Global Alignment and Structure from Motion

Computer Vision

CSE576, Spring 2005

Richard Szeliski

Today's lecture

Rotational alignment ("3D stitching") [Project 3]

- pairwise alignment (Procrustes)
- global alignment (linearized least squares)

Calibration

- camera matrix (Direct Linear Transform)
  - non-linear least squares
- separating intrinsics and extrinsics
- focal length and optic center

Today's lecture

Structure from Motion

- triangulation and pose
- two-frame methods
- factorization
- bundle adjustment
- robust statistics

Global rotational alignment

Fully Automated Panoramic Stitching

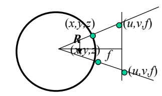
[Project 3]

AutoStitch [Brown & Lowe'03]

Stitch panoramic image from an *arbitrary* collection of photographs (known focal length)

1. Extract and (pairwise) match features
2. Estimate pairwise rotations using RANSAC
3. Add to stitch and re-run *global alignment*
4. Warp images to sphere and blend

3D Rotation Model



Projection equations

1. Project from image to 3D ray
   $(x_0,y_0,z_0) = (u_0-u_c,v_0-v_c,f)$
2. Rotate the ray by camera motion
   $(x_1,y_1,z_1) = \boldsymbol{R}_{01}\,(x_0,y_0,z_0)$
3. Project back into new (source) image
   $(u_1,v_1) = (fx_1/z_1+u_c,fy_1/z_1+v_c)$

Pairwise alignment

Absolute orientation [Arun *et al.*, PAMI 1987] [Horn *et al.*, JOSA A 1988], Procrustes Algorithm [Golub & VanLoan]

Given two sets of matching points, compute R

$p_i' = \boldsymbol{R}\,p_i$ with 3D rays

$p_i = N(x_i,y_i,z_i) = N(u_i-u_c,v_i-v_c,f)$

$\boldsymbol{A} = \Sigma_i\, p_i\, p_i'^T = \Sigma_i\, p_i\, p_i^T \boldsymbol{R}^T = \boldsymbol{U}\,\boldsymbol{S}\,\boldsymbol{V}^T = (\boldsymbol{U}\,\boldsymbol{S}\,\boldsymbol{U}^T)\,\boldsymbol{R}^T$

$\boldsymbol{V}^T = \boldsymbol{U}^T \boldsymbol{R}^T$

$\boldsymbol{R} = \boldsymbol{V}\,\boldsymbol{U}^T$

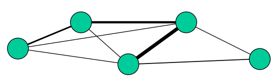
Pairwise alignment

RANSAC loop:

1. Select two feature pairs (at random)
   $p_i = N(u_i-u_c,v_i-v_c,f),\ p_i' = N(u_i'-u_c,v_i'-v_c,f),\ i=0,1$
2. Compute outer product matrix $\boldsymbol{A} = \Sigma_i\, p_i\, p_i'^T$
3. Compute R using SVD, $\boldsymbol{A} = \boldsymbol{U}\,\boldsymbol{S}\,\boldsymbol{V}^T$, $\boldsymbol{R} = \boldsymbol{V}\,\boldsymbol{U}^T$
4. Compute *inliers* where $f\,|p_i' - \boldsymbol{R}\,p_i| < \varepsilon$
5. Keep largest set of inliers
6. Re-compute least-squares SVD estimate on all of the inliers, $i=0..n$

Automatic stitching

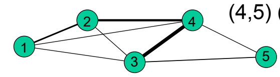
1. Match *all* pairs and keep the good ones (# inliers > threshold)
2. Sort pairs by *strength* (# inliers)
3. Add in next strongest match (and other relevant matches) to current stitch
4. Perform *global alignment*



Incremental selection & addition

	15	9	9	
11		15	18	
10	12		27	11
8	16	25		10
		12	8	

[3]
[4] (3,4) (4,3)
[2] (2,4) (4,2)
(2,3) (3,2)
[1] (1,2) (2,1)
(1,4) (4,1) (1,3) (3,1)
[5] (5,3) (3,5)
(4,5) (5,4)



Global alignment

Task: Compute globally consistent set of rotations $\{\boldsymbol{R}_i\}$ such that

$$\boldsymbol{R}_j \boldsymbol{p}_{ij} \approx \boldsymbol{R}_k \boldsymbol{p}_{ik} \quad \text{or} \quad \min |\boldsymbol{R}_j \boldsymbol{p}_{ij} - \boldsymbol{R}_k \boldsymbol{p}_{ik}|^2$$

1. Initialize "*first*" frame $\boldsymbol{R}_i = \boldsymbol{I}$
2. Multiply "*next*" frame by pairwise rotation $\boldsymbol{R}_{ij}$
3. Globally update all of the current $\{\boldsymbol{R}_i\}$

Q: How to parameterize and update the $\{\boldsymbol{R}_i\}$?

Parameterizing rotations

How do we parameterize **R** and Δ**R**?

- Euler angles: bad idea
- quaternions: 4-vectors on unit sphere
- use incremental rotation **R**(**I** + Δ**R**)

$$\Delta \mathrm{R} = [\omega]_\times = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$$

- update with Rodriguez formula

$$\mathrm{R}(\hat{\mathrm{n}}, \theta) = \mathrm{I} + \sin\theta\,[\hat{\mathrm{n}}]_\times + (1 - \cos\theta)\,[\hat{\mathrm{n}}]_\times^2, \quad \boldsymbol{\omega} = \theta \hat{\mathrm{n}}$$

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

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

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

Iterative focal length adjustment

(Optional) [Szeliski & Shum’97; MSR-TR-03]

Simplest approach:

$\arg\min_f f\ |\boldsymbol{R}_j \boldsymbol{p}_{ij} - \boldsymbol{R}_k \boldsymbol{p}_{ik}|^2$

More complex approach:

full bundle adjustment (op. cit. & later in talk)

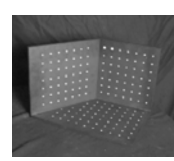
Camera Calibration

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?

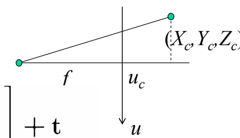


Camera calibration – approaches

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



$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} \mathbf{R} \end{bmatrix}_{3\times3} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \mathbf{t}$$

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

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?

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

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

- Least squares (pseudo-inverse)
- Is this good enough?

Optimal estimation

Feature measurement equations

$$u_i = f(\mathbf{M}, \mathbf{x}_i) + n_i = \hat{u}_i + n_i, \quad n_i \sim N(0, \sigma)$$

$$v_i = g(\mathbf{M}, \mathbf{x}_i) + m_i = \hat{v}_i + m_i, \quad m_i \sim N(0, \sigma)$$

Likelihood of ***M*** given {(u_i,v_i)}

$$L = \prod_i p(u_i|\hat{u}_i)p(v_i|\hat{v}_i) = \prod_i e^{-(u_i-\hat{u}_i)^2/\sigma^2} e^{-(v_i-\hat{v}_i)^2/\sigma^2}$$

Optimal estimation

Log likelihood of ***M*** given {(u_i,v_i)}

$$C = -\log L = \sum_i (u_i - \hat{u}_i)^2/\sigma_i^2 + (v_i - \hat{v}_i)^2/\sigma_i^2$$

How do we minimize *C*?

Non-linear regression (least squares), because $\hat{u}_i$ and v_i are non-linear functions of ***M***

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 function in **Δm**

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

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 **Δm** (my favorite)
- Use line search
- Other ideas?

Uncertainty analysis: covariance $\Sigma = A^{-1}$

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
- more unknowns than true degrees of freedom
- need a separate camera matrix for each new view

Separate intrinsics / extrinsics

New feature measurement equations

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

Use non-linear minimization

Standard technique in photogrammetry, computer vision, computer graphics

- [Tsai 87] – also estimates κ_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:

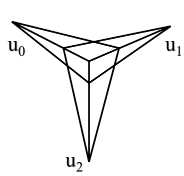
- can solve for more than one camera pose at a time
- potentially fewer degrees of freedom

Disadvantages:

- more complex update rules
- need a good initialization (recover **K** [**R** | **t**] from **M**)

Vanishing Points

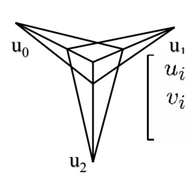
Determine focal length *f* and optical center (u_c,v_c) from image of cube's (or building's) *vanishing points*
[Caprile '90][Antone & Teller '00]



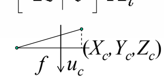
Vanishing Points

X, *Y*, and *Z* directions, $\mathbf{X}_i$ = (1,0,0,0) … (0,0,1,0) correspond to vanishing points that are scaled version of the rotation matrix:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & u_c \\ 0 & f & v_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} | \mathbf{t} \end{bmatrix} \mathbf{X}$$



$$\begin{bmatrix} u_i - u_c \\ v_i - v_c \\ f \end{bmatrix} \sim \begin{bmatrix} \mathbf{R} & | & \mathbf{t} \end{bmatrix} \mathbf{X}_i = \mathbf{r}_i,$$



Vanishing Points

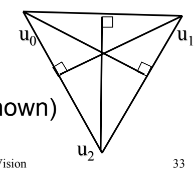
Orthogonality conditions on rotation matrix ***R***,

$\boldsymbol{r}_i \cdot \boldsymbol{r}_j = \delta_{ij}$

$$(u_i - u_c, v_i - v_c, f) \cdot (u_j - u_c, v_j - v_c, f) = 0,\ i \neq j$$

Determine (u_c, v_c) from *orthocenter* of vanishing point triangle

Then, determine f^2 from two equations
(only need 2 v.p.s if (u_c, v_c) known)



Vanishing point calibration

Advantages:

- only need to see vanishing points
  (e.g., architecture, table, …)

Disadvantages:

- not that accurate
- need rectahedral object(s) in scene

Single View Metrology

A. Criminisi, I. Reid and A. Zisserman (ICCV 99)

Make scene measurements from a single image

- Application: 3D from a single image

Assumptions

1. 3 orthogonal sets of parallel lines
2. 4 known points on ground plane
3. 1 height in the scene

Can still get an *affine reconstruction* without 2 and 3

Criminisi et al., ICCV 99

Complete approach

- Load in an image
- Click on parallel lines defining X, Y, and Z directions
- Compute vanishing points
- Specify points on reference plane, ref. height
- Compute 3D positions of several points
- Create a 3D model from these points
- Extract texture maps
- Output a VRML model

3D Modeling from a Photograph

3D Modeling from a Photograph

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 $\boldsymbol{K}^{-T}\boldsymbol{K}^{-1}$ from $\boldsymbol{H}_k$'s
  – 1plane if only f unknown
  – 2 planes if (f,u_c,v_c) unknown
  – 3+ planes for full $\boldsymbol{K}$

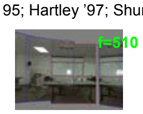
- Code available from Zhang and OpenCV

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

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

[Tsai87] [Bogart91]

Application: superimpose 3D graphics onto video

How do we initialize (***R***,***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]

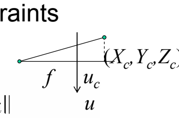
Pose estimation

Solve orthographic problem, iterate

[DeMenthon 95]

Use inter-point distance constraints

[Quan 99][Ameller 00]



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

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

Solve set of polynomial equations in x_i^{2p}

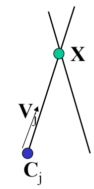
Triangulation

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 ($\mathbf{K}$, $\mathbf{R}_j$, $\mathbf{t}_j$)

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

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

Structure from motion

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

How many points do we need to match?

- 2 frames:
  ($\boldsymbol{R},\boldsymbol{t}$): 5 dof + $3n$ point locations $\leq$
  $4n$ point measurements $\Rightarrow$
  $n \geq 5$
- k frames:
  $6(k-1)-1 + 3n \leq 2kn$
- always want to use many more

Two-frame methods

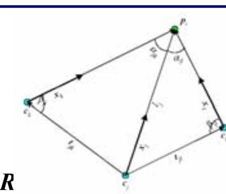
Two main variants:

1. Calibrated: "Essential matrix" $\boldsymbol{E}$
   use ray directions ($\boldsymbol{x}_i$, $\boldsymbol{x}_i'$)
2. Uncalibrated: "Fundamental matrix" $\boldsymbol{F}$

[Hartley & Zisserman 2000]

Essential matrix

Co-planarity constraint:



$$\boldsymbol{x}' \approx \boldsymbol{R}\,\boldsymbol{x} + \boldsymbol{t}$$
$$[\boldsymbol{t}]_\times \boldsymbol{x}' \approx [\boldsymbol{t}]_\times \boldsymbol{R}\,\boldsymbol{x}$$
$$\boldsymbol{x}'^T [\boldsymbol{t}]_\times \boldsymbol{x}' \approx \boldsymbol{x}'^T [\boldsymbol{t}]_\times \boldsymbol{R}\,\boldsymbol{x}$$
$$\boldsymbol{x}'^T \boldsymbol{E}\,\boldsymbol{x} = 0 \text{ with } \boldsymbol{E} = [\boldsymbol{t}]_\times \boldsymbol{R}$$

- Solve for $\boldsymbol{E}$ using least squares (SVD)
- $\boldsymbol{t}$ is the least singular vector of $\boldsymbol{E}$
- $\boldsymbol{R}$ obtained from the other two s.v.s

Fundamental matrix

Camera calibrations are unknown

$x' F x = 0$ with $F = [e]_\times H = K'[t]_\times R K^{-1}$

- Solve for F using least squares (SVD)
  - re-scale (x_i, x_i') so that $|x_i| \approx 1/2$ [Hartley]
- e (epipole) is *still* the least singular vector of F
- H obtained from the other two s.v.s
- "plane + parallax" (projective) reconstruction
- use self-calibration to determine K [Pollefeys]

Three-frame methods

Trifocal tensor

[Hartley & Zisserman 2000]

Multi-frame Structure from Motion

Factorization

[Tomasi & Kanade, IJCV 92]

Structure [from] Motion

Given a set of feature tracks,
estimate the 3D structure and 3D (camera) motion.

Assumption: orthographic projection

Tracks: (u_{fp}, v_{fp}), f: frame, p: point

Subtract out mean 2D position…

$u_{fp} = \boldsymbol{i}_f^T \boldsymbol{s}_p$ $\qquad \boldsymbol{i}_f$: rotation, $\boldsymbol{s}_p$: position

$v_{fp} = \boldsymbol{j}_f^T \boldsymbol{s}_p$

Measurement equations

Measurement equations

$u_{fp} = \boldsymbol{i}_f^T \boldsymbol{s}_p$ $\qquad \boldsymbol{i}_f$: rotation, $\boldsymbol{s}_p$: position

$v_{fp} = \boldsymbol{j}_f^T \boldsymbol{s}_p$

Stack them up…

$\boldsymbol{W} = \boldsymbol{R}\,\boldsymbol{S}$

$\boldsymbol{R} = (\boldsymbol{i}_1, \ldots, \boldsymbol{i}_F, \boldsymbol{j}_1, \ldots, \boldsymbol{j}_F)^T$

$\boldsymbol{S} = (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_P)$

Factorization

$\boldsymbol{W} = \boldsymbol{R}_{2F\times 3}\,\boldsymbol{S}_{3\times P}$

SVD

$\boldsymbol{W} = \boldsymbol{U}\,\boldsymbol{\Lambda}\,\boldsymbol{V}$ $\qquad$ **Λ must be rank 3**

$\boldsymbol{W'} = (\boldsymbol{U}\,\boldsymbol{\Lambda}^{1/2})(\boldsymbol{\Lambda}^{1/2}\,\boldsymbol{V}) = \boldsymbol{U'}\,\boldsymbol{V'}$

Make $\boldsymbol{R}$ orthogonal

$\boldsymbol{R} = \boldsymbol{Q}\boldsymbol{U'}$, $\boldsymbol{S} = \boldsymbol{Q}^{-1}\boldsymbol{V'}$

$\boldsymbol{i}_f^T \boldsymbol{Q}^T \boldsymbol{Q} \boldsymbol{i}_f = 1 \ldots$

Results

Look at paper figures…

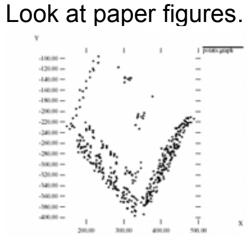


Figure 4.5: A view of the computed shape from approximately above the building (compare with figure 4.6).

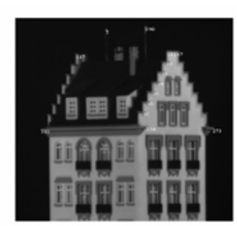


Figure 4.7: For a quantitative evaluation, distances between the features shown in the picture were measured on the actual model, and compared with the computed results. The comparison is shown in figure 4.8.

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

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

What makes this non-linear minimization hard?

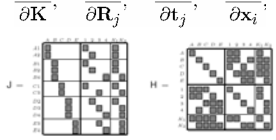
- many more parameters: potentially slow
- poorer conditioning (high correlation)
- potentially lots of outliers
- gauge (coordinate) freedom

Lots of parameters: sparsity

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

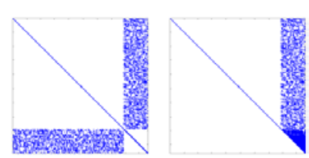
Only a few entries in Jacobian are non-zero

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



Sparse Cholesky (skyline)

First used in finite element analysis
Applied to SfM by [Szeliski & Kang 1994]



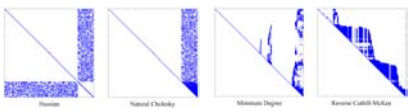
Hessian Natural Cholesky

structure | motion fill-in

Conditioning and gauge freedom

Poor conditioning:

- use 2nd order method
- use Cholesky decomposition



Gauge freedom

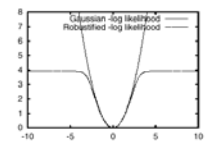
- fix certain parameters (orientation) *or*
- zero out last few rows in Cholesky decomposition

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]

Correspondences

Can refine feature matching <u>*after*</u> a structure and motion estimate has been produced

- decide which ones obey the *epipolar geometry*
- decide which ones are *geometrically consistent*
- (optional) iterate between correspondences and SfM estimates using MCMC [Dellaert *et al.*, Machine Learning 2003]

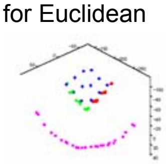
Structure from motion: limitations

Very difficult to reliably estimate <u>*metric*</u> structure and motion unless:

- large (*x* or *y*) rotation *or*
- large field of view and depth variation

Camera calibration important for Euclidean reconstructions

Need good feature tracker



Bibliography

M.-A. Ameller, B. Triggs, and L. Quan.
Camera pose revisited -- new linear algorithms.
http://www.inrialpes.fr/movi/people/Triggs/home.html, 2000.

M. Antone and S. Teller.
Recovering relative camera rotations in urban scenes.
In IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'2000), volume 2, pages 282--289, Hilton Head Island, June 2000.

S. Becker and V. M. Bove.
Semiautomatic {3-D model extraction from uncalibrated 2-d camera views.
In SPIE Vol. 2410, Visual Data Exploration and Analysis {II, pages 447--461, San Jose, CA, February 1995. Society of Photo-Optical Instrumentation Engineers.

R. G. Bogart.
View correlation.
In J. Arvo, editor, Graphics Gems II, pages 181--190. Academic Press, Boston, 1991.

Bibliography

D. C. Brown.
Close-range camera calibration.
Photogrammetric Engineering, 37(8):855--866, 1971.

B. Caprile and V. Torre.
Using vanishing points for camera calibration.
International Journal of Computer Vision, 4(2):127--139, March 1990.

R. T. Collins and R. S. Weiss.
Vanishing point calculation as a statistical inference on the unit sphere.
In Third International Conference on Computer Vision (ICCV'90), pages 400--403, Osaka, Japan, December 1990. IEEE Computer Society Press.

A. Criminisi, I. Reid, and A. Zisserman.
Single view metrology.
In Seventh International Conference on Computer Vision (ICCV'99), pages 434--441, Kerkyra, Greece, September 1999.

Bibliography

L. {de Agapito, R. I. Hartley, and E. Hayman.
Linear calibration of a rotating and zooming camera.
In IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99), volume 1, pages 15--21, Fort Collins, June 1999.

D. I. DeMenthon and L. S. Davis.
Model-based object pose in 25 lines of code.
International Journal of Computer Vision, 15:123--141, June 1995.

M. Gleicher and A. Witkin.
Through-the-lens camera control.
Computer Graphics (SIGGRAPH'92), 26(2):331--340, July 1992.

R. I. Hartley.
An algorithm for self calibration from several views.
In IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'94), pages 908--912, Seattle, Washington, June 1994. IEEE Computer Society.

Bibliography

R. I. Hartley.
Self-calibration of stationary cameras.
International Journal of Computer Vision, 22(1):5--23, 1997.

R. I. Hartley, E. Hayman, L. {de Agapito, and I. Reid.
Camera calibration and the search for infinity.
In IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'2000), volume 1, pages 510--517, Hilton Head Island, June 2000.

R. I. Hartley. and A. Zisserman.
Multiple View Geometry.
Cambridge University Press, 2000.

B. K. P. Horn.
Closed-form solution of absolute orientation using unit quaternions.
Journal of the Optical Society of America A, 4(4):629--642, 1987.

Bibliography

S. B. Kang and R. Weiss.
Characterization of errors in compositing panoramic images.
Computer Vision and Image Understanding, 73(2):269--280, February 1999.

M. Pollefeys, R. Koch and L. Van Gool.
Self-Calibration and Metric Reconstruction in spite of Varying and Unknown Internal Camera Parameters.
International Journal of Computer Vision, 32(1), 7-25, 1999. [pdf]

L. Quan and Z. Lan.
Linear N-point camera pose determination.
IEEE Transactions on Pattern Analysis and Machine Intelligence, 21(8):774--780, August 1999.

G. Stein.
Accurate internal camera calibration using rotation, with analysis of sources of error.
In Fifth International Conference on Computer Vision (ICCV'95), pages 230--236, Cambridge, Massachusetts, June 1995.

Bibliography

Stewart, C. V. (1999). Robust parameter estimation in computer vision. SIAM Reviews, 41(3), 513–537.

R. Szeliski and S. B. Kang.
Recovering 3D Shape and Motion from Image Streams using Nonlinear Least Squares
Journal of Visual Communication and Image Representation, 5(1):10-28, March 1994.

R. Y. Tsai.
A versatile camera calibration technique for high-accuracy {3D machine vision metrology using off-the-shelf {TV cameras and lenses.
IEEE Journal of Robotics and Automation, RA-3(4):323--344, August 1987.

Z. Zhang.
Flexible camera calibration by viewing a plane from unknown orientations.
In Seventh International Conference on Computer Vision (ICCV'99), pages 666--687, Kerkyra, Greece, September 1999.