Global Alignment and Structure from Motion

CSE576, Spring 2009
Sameer Agarwal

Overview

1. Global refinement for Image stitching
2. Camera calibration
3. Pose estimation and Triangulation
4. Structure from Motion

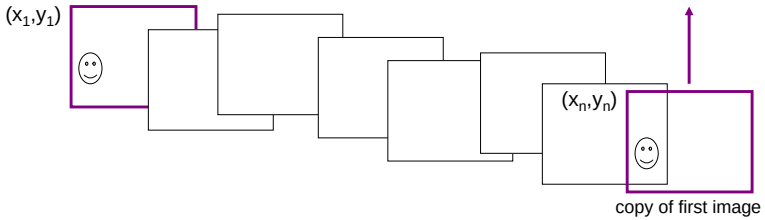
Readings

Chapter 3, Noah Snavely's thesis

Supplementary readings:

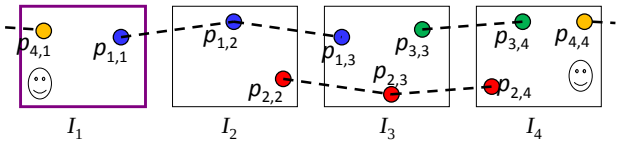
- Hartley & Zisserman, **Multiview Geometry,** Appendices 5 and 6.
- Brown & Lowe, **Recognizing Panoramas**, ICCV 2003

Problem: Drift



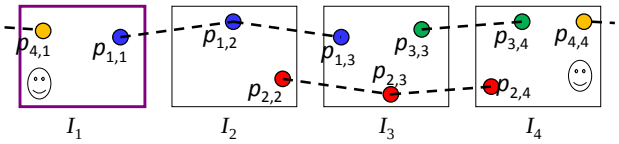
– add another copy of first image at the end
– this gives a constraint: $y_n = y_1$
– there are a bunch of ways to solve this problem
- add displacement of $(y_1 - y_n)/(n - 1)$ to each image after the first
- compute a global warp: $y' = y + ax$
- run a big optimization problem, incorporating this constraint
  – best solution, but more complicated
  – known as "bundle adjustment"

Global optimization



- Minimize a global energy function:
  - What are the variables?
    - The translation $t_j = (x_j, y_j)$ for each image
  - What is the objective function?
    - We have a set of matched features $p_{i,j} = (u_{i,j}, v_{i,j})$
    - For each point match $(p_{i,j}, p_{i,j+1})$:
      - $p_{i,j+1} - p_{i,j} = t_{j+1} - t_j$

Global optimization



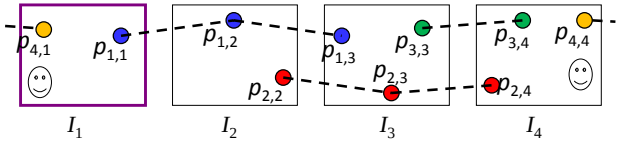
w_{ij} = 1 if feature i is visible in images j and j+1
0 otherwise

$$p_{1,2} - p_{1,1} = t_2 - t_1$$
$$p_{1,3} - p_{1,2} = t_3 - t_2$$
$$p_{2,3} - p_{2,2} = t_3 - t_2$$
$$\dots$$
$$v_{4,1} - v_{4,4} = y_1 - y_4$$

minimize

$$\sum_{i=1}^{m}\sum_{j=1}^{n-1} w_{ij} \cdot \left\|(p_{i,j+1} - p_{i,j}) - (t_{j+1} - t_j)\right\|^2 + \sum_{i=1}^{m} w_{in} \cdot \left\|(v_{i,1} - v_{i,n}) - (y_1 - y_n)\right\|^2$$

Global optimization



$$\begin{bmatrix} -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ & & & \dots & & & & \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \\ x_4 \\ y_4 \end{bmatrix} = \begin{bmatrix} u_{1,2} - u_{1,1} \\ v_{1,2} - v_{1,1} \\ \vdots \\ v_{4,1} - v_{4,4} \end{bmatrix}$$

A (2m x 2n) **x** (2n x 1) **b** (2m x 1)

Global optimization

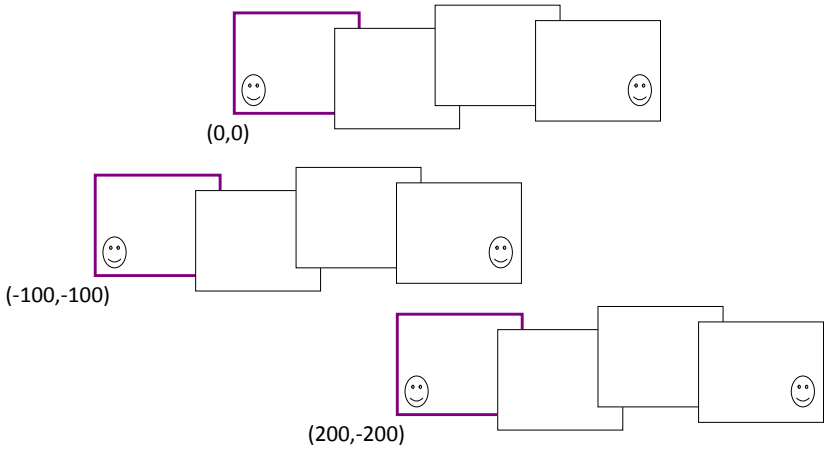
$$\begin{bmatrix} -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ & & & \dots & & & & \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \\ x_4 \\ y_4 \end{bmatrix} = \begin{bmatrix} u_{1,2} - u_{1,1} \\ v_{1,2} - v_{1,1} \\ \vdots \\ v_{4,1} - v_{4,4} \end{bmatrix}$$

A (2m x 2n) **x** (2n x 1) **b** (2m x 1)

Defines a least squares problem: minimize $\|\mathbf{Ax} - \mathbf{b}\|$

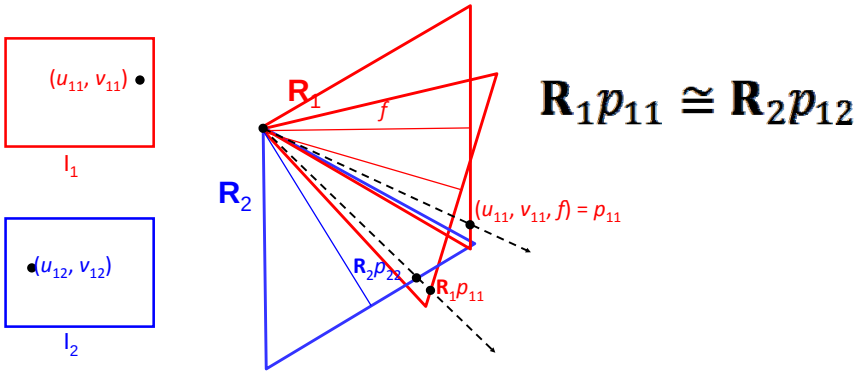
- Solution: $\hat{\mathbf{x}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$
- Problem: there is no unique solution for $\hat{\mathbf{x}}$ (det $(\mathbf{A}^T\mathbf{A}) = 0$)
- We can add a global offset to a solution $\hat{\mathbf{x}}$ and get the same error

Ambiguity in global location



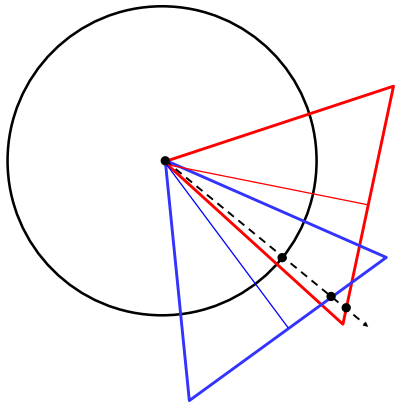
- Each of these solutions has the same error
- Called the *gauge ambiguity*
- Solution: fix the position of one image (e.g., make the origin of the 1st image (0,0))

Solving for rotations



$$\mathbf{R}_1 p_{11} \cong \mathbf{R}_2 p_{12}$$

Solving for rotations



$$\mathbf{R}_1 p_{11} \cong \mathbf{R}_2 p_{12}$$

$$\mathbf{R}_1 \hat{p}_{11} = \mathbf{R}_2 \hat{p}_{12}$$

$$\text{minimize} \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} \cdot \left\| \mathbf{R}_{j+1} \hat{p}_{i,j+1} - \mathbf{R}_j \hat{p}_{i,j} \right\|^2$$

Parameterizing rotations

- How do we parameterize **R** and Δ**R**?
  - Euler angles: bad idea
  - quaternions: 4-vectors on unit sphere
  - Axis-angle representation (Rodriguez Formula)

$$x = \theta \hat{x}, \quad x \in \mathbb{R}^3$$

$$[\hat{x}]_\times = \begin{bmatrix} 0 & -\hat{x}_3 & \hat{x}_2 \\ \hat{x}_3 & 0 & -\hat{x}_1 \\ -\hat{x}_2 & \hat{x}_1 & 0 \end{bmatrix}$$

$$R(x) = I + \sin\theta [\hat{x}]_\times + (1 - \cos\theta)[\hat{x}]_\times^2$$

Nonlinear Least Squares

Global alignment

- Least-squares solution of $\min |\boldsymbol{R}_j\boldsymbol{p}_{ij} - \boldsymbol{R}_k\boldsymbol{p}_{ik}|^2$ or $\boldsymbol{R}_j\boldsymbol{p}_{ij} - \boldsymbol{R}_k\boldsymbol{p}_{ik} = 0$

1. Use the linearized update $(\boldsymbol{I}+[\boldsymbol{\omega}_j]_\times)\boldsymbol{R}_j\boldsymbol{p}_{ij} - (\boldsymbol{I}+[\boldsymbol{\omega}_k]_\times)\boldsymbol{R}_k\boldsymbol{p}_{ik} = 0$

- or
- $[\boldsymbol{q}_{ij}]_\times\boldsymbol{\omega}_j - [\boldsymbol{q}_{ik}]_\times\boldsymbol{\omega}_k = \boldsymbol{q}_{ij}-\boldsymbol{q}_{ik}, \quad \boldsymbol{q}_{ij} = \boldsymbol{R}_j\boldsymbol{p}_{ij}$

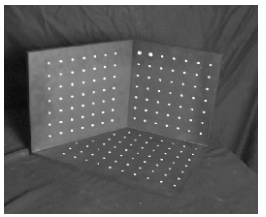
2. Estimate least square solution over $\{\boldsymbol{\omega}_i\}$
3. Iterate a few times (updating the $\{\boldsymbol{R}_i\}$)

Camera Calibration

Camera calibration

- Determine camera parameters from *known* 3D points or calibration object(s)

1. *internal* or *intrinsic* parameters such as focal length, optical center, aspect ratio: *what kind of camera?*
2. *external* or *extrinsic* (pose) parameters: *where is the camera?*



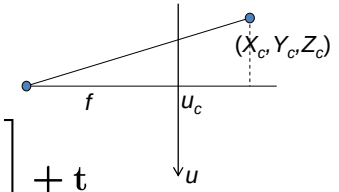
- How can we do this?

Camera calibration – approaches

- Possible approaches:

1. linear regression (least squares)
2. non-linear optimization
3. vanishing points
4. multiple planar patterns
5. panoramas (rotational motion)

Image formation equations



$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} \mathbf{R} \end{bmatrix}_{3\times 3} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \mathbf{t}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} f & 0 & u_c \\ 0 & f & v_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

Calibration matrix

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & u_c \\ 0 & f & v_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \mathbf{K}\ \mathbf{X}_c$$

- Is this form of K good enough?
- non-square pixels (digital video)
- skew
- radial distortion

$$\mathbf{K} = \begin{bmatrix} fa & s & u_c \\ 0 & f & v_c \\ 0 & 0 & 1 \end{bmatrix}$$

Camera matrix

- Fold *intrinsic* calibration matrix **K** and *extrinsic* pose parameters (**R**,**t**) together into a *camera matrix*
- **M** = **K** [**R** | **t**]

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- (put 1 in lower r.h. corner for 11 d.o.f.)

Camera matrix calibration

- Directly estimate 11 unknowns in the **M** matrix using known 3D points (X_i,Y_i,Z_i) and measured feature positions (u_i,v_i)

$$u_i = \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1}$$

$$v_i = \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1}$$

Camera matrix calibration

- Linear regression:
  - Bring denominator over, solve set of (over-determined) linear equations. How?

$$u_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1) = m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}$$

$$v_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1) = m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}$$

  - Le
  - Is this good enough?

Levenberg-Marquardt

- Iterative non-linear least squares [Press'92]
  - Linearize measurement equations

$$\hat{u}_i = f(\mathbf{m}, \mathbf{x}_i) + \frac{\partial f}{\partial \mathbf{m}}\Delta\mathbf{m}$$

$$\hat{v}_i = g(\mathbf{m}, \mathbf{x}_i) + \frac{\partial g}{\partial \mathbf{m}}\Delta\mathbf{m}$$

  - Substitute into log-likelihood equation: quadratic cost fu

$$\sum_i \sigma_i^{-2}(\hat{u}_i - u_i + \frac{\partial f}{\partial \mathbf{m}}\Delta\mathbf{m})^2 + \cdots$$

Levenberg-Marquardt

- Iterative non-linear least squares [Press'92]
  - Solve for minimum $\frac{\partial C}{\partial \mathbf{m}} = 0$

$$\mathbf{A}\Delta\mathbf{m} = \mathbf{b}$$

Hessian: $$\mathbf{A} = \left[\sum_i \sigma_i^{-2}\frac{\partial f}{\partial \mathbf{m}}\left(\frac{\partial f}{\partial \mathbf{m}}\right)^T + \cdots\right]$$

error: $$\mathbf{b} = \left[\sum_i \sigma_i^{-2}\frac{\partial f}{\partial \mathbf{m}}(u_i - \hat{u}_i) + \cdots\right]$$

Levenberg-Marquardt

- What if it doesn't converge?
  - Multiply diagonal by (1 + λ), increase λ until it does
  - Halve the step size ⊗**m** (my favorite)
  - Use line search
  - Other ideas?
- Uncertainty analysis: covariance Σ = A^{-1}
- Is *maximum* likelihood the best idea?
- How to start in vicinity of global minimum?

Camera matrix calibration

- Advantages:
  - very simple to formulate and solve
  - can recover **K** [**R** | **t**] from **M** using QR decomposition [Golub & VanLoan 96]
- Disadvantages:
  - doesn't compute internal parameters
  - can give garbage results
  - more unknowns than true degrees of freedom
  - need a separate camera matrix for each new view

Separate intrinsics / extrinsics

- New feature measurement equations

$$\begin{aligned} \hat{u}_{ij} &= f(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i) \\ \hat{v}_{ij} &= g(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i) \end{aligned}$$

- Use non-linear minimization
- Standard technique in photogrammetry, computer vision, computer graphics
  - [Tsai 87] – also estimates $|_1$ (freeware @ CMU) http://www.cs.cmu.edu/afs/cs/project/cil/ftp/html/v-source.html
  - [Bogart 91] – *View Correlation*

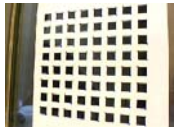
Intrinsic/extrinsic calibration

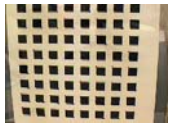
- Advantages:
  - can solve for more than one camera pose at a time
  - potentially fewer degrees of freedom
- Disadvantages:
  - more complex update rules
  - need a good initialization (recover **K** [**R** | **t**] from **M**)

Multi-plane calibration

- Use several images of planar target held at *unknown* orientations [Zhang 99]
  - Compute plane homographies

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \sim \mathrm{K} \begin{bmatrix} \mathrm{r}_1 & \mathrm{r}_2 & \mathrm{t} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \sim \mathrm{HX}$$



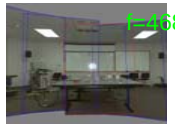


  - Solve for $\boldsymbol{K}^{-\mathsf{T}}\boldsymbol{K}^{-1}$ from $\boldsymbol{H}_k$'s
    - 1 plane if only *f* unknown
    - 2 planes if (f,u_c,v_c) unknown
    - 3+ planes for full $\boldsymbol{K}$
  - Code available from Zhang and OpenCV

Rotational motion

- Use pure rotation (large scene) to estimate *f*
  1. estimate *f* from pairwise homographies
  2. re-estimate *f* from 360º "gap"
  3. optimize over all $\{\boldsymbol{K},\boldsymbol{R}_j\}$ parameters

     [Stein 95; Hartley '97; Shum & Szeliski '00; Kang & Weiss '99]



- Most accurate way to get *f*, short of surveying distant points

Pose estimation and triangulation

Pose estimation

- Once the internal camera parameters are known, can compute camera pose

$$\begin{aligned} \hat{u}_{ij} &= f(\mathrm{K}, \mathrm{R}_j, \mathrm{t}_j, \mathrm{x}_i) \\ \hat{v}_{ij} &= g(\mathrm{K}, \mathrm{R}_j, \mathrm{t}_j, \mathrm{x}_i) \end{aligned}$$

  – [Tsai87] [Bogart91]
- Application: superimpose 3D graphics onto video

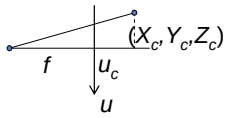
- How do we initialize $(\boldsymbol{R},\boldsymbol{t})$?

Pose estimation

- Previous initialization techniques:
  - vanishing points [Caprile 90]
  - planar pattern [Zhang 99]
- Other possibilities
  - *Through-the-Lens Camera Control* [Gleicher92]: differential update
  - 3+ point "linear methods":
  - [DeMenthon 95][Quan 99][Ameller 00]

Pose estimation

- Use inter-point distance constraints
  - [Quan 99][Ameller 00]



$$\mathbf{u}_i = \begin{bmatrix} u_i - u_c \\ v_i - v_c \\ f \end{bmatrix}, \quad x_i = \|\mathbf{X}_i\|$$

$$d_{ij}^2 = \|\mathbf{X}_i - \mathbf{X}_j\| = x_i^2 + x_j^2 - 2x_i x_j cos\theta_{ij}$$

  - Solve set of polynomial equations in x_i^{2p}
  - Recover R,t using procrustes analysis.

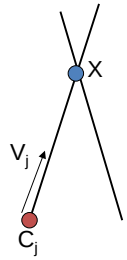
Triangulation

- Problem: Given some points in *correspondence* across two or more images (taken from calibrated cameras), $\{(u_j,v_j)\}$, compute the 3D location **X**

Triangulation

- **Method I**: intersect viewing rays in 3D, minimize:

$$\arg\min_{\mathbf{X}} \sum_j \|\mathbf{C}_j + s\mathbf{V}_j - \mathbf{X}\|$$



  - **X** is the unknown 3D point
  - $\mathbf{C}_j$ is the optical center of camera *j*
  - $\mathbf{V}_j$ is the *viewing ray* for pixel (u_j,v_j)
  - s_j is unknown distance along $\mathbf{V}_j$
- Advantage: geometrically intuitive

Triangulation

- **Method II**: solve linear equations in **X**
  - advantage: very simple

$$u_i = \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1}$$
$$v_i = \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1}$$

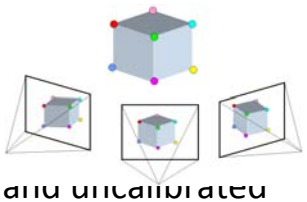
- **Method III**: non-linear minimization
  - advantage: most accurate (image plane error)

Structure from Motion

Structure from motion

- Given many points in *correspondence* across several images, $\{(u_{ij}, v_{ij})\}$, simultaneously compute the 3D location $\mathbf{x}_i$ and camera (or *motion*) parameters ($\mathbf{K}$, $\mathbf{R}_j$, $\mathbf{t}_j$)

$$\hat{u}_{ij} = f(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i)$$
$$\hat{v}_{ij} = g(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i)$$



- Two main variants: calibrated, and uncalibrated (sometimes associated with Euclidean and projective reconstructions)

Orthographic SFM

[Tomasi & Kanade, IJCV 92]

Results

- Look at paper figures…

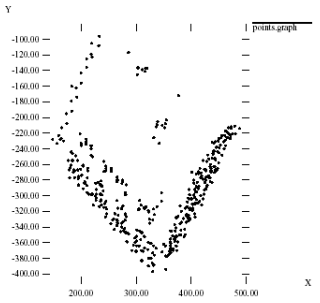


Figure 4.5: A view of the computed shape from approximately above the building (compare with figure 4.6).

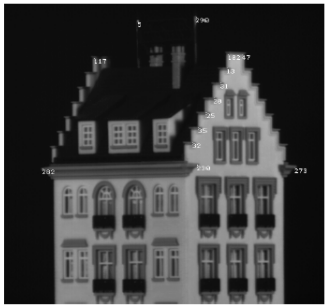


Figure 4.7: For a quantitative evaluation, distances between the features shown in the picture were measured on the actual model, and compared with the computed results. The comparison is shown in figure 4.8.

Structure from motion

$$\begin{aligned} \hat{u}_{ij} &= f(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i) \\ \hat{v}_{ij} &= g(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i) \end{aligned}$$

- How many points do we need to match?
- 2 frames:
  - ($\boldsymbol{R}$,$\boldsymbol{t}$): 5 dof + $3n$ point locations δ $4n$
  - point measurements ® n ε 5
- k frames: $6(k–1)$-1 + $3n$ δ $2kn$
- always want to use many more

Extensions

- Paraperspective
- [Poelman & Kanade, PAMI 97]
- Sequential Factorization
- [Morita & Kanade, PAMI 97]
- Factorization under perspective
- [Christy & Horaud, PAMI 96]
- [Sturm & Triggs, ECCV 96]
- Factorization with Uncertainty
- [Anandan & Irani, IJCV 2002]

Bundle Adjustment

$$\begin{aligned} \hat{u}_{ij} &= f(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i) \\ \hat{v}_{ij} &= g(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i) \end{aligned}$$

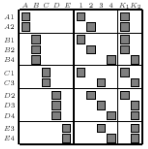
- What makes this non-linear minimization hard?
  - many more parameters: potentially slow
  - poorer conditioning (high correlation)
  - potentially lots of outliers
  - gauge (coordinate) freedom

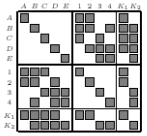
Lots of parameters: sparsity

$$\begin{aligned}\hat{u}_{ij} &= f(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i) \\ \hat{v}_{ij} &= g(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i)\end{aligned}$$

- Only a few entries in Jacobian are non-zero

$$\frac{\partial \hat{u}_{ij}}{\partial \mathbf{K}}, \quad \frac{\partial \hat{u}_{ij}}{\partial \mathbf{R}_j}, \quad \frac{\partial \hat{u}_{ij}}{\partial \mathbf{t}_j}, \quad \frac{\partial \hat{u}_{ij}}{\partial \mathbf{x}_i},$$

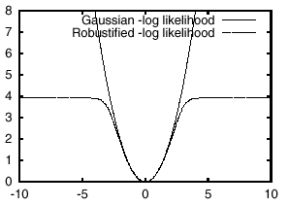
J = 

H = 

Robust error models

- Outlier rejection
  - use robust penalty applied to each set of joint measurements

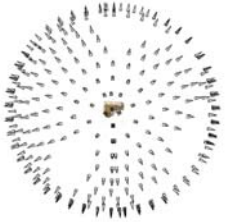
$$\sum_i \sigma_i^{-2} \rho\left(\sqrt{(u_i - \hat{u}_i)^2 + (v_i - \hat{v}_i)^2}\right)$$



  - for extremely bad data, use random sampling [RANSAC, Fischler & Bolles, CACM'81]

Structure from motion

Reconstruction (side)



(top)

- Input: images with points in correspondence
  $p_{i,j} = (u_{i,j}, v_{i,j})$
- Output
- structure: 3D location $\mathbf{x}_i$ for each point p_i
- motion: camera parameters $\mathbf{R}_j$, $\mathbf{t}_j$
- Objective function: minimize *reprojection error*

SfM objective function

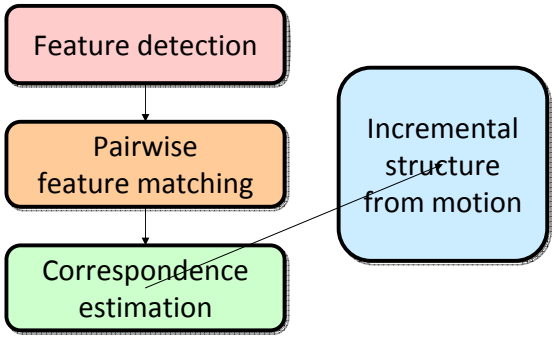
- Given point **x** and rotation and translation **R**, **t**

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \mathbf{R}\mathbf{x} + \mathbf{t} \qquad u' = \frac{fx'}{z'} \qquad v' = \frac{fy'}{z'} \qquad \begin{bmatrix} u' \\ v' \end{bmatrix} = \mathbf{P}(\mathbf{x}, \mathbf{R}, \mathbf{t})$$

- Minimize sum of squared reprojection errors:

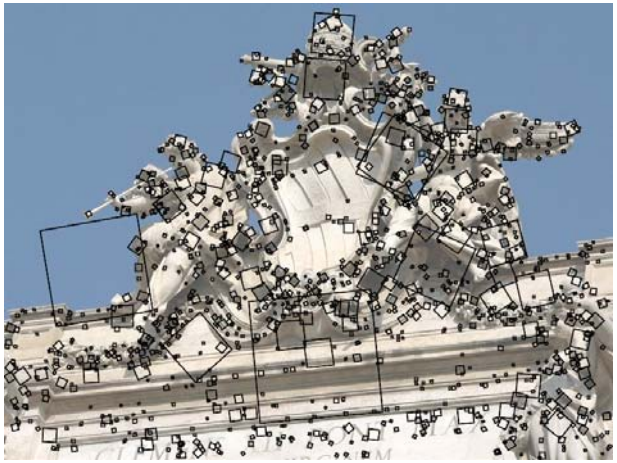
$$g(\mathbf{X}, \mathbf{R}, \mathbf{T}) = \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} \cdot \left\| \underbrace{\mathbf{P}(\mathbf{x}_i, \mathbf{R}_j, \mathbf{t}_j)}_{\substack{\textit{predicted} \\ \text{image location}}} - \underbrace{\begin{bmatrix} u_{i,j} \\ v_{i,j} \end{bmatrix}}_{\substack{\textit{observed} \\ \text{image location}}} \right\|^2$$

Scene reconstruction



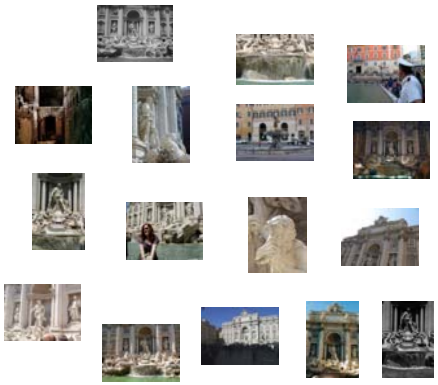
Feature detection

Detect features using SIFT [Lowe, IJCV 2004]



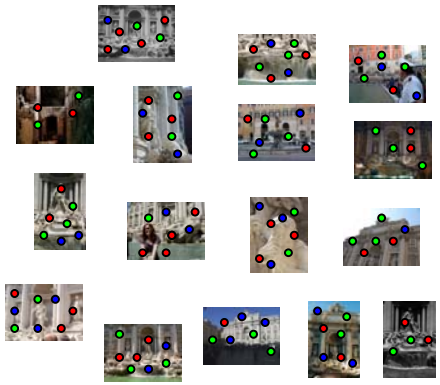
Feature detection

Detect features using SIFT [Lowe, IJCV 2004]



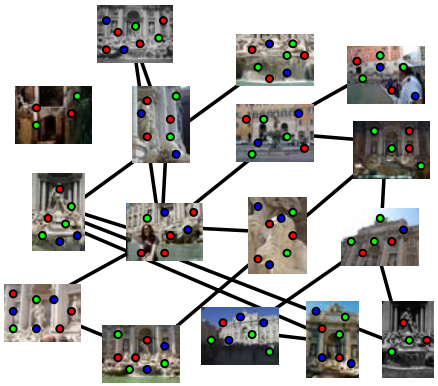
Feature detection

- Detect features using SIFT [Lowe, IJCV 2004]



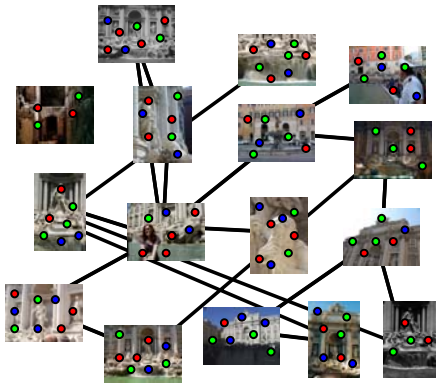
Feature matching

- Match features between each pair of images



Feature matching

Refine matching using RANSAC [Fischler & Bolles 1987] to estimate fundamental matrices between pairs



Reconstruction

1. Choose two/three views to seed the reconstruction.
2. Add 3d points via triangulation.
3. Add cameras using pose estimation.
4. Bundle adjustment
5. Goto step 2.

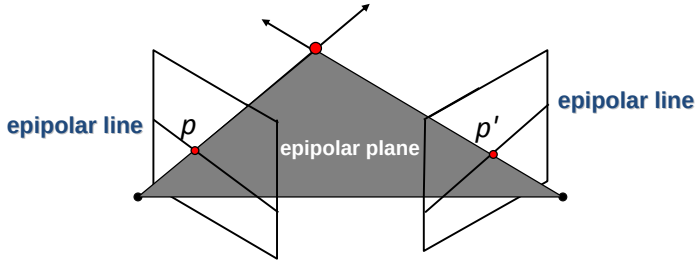
Two-view structure from motion

- Simpler case: can consider *motion* independent of structure

$$p = \begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \underbrace{\begin{bmatrix} -fs_x & 0 & x'_c \\ 0 & -fs_y & y'_c \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \left(\mathbf{R}_{3\times 3}\mathbf{x} + \mathbf{t}_{3\times 1}\right)$$

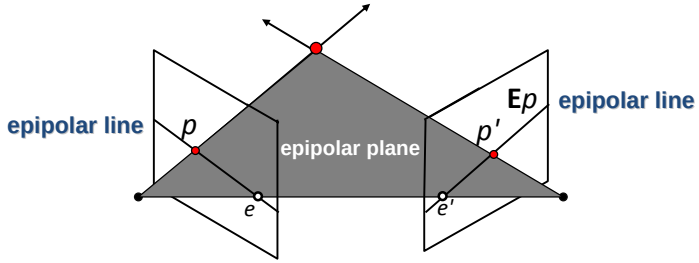
- Let's first consider the case where **K** is known
  - Each image point ($u_{i,j}$, $v_{i,j}$, 1) can be multiplied by $\mathbf{K}^{-1}$ to form a 3D ray
  - We call this the *calibrated* case

Notes on two-view geometry



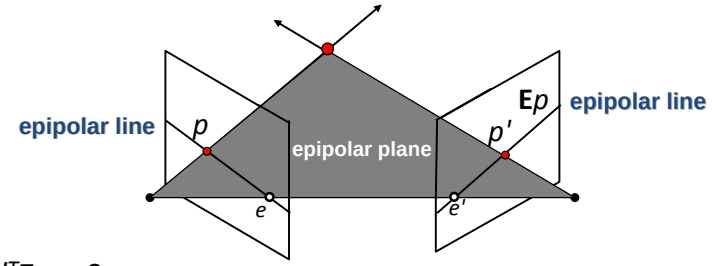
- How can we express the epipolar constraint?
- Answer: there is a 3x3 matrix **E** such that
- $p'^T \mathbf{E} p = 0$
- **E** is called the essential matrix

Properties of the essential matrix



$$E = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} R$$

Properties of the essential matrix



- $p'^T \mathbf{E} p = 0$
- $\mathbf{E}p$ is the epipolar line associated with p
- e and e' are called *epipoles*: $\mathbf{E}e = \mathbf{0}$ and $\mathbf{E}^T e' = \mathbf{0}$
- **E** can be solved for with 5 point matches
  - see Nister, **An efficient solution to the five-point relative pose problem**. *PAMI* 2004.

The Fundamental matrix

- If **K** is not known, then we use a related matrix called the *Fundamental matrix*, **F**
  - Called the *uncalibrated* case

$$\begin{aligned} p_2^T E p_1 &= 0 \\ p_1^\top K^\top K^{-\top} E K_1 - \top K^\top p_2 &= 0 \\ \hat{p}_1^\top F \hat{p}_2 &= 0 \end{aligned}$$

- **F** can be solved for linearly with eight points, or non-linearly with six or seven points

Photo Tourism overview

Input photographs

Scene reconstruction

Relative camera positions and orientations

Point cloud

Sparse correspondence

Photo Explorer