Quiz & Multiple View Geometry

METR 4202: **Robotics** & Automation

Dr Surya Singh -- Lecture # 9

September 20, 2017

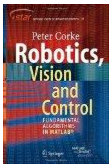
metr4202-staff@itee.uq.edu.au

http://robotics.itee.uq.edu.au/~metr4202/ [http://**metr4202.com**]

Lecture Schedule

Week	Date	Lecture (W: 3:05p-4:50, 7-222)
1	26-Jul	Introduction + Representing Position & Orientation & State
2	2-Aug	Robot Forward Kinematics (Frames, Transformation Matrices & Affine Transformations)
3	9-Aug	Robot Inverse Kinematics & Dynamics (Jacobians)
4	16-Aug	*Ekka Day* (Robot Kinematics & Kinetics Review)
5	23-Aug	Jacobians & Robot Sensing Overview
6	30-Aug	Robot Sensing: Single View Geometry & Lines
7	6-Sep	Robot Sensing: Basic Feature Detection
8	13-Sep	Robot Sensing: Scalable Feature Detection
9	20-Sep	***Mid-Semester Exam*** **& Multiple View Geometry**
	27-Sep	*Study break*
10	4-Oct	Motion Planning
11	11-Oct	Probabilistic Robotics: Localization & SLAM
12	18-Oct	Probabilistic Robotics: Planning & Control (State-Space/Shaping the Dynamic Response/LQR)
13	25-Oct	The Future of Robotics/Automation + Challenges + Course Review

Follow Along Reading:



Robotics, Vision & Control
by Peter Corke

Also online:SpringerLink

UQ Library eBook:
364220144X

Today

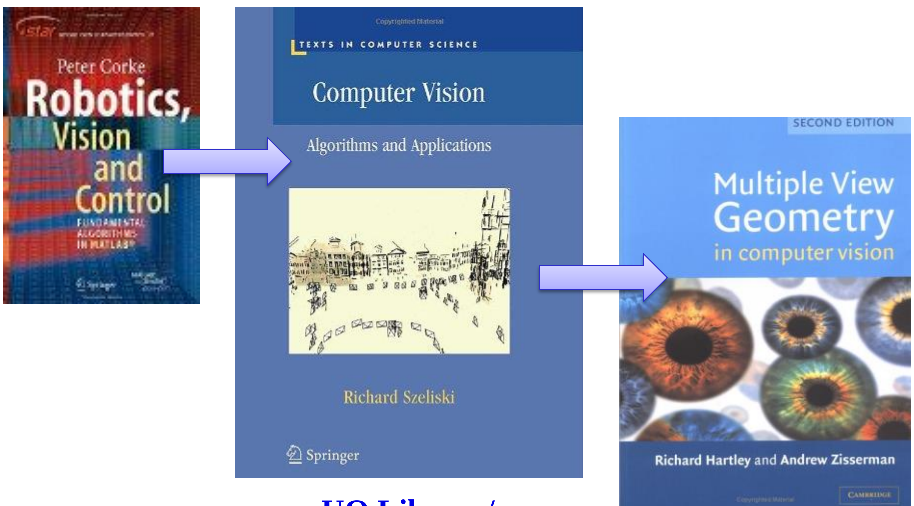
➔ Multiple View Geometry ←

- A simple little quiz ☺
- RVC
  - §14.1-14.4: Multiple Images

- Planning
  - pp. 91-103
    (Yup! That's all Peter Corke has to say on that – which explains why there is no planning at ACRV ☺).

Next Time

Reference Material

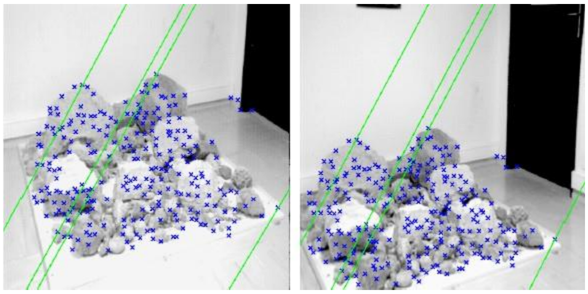


UQ Library/ SpringerLink

UQ Library (ePDF)

Multiple Cameras: **Stereo** (Feature-based)

- Match "corner" (interest) points



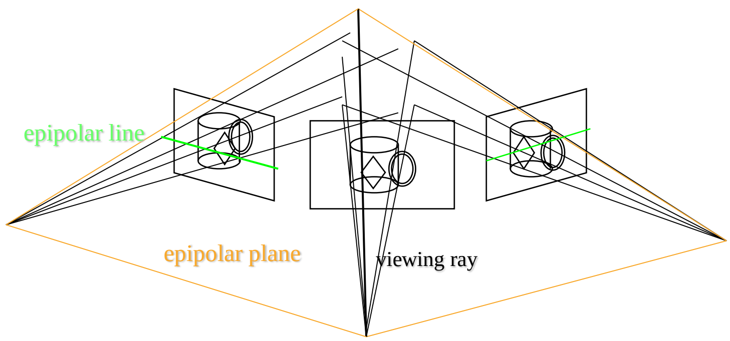
- Interpolate complete solution

Slide from Szeliski, *Computer Vision: Algorithms and Applications*

"Fundamental" Multi-View Geometry

Stereo Geometry → Epipolar Geometry

- Match features along epipolar lines



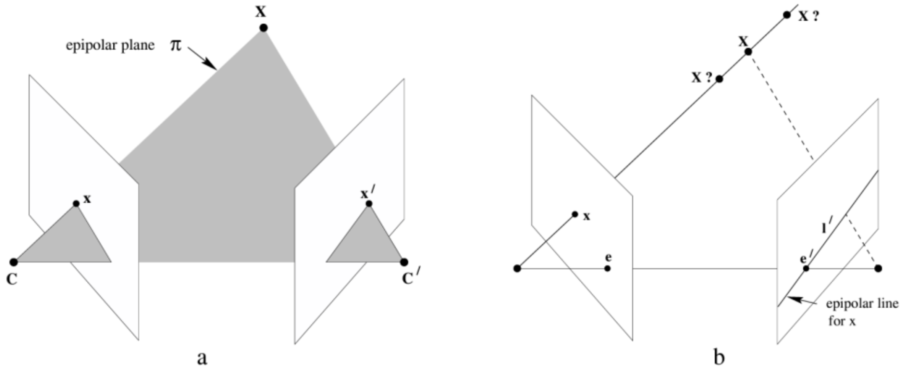
Slide from Szeliski, *Computer Vision: Algorithms and Applications*

Stereo Geometry → Epipolar geometry

- **Epipolar lines :=**
  are the projection of the pencil of planes passing through the centers
- For 2 images (or images with collinear camera centers):
  We can find epipolar lines that intersect and thus "simplify" the stereo **feature matching and correspondence problem**
- **Rectification :=**
  warping the input images (perspective transformation) so that epipolar lines are horizontal

Slide from Szeliski, *Computer Vision: Algorithms and Applications*

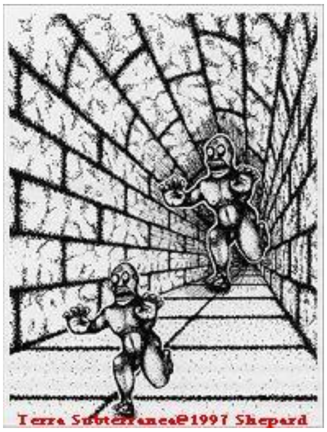
Two-View Geometry: Epipolar Plane

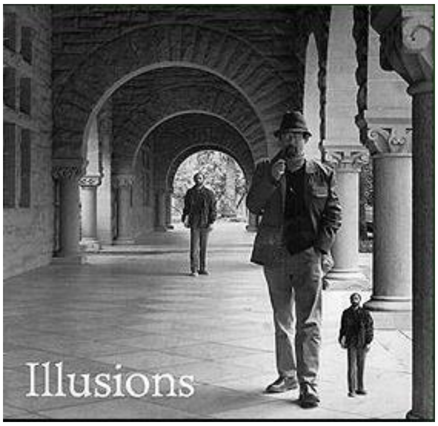


- **Epipole:** The *point* of intersection of the line joining the camera centres (the baseline) with the image plane. Equivalently, the epipole is the image in one view of the camera centre of the other view.
- **Epipolar plane** is a plane containing the baseline. There is a one-parameter family (a pencil) of epipolar planes
- **Epipolar line** is the intersection of an epipolar plane with the image plane. All epipolar lines intersect at the epipole. An epipolar plane intersects the left and right image planes in epipolar lines, and defines the correspondence between the lines.

Key Correlator: Vanishing Points

- Vanishing Points can be fun…

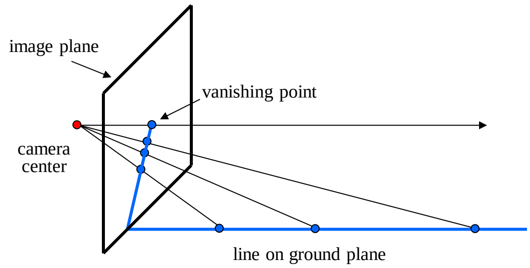




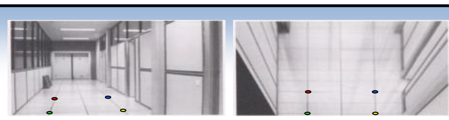
- **They also hold a key to correlating views!**

Slide from Szeliski, *Computer Vision: Algorithms and Applications*

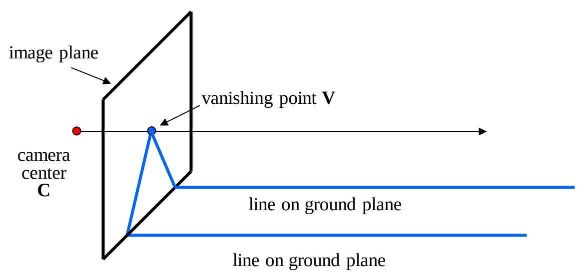
Vanishing Points (2D)



Vanishing Points

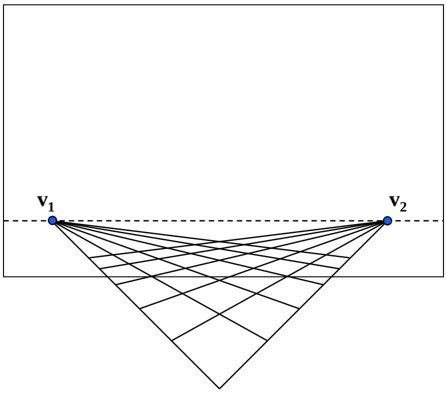


- Properties
  - Any two parallel lines have the same vanishing point
  - The ray from C through v point is parallel to the lines
  - An image may have more than one vanishing point



Vanishing Lines

- Multiple Vanishing Points
  - Any set of parallel lines on the plane define a vanishing point
  - The union of all of these vanishing points is the horizon line



Back to Two-frames

Two classes of two-frame main variants:

I. Calibrated: "Essential matrix" $\boldsymbol{E}$

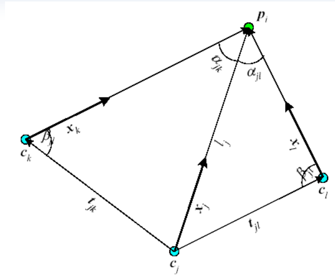
– Use ray directions $(\boldsymbol{x_i}, \boldsymbol{x_i}')$

II. Uncalibrated: "Fundamental matrix" $\boldsymbol{F}$

THE reference: [Hartley & Zisserman, Chapter 9]

I. Essential matrix

- Co-planarity constraint:
  - $x' \approx R\,x + t$
  - $[t]\times x' \approx [t]\times R\,x$
  - $x'\,[t]\times x' \approx x'\,[t]\times R\,x$
  - $x'\,E\,x = 0$ with $E = [t]\times R$



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

From Szeliski, *Computer Vision: Algorithms and Applications*

II. Fundamental Matrix

- The fundamental matrix is the algebraic representation of epipolar geometry.

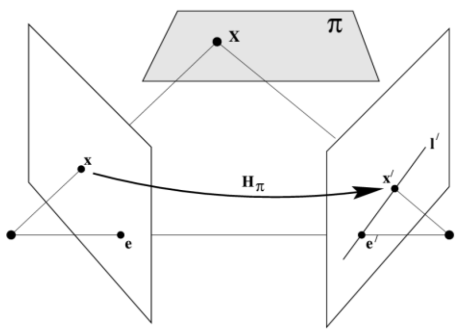


Fig. 9.5. *A point* $\mathbf{x}$ *in one image is transferred via the plane* π *to a matching point* $\mathbf{x}'$ *in the second image. The epipolar line through* $\mathbf{x}'$ *is obtained by joining* $\mathbf{x}'$ *to the epipole* $\mathbf{e}'$*. In symbols one may write* $\mathbf{x}' = H_\pi \mathbf{x}$ *and* $\mathbf{l}' = [\mathbf{e}']_\times \mathbf{x}' = [\mathbf{e}']_\times H_\pi \mathbf{x} = F\mathbf{x}$ *where* $F = [\mathbf{e}']_\times H_\pi$ *is the fundamental matrix.*

Reference: [Hartley & Zisserman, Chapter 9, § 9.2-9.4]

Fundamental matrix

- Camera calibrations are unknow

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

- Solve for **F** using least squares (SVD)
  - re-scale (x_i, x_i') so that $|x_i| \approx \frac{1}{2}$ [Hartley]
- $\boldsymbol{e}$ (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]

Reference: [Hartley & Zisserman, Chapter 9, p. 246

Fundamental Matrix Example

- Suppose the camera matrices are those of a calibrated stereo rig with the world origin at the first camera

$$\mathrm{P} = \mathrm{K}[\mathrm{I} \mid \mathbf{0}] \qquad \mathrm{P}' = \mathrm{K}'[\mathrm{R} \mid \mathbf{t}].$$

- Then:

$$\mathrm{P}^{+} = \begin{bmatrix} \mathrm{K}^{-1} \\ \mathbf{0}^{\mathsf{T}} \end{bmatrix} \qquad \mathbf{C} = \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix}$$

- Epipoles are at:

$$\mathbf{e} = \mathrm{P}\begin{pmatrix} -\mathrm{R}^{\mathsf{T}}\mathbf{t} \\ 1 \end{pmatrix} = \mathrm{K}\mathrm{R}^{\mathsf{T}}\mathbf{t} \qquad \mathbf{e}' = \mathrm{P}'\begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix} = \mathrm{K}'\mathbf{t}.$$

∴

$$\mathrm{F} = [\mathbf{e}']_{\times}\mathrm{K}'\mathrm{R}\mathrm{K}^{-1} = \mathrm{K}'^{-\mathsf{T}}[\mathbf{t}]_{\times}\mathrm{R}\mathrm{K}^{-1} = \mathrm{K}'^{-\mathsf{T}}\mathrm{R}[\mathrm{R}^{\mathsf{T}}\mathbf{t}]_{\times}\mathrm{K}^{-1} = \mathrm{K}'^{-\mathsf{T}}\mathrm{R}\mathrm{K}^{\mathsf{T}}[\mathbf{e}]_{\times}$$

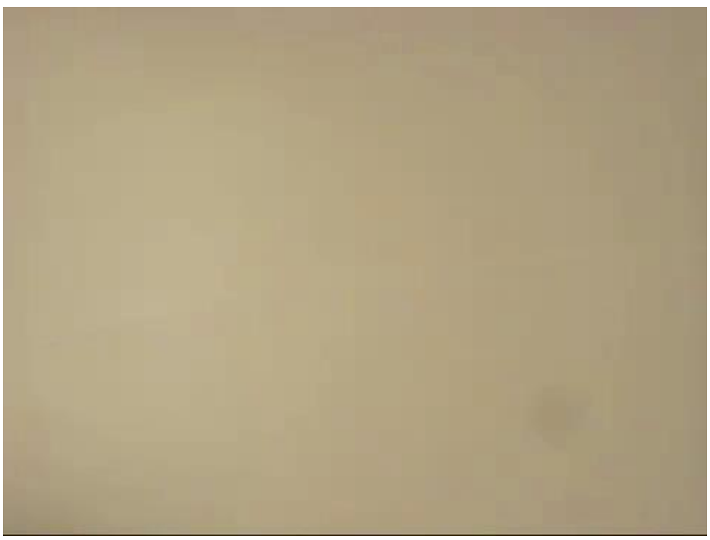
Reference: [Hartley & Zisserman, Chapter 9, § 9.2-9.4]

Summary of fundamental matrix properties

- F is a rank 2 homogeneous matrix with 7 degrees of freedom.
- **Point correspondence**: If $\mathbf{x}$ and $\mathbf{x}'$ are corresponding image points, then $\mathbf{x}'^{\mathsf{T}} \mathtt{F} \mathbf{x} = 0$.
- **Epipolar lines**:
  - ◇ $\mathbf{l}' = \mathtt{F}\mathbf{x}$ is the epipolar line corresponding to $\mathbf{x}$.
  - ◇ $\mathbf{l} = \mathtt{F}^{\mathsf{T}}\mathbf{x}'$ is the epipolar line corresponding to $\mathbf{x}'$.
- **Epipoles**:
  - ◇ $\mathtt{F}\mathbf{e} = \mathbf{0}$.
  - ◇ $\mathtt{F}^{\mathsf{T}}\mathbf{e}' = \mathbf{0}$.
- **Computation from camera matrices** $\mathtt{P}, \mathtt{P}'$:
  - ◇ General cameras,
    $\mathtt{F} = [\mathbf{e}']_\times \mathtt{P}'\mathtt{P}^+$, where $\mathtt{P}^+$ is the pseudo-inverse of $\mathtt{P}$, and $\mathbf{e}' = \mathtt{P}'\mathbf{C}$, with $\mathtt{P}\mathbf{C} = \mathbf{0}$.
  - ◇ Canonical cameras, $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$, $\mathtt{P}' = [\mathtt{M} \mid \mathbf{m}]$,
    $\mathtt{F} = [\mathbf{e}']_\times \mathtt{M} = \mathtt{M}^{-\mathsf{T}}[\mathbf{e}]_\times$, where $\mathbf{e}' = \mathbf{m}$ and $\mathbf{e} = \mathtt{M}^{-1}\mathbf{m}$.
  - ◇ Cameras not at infinity $\mathtt{P} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}]$, $\mathtt{P}' = \mathtt{K}'[\mathtt{R} \mid \mathbf{t}]$,
    $\mathtt{F} = \mathtt{K}'^{-\mathsf{T}}[\mathbf{t}]_\times \mathtt{R}\mathtt{K}^{-1} = [\mathtt{K}'\mathbf{t}]_\times \mathtt{K}'\mathtt{R}\mathtt{K}^{-1} = \mathtt{K}'^{-\mathsf{T}}\mathtt{R}\mathtt{K}^{\mathsf{T}}[\mathtt{K}\mathtt{R}^{\mathsf{T}}\mathbf{t}]_\times$.

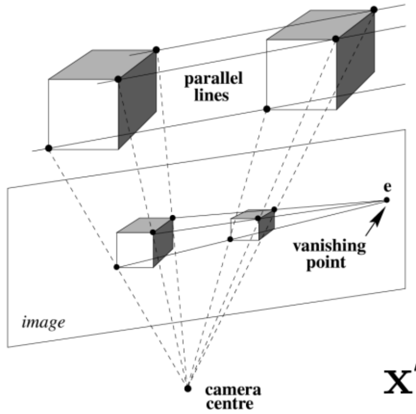
Reference: [Hartley & Zisserman, Chapter 9, p. 246

Cool Robotics Share: **Fundamental Matrix** Song



D. Wedge, *The Fundamental Matrix Song*

Fundamental Matrix & Motion



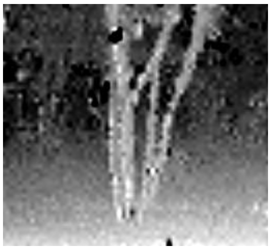
$$x'^{\mathsf{T}}Fx = 0.$$

- Under a pure translational camera motion, 3D points appear to slide along parallel rails. The images of these parallel lines intersect in a vanishing point corresponding to the translation direction. The epipole **e** is the vanishing point.

Correspondences

Finding correspondences

- Apply feature matching criterion (e.g., correlation or Lucas-Kanade) at all pixels simultaneously
- Search only over epipolar lines (many fewer candidate positions)



Slide from Szeliski, *Computer Vision: Algorithms and Applications*

Matching criteria

- Raw pixel values (correlation)
- Band-pass filtered images [Jones & Malik 92]
- “Corner” like features [Zhang, …]
- Edges [many people…]
- Gradients [Seitz 89; Scharstein 94]
- Rank statistics [Zabih & Woodfill 94]

Slide from Szeliski, *Computer Vision: Algorithms and Applications*

Stereo matching framework

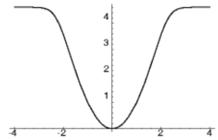
- For every disparity, compute raw matching costs

Why use a robust function?

  – occlusions, other outliers

$$E_0(x, y; d) = \rho(I_L(x' + d, y') - I_R(x', y'))$$

- Can also use alternative match criteria

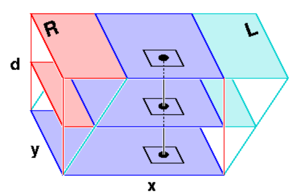


Stereo matching framework

- Aggregate costs spatially

$$E(x, y; d) = \sum_{(x', y') \in N(x, y)} E_0(x', y', d)$$

- Here, we are using a box filter (efficient moving average implementation)
- Can also use weighted average, [non-linear] diffusion…

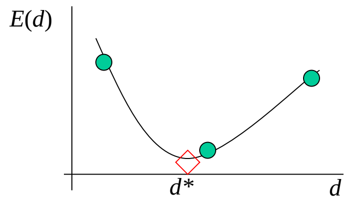


Stereo matching framework

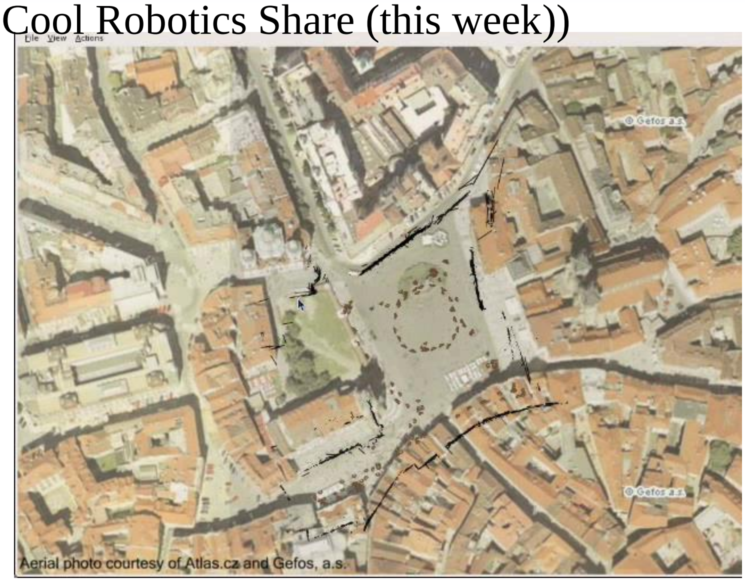
- Choose winning disparity at each pixel

$$d(x,y) = \arg\min_{d} E(x,y;d)$$

- Interpolate to sub-pixel accuracy



SFM: Structure from Motion
(& Cool Robotics Share (this week))



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…

  $\mathbf{i}_f$: rotation, $\mathbf{s}_p$: position

$$u_{fp} = i_f^T s_p, v_{fp} = j_f^T s_p$$

From Szeliski, *Computer Vision: Algorithms and Applications*

Structure from motion

- How many points do we need to match?
- 2 frames:
  - (R,t): 5 dof + 3n point locations ≤
  - 4n point measurements ⇒
  - n ≥ 5
- k frames:
  - 6(k–1)-1 + 3n ≤ 2kn
- always want to use many more

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

From Szeliski, *Computer Vision: Algorithms and Applications*

Measurement equations

- Measurement equations

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

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

- Stack them up…

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

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

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

From Szeliski, [Computer Vision: Algorithms and Applications]

Factorization

$\boldsymbol{W} = \boldsymbol{R}_{2F\times3}\,\boldsymbol{S}_{3\times P}$

SVD

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

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

Make $\boldsymbol{R}$ orthogonal

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

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

From Szeliski, [Computer Vision: Algorithms and Applications]

Results

- Look at paper figures…

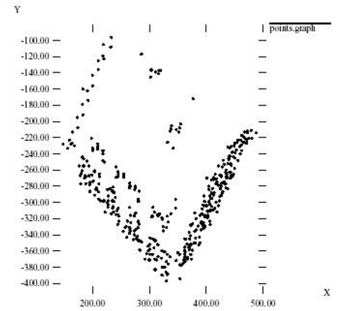


Figure 4.5: A view of the computed shape from approximately above the building (compare with figure 4.6).

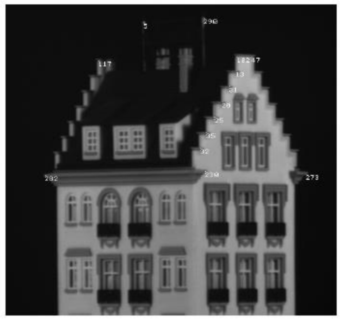


Figure 4.7: For a quantitative evaluation, distances between the features shown in the picture were measured on the actual model, and compared with the computed results. The comparison is shown in figure 4.8.

From Szeliski, *Computer Vision: Algorithms and Applications*

Bundle Adjustment

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

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

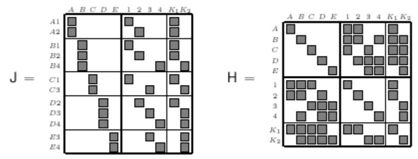
From Szeliski, *Computer Vision: Algorithms and Applications*

Lots of parameters: sparsity

- Only a few entries in Jacobian are non-zero

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

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

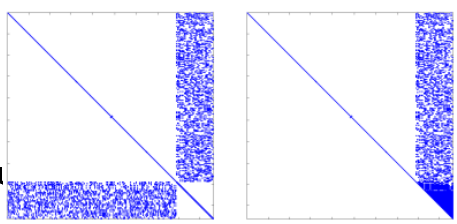


From Szeliski, *Computer Vision: Algorithms and Applications*

Sparse Cholesky (skyline)

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

structu

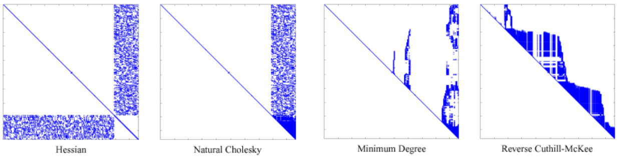


Hessian Natural Cholesky

From Szeliski, *Computer Vision: Algorithms and Applications*

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

From Szeliski, *Computer Vision: Algorithms and Applications*

Cool Robotics Share!: Photosynth & Bundler