbf{X} \) correspond to \( \mathbf{u} \)?

All 3D points corresponding to pixel \( \mathbf{u} \) lies in 1D linear subspace (ray) of 3D space (2 linear equations with 3 unknowns):

\[
\begin{align*}
\mathbf{u}_1 \mathbf{p}_3^\top \mathbf{X} &= \mathbf{p}_1^\top \mathbf{X}, \\
\mathbf{u}_2 \mathbf{p}_3^\top \mathbf{X} &= \mathbf{p}_2^\top \mathbf{X}
\end{align*}
\]

\[
\Rightarrow \begin{bmatrix} \mathbf{u}_1 \mathbf{p}_3^\top - \mathbf{p}_1^\top \\ \mathbf{u}_2 \mathbf{p}_3^\top - \mathbf{p}_2^\top \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = 0
\]
**Fundamental matrix**

- Projection of the ray from \( u \) into a second camera is called epipolar line

\[
\{ v \mid u^T F v = 0 \},
\]

- where matrix \( F = K^T (R \times t) K \) is called fundamental matrix.
Essential matrix

- We assume that $K$ is known (i.e. the camera is calibrated).
Essential matrix

- We assume that $K$ is known (i.e. the camera is calibrated).
- We normalize coordinates $u_n = K^{-1}u$, $v_n = K^{-1}v$ and pretend that $K$ is identity.
**Essential matrix**

- We assume that $K$ is known (i.e. the camera is calibrated).
- We normalize coordinates $\mathbf{u}_n = K^{-1} \mathbf{u}$, $\mathbf{v}_n = K^{-1} \mathbf{v}$ and pretend that $K$ is identity.
- Epipolar line wrt normalized coordinates is $\{ \mathbf{v}_n \mid \mathbf{u}_n^\top E \mathbf{v}_n = 0 \}$, where matrix $E = R \times t$ is called essential matrix.
What is the essential matrix good for?

Important result 1:

- If camera motion is **known** (e.g. stereo), then

- all possible correspondences of point \( \mathbf{u} \) lie on the epipolar line (i.e. either \( \{ \mathbf{v} \mid \mathbf{u}^\top \mathbf{F} \mathbf{v} = 0 \} \) or \( \{ \mathbf{v}_n \mid \mathbf{u}_n^\top \mathbf{E} \mathbf{v}_n = 0 \} \)).
What is the essential matrix good for?

◆ Important result 1:

- If camera motion is **known** (e.g. stereo), then
  - all possible correspondences of point $u$ lie on the epipolar line (i.e. either $\{v \mid u^\top F v = 0\}$ or $\{v_n \mid u_n^\top E v_n = 0\}$).

◆ Important result 2:

- If camera motion is **unknown** (e.g. motion of a single camera), then
  - the essential matrix determines relative position of cameras (i.e. motion), since there exist unique decomposition $E = R \times t$. 
What is the essential matrix good for?

Important result 1:

- If camera motion is known (e.g. stereo), then
  - all possible correspondences of point \( u \) lie on the epipolar line (i.e. either \( \{ v : \mathbf{u}^\top \mathbf{F} \mathbf{v} = 0 \} \) or \( \{ \mathbf{v}_n : \mathbf{u}_n^\top \mathbf{E} \mathbf{v}_n = 0 \} \)).

Important result 2:

- If camera motion is unknown (e.g. motion of a single camera), then
  - the essential matrix determines relative position of cameras (i.e. motion), since there exist unique decomposition \( \mathbf{E} = \mathbf{R} \times \mathbf{t} \).

From now on, we drop the index \( n \) in normalized coordinates.

How do we obtain the essential/fundamental matrix?
Compute essential matrix by minimizing L2-norm

- Let us assume that we have several correct correspondences.
Compute essential matrix by minimizing L2-norm

- Let us assume that we have several correct correspondences.
- Essential matrix $E$ is just a solution of (overdetermined) homogeneous system of linear equations.
Compute essential matrix by minimizing L2-norm

Let us assume that we have several correct correspondences.

Essential matrix \( E \) is just a solution of (overdetermined) homogeneous system of linear equations.

For each correspondence pair \( u, v \), the following holds:

\[
\begin{align*}
    u^T E v &= u^T \begin{bmatrix} e_1^T \\ e_2^T \\ e_3^T \end{bmatrix} v \\
    &= u^T \begin{bmatrix} e_1^T v \\ e_2^T v \\ e_3^T v \end{bmatrix} \\
    &= [u_1 e_1^T v + u_2 e_2^T v + u_3 e_3^T v]
\end{align*}
\]
Compute essential matrix by minimizing L2-norm

Let us assume that we have several correct correspondences.

Essential matrix $E$ is just a solution of (overdetermined) homogeneous system of linear equations.

For each correspondence pair $u_i, v_i$, the following holds:

$$u^T E v = u^T \begin{bmatrix} e_1^T \\ e_2^T \\ e_3^T \end{bmatrix} v = u^T \begin{bmatrix} e_1^T v \\ e_2^T v \\ e_3^T v \end{bmatrix} = [u_1 e_1^T v + u_2 e_2^T v + u_3 e_3^T v] =$$

$$= [u_1 v^T \ u_2 v^T \ u_3 v^T] \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = 0$$

It must hold for all correspondence pairs $u_i, v_i$, therefore:

$$\begin{bmatrix} u_{11} v_1^T \ & u_{12} v_1^T \ & u_{13} v_1^T \\ u_{21} v_2^T \ & u_{22} v_2^T \ & u_{23} v_2^T \\ \cdots \ & \cdots \ & \cdots \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = 0$$
Compute essential matrix by minimizing L2-norm

- It is just homogeneous set of linear equations:

\[
\begin{bmatrix}
    u_{11}v_1^\top & u_{12}v_1^\top & u_{13}v_1^\top \\
    u_{21}v_2^\top & u_{22}v_2^\top & u_{23}v_2^\top \\
    \vdots
\end{bmatrix}
\begin{bmatrix}
e_1 \\
e_2 \\
e_3
\end{bmatrix} = 0
\]

- We want to avoid trivial solution \( e_1 = e_2 = e_3 = 0 \),

- therefore the following optimization task (constrained LSQ) is solved:

\[
\arg \min \|Ae\| \quad \text{subject to} \quad \|e\| = 1
\]

- the solution is singular vector of matrix \( A \) corresponding to the smallest singular value (can be found via SVD or eigenvectors/eigenvalues of \( AA^\top \))
Compute essential matrix by minimizing L2-norm

- The same is valid for the estimation of the fundamental matrix from not normalized coordinates.
Compute essential matrix by minimizing L2-norm

- The same is valid for the estimation of the fundamental matrix from not normalized coordinates.
- $L_2$-norm works only in a controlled environment (e.g. offline stereo calibration).
Compute essential matrix by minimizing L2-norm

- The same is valid for the estimation of the fundamental matrix from not normalized coordinates.
- $L_2$-norm works only in a controlled environment (e.g. offline stereo calibration).
- I will show how essential/fundamental matrix allows to estimate correspondences in state-of-the-art depth (3D) sensors.
Stereo

- Pair of cameras mounted on a rigid body, which provides depth (3D points) of the scene (simulates human binocular vision).
- Relative position of cameras fixed

---

0 Courtesy of prof. Boris Flach for original stereo images and depth images
Stereo

- Pair of cameras mounted on a rigid body, which provides depth (3D points) of the scene (simulates human binocular vision).
- Relative position of cameras fixed
- **offline**: fundamental matrix estimated from known correspondences.

---

0 Courtesy of prof. Boris Flach for original stereo images and depth images
Stereo

- Pair of cameras mounted on a rigid body, which provides depth (3D points) of the scene (simulates human binocular vision).
- Relative position of cameras fixed
- **offline**: fundamental matrix estimated from known correspondences.
- **online**: correspondences searched along epipolar lines.

---

0 Courtesy of prof. Boris Flach for original stereo images and depth images
Stereo

Block-matching energy function: \( E(u) = \sum_{x \in W} (I_L(x) - I_R(x + u))^2 \)
Stereo

Block-matching energy function: \( E(u) = \sum_{x \in W} (I_L(x) - I_R(x + u))^2 \)
Stereo

Block-matching energy function: $E(u) = \sum_{x \in W} (I_L(x) - I_R(x + u))^2$
Stereo

Correspondence for each pixel estimated separately: $u^* = \arg\min_u E(u)$
Stereo

Correspondence for each pixel estimated separately: $u^* = \arg \min_u E(u)$
Stereo

How can we improve the result?
Stereo

Energy with horizontal smoothness term: $E(u_1, u_2) = E(u_2) + C \cdot (u_2 - u_1)^2$
Stereo

Dynamic programming solves each line of $N$ pixels separately:

$$U^* = \arg \min_{U \in \mathcal{R}^N} \sum_{i=1}^{N-1} E(u_i, u_{i+1})$$
Stereo

What else can we do?

Image  Block matching  Dynamic programming
Stereo

Enforce also vertical smoothness $\Rightarrow$ graph energy minimization (computationally demanding optimization solved on specialized chips).
Stereo

Enforce also vertical smoothness $\Rightarrow$ graph energy minimization (computationally demanding optimization solved on specialized chips).

- **Limitation:** usually works only on sufficiently rich patterns and sufficiently smooth depths.
Stereo competition

- Do you have your own idea how to estimate the depth from stereo images?
- http://vision/middlebury.edu/stereo/data/2014/
**Kinect (structured-light approach)**

- **Stereo** looks at the same object two-times and estimates the correspondence from two passive RGB images.

- **Kinect** avoids ambiguity by actively projecting a unique IR pattern on the surface and search for its known appearance in the IR camera.
Since camera-projector relative position is known, correspondence between projected pixel and observed pixel lies again on epipolar lines.
Kinect

- Unique IR speckle-pattern: no two sub-windows with the same pattern
- Energy along epipolar line has only one strong minimum.
- **Limitation**: works only indoor.
**RealSense**

- Hybrid approach one IR projector and two IR cameras.
- Combines advantages of stereo and structured light approach. So far best solution for robotics.
Depth from a single camera

- Is it possible to get the 3D points from a single camera?
Depth from a single camera

- Is it possible to get the 3D points from a single camera?
- Theoretically yes (if scene is static and the camera moves around sufficiently).
Depth from a single camera

- Is it possible to get the 3D points from a single camera?

- Theoretically yes (if scene is static and the camera moves around sufficiently).

- We have also two cameras. Main difference is that they have been captured in different times and the relative motion (i.e. epipolar geometry) is unknown.
Depth from a single camera

- Is it possible to get the 3D points from a single camera?
- Theoretically yes (if scene is static and the camera moves around sufficiently).
- We have also two cameras. Main difference is that they have been captured in different times and the relative motion (i.e. epipolar geometry) is unknown.
- The second part of this lecture is about how to estimate online both the relative motion of the camera and the 3D model of the world from captured images.
Depth from a single camera

- Is it possible to get the 3D points from a single camera?
- Theoretically yes (if scene is static and the camera moves around sufficiently).
- We have also two cameras. Main difference is that they have been captured in different times and the relative motion (i.e. epipolar geometry) is unknown.
- The second part of this lecture is about how to estimate online both the relative motion of the camera and the 3D model of the world from captured images.
- We assume, that at least the camera intrinsic parameters $K$ has been calibrated offline.
Algorithm at glance

1. Get image $I_k$.

2. Estimate tentative correspondences between $I_{k-1}$ and $I_k$.

3. Find correct correspondences and robustly estimate essential matrix $E$.

4. Decompose $E$ into $R_k$ and $t_k$.

5. Compute 3D model (points $X$).

6. Rescale $t_k$ according to relative scale $r$.

7. $k = k + 1$
Feature point detection

- Which points are suitable?
Feature point detection

- Feature points must be well distinguishable from its neighbourhood.

\[
E(u, v) = \sum_{x, y} \left( I(x + u, y + v) - I(x, y) \right)^2 \approx [u \ v] M [u \ v]
\]

\( \lambda_1 \text{ and } \lambda_2 \text{ are large} \)
Feature point detection

Feature points must be well distinguishable from its neighbourhood.

\[ E(u, v) = \sum_{x, y} \left( I(x + u, y + v) - I(x, y) \right)^2 \approx [u \ v] \ M \ [u \ v] \]

large \( \lambda_1 \), small \( \lambda_2 \)
Feature point detection

- Feature points must be well distinguishable from its neighbourhood.

\[ E(u, v) = \sum_{x,y} \left( I(x + u, y + v) - I(x, y) \right)^2 \approx [u \ v] M [u \ v] \]
Estimate tentative correspondences

- Estimate tentative correspondences by matching pixel neighbourhoods.
- Matching pixels: Tracking - for high temporal resolution
  OpenCV Lucas-Kanade tracker
- Matching invariant descriptors: Detection - for high spatial resolution
  OpenCV: SIFT, SURF etc ...
Algorithm at glance

1. Get image $I_k$.

2. Estimate tentative correspondences between $I_{k-1}$ and $I_k$.

3. Find correct correspondences and robustly estimate essential matrix $E$.

4. Decompose $E$ into $R_k$ and $t_k$.

5. Compute 3D model (points $X$).

6. Rescale $t_k$ according to relative scale $r$.

7. $k = k + 1$
Estimate essential matrix

- most of the tentative correspondences is **incorrect**,
- $L_2$-norm is very sensitive to such incorrect correspondence (i.e. outliers).
- Direct minimization of the $L_2$-norm, yields poor essential matrix

$$
e^* = \arg \min_e \|Ae\|
\text{ s.t. } \|e\| = 1$$
Estimate essential matrix by minimizing box-penalty function

- We will use outlier-insensitive estimation which will find both:
  - the correct essential matrix and
  - the set of correct correspondences (i.e. inliers).
Estimate essential matrix by minimizing box-penalty function

- What makes the $L_2$-norm outlier-sensitive?
Estimate essential matrix by minimizing box-penalty function

- What makes the $L_2$-norm outlier-sensitive?

- $L_2$-norm:

  $$\arg \min_{e} \| Ae \|
  \text{ s.t. } \| e \| = 1$$
Estimate essential matrix by minimizing box-penalty function

- What makes the $L_2$-norm outlier-sensitive?

- $L_2$-norm:
  \[
  \arg \min_e \|Ae\| \\
  \text{s.t. } \|e\| = 1
  \]

- Box-penalty:
  \[
  \arg \min_e 1 - \rho(Ae) \\
  \text{s.t. } \|e\| = 1
  \]
RANSAC algorithm

We solve the following not-convex and not-differentiable optimization task:

\[
\arg\min_e 1 - \rho(Ae) = \arg\max_e \rho(Ae)
\]

\[
\text{s.t. } \|e\| = 1 \quad \text{s.t. } \|e\| = 1
\]
RANSAC algorithm

- We solve the following not-convex and not-differentiable optimization task:

$$\arg\min_{e} 1 - \rho(Ae) = \arg\max_{e} \rho(Ae)$$

s.t. $\|e\| = 1$  \hspace{1cm} s.t. $\|e\| = 1$

- RANSAC (RAandom SAmples Consensus) algorithm:
  1. Randomly choose minimal subset of equations (rows) $B$ from $A$.
  2. Solve constrained LSQ problem by SVD decomposition:

$$e^* = \arg\min_{e} \|Be\|$$

s.t. $\|e\| = 1$

3. Estimate $\rho(Ae^*)$ as the number of rows $a_i^T$ of $A$ which satisfy $|a_i^T e^*| < \epsilon$.

4. If $\rho_{\text{max}} > \rho(e^*)$ then $\rho_{\text{max}} = \rho(e^*)$ and $e_{\text{max}} = e^*$.

5. Repeat from 1 until the optimum is found with sufficient probability.
RANSAC properties

**Important result 3:** Let us denote

- $N$ . . . number of data points.
- $w$ . . . fraction of inliers.
- $s$ . . . size of the sample
- $K$ . . . number of trials.
- $p$ . . . probability to select uncontaminated samples at least once

then

$$K = \frac{\log(1 - p)}{\log(1 - w^s)}$$
RANSAC properties

**Important result 3:** Let us denote

- \( N \) . . . number of data points.
- \( w \) . . . fraction of inliers.
- \( s \) . . . size of the sample
- \( K \) . . . number of trials.
- \( p \) . . . probability to select uncontaminated samples at least once

then

\[
K = \frac{\log(1 - p)}{\log(1 - w^s)}
\]

We search for 8 unknowns (\( \dim(e) = 9 \) minus scale) ⇒ at least 8 correspondences needed ⇒ \( s = 8 \) ⇒ \( K \) grows fast with \( s \).

However you want to find only camera translation (3 DoFs) and rotation (3 DoFs) minus scale ⇒ 5-point algorithm [Nister 2003].
Algorithm at glance

1. Get image $I_k$.
2. Estimate tentative correspondences between $I_{k-1}$ and $I_k$.
3. Find correct correspondences and compute essential matrix $E$.
4. Decompose $E$ into $R_k$ and $t_k$.
5. Compute 3D model (points $X$).
6. Rescale $t_k$ according to relative scale $r$.
7. $k = k + 1$
Decompose $E$ into $R$ and $t$

Once you find $E$, you can estimate camera motion by SVD ($E = U\Sigma V^T$) as follows: $[t]_\times = VW\Sigma V^T$, $R = UW^{-1}V^T$, but !!!:
Decompose $E$ into $R$ and $t$

- Once you find $E$, you can estimate camera motion by SVD ($E = U\Sigma V^T$) as follows: $[t]_\times = VW\Sigma V^T$, $R = UW^{-1}V^T$, but !!!:

- Scale $r$ is unknown (if $\|A \cdot e^*\| \approx 0$, then $\|A \cdot (re^*)\| \approx 0$).
Decompose $E$ into $R$ and $t$

- Once you find $E$, you can estimate camera motion by SVD ($E = U \Sigma V^T$) as follows: $[t]_\times = VW\Sigma V^T$, $R = UW^{-1}V^T$, **but !!!:**

- Scale $r$ is unknown (if $\|A \cdot e^*\| \approx 0$, then $\|A \cdot (re^*)\| \approx 0$). 

![Diagram showing the relationship between camera positions and projections on the image plane](image.png)
Algorithm at glance

1. Get image $I_k$.
2. Estimate tentative correspondences between $I_{k-1}$ and $I_k$ (either feature matching or tracking).
3. Find correct correspondences and compute essential matrix $E$.
4. Decompose $E$ into $R_k$ and $t_k$.
5. Compute 3D model (points $X$).
6. Rescale $t_k$ according to relative scale $r$.
7. $k = k + 1$
Compute 3D model

- Scene point $X$ is observed by two cameras $P$ and $Q$.
- Let $u = [u_1 \ u_2]^\top$ and $v = [v_1 \ v_2]^\top$ are projections of $X$ in $P$ and $Q$,
- then

$$u_1 = \frac{p_1^\top X}{p_3^\top X} \Rightarrow u_1 p_3^\top X - p_1^\top X = 0$$
Compute 3D model

- Scene point $X$ is observed by two cameras $P$ and $Q$.

- Let $u = [u_1 \ u_2]^\top$ and $v = [v_1 \ v_2]^\top$ be a correspondence pair (i.e. projections of $X$ in $P$ and $Q$).

- Then

$$u_1 = \frac{p_1^\top X}{p_3^\top X} \Rightarrow u_1 p_3^\top X - p_1^\top X = 0$$

- and similarly ...

$$u_2 = \frac{p_2^\top X}{p_3^\top X} \Rightarrow u_2 p_3^\top X - p_2^\top X = 0$$

$$v_1 = \frac{q_1^\top X}{q_3^\top X} \Rightarrow v_1 q_3^\top X - q_1^\top X = 0$$

$$v_2 = \frac{q_2^\top X}{q_3^\top X} \Rightarrow v_2 q_3^\top X - q_2^\top X = 0$$
Compute 3D model

- Which is $4 \times 4$ homogeneous system of linear equations:

$$
\begin{bmatrix}
    u_1 p_3^\top - p_1^\top \\
    u_2 p_3^\top - p_2^\top \\
    v_1 q_3^\top - q_1^\top \\
    v_2 q_3^\top - q_2^\top 
\end{bmatrix} \mathbf{X} = 0
$$
Algorithm at glance

1. Get image $I_k$.
2. Compute correspondences between $I_{k-1}$ and $I_k$ (either feature matching or tracking).
3. Find correct correspondences and compute essential matrix $E$.
4. Decompose $E$ into $R_k$ and $t_k$.
5. Compute 3D model (points $X$).
6. Rescale $t_k$ according to relative scale $r$.
7. $k = k + 1$
Estimating camera motion - relative scale

1. You cannot get absolute scale (without a calibration object).
Estimating camera motion - relative scale

1. You cannot get absolute scale (without a calibration object).
2. If you estimate motion (and 3D model) from $C_1, C_2$
Estimating camera motion - relative scale

1. You cannot get absolute scale (without calibration object).

2. If you estimate motion (and 3D model) from $C_1, C_2$ and then from $C_2, C_3$ you can have completely different scale.
Estimating camera motion - relative scale

1. You cannot get absolute scale (without calibration object).
2. If you estimate motion (and 3D model) from $C_1, C_2$ and then from $C_2, C_3$ you can have completely different scale.
3. You want to keep the same relative scale $r$ by rescaling $t$ (and 3D)

$$r = \frac{d_k}{d_{k-1}} = \frac{\|X_k - Y_k\|}{\|X_{k-1} - Y_{k-1}\|}$$
What we did not speak about.

- Result is usually improved by gradient descent of the reprojection error (bundle adjustment).
- Error accumulates over time $\Rightarrow$ drift $\Rightarrow$ loop-closure needed.
- Avoid motion estimation for small motions or pure rotation (keyframe detection)
- Single camera is usually fused with IMU (e.g. Google project Tango).
- Many papers about clever similarity measure for tentative correspondences.

Before loop closing

After loop closing