Lecture 07: Multi-View Geometry I

1 Multi-View Geometry 1

One can define different problem formulations:

- **3D reconstruction from multiple views:**
  - **Assumptions:** K, T, R are **known**.
  - **Goal:** Recover the 3D structure from images.
- **Structure from motion**
  - **Assumptions:** K, T, R are **unknown**.
  - **Goal:** Recover simultaneously 3D scene structure and camera poses (up to scale) from multiple images.

For a 2-view geometry, we can define the same problems to be:

- **Depth from stereo (stereo vision)**
  - **Assumptions:** K, T, R are known.
  - **Goal:** Recover the 3D structure from images.
- **2-view structure from motion**
  - **Assumptions:** K, T, R are **unknown**.
  - **Goal:** Recover simultaneously 3D scene structure, camera poses (up to scale) and intrinsic parameters from two different views of the scene.

1.1 Depth from Stereo

From a single camera, we can only compute the **ray** on which each image point lies. With a stereo camera (binocular), we can solve for the intersection of the rays and eventually recover the 3D structure.

1.1.1 The human binocular system

Stereopsys: the brain allows us to see the left and right retinal images as a single 3D image. The images project on our retina up-side-down but our brains let us perceive them as straight. Radial distortion is removed. This process is also known as **rectification**. The distance of two seen images is called **disparity** (allows us to perceive the depth, i.e. the **smaller** the disparity, the **farther** the object). An application of this concept are **stereograms**.
As simple test, fix an object, open and close alternatively the left and the right eyes: you will observe an horizontal displacement. This displacement is called **disparity**. Note that the smaller the disparity, the farther the object. Another applications is stereo photography, stereo viewers. Take two pictures of the same subject from two different viewpoints and display them so that each eye sees only one of the images.

1.1.2 Stereo Vision: basics

The basic principle behind stereo vision is **triangulation**.

- Gives reconstruction as intersection of two rays.
- Requires **camera pose (calibration) and point correspondence** (matching pairing points of the two images which are the projection of the same point in the scene).

There are basically two cases

1. Simplified case: identical cameras are **aligned**.
2. General case: different cameras are **not aligned**.

Simplified Case

The two cameras are identical, meaning that they have the same **focal length**, and are **aligned** with the x-axis. If we have a world point P_w, a distance from the axis to the point Z_P, a distance between the cameras b and focal length f, we can use similar triangles from Figure 1 and get

$$\begin{aligned}\frac{f}{Z_P} &= \frac{u_l}{X_P}\\ \frac{f}{Z_P} &= \frac{-u_r}{b-X_P}\\ \Rightarrow X_P &= \frac{u_l\cdot b}{u_l-u_r}\\ \Rightarrow Z_P &= \frac{b\cdot f}{u_l-u_r}.\end{aligned} \tag{1.1}$$

The difference $u_l - u_r$ is called **disparity**: difference in image location of the projection of a 3D point on two image planes. Observations from this equation are

- Distance is inversely proportional to disparity: the distance to near objects can be measured more accurately than that to distant objects.
- Disparity is proportional to b. For a given disparity error, the accuracy of the depth estimate increases with increasing baseline b.
- As b is increased, since the distance of the two cameras is increased, some objects may appear in one camera but not in the other one (field of view of cameras). These objects won't have disparity.
- If the baseline b is unknown it is possible to reconstruct the scene up to a scale (structure from motion!)

Now, based on these observations, what is the optimal baseline?

- **If we choose it too small:**
  - Large depth error

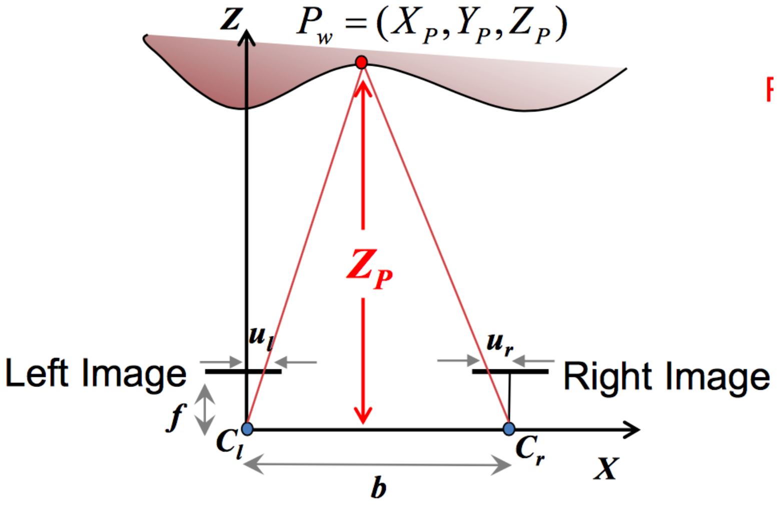

Figure 1: Simplified case for stereo vision.

  - Quantification of error as a function of the disparity?

- **If we choose it too large:**
  - Minimum measurable distance increases.
  - Difficult search problem for close objects.

General Case: triangulation

In reality cameras are not identical and aligning two of them on a horizontal axis is impossible. **Why?**

- There will always be differences in focal lengths due to manufacturing.
- The internal orientation of the CCD (elements which accumulate charge) in the camera package is unknown, theoretically orientated, but not in reality.

. In order to be able to use a stereo camera, we need to compute:

- The extrinsic parameters (relative rotation and translation) and
- the intrinsic parameters (focal length, optical center, radial distortion of each camera).

In order to do this we use calibration methods (Tsai or homographies). How do we get the **relative pose?** Lines do not always interset in the 3D space: we want to minimize the error. Assuming the two cameras we have

$$\tilde{p}_l = \lambda_l \cdot \begin{pmatrix} u_l \\ v_l \\ 1 \end{pmatrix} = K_l \cdot \begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix}, \quad \tilde{p}_r = \lambda_r \cdot \begin{pmatrix} u_r \\ v_r \\ 1 \end{pmatrix} = K_r \cdot R \cdot \begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} + T. \tag{1.2}$$

Triangulation: The problem of determining the 3D position of a point given a set of corresponding image locations and known camera poses.

In order to triangulate, we use **least-squares** approximation:

$$\begin{aligned}
\lambda_1 \cdot \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} &= K \cdot [I|0] \cdot \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} \Rightarrow \lambda_1 p_1 = M_1 \cdot P \qquad \text{left camera.} \\
\lambda_2 \cdot \begin{pmatrix} u_2 \\ v_2 \\ 1 \end{pmatrix} &= K \cdot [R|T] \cdot \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} \Rightarrow \lambda_2 p_2 = M_2 \cdot P \qquad \text{right camera.}
\end{aligned} \tag{1.3}$$

We solve for P and get a system of the form $A \cdot \begin{pmatrix} \lambda_1 \\ \lambda_2 \end{pmatrix} = b$, which cannot be inverted (A is a 3×2 matrix). We use pseudoinverse approximation (least squares) and get

$$A^T \cdot A \cdot \begin{pmatrix} \lambda_1 \\ \lambda_2 \end{pmatrix} = A^T \cdot b \Rightarrow \begin{pmatrix} \lambda_1 \\ \lambda_2 \end{pmatrix} = (A^T \cdot A)^{-1} \cdot A^T \cdot b. \tag{1.4}$$

Remark. This is a problem with 6 equations and 5 unknowns: the 3 elements of the coordinates of the world point and the two depth factors λ_1, λ_2.

$$^{232}_{9}\mathrm{Th} \longrightarrow {}^{228}_{8}\mathrm{Ra} + {}^{2}_{4}\mathrm{He} \tag{1.5}$$

Interpretation: Given the projections $p_{1,2}$ of a 3D point P in two or more images, we want to find the coordinates of the 3D point by intersecting the two rays corresponding to the projections. We want to find the shortest segment connecting the two viewing rays and let P be the **midpoint** of the segment. The two rays won't meet exactly because of **noise** and **numerical errors**.

Triangulation: nonlinear approach

We want to find the point P that minimizes the **sum of squared reprojection errors**

$$SSRE = d^2\left(p_1, \pi_1(P)\right) + d^2\left(p_2, \pi_2(P)\right), \tag{1.6}$$

where

$$d\left(p_1, \pi_1(P)\right) = \|p_1 - \pi_1(P)\| \tag{1.7}$$

is called **reprojection error**. The observed points are p_1, p_2 and the reprojected ones are M_1P, M_2P. In practice, this is done by initializing P using a linear approach and then minimize the SSRE using Gauss-Newton or Levenberg-Marquardt.

1.1.3 Correspondence Problem

Given a point p in a first image, where is its corresponding point p' in the right image?
Correspondence Search: Identify image patches in the left and in the right images, corresponding to the same scene structure. Similarity measures include:

- (Z)ZNCC
- (Z)SSD

- (Z)SAD
- Census Transform

This brings up an issue: exhaustive image search can be computationally very expensive! Can we do that in 1D? Potential matches for p **have to lie** on the corresponding epipolar line l'. Considering Figure 2, one can note the following:

- The **epipolar line** is the projection of the infinite ray $\pi^{-1}(p)$ corresponding to p in the other camera image. Since

  $$\pi(p) = \lambda K, \tag{1.8}$$

  one can write

  $$\pi(p)^{-1} = \lambda K^{-1} p. \tag{1.9}$$

  This makes sense: if we observe a projection p_1 in the left camera, this can correspond to each world point lying on the infinite ray. In fact, every one of these points would project into p_1. All these points have a different projection on the right camera, which in 2D forms the epipolar line.

- The **epipole** is the projection of the optical center on the other camera image.
- A **stereo camera** has two epipoles!

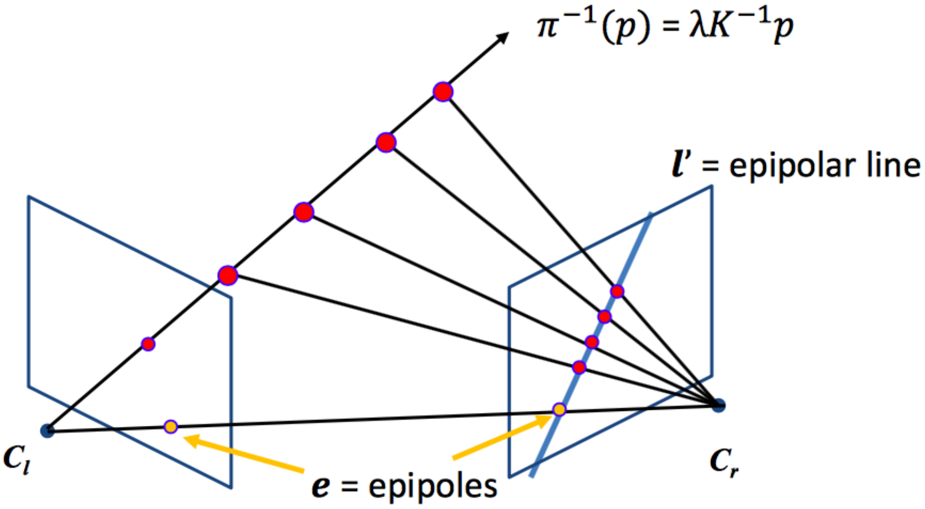

Figure 2: Epipolar lines and epipoles.

1.1.4 The Epipolar Constraint

The epipolar plane is uniquely defined by the two optical centers C_l, C_r and one image point p. The **epipolar constraint** constraints the location, in the second view, of the corresponding point to a given point in the first view. This reduces the search to a 1D problem along conjugate epipolar lines.

Example 1. Converging Cameras. All epipolar lines intersect at the epipole! As the position od the 3D point varies, the epipolar line *rotates* about the baseline.

Example 2. Identical and horizontally-aligned cameras e and e' are at infinity.

Example 3. Forward motion. Epipoles have the same coordinates in both images: points move along lines radiating from e: *Focus of expansion*.

Stereo Rectification

We can note the following:

- Even in commercial stereo cameras, left and right images are not aligned.
- It is convenient if image scanlines are the epipolar lines.
- Stereo rectification warps left and right images into new *rectified* images, whose epipolar lines are aligned to the baseline.

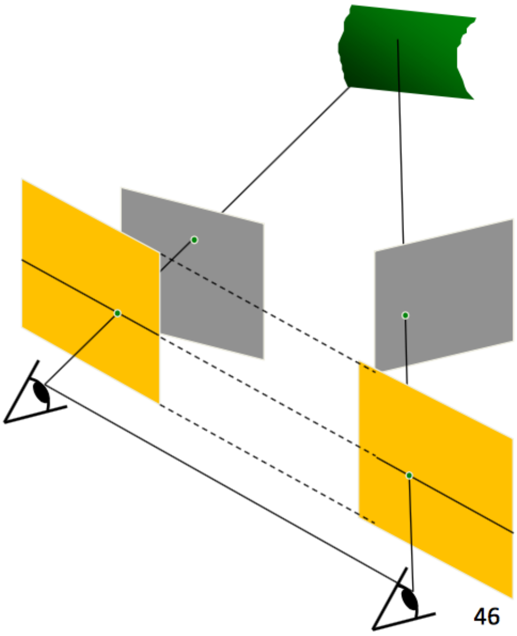

Figure 3: Stereo Rectification.

- It Reprojects image planes onto a common plane parallel to the baseline.
- It works by computing two homographies, one for each input image reprojection.
- Then, scanlines are **aligned** and epipolar lines are **horizontal**.

Idea: we define two new Perspective Projection Matrices obtained by rotating the old ones around their optical centers, until focal planes become coplanar (containing the baseline). This ensures **epipoles at infinity**. In order to have horizontal epipolar lines, the baseline should be *parallel* to the new X axis of both cameras. Moreover, corresponding points should have **the same vertical coordinate**. This is obtained by having same intrinsic parameters for the new cameras. Since the the focal length is the same, the new image planes are coplanar. PPMs are the same as the old cameras, whereas the new orientation (the same for both cameras) differs from the old ones by suitable rotations. For both cameras, **intrinsic parameters are the same**.

Implementation

1. The perspective equation for a point in the world is

$$\text{image point} = \tilde{p} = \begin{pmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{pmatrix} = \lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K[R|T] \cdot \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}. \tag{1.10}$$

   This can be rewritten in a more convenient way by considering $[R|T]$ as the transformation from the world to the Camera frame (T expressed as C):

$$\lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K \cdot R^{-1} \cdot \left(\begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} - C \right) \tag{1.11}$$

2. We can then write the Perspective Equation for the Left and Right cameras: we assume for generality that they have the same intrinsic parameters:

$$\begin{aligned} \lambda_L \cdot \begin{pmatrix} u_L \\ v_L \\ 1 \end{pmatrix} &= K_L \cdot R_L^{-1} \cdot \left(\begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} - C_L \right) \quad \text{left camera} \\ \lambda_R \cdot \begin{pmatrix} u_R \\ v_R \\ 1 \end{pmatrix} &= K_R \cdot R_R^{-1} \cdot \left(\begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} - C_R \right) \quad \text{right camera} \end{aligned} \tag{1.12}$$

3. The goal of stereo rectification is to warp left and right camera images such that their focal planes are **coplanar** and the intrinsic parameters are identical. It follows

$$\begin{aligned} \lambda_L \cdot \begin{pmatrix} u_L \\ v_L \\ 1 \end{pmatrix} = K_L \cdot R_L^{-1} \cdot \left(\begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} - C_L \right) &\to \overline{\lambda}_L \cdot \begin{pmatrix} \overline{u}_L \\ \overline{v}_L \\ 1 \end{pmatrix} = \overline{K} \cdot \overline{R}^{-1} \cdot \left(\begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} - C_L \right) \\ \lambda_R \cdot \begin{pmatrix} u_R \\ v_R \\ 1 \end{pmatrix} = K_R \cdot R_R^{-1} \cdot \left(\begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} - C_R \right) &\to \overline{\lambda}_R \cdot \begin{pmatrix} \overline{u}_R \\ \overline{v}_R \\ 1 \end{pmatrix} = \overline{K} \cdot \overline{R}^{-1} \cdot \left(\begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} - C_R \right). \end{aligned} \tag{1.13}$$

4. By solving for $\begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix}$ for each camera, we can compute the Homography (or warping) that needs to be applied to rectify each camera image:

$$\begin{aligned} \overline{\lambda}_L \cdot \begin{pmatrix} \overline{u}_L \\ \overline{v}_L \\ 1 \end{pmatrix} &= \lambda_L \cdot \underbrace{\overline{K} \cdot \overline{R}^{-1} \cdot R_L \cdot K_L^{-1}}_{\text{homography left camera}} \cdot \begin{pmatrix} u_L \\ v_L \\ 1 \end{pmatrix} \\ \overline{\lambda}_R \cdot \begin{pmatrix} \overline{u}_R \\ \overline{v}_R \\ 1 \end{pmatrix} &= \lambda_R \cdot \underbrace{\overline{K} \cdot \overline{R}^{-1} \cdot R_R \cdot K_R^{-1}}_{\text{homography right camera}} \cdot \begin{pmatrix} u_R \\ v_R \\ 1 \end{pmatrix} \end{aligned} \tag{1.14}$$

5. The new $\overline{K}, \overline{R}$ can be chosen as

$$\begin{aligned} \overline{K} &= \frac{K_L + K_R}{2} \\ \overline{R} &= [\overline{r}_1, \overline{r}_2, \overline{r}_3], \end{aligned} \tag{1.15}$$

where $\overline{r}_1, \overline{r}_2, \overline{r}_3$ are the column vectors of $\overline{R}$. These can be computed as

$$\begin{aligned} &\overline{r}_1 = \frac{C_2 - C_1}{\|C_2 - C_1\|} \\ &\overline{r}_2 = r_3 \times \overline{r_1} \text{ where } r_3 \text{ is the third column of } R_L \\ &\overline{r}_3 = \overline{r}_1 \times \overline{r}_2. \end{aligned} \tag{1.16}$$

For more details have a look at `http://rpg.ifi.uzh.ch/docs/teaching/2016/rectification.zip`.

In order to solve the correspondence problem, we want to average noise effects. One can use a window around the point of interest and use similarity measures to find neighborhoods (these should be similar in intensity patterns).

Correlation-based window matching

Problem: textureless regions (**aperture problem**) lead to high ambiguity! **Solution:** increase window size to distinguish them! This can be achieved with:

- Smaller window: **more detail but more noise**!
- Larger window: **smoother disparity maps but less detail**.

Disparity Map

A disparity map appears as a grayscale image where the intensity of every pixel point is proportional to the disparity of that pixel in the left and right image: objects that are closer to the camera appear lighter, whiler farther objects appear darker. This is used as input to dense 3D reconstruction. Its computation works as follows:

1. For each pixel in the left image, find its corresponding point in the right image.
2. Compute the disparity for each pair of correspondences.
3. Visualized in gray-scale or color coded image.

Close objects experience bigger disparity: they appear brighter in disparity map. The depth Z can be computed from the disparity by recalling that

$$Z_P = \frac{bf}{u_l - u_r}. \tag{1.17}$$

This is really useful for obstacle avoidance. Challenges include: occlusion, repetition, non-lambertian surfaces (specularities), textureless surfaces.

Improvements:

Multiple matches could satisfy the epipolar constraint. A good way to address the problem would be to have:

- Uniqueness: only one match in right image for every point in left image.
- Ordering: points on same surface will be in same order in both views
- Disparity gradient: disparity changes smoothly between points on the same surface.

Sparse Stereo Correspondence restricts search to sparse set.

1.2 Understanding Check

Are you able to answer the following questions?

- *Can you relate Structure from Motion to 3D reconstruction? In what they differ?*
- *Can you define disparity in both the simplified and the general case?*
- *Can you provide a mathematical expression of depth as a function of the baseline, the disparity and the focal length?*
- *Can you apply error propagation to derive an expression for depth uncertainty? How can we improve the uncertainty?*
- *Can you analyse the effects of a large/small baseline?*
- *What is the closest depth that a stereo camera can measure?*
- *Are you able to show mathematically how to compute the intersection of two lines (linearly and non-linearly)?*
- *What is the geometric interpretation of the linear and non-linear approaches and what error do they minimize?*
- *Are you able to provide a definition of epipole, epipolar line and epipolar plane?*
- *Are you able to draw the epipolar lines for two converging cameras, for a forward motion situation, and for a side-moving camera?*
- *Are you able to define stereo rectification and to derive mathematically the rectifying homographies?*
- *How is the disparity map computed?*
- *How can one establish stereo correspondences with subpixel accuracy?*
- *Describe one or more simple ways to reject outliers in stereo correspondences.*
- *Is stereo vision the only way of estimating depth information? If not, are you able to list alternative options?*