Computer Vision
cmput 428/615

3D Modeling from images

Martin Jagersand

3D from video example
Capture objects

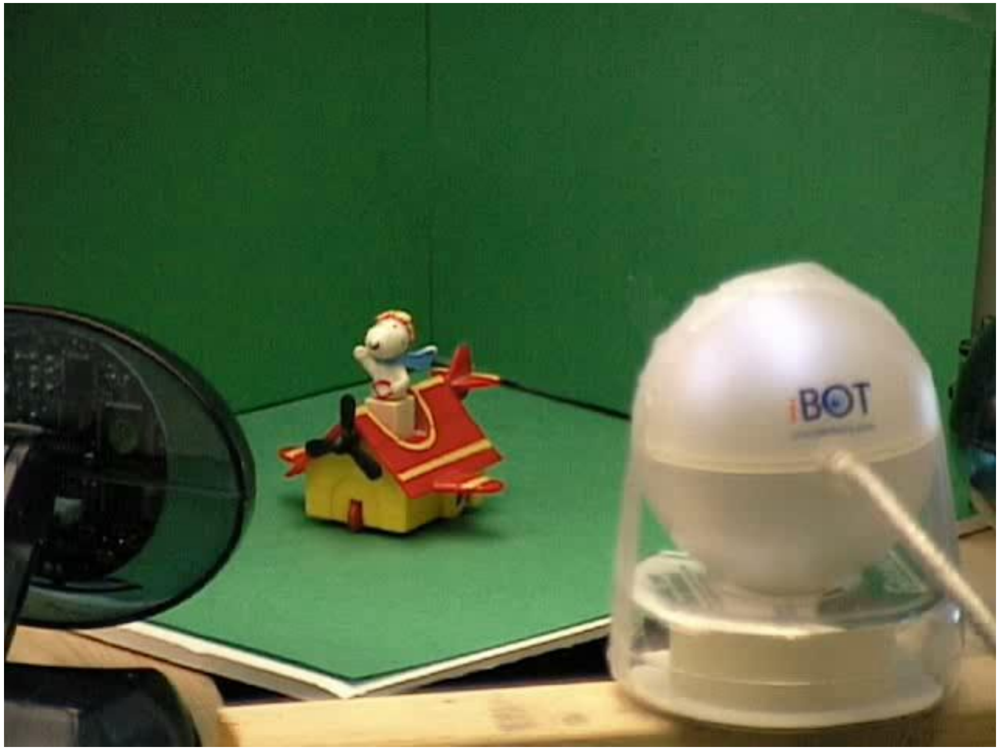
- Inexpensive

- Quick and convenient for the user

- Integrates with existing SW e.g. Blender, Maya

3D from video: Low budget

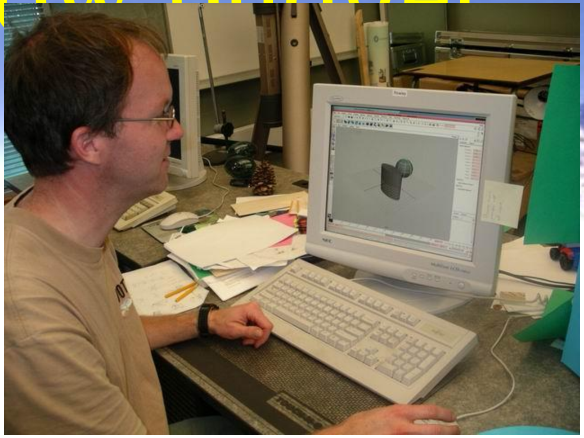
- Inexpensive



$100: Webcams, Digital Cams

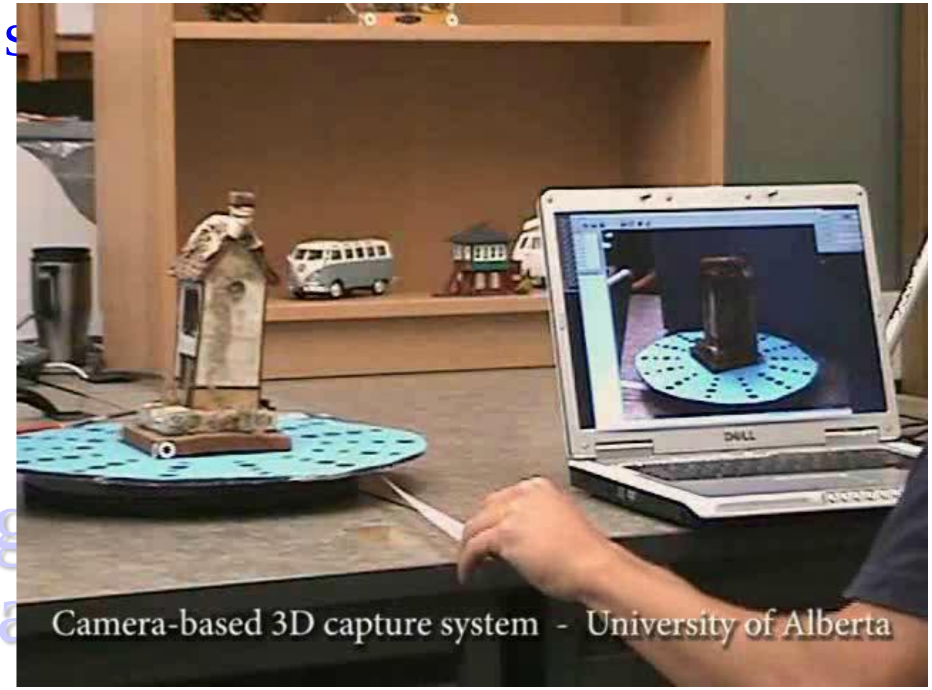
$100,000 Laser scanners etc.

3D from video Low budget



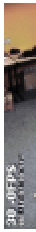
Modeling geom primitives into s

- Inexpensive
- Quick and convenient for the user
- Integrates with existing SW e.g. Blender, Maya



Capturing 3D from 2D video:

Low budget 3D from video



- Inexpensive
- Quick and convenient for the user
- Integrates with existing SW e.g. Blender, Maya

Application Case Study
Modeling Inuit Artifacts

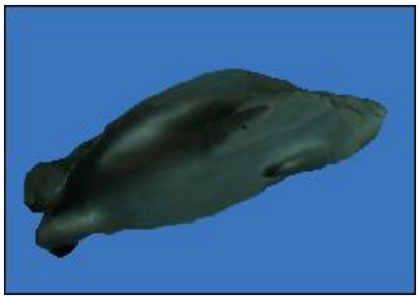
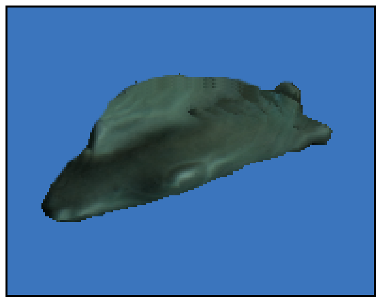
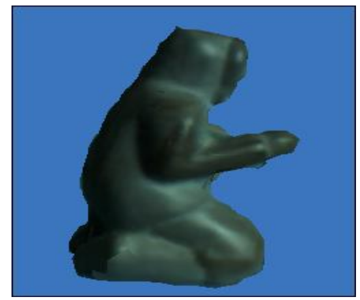
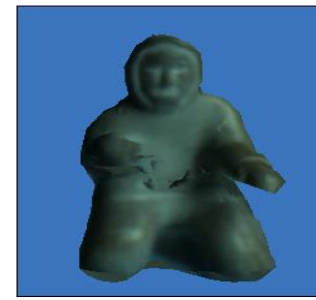
- New acquisition at the UofA: A group of 8 sculptures depicting Inuit seal hunt
- Acquired from sculptor by Hudson Bay Company



Application Case Study
Modeling Inuit Artifacts

Results:

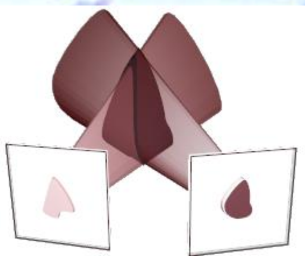
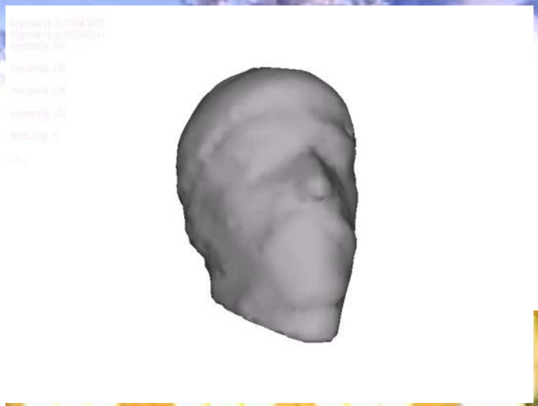
1. A collection of 3D models of each component

2. Assembly of the individual models into animations and Internet web study material.

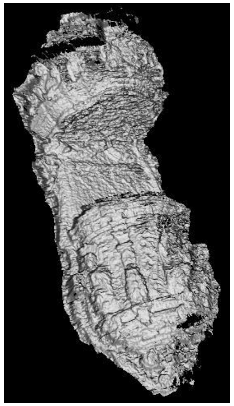



Preliminaries:
Capturing **Macro** geometry:

- Shape From Silhouette
  - Works for objects
  - Robust
  - Visual hull not true object surface

- Structure From Motion
  - Works for Scenes
  - Typically sparse
  - Sometimes fragile (no salient points in scene)
- Space carving
  - Use free space constraints
- (Dense “Stereo” -- later)
  - Use as second refinement step

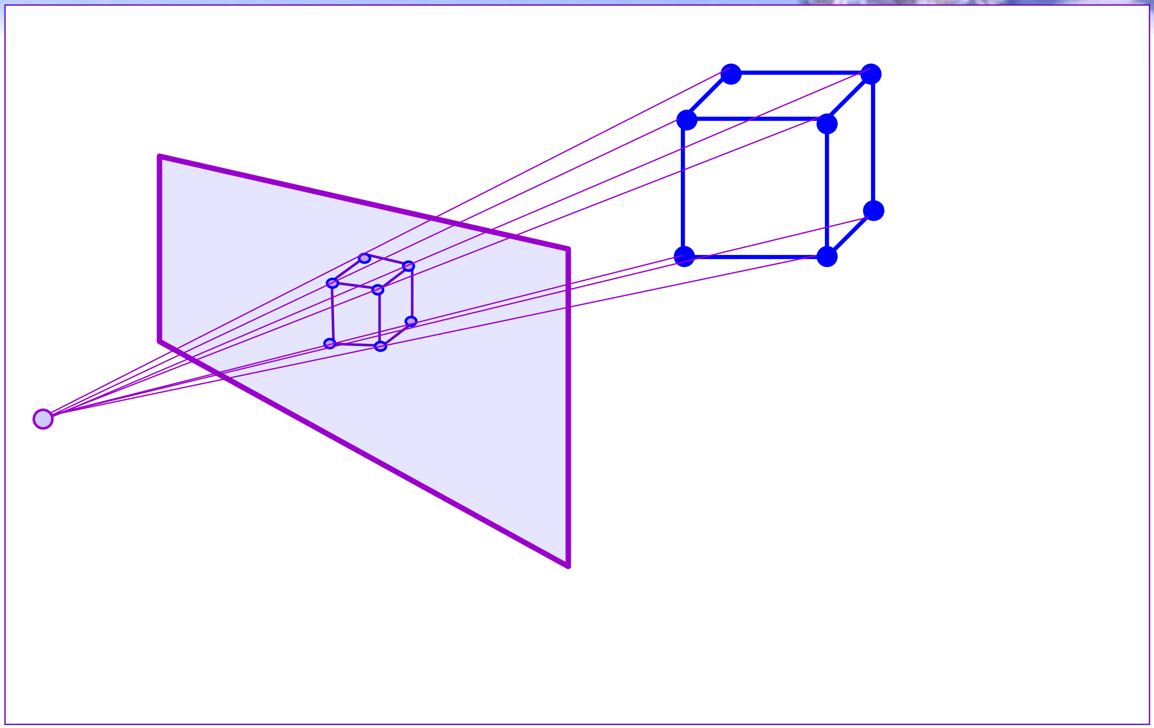



Multi-view geometry - resection

- Projection equation

  $x_i = P_i X$

- Resection:
  - $x_i, X \longrightarrow P_i$



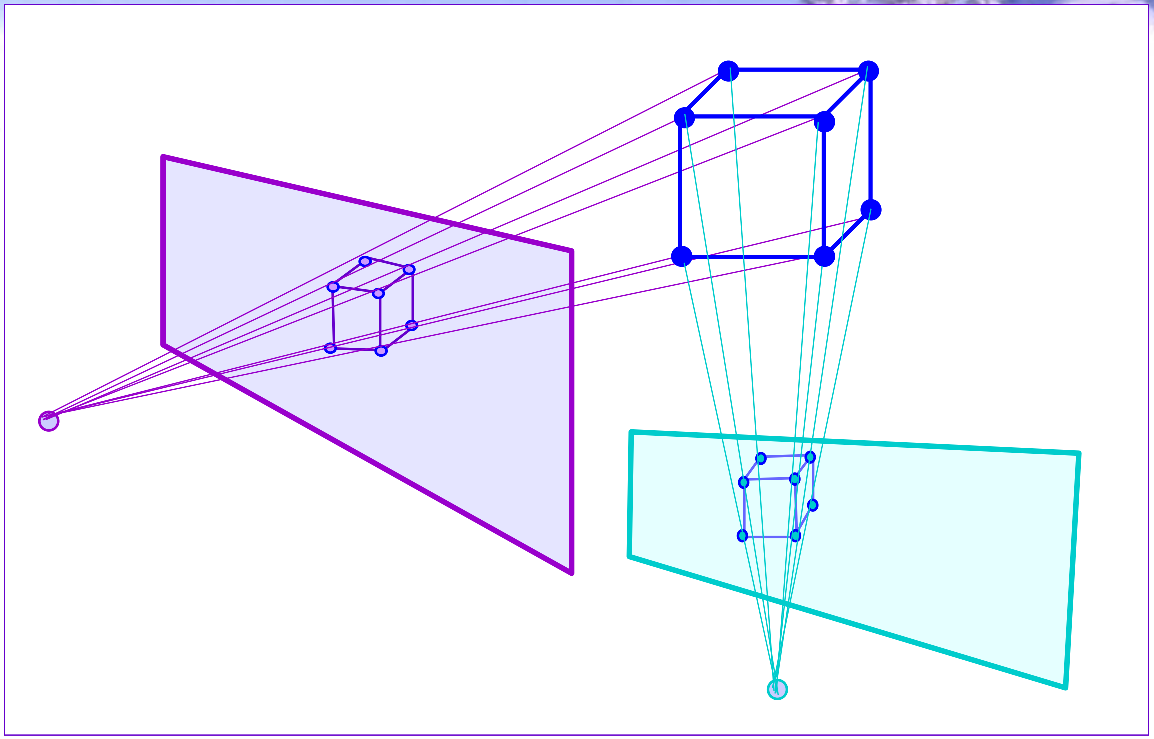
Given image points and 3D points calculate camera projection matrix.

Multi-view geometry - intersection

- Projection equation

  $x_i = P_i X$

- Intersection:
  - $x_i, P_i \longrightarrow X$



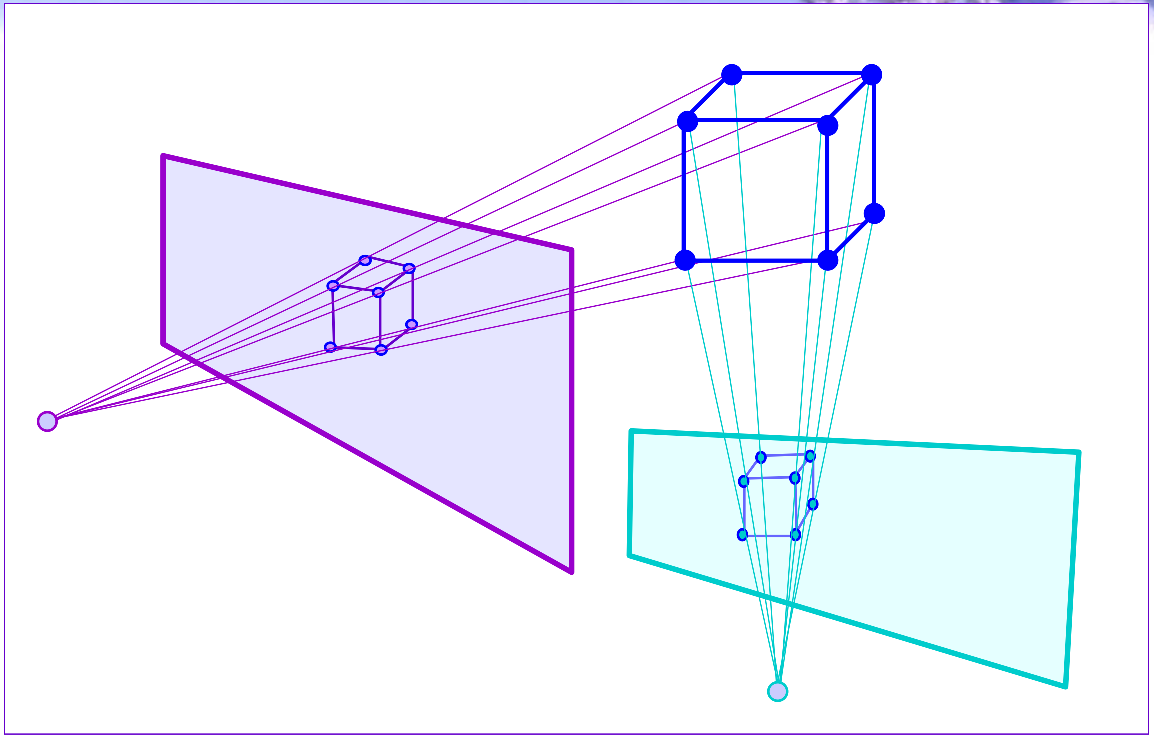
Given image points and camera projections in at least 2 views calculate the 3D points (structure)

Multi-view geometry - SFM

- Projection equation

  $x_i = P_i X$

- Structure from motion (SFM)
  - $x_i \longrightarrow P_i, X$



Given image points in at least 2 views calculate the 3D points (structure) and camera projection matrices (motion)

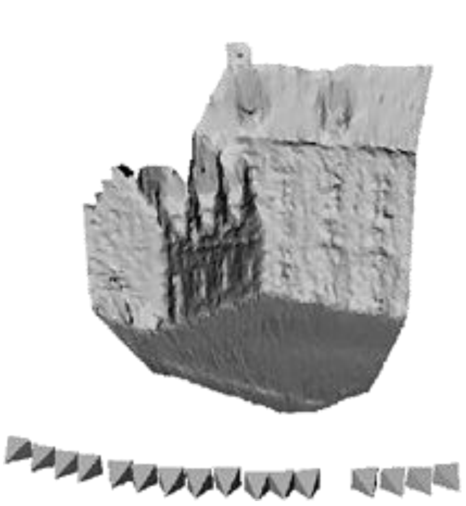
- Estimate projective structure
- Rectify the reconstruction to metric (autocalibration)

Examples: geometric modeling

Debevec and Taylor: Façade

Examples: geometric modeling

Pollefeys: Arenberg Castle



Examples – modeling with dynamic texture

Cobzas,Yerex,Jagersand

Ways to 3D pointcloud from images

- Aligned Cameras: Stereo camera (Lab 3)
- Know cameras beforehand (Photogrammetry)
- First compatible cameras, then 3D Geom
  - Fundametal matrix F -> Pcompat -> 3D triang
  - All of this in projective geom.
- Simultaneous computation of cameras and 3D
  - Compact math formulation. Easy to understand?

Pinhole camera

- Central projection

$$(X,Y,Z)^T \to (fX/Z, fY/Z)^T$$

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \sim \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

- Principal point & aspect

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \sim \begin{pmatrix} \frac{1}{p_x} x + c_x \\ \frac{1}{p_y} y + c_y \\ 1 \end{pmatrix} = \begin{bmatrix} \frac{1}{p_x} & 0 & c_x \\ 0 & \frac{1}{p_y} & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

The projection matrix:

$$\mathbf{x} = \begin{bmatrix} \frac{f}{p_x} & 0 & c_x & 0 \\ 0 & \frac{f}{p_y} & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{X}_{cam}$$

$$\mathbf{x} = K[I \mid \mathbf{0}]\mathbf{X}_{cam}$$

Projective camera

- Camera rotation and translation

$$\mathbf{X} = [R \quad \mathbf{t}]\mathbf{X}_{cam} \qquad \mathbf{X}_{cam} = [R^T \quad -R^T\mathbf{t}]\mathbf{X}$$

- The projection matrix

$$\mathbf{x} = \underbrace{KR^T[I \quad -\mathbf{t}]}_{P}\mathbf{X}$$

In general:

•P is a 3x4 matrix with 11 DOF

•Finite: left 3x3 matrix non-singular

•Infinite: left 3x3 matrix singular

Properties: $P=[M\ p_4]$

•Center: $P\mathbf{C} = 0$

$$\mathbf{C} = \begin{pmatrix} -M^{-1}\mathbf{p}_4 \\ 1 \end{pmatrix} \qquad \mathbf{C} = \begin{pmatrix} \mathbf{d} \\ 0 \end{pmatrix}, M\mathbf{d} = 0$$

•Principal ray (projection direction)

$$\mathbf{v} = \det(M)\mathbf{m}^3$$

Affine cameras

- Infinite cameras where the last row of P is (0,0,0,1)
- Points at infinity are mapped to points at infinity

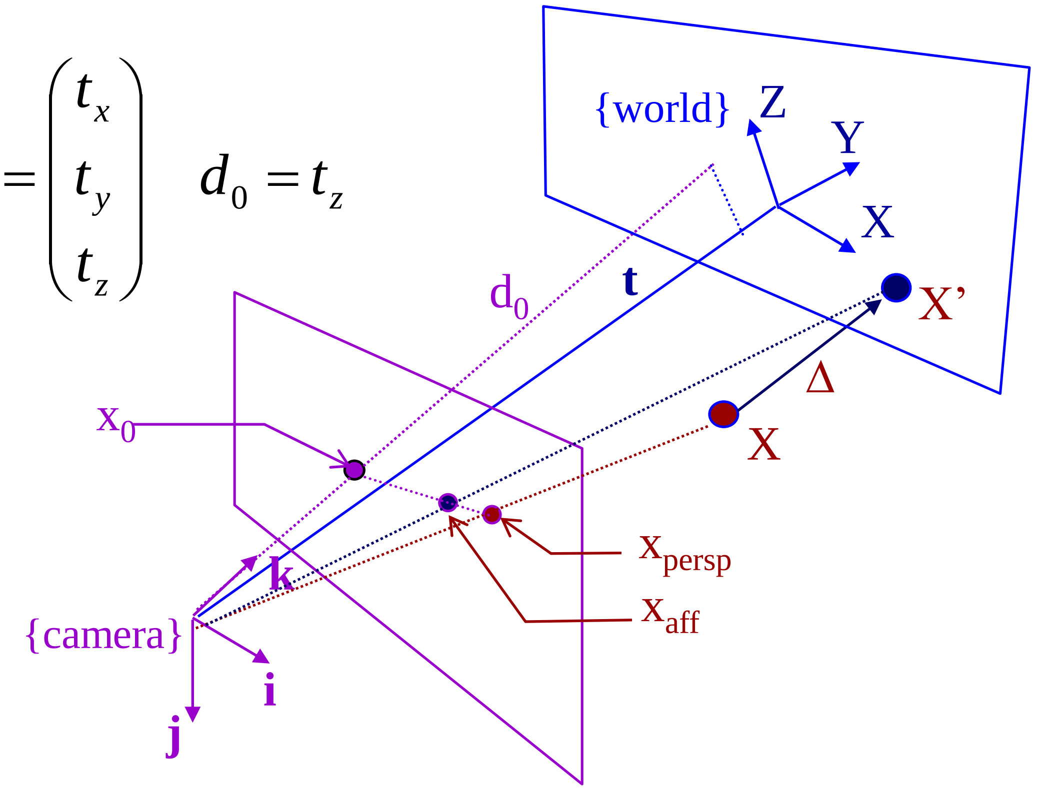
$$P_\infty = K\begin{bmatrix} \mathbf{i} & t_x \\ \mathbf{j} & t_y \\ \mathbf{0}^T & d_0 \end{bmatrix} \quad R = \begin{pmatrix} \mathbf{i} \\ \mathbf{j} \\ \mathbf{k} \end{pmatrix} \quad \mathbf{t} = \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \quad d_0 = t_z$$

- Error

$$\mathbf{x}_{aff} - \mathbf{x}_{persp} = \frac{\Delta}{d_0}(\mathbf{x}_{proj} - \mathbf{x}_0)$$

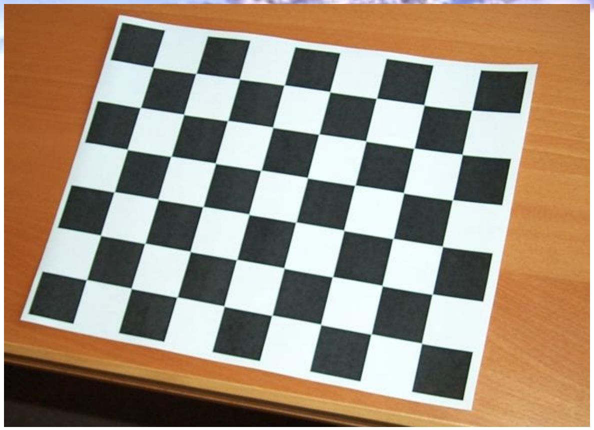
Good approximation:

- Δ small compared to d_0
- point close to principal ray



Camera calibration

$$\underbrace{\mathbf{x}_i}_{\text{known}} = \underbrace{P}_{?} \underbrace{\mathbf{X}_i}_{\text{known}}$$

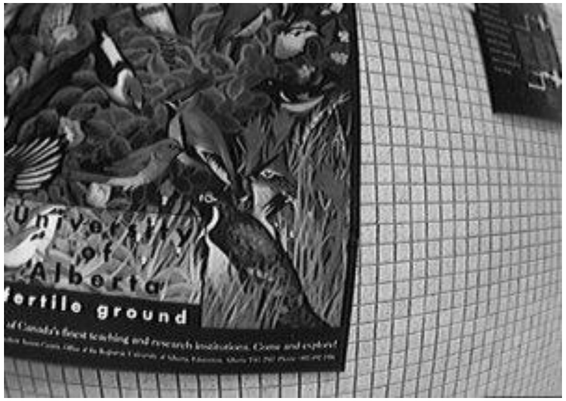


- 11 DOF => at least 6 points
- Linear solution
  - Normalization required
  - Minimizes algebraic error

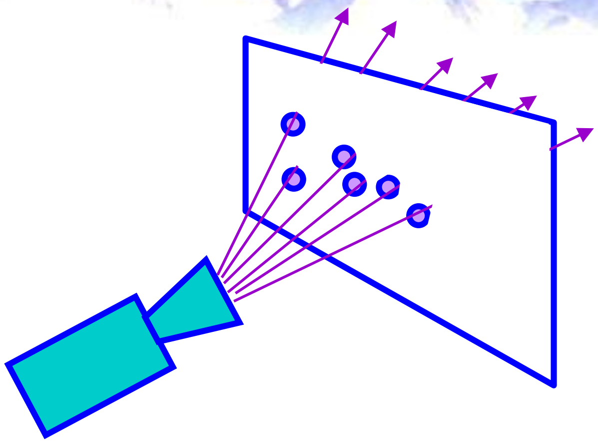
$$\begin{cases} \min \ A\mathbf{p} = 0 \\ \|\mathbf{p}\| = 1 \end{cases}$$

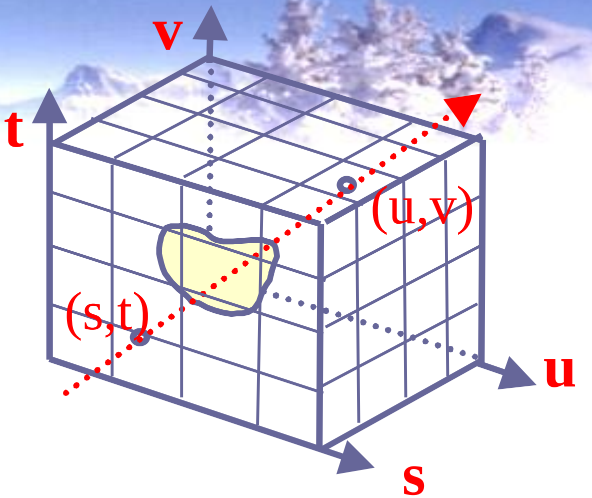
- Nonlinear solution
  - Minimize geometric error (pixel re-projection)
- Radial distortion
  - Small near the center, increase towards periphery

$$\delta r = 1 + K_1 r + K_2 r^2 + \ldots$$



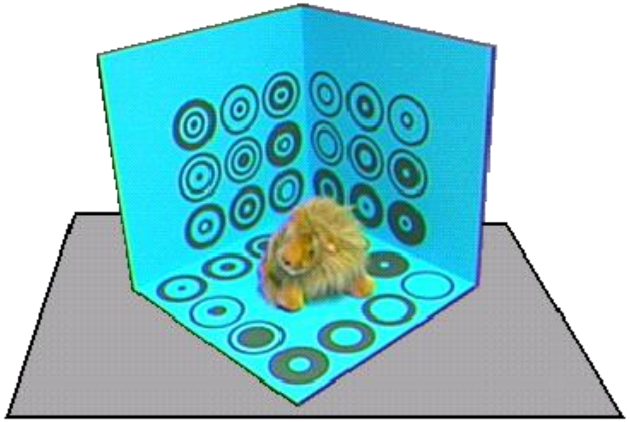
Application: raysets

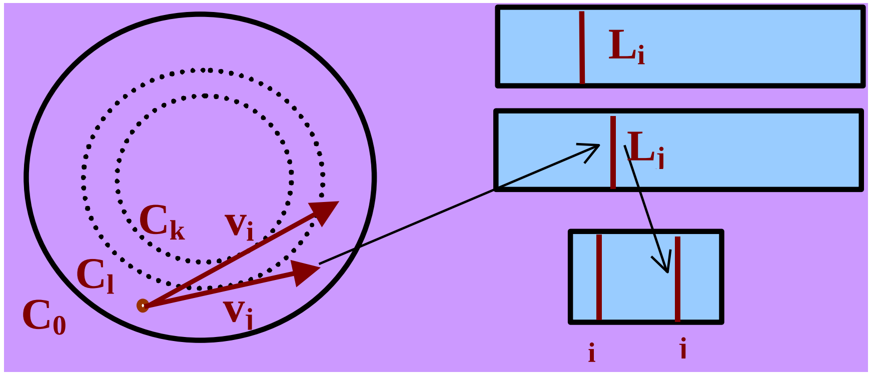




Gortler and *al*.; Microsoft Lumigraph

H-Y Shum, L-W He; Microsoft Concentric mosaics



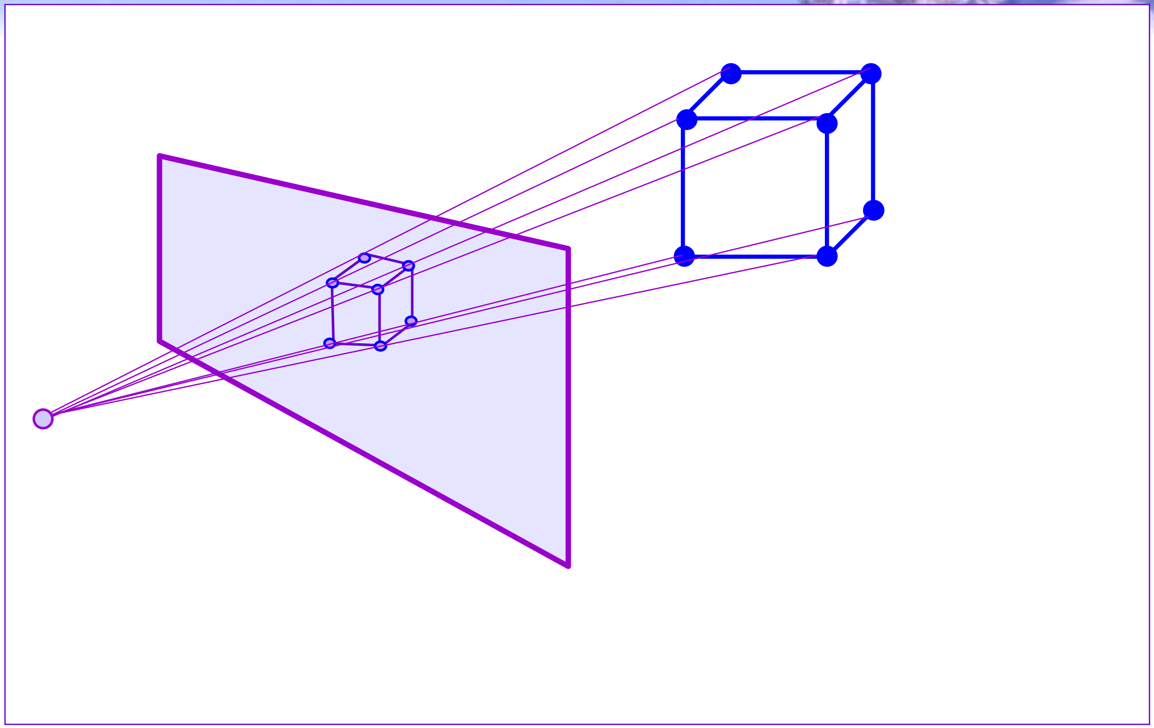


Multi-view geometry - resection

- Projection equation

  $x_i = P_i X$

- Resection:
  - $x_i, X \longrightarrow P_i$



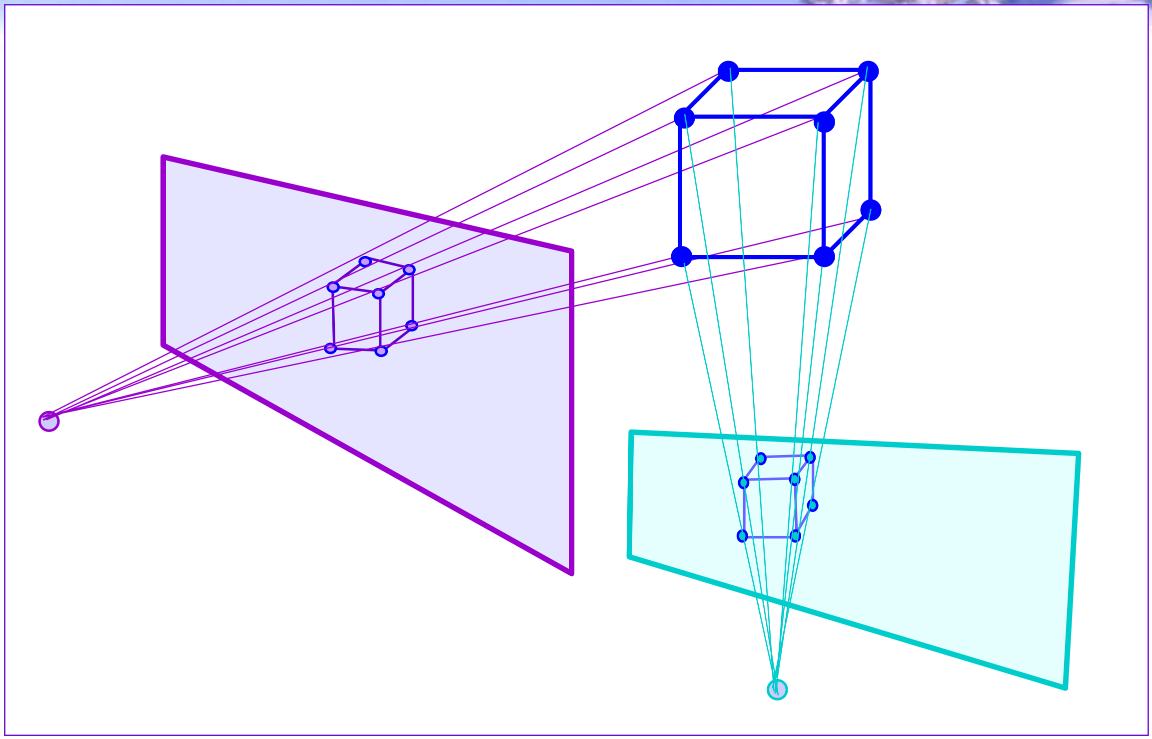
Given image points and 3D points calculate camera projection matrix.

Multi-view geometry - intersection

- Projection equation

  $x_i = P_i X$

- Intersection:
  - $x_i, P_i \longrightarrow X$



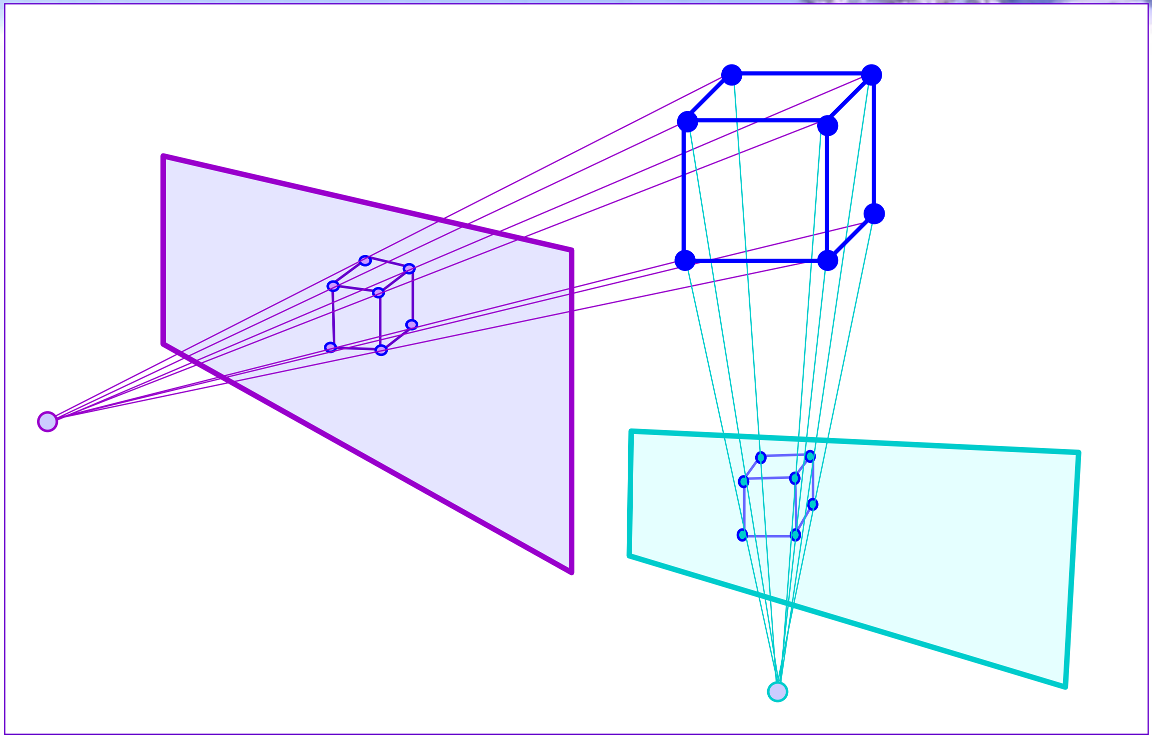
Given image points and camera projections in at least 2 views calculate the 3D points (structure)

Multi-view geometry - SFM

- Projection equation

  $x_i = P_i X$

- Structure from motion (SFM)
  - $x_i \longrightarrow P_i, X$

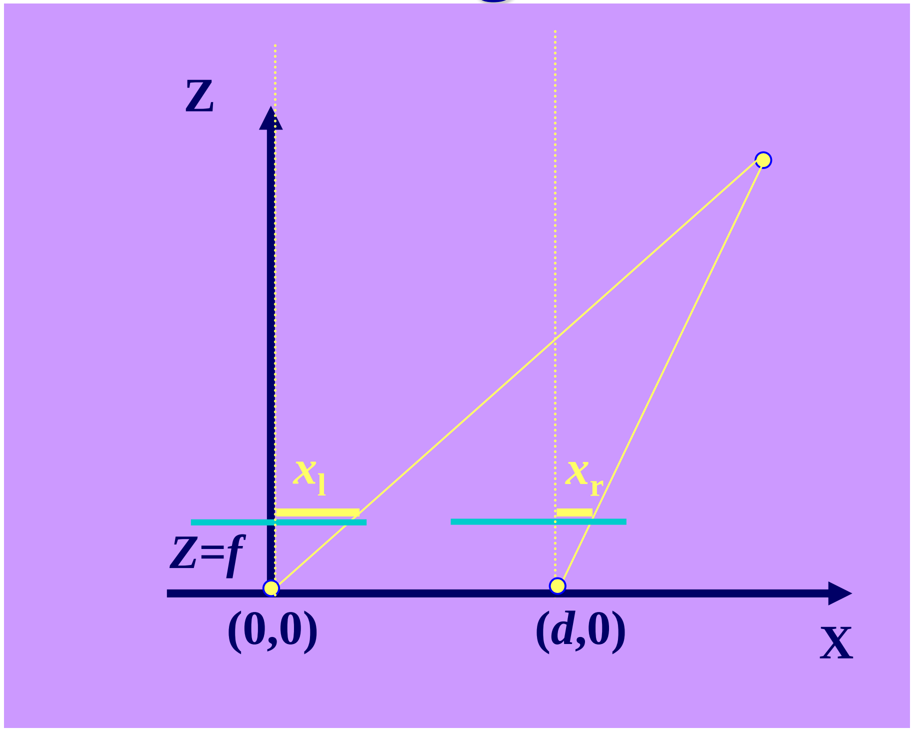


Given image points in at least 2 views calculate the 3D points (structure) and camera projection matrices (motion)

- Estimate projective structure
- Rectify the reconstruction to metric (autocalibration)

Depth from stereo

- Calibrated aligned cameras



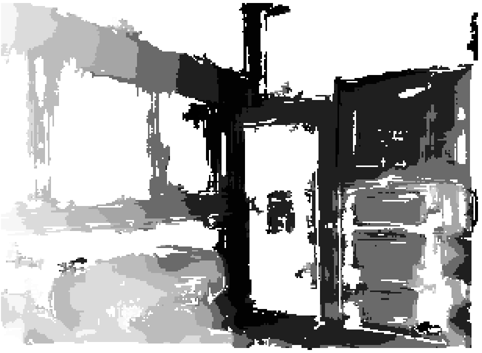
Disparity d

$$Z = \frac{f}{x_l} X = \frac{f}{x_r}(X - d)$$

$$Z = \frac{df}{x_l - x_r}$$

Trinocular Vision System

(Point Grey Research)



Application: depth based reprojection

3D warping, McMillan

Plenoptic modeling, McMillan & Bishop

Application: depth based reprojection

Layer depth images, Shade et al.

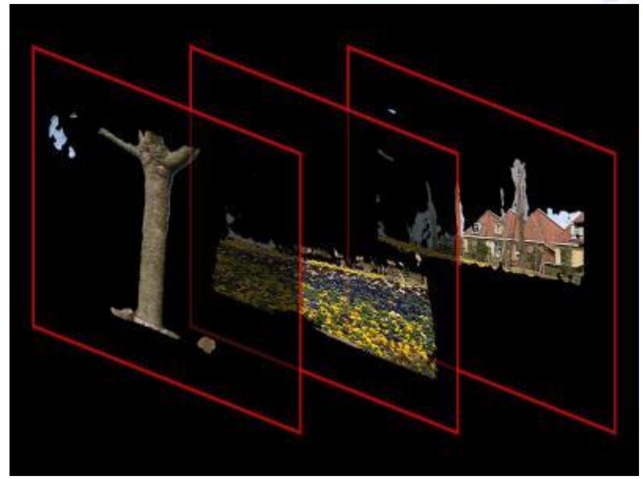
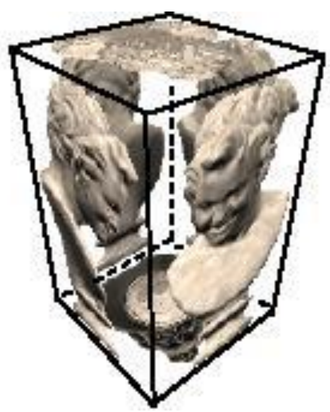


Image based objects, Oliveira & Bishop

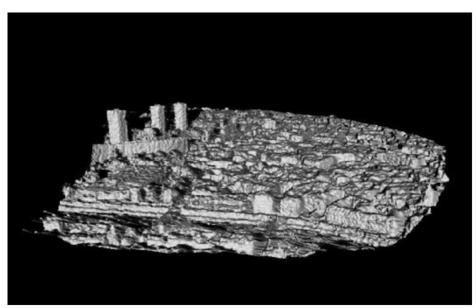


Sequential 3D structure from motion using 2 and 3 view geom

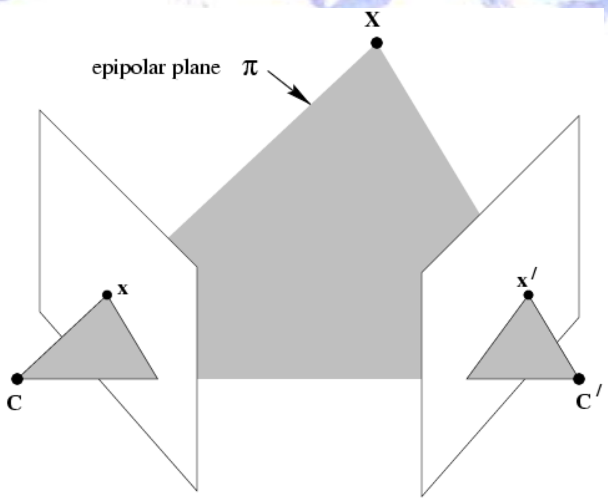
- Initialize structure and motion from two views
- For each additional view
  - Determine pose
  - Refine and extend structure

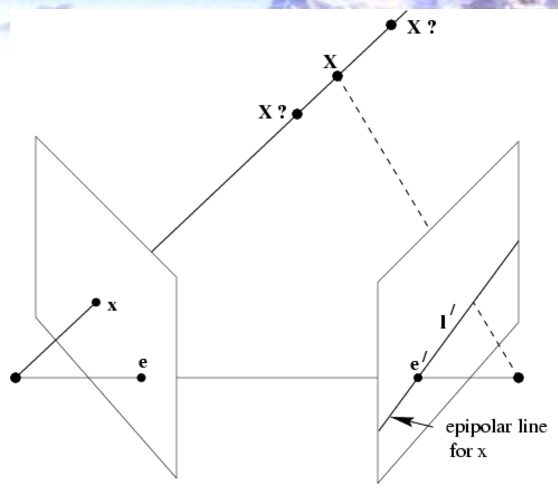
Images

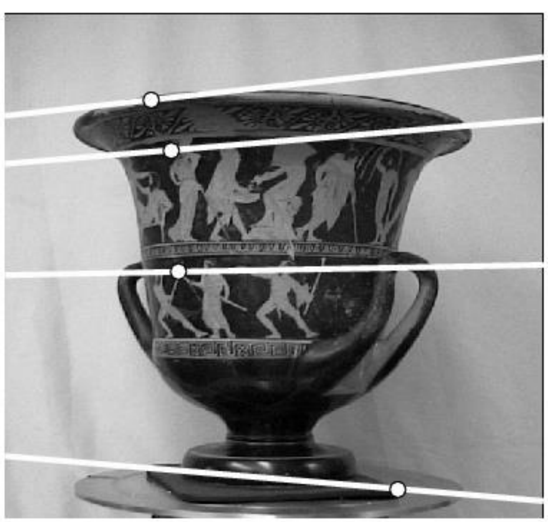
- Determine correspondences robustly by jointly estimating matches and epipolar geometry



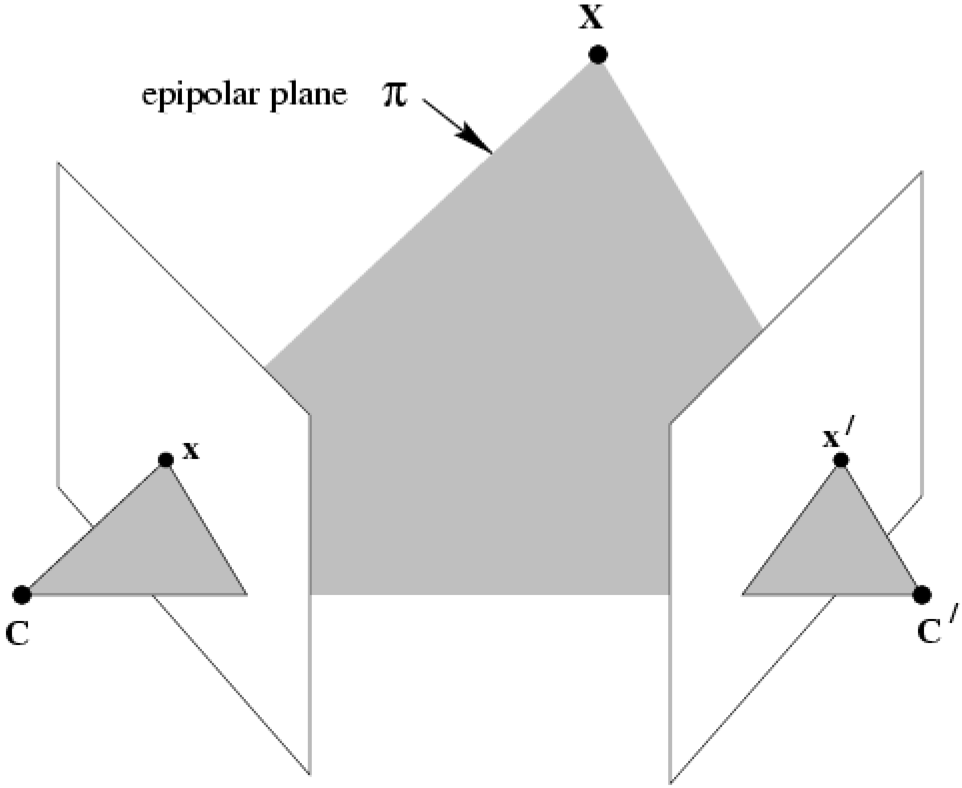
2 view geometry
Epipolar geometry and Fundamental matrix F





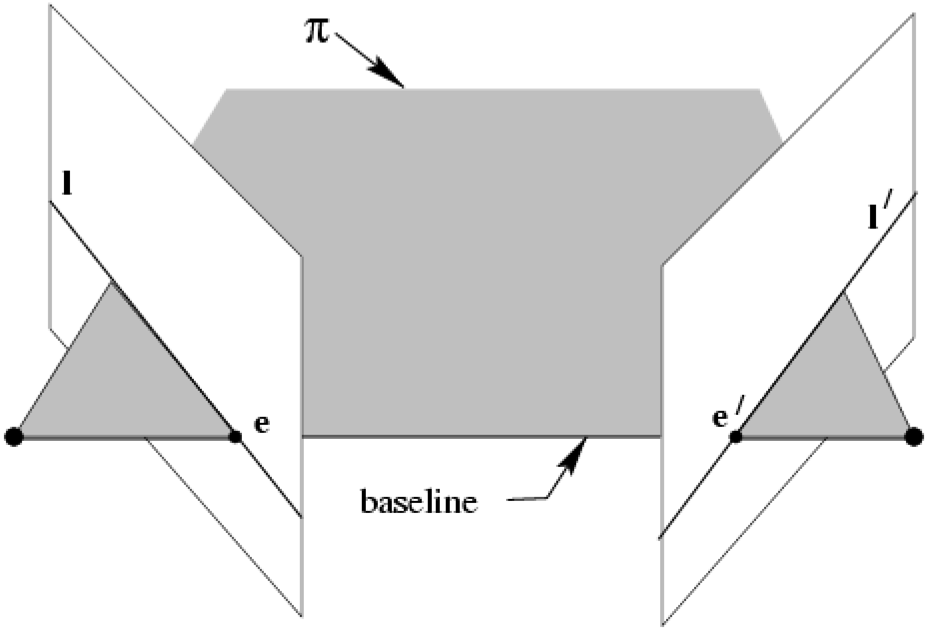


The epipolar geometry



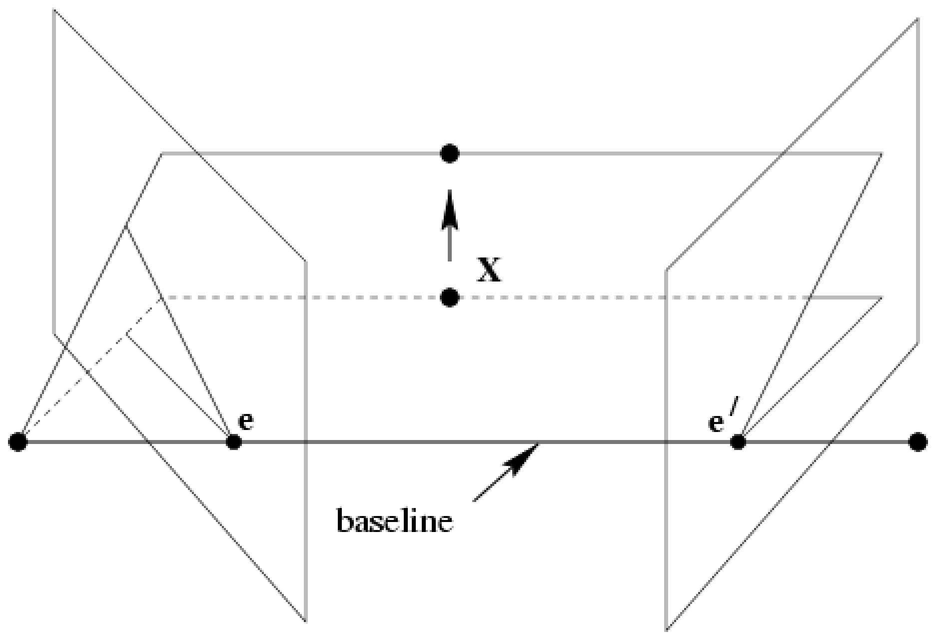
C,C',x,x' and X are coplanar

The epipolar plane



All points on π project on l and l'

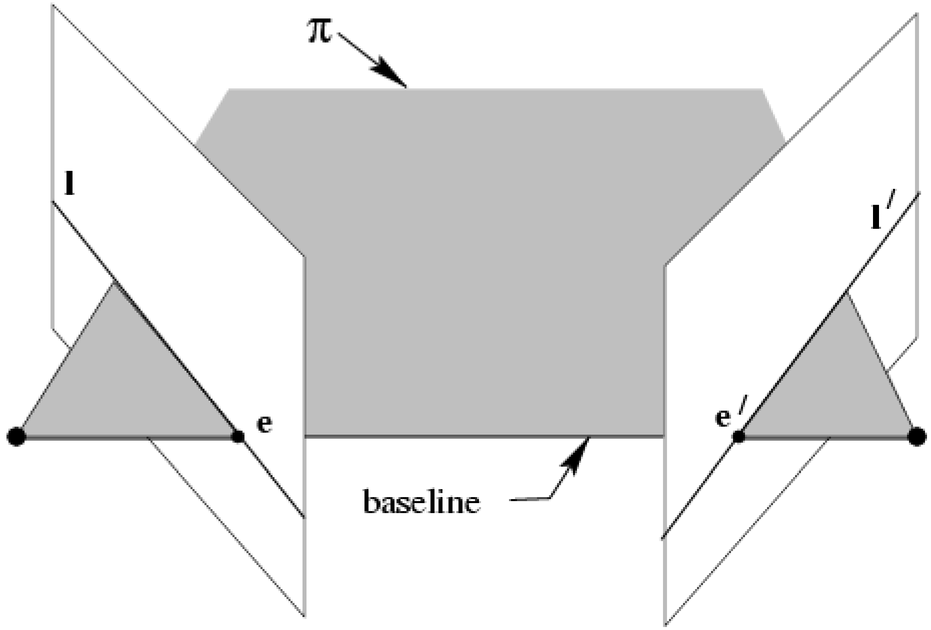
The epipolar planes



Family of planes π and lines l and l'
Intersection in e and e'

The epipoles

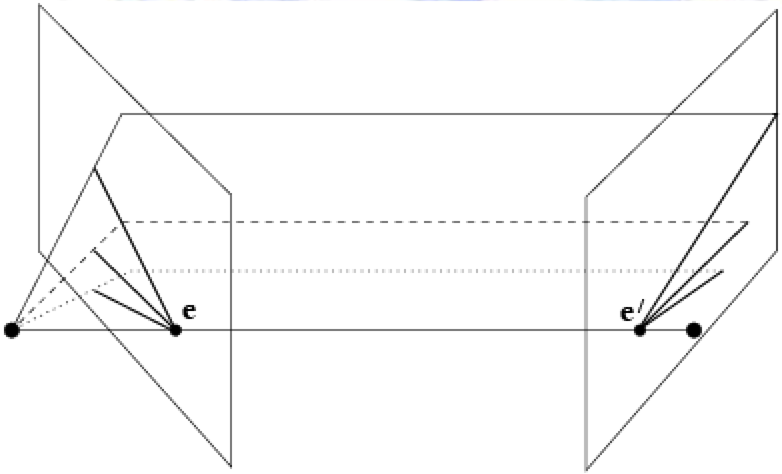
epipoles e,e’
= intersection of baseline with image plane
= projection of projection center in other image
= vanishing point of camera motion direction

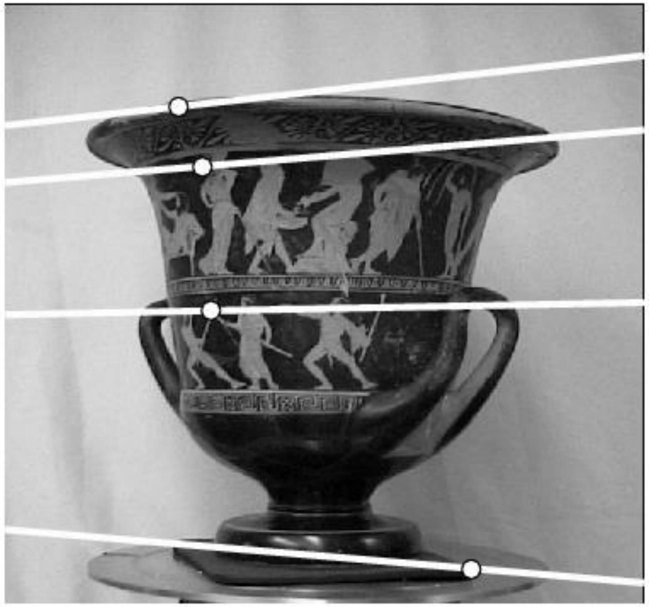


an epipolar plane = plane containing baseline (1-D family)

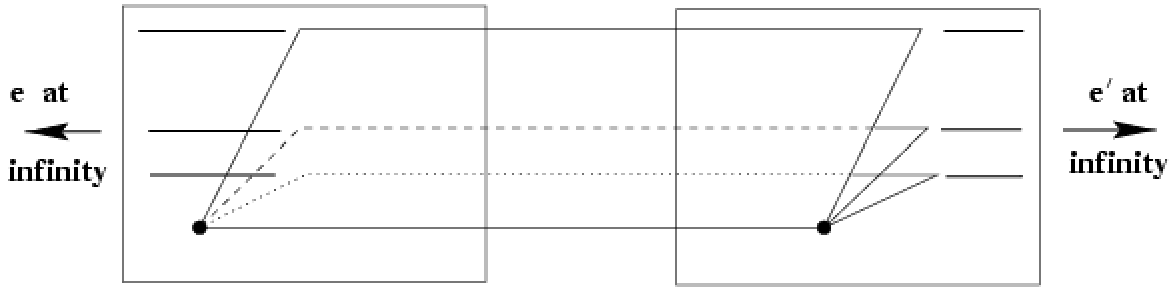
an epipolar line = intersection of epipolar plane with image
(always come in corresponding pairs)

Example: converging cameras

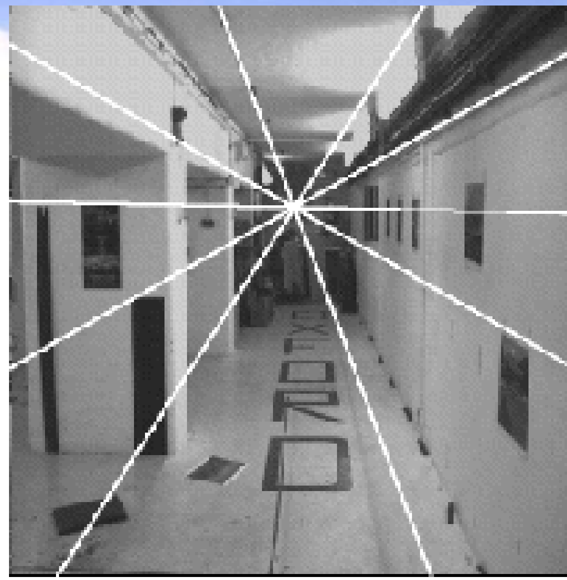


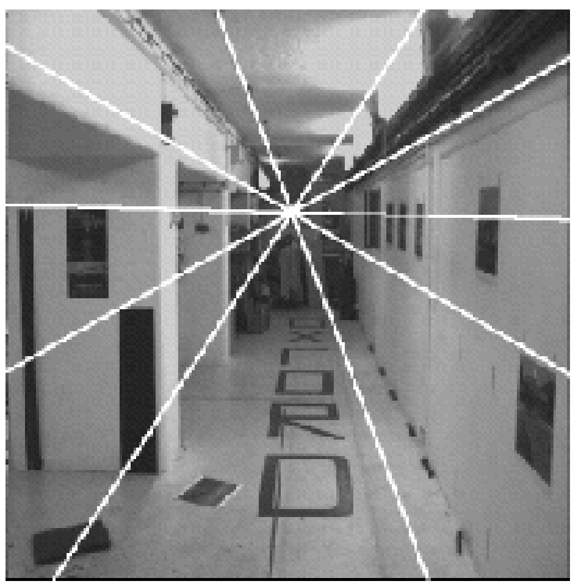


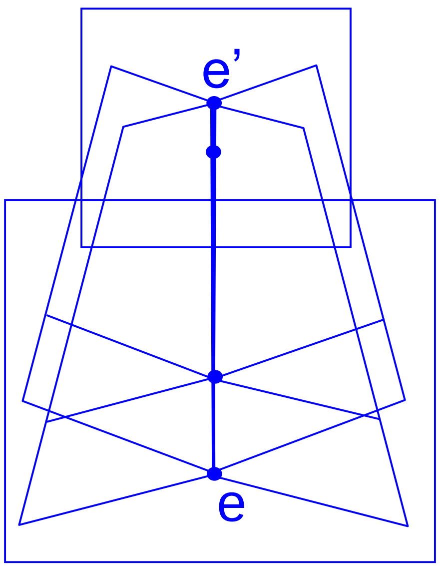
Example: motion parallel with image plane



Example: forward motion







The fundamental matrix F

algebraic representation of epipolar geometry

$$x \mapsto l'$$

we will see that this mapping is (singular) correlation (i.e. projective mapping from points to lines) represented by the fundamental matrix F

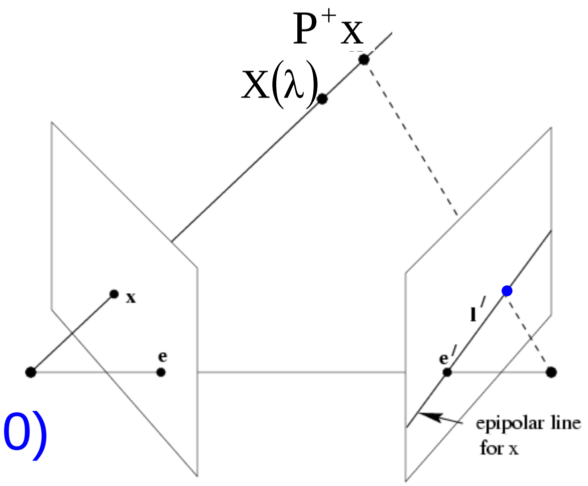
The fundamental matrix F

algebraic derivation (of existence)

$$X(\lambda) = P^{+}x + \lambda C \qquad \left(P^{+}P = I\right)$$

$$l' = P'C \times P'P^{+}x$$

$$F = [e']_{\times} P'P^{+}$$



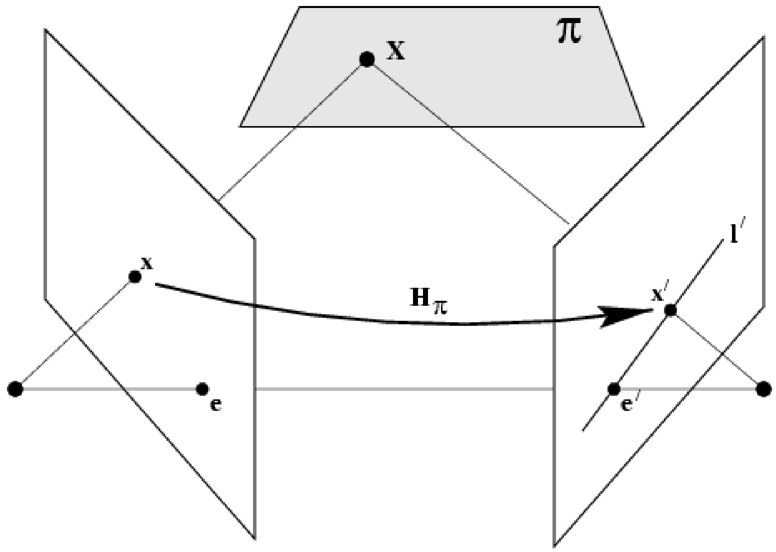
(note: doesn't work for C=C' $\Rightarrow$ F=0)

Alternatively can write:

$$F = [e']_{\times} H_{\infty} \qquad \left(H_{\infty} = K^{-1}RK\right)$$

The fundamental matrix F

geometric derivation

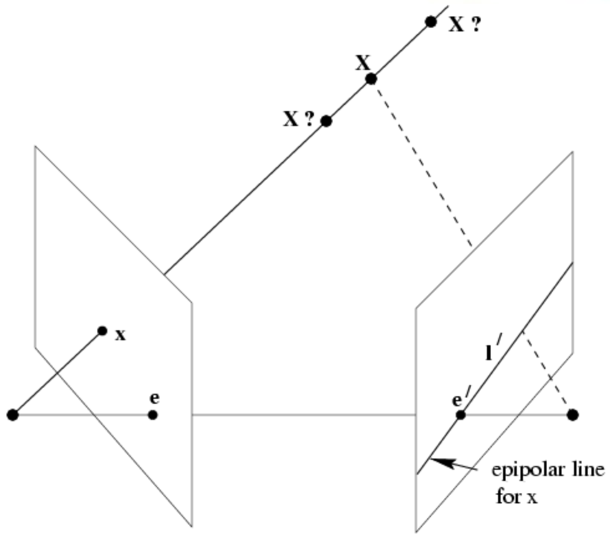


Step 1: X on a plane π $\quad x' = H_\pi x$

Step 2: epipolar line l' $\quad l' = e' \times x' = [e']_\times H_\pi x = F x$

mapping from 2-D to 1-D family (rank 2)

The fundamental matrix F



correspondence condition

The fundamental matrix satisfies the condition that for any pair of corresponding points x↔x' in the two images

$$x'^T F x = 0 \qquad \left(x'^T l' = 0\right)$$

The fundamental matrix F

F is the unique 3x3 rank 2 matrix that satisfies $x'^T F x = 0$ for all $x \leftrightarrow x'$

(i) **Transpose:** if F is fundamental matrix for (P,P'), then F^T is fundamental matrix for (P',P)

(ii) **Epipolar lines:** $l' = Fx$ & $l = F^T x'$

(iii) **Epipoles:** on all epipolar lines, thus $e'^T F x = 0, \forall x$ $\Rightarrow e'^T F = 0$, similarly $Fe = 0$

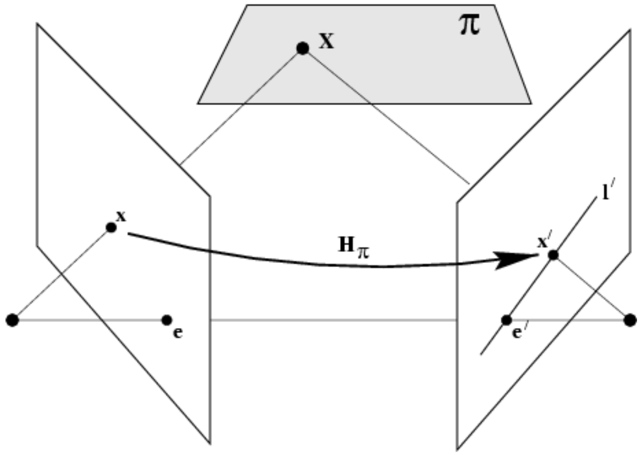
(iv) **F** has 7 d.o.f. , i.e. 3x3-1(homogeneous)-1(rank2)

(v) **F** is a correlation, projective mapping from a point x to a line $l' = Fx$ (not a proper correlation, i.e. not invertible)

Fundamental matrix, summary

[Faugeras ’92, Hartley ’92]

- Algebraic representation of epipolar geometry



Step 1: X on a plane π $\mathbf{x}' = H\mathbf{x}$

Step 2: epipolar line l’ $\mathbf{l}' = \mathbf{e}' \times \mathbf{x}' = [\mathbf{e}']_\times \mathbf{x}'$

$$= [\mathbf{e}']_\times H\mathbf{x} = F\mathbf{x}$$

$$\mathbf{x}'^T F\mathbf{x} = 0$$

F

- 3x3, Rank 2, det(F)=0
- Linear sol. – 8 corr. Points (unique)
- Nonlinear sol. – 7 corr. points (3sol.)
- Very sensitive to noise & outliers

Epipolar lines: $\mathbf{l}' = F\mathbf{x} \quad \mathbf{l} = F^T\mathbf{x}'$

Epipoles: $F\mathbf{e} = 0 \quad F^T\mathbf{e}' = 0$

Projection matrices: $P = [I \mid \mathbf{0}]$

$$P' = \left[[\mathbf{e}']_\times F + \mathbf{e}'\mathbf{v}^T \mid \lambda\mathbf{e}'\right]$$

Relating 3D geometry and 2D images
The Fundamental Matrix F

F Relates to three questions:

(i) Correspondence geometry: Given an image point x in the first view, how does this constrain the position of the corresponding point x' in the second image?

(ii) Camera geometry (motion): Given a set of corresponding image points $\{x_i \leftrightarrow x'_i\}$, i=1,…,n, what are the cameras P and P' for the two views?

(iii) Scene geometry (structure): Given corresponding image points $x_i \leftrightarrow x'_i$ and cameras P, P', what is the position of (their pre-image) X in space?

Computing F; 8 pt alg

$$\mathrm{x}'^{\mathrm{T}}\mathrm{F}\mathrm{x} = 0$$

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + xf_{31} + yf_{32} + f_{33} = 0$$

separate known from unknown

$$[x'x, x'y, x', y'x, y'y, y', x, y, 1][f_{11}, f_{12}, f_{13}, f_{21}, f_{22}, f_{23}, f_{31}, f_{32}, f_{33}]^{\mathrm{T}} = 0$$

(data) (unknowns) (linear)

$$\begin{bmatrix} x'_1 x_1 & x'_1 y_1 & x'_1 & y'_1 x_1 & y'_1 y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_n x_n & x'_n y_n & x'_n & y'_n x_n & y'_n y_n & y'_n & x_n & y_n & 1 \end{bmatrix} \mathrm{f} = 0$$

$$\mathrm{A}\mathrm{f} = 0$$

8-point algorithm

$$\begin{bmatrix} x_1x_1' & y_1x_1' & x_1' & x_1y_1' & y_1y_1' & y_1' & x_1 & y_1 & 1 \\ x_2x_2' & y_2x_2' & x_2' & x_2y_2' & y_2y_2' & y_2' & x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_nx_n' & y_nx_n' & x_n' & x_ny_n' & y_ny_n' & y_n' & x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0$$

$$\mathrm{Af} = 0$$

Solve for nontrivial solution using SVD:

$$\mathbf{A} = \mathbf{USV}^{\mathbf{T}} \qquad \left\|\mathbf{USV}^{\mathbf{T}}\right\| = \left\|\mathbf{SV}^{\mathrm{T}}\right\| \qquad \|\mathbf{x}\| = \|\mathbf{Vx}\|$$

Var subst: $\mathbf{y} = \mathbf{Vx}$ Now Min $\mathbf{Sy} \Leftrightarrow \mathbf{y} = [0,0,\ldots,0,1]^T$

Hence x = last vector in V

(Also 3,4,N view geometry. HZ 15,16)

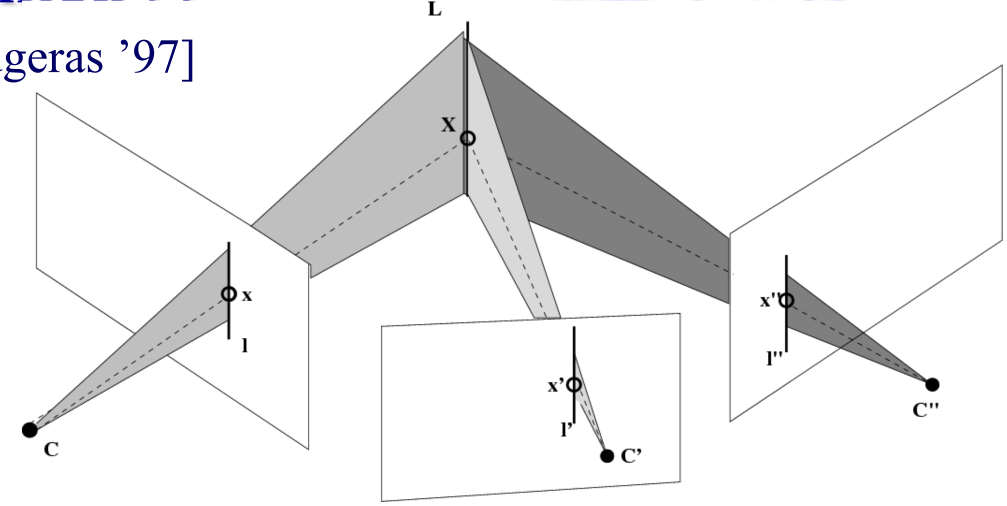
- Trifocal tensor (3 view geometry)

[Hartley '97][Torr & Zisserman '97][Faugeras '97]

$\mathbf{T}:[T_1,T_2,T_3]$ 3x3x3 tensor;
27 params. (18 indep.)

$\mathbf{l}'[T1,T2,T3]\mathbf{l}''=\mathbf{l}^T$ lines

$[\mathbf{x}']_\times(\sum_i \mathbf{x}^i T_i)[\mathbf{x}'']_\times = 0$ points



- Quadrifocal tensor (4 view geometry) [Triggs '95]
- Multiview tensors [Hartley'95][Hayden '98]

There is no additional constraint between more than 4 images. All the constraints can be expressed using F,triliear tensor or quadrifocal tensor.

Using Fundamental Matrix F to compute structure and motion

Epipolar geometry ↔ Projective calibration

$$\mathbf{m}_2^{\mathsf{T}}\mathbf{F}\mathbf{m}_1 = 0$$

$$\mathbf{P}_1 = \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix}$$

$$\mathbf{P}_2 = \begin{bmatrix} [\mathbf{e}]_x \mathbf{F} + \mathbf{e}\mathbf{a}^{\mathrm{T}} & \mathbf{e} \end{bmatrix}$$

compatible with F

Yields correct projective camera setup

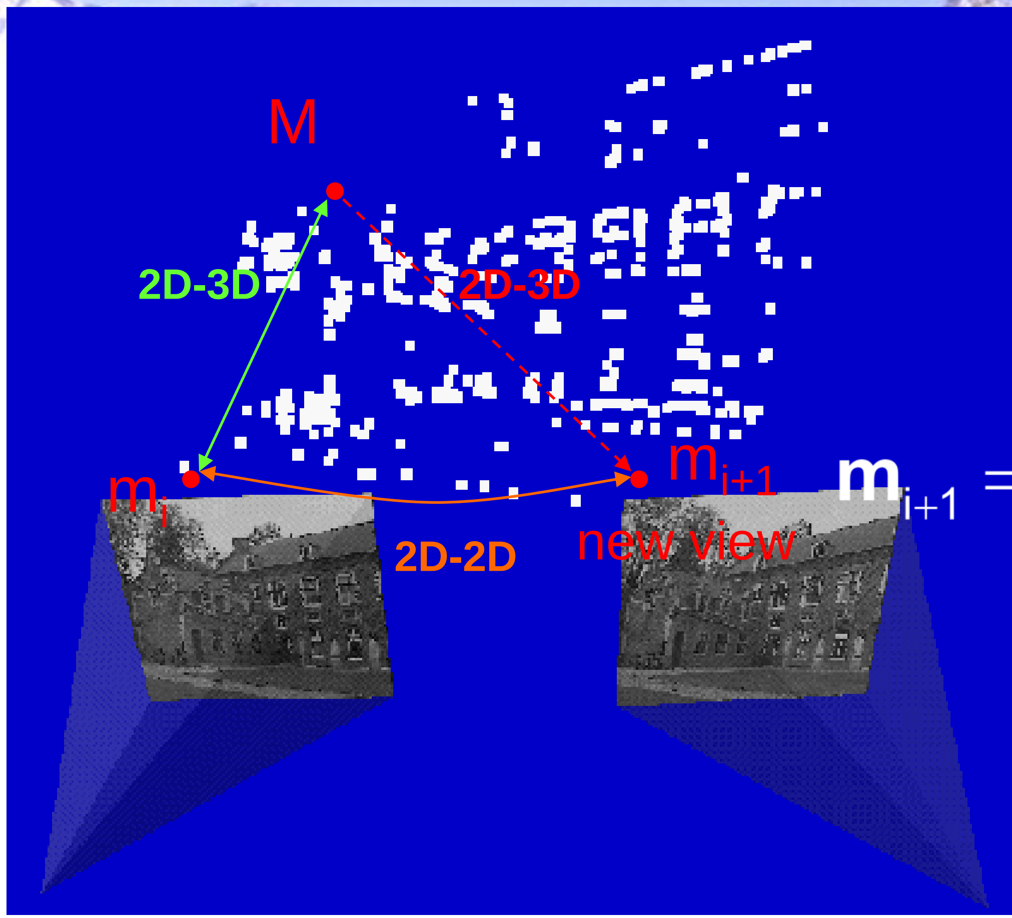
(Faugeras´92,Hartley´92)

Obtain structure through triangulation

Use reprojection error for minimization

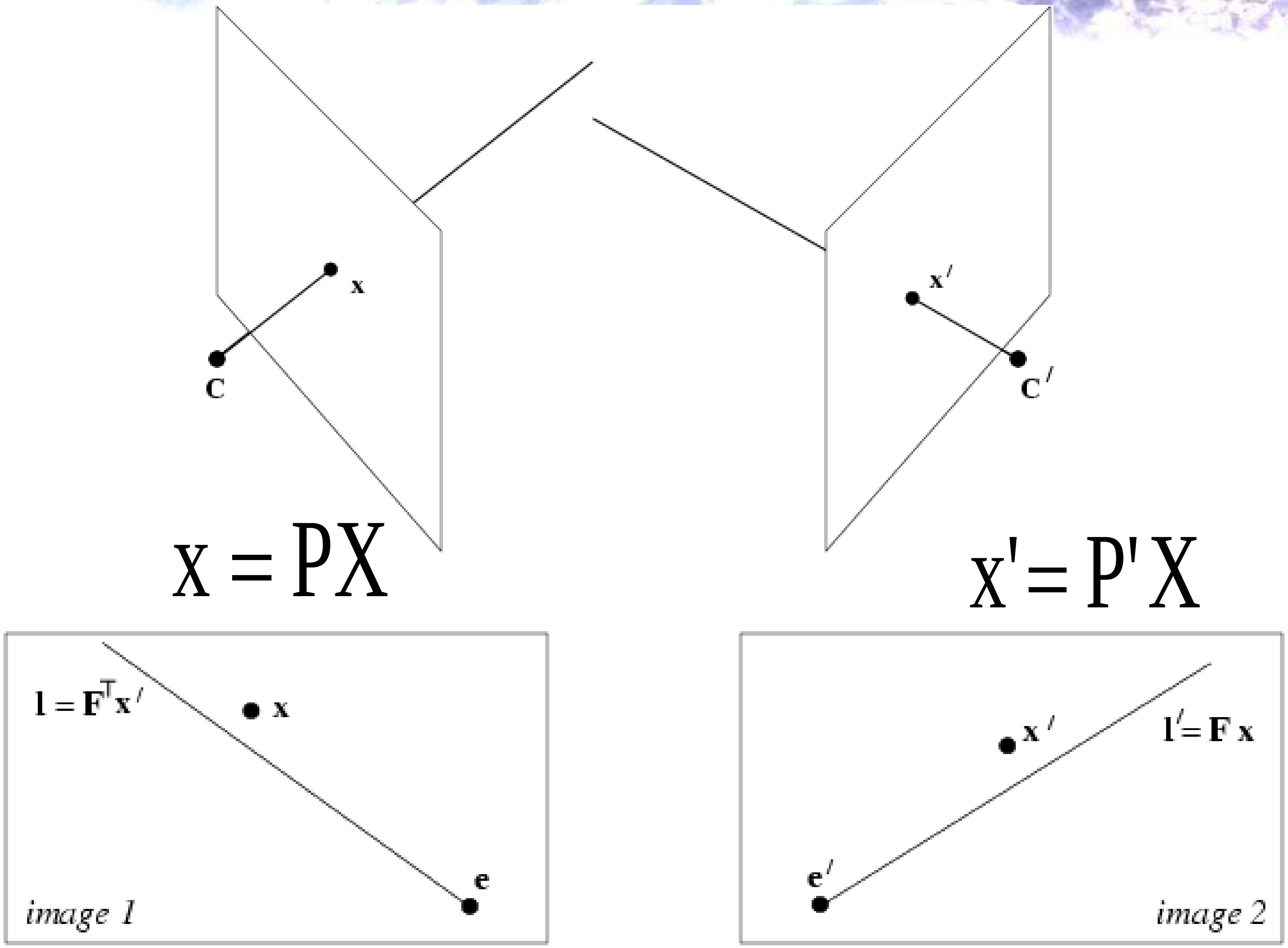
Avoid measurements in projective space

Determine coordinates of 3D Points compatible with $\mathbf{P}_1$ and $\mathbf{P}_2$



1. Compute $\mathbf{P}_1$ and $\mathbf{P}_2$
2. Triangulate 3D points

Structure from images:
3D Point reconstruction



linear triangulation

$$\mathbf{x} = \mathrm{P}\mathbf{X} \quad \mathbf{x}' = \mathrm{P}'\mathbf{X}$$

$$\mathbf{x} \times \mathrm{P}'\mathbf{X} = 0$$

$$x\left(\mathbf{p}^{3T}\mathbf{X}\right) - \left(\mathbf{p}^{1T}\mathbf{X}\right) = 0$$

$$y\left(\mathbf{p}^{3T}\mathbf{X}\right) - \left(\mathbf{p}^{2T}\mathbf{X}\right) = 0$$

$$x\left(\mathbf{p}^{2T}\mathbf{X}\right) - y\left(\mathbf{p}^{1T}\mathbf{X}\right) = 0$$

$$\mathrm{A}\mathbf{X} = 0$$

$$\mathrm{A} = \begin{bmatrix} x\mathbf{p}^{3T} - \mathbf{p}^{1T} \\ y\mathbf{p}^{3T} - \mathbf{p}^{2T} \\ x'\mathbf{p}'^{3T} - \mathbf{p}'^{1T} \\ y'\mathbf{p}'^{3T} - \mathbf{p}'^{2T} \end{bmatrix}$$

homogeneous

$$\left\|\mathbf{X}\right\| = 1$$

invariance?

$$(\mathrm{A}\mathrm{H}^{-1})(\mathrm{H}\mathbf{X}) = \mathbf{e}$$

inhomogeneous

$$(X, Y, Z, 1)$$

algebraic error yes,
constraint no
(except for affine)

linear triangulation

$$\mathrm{x} = \mathrm{P}\mathrm{X} \qquad \mathrm{x}' = \mathrm{P}'\mathrm{X}$$

$$\mathrm{x} \times \mathrm{P}'\mathrm{X} = 0$$

$$x\left(\mathrm{p}^{3T}\mathrm{X}\right) - \left(\mathrm{p}^{1T}\mathrm{X}\right) = 0$$
$$y\left(\mathrm{p}^{3T}\mathrm{X}\right) - \left(\mathrm{p}^{2T}\mathrm{X}\right) = 0$$
$$x\left(\mathrm{p}^{2T}\mathrm{X}\right) - y\left(\mathrm{p}^{1T}\mathrm{X}\right) = 0$$

$$\mathrm{A}\mathrm{X} = 0$$

$$\mathrm{A} = \begin{bmatrix} x\mathrm{p}^{3T} - \mathrm{p}^{1T} \\ y\mathrm{p}^{3T} - \mathrm{p}^{2T} \\ x'\mathrm{p}'^{3T} - \mathrm{p}'^{1T} \\ y'\mathrm{p}'^{3T} - \mathrm{p}'^{2T} \end{bmatrix}$$

homogeneous

$$\|\mathrm{X}\| = 1$$

inhomogeneous

$$(X, Y, Z, 1)$$

invariance?

algebraic error yes, constraint no (except for affine)

Linear triangulation

Alternative way of linear intersection:

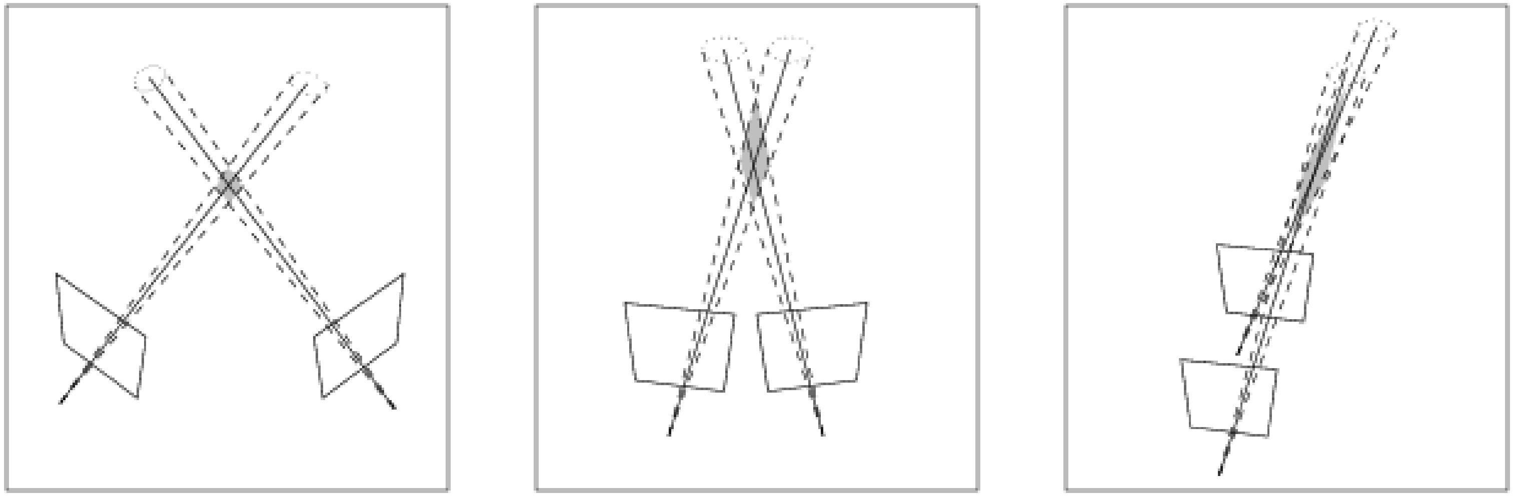
- Formulate a set of linear equations explicitly solving for λ's

$\lambda_1 \mathbf{x}_1 = P_1 \mathbf{X}$ and $\lambda_2 \mathbf{x}_2 = P_2 \mathbf{X}$ and rewrite

$$0 = \begin{bmatrix} P_1 & \mathbf{x}_1 & \mathbf{0}^T \\ P_2 & \mathbf{0}^T & \mathbf{x}_2 \end{bmatrix} \begin{bmatrix} \mathbf{X} \\ \lambda_1 \\ \lambda_2 \end{bmatrix}$$

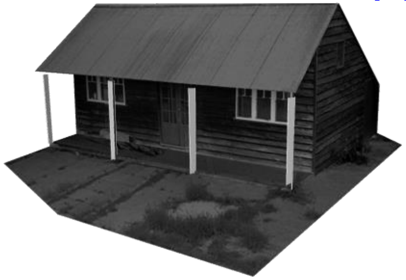
See our VR2003 tutorial p. 26

Reconstruction uncertainty



consider angle between rays

Summary: 2view Reconstuction



Objective

Given two uncalibrated images compute (P_M,P'_M,{X_{Mi}})
(i.e. within similarity of original scene and cameras)

Algorithm

(i) Compute projective reconstruction (P,P',{X_i})
  (a) Compute F from $x_i \leftrightarrow x'_i$
  (b) Compute P,P' from F
  (c) Triangulate X_i from $x_i \leftrightarrow x'_i$

(ii) Rectify reconstruction from projective to metric

Direct method: compute H from control points $X_{Ei} = HX_i$

$$P_M = PH^{-1} \quad P'_M = P'H^{-1} \quad X_{Mi} = HX_i$$

Stratified method:

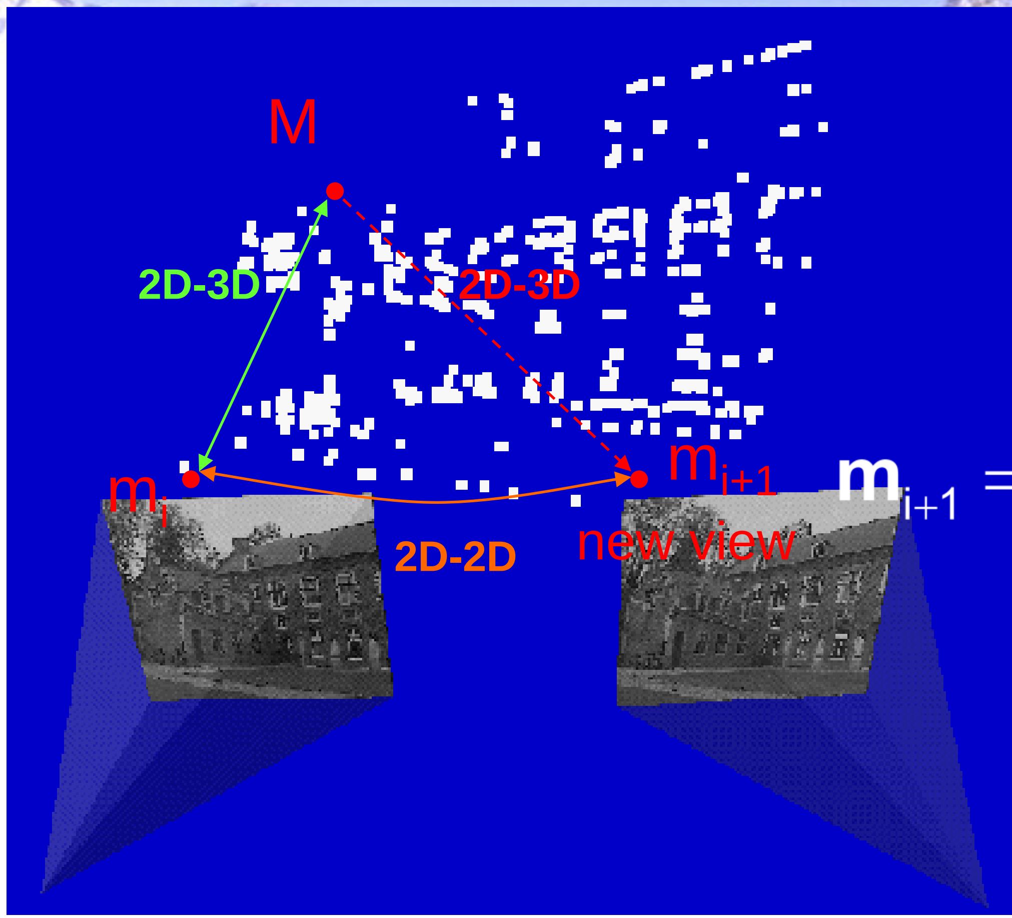
(a) Affine reconstruction: compute π_∞

$$H = \begin{bmatrix} I \mid 0 \\ \pi_\infty \end{bmatrix}$$

(b) Metric reconstruction: compute IAC ω

$$H = \begin{bmatrix} A^{-1} & 0 \\ 0 & 1 \end{bmatrix} \quad AA^T = \left(M^T \omega M\right)^{-1}$$

Determine pose towards existing structure



Compute $\mathbf{P}_{i+1}$ using robust approach
Find additional matches using predicted projection
Extend, correct and refine reconstruction

Affine camera factorization
3D structure from many images

The affine projection equations are

$$\begin{bmatrix} x_{ij} \\ y_{ij} \end{bmatrix} = \begin{bmatrix} P_i^x \\ P_i^y \end{bmatrix} \begin{bmatrix} X_j \\ Y_j \\ Z_j \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x_{ij} \\ y_{ij} \\ 1 \end{bmatrix} = \begin{bmatrix} P_i^x \\ P_i^y \\ 0001 \end{bmatrix} \begin{bmatrix} X_j \\ Y_j \\ Z_j \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x_{ij} - P_i^{x4} \\ y_{ij} - P_i^{y4} \end{bmatrix} = \begin{bmatrix} \tilde{x}_{ij} \\ \tilde{y}_{ij} \end{bmatrix} = \begin{bmatrix} \bar{P}_i^x \\ \bar{P}_i^y \end{bmatrix} \begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix}$$

Orthographic factorization

(Tomasi Kanade'92)

The ortographic projection equations are

$$\bar{\mathrm{m}}_{ij} = \bar{\mathbf{P}}_i \bar{\mathrm{M}}_j, i = 1,\ldots,m, \; j = 1,\ldots,n$$

where

$$\bar{\mathrm{m}}_{ij} = \begin{bmatrix} \tilde{x}_{ij} \\ \tilde{y}_{ij} \end{bmatrix}, \bar{\mathbf{P}}_i = \begin{bmatrix} \bar{P}_i^x \\ \bar{P}_i^y \end{bmatrix}, \bar{\mathrm{M}}_j = \begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix}$$

All equations can be collected for all *i* and *j*

$$\bar{\mathbf{m}} = \bar{\mathbf{P}} \bar{\mathbf{M}}$$

where

$$\bar{\mathbf{m}} = \begin{bmatrix} \bar{\mathrm{m}}_{11} & \bar{\mathrm{m}}_{12} & \cdots & \bar{\mathrm{m}}_{1n} \\ \bar{\mathrm{m}}_{21} & \bar{\mathrm{m}}_{22} & \cdots & \bar{\mathrm{m}}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{\mathrm{m}}_{m1} & \bar{\mathrm{m}}_{m2} & \cdots & \bar{\mathrm{m}}_{mn} \end{bmatrix}, \quad \bar{\mathbf{P}} = \begin{bmatrix} \bar{\mathbf{P}}_1 \\ \bar{\mathbf{P}}_2 \\ \vdots \\ \bar{\mathbf{P}}_m \end{bmatrix}, \bar{\mathbf{M}} = [\mathrm{M}_1, \mathrm{M}_2, \ldots, \mathrm{M}_n]$$

Note that **P** and **M** are resp. 2*m*x3 and 3x*n* matrices and therefore the rank of **m** is at most 3

Orthographic factorization

(Tomasi Kanade'92)

Factorize **m** through singular value decomposition

$$\bar{\mathbf{m}} = \mathbf{U}\Sigma\mathbf{V}^{\mathsf{T}}$$

An affine reconstruction is obtained as follows

$$\tilde{\mathbf{P}} = \mathbf{U}, \tilde{\mathbf{M}} = \Sigma\mathbf{V}^{\mathsf{T}}$$

Closest rank-3 approximation yields MLE!

$$\min \left\| \begin{bmatrix} \bar{\mathrm{m}}_{11} & \bar{\mathrm{m}}_{12} & \cdots & \bar{\mathrm{m}}_{1n} \\ \bar{\mathrm{m}}_{21} & \bar{\mathrm{m}}_{22} & \cdots & \bar{\mathrm{m}}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{\mathrm{m}}_{m1} & \bar{\mathrm{m}}_{m2} & \cdots & \bar{\mathrm{m}}_{mn} \end{bmatrix} - \begin{bmatrix} \bar{\mathbf{P}}_1 \\ \bar{\mathbf{P}}_2 \\ \vdots \\ \bar{\mathbf{P}}_m \end{bmatrix} [\mathrm{M}_1, \mathrm{M}_2, \ldots, \mathrm{M}_n] \right\|$$

Orthographic factorization

(Tomasi Kanade'92)

Factorize **m** through singular value decomposition

$$\bar{\mathbf{m}} = \mathbf{U}\Sigma\mathbf{V}^T$$

An affine reconstruction is obtained as follows

$$\tilde{\mathbf{P}} = \mathbf{U}, \tilde{\mathbf{M}} = \Sigma\mathbf{V}^T$$

A metric reconstruction is obtained as follows

$$\bar{\mathbf{P}} = \tilde{\mathbf{P}}\mathbf{Q}^{-1}, \bar{\mathbf{M}} = \mathbf{Q}\tilde{\mathbf{M}}$$

Where A is computed from

$$\bar{P}_i^x \bar{P}_i^{xT} = \tilde{P}_i^x \mathbf{Q}^{-1}\mathbf{Q}^{-T} \tilde{P}_i^{xT} = 1$$

$$\bar{P}_i^y \bar{P}_i^{yT} = \tilde{P}_i^y \mathbf{Q}^{-1}\mathbf{Q}^{-T} \tilde{P}_i^{yT} = 1$$

$$\bar{P}_i^x \bar{P}_i^{yT} = \tilde{P}_i^x \mathbf{Q}^{-1}\mathbf{Q}^{-T} \tilde{P}_i^{yT} = 0$$

3 linear equations per view on symmetric matrix **C** (6DOF)

Q can be obtained from **C** through Cholesky factorisation and inversion

Weak perspective factorization

[D. Weinshall]

- Weak perspective camera $M = \begin{bmatrix} s\mathbf{i} \\ s\mathbf{j} \end{bmatrix}$

- Affine ambiguity $\hat{W} = \hat{M}QQ^{-1}\hat{X} = (\hat{M}Q)(Q^{-1}\hat{X})$

- Metric constraints

$$s\hat{\mathbf{i}}^T QQ^T s\hat{\mathbf{i}} = s\hat{\mathbf{j}}^T QQ^T s\hat{\mathbf{j}} = s^2$$

$$s\hat{\mathbf{i}}^T QQ^T s\hat{\mathbf{j}} = 0$$

Extract motion parameters

– Eliminate scale

– Compute direction of camera axis k = i x j

– parameterize rotation with Euler angles

Full perspective factorization

The camera equations

$$\lambda_{ij} \mathrm{m}_{ij} = \mathbf{P}_i \mathrm{M}_j, i = 1, \ldots, m, j = 1, \ldots, n$$

for a fixed image *i* can be written in matrix form as

$$\mathbf{m}_i \Lambda_i = \mathbf{P}_i \mathbf{M}$$

where

$$\mathbf{m}_i = [m_{i1}, m_{i2}, \ldots, m_{im}], \quad \mathbf{M} = [\mathrm{M}_1, \mathrm{M}_2, \ldots, \mathrm{M}_m]$$

$$\Lambda_i = \mathrm{diag}(\lambda_{i1}, \lambda_{i2}, \ldots, \lambda_{im})$$

Perspective factorization

All equations can be collected for all *i* as

$$\mathbf{m} = \mathbf{P}\,\mathbf{M}$$

where

$$\mathbf{m} = \begin{bmatrix} \mathbf{m}_1\Lambda_1 \\ \mathbf{m}_2\Lambda_2 \\ \cdots \\ \mathbf{m}_n\Lambda_n \end{bmatrix}, \quad \mathbf{P} = \begin{bmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \cdots \\ \mathbf{P}_m \end{bmatrix}$$

In these formulas m are known, but Λ_i,**P** and **M** are unknown

Observe that **PM** is a product of a 3mx4 matrix and a 4xn matrix, i.e. it is a rank 4 matrix

Perspective factorization algorithm

Assume that Λ_i are known, then **PM** is known.

Use the singular value decomposition

$$\mathbf{PM} = U \Sigma V^T$$

In the noise-free case

$$S = \mathrm{diag}(\sigma_1, \sigma_2, \sigma_3, \sigma_4, 0, \ldots, 0)$$

and a reconstruction can be obtained by setting:

P=the first four columns of $U\Sigma$.

M=the first four rows of V.

Iterative perspective factorization

When Λ_i are unknown the following algorithm can be used:

1. Set λ_{ij}=1 (affine approximation).

2. Factorize **PM** and obtain an estimate of **P** and **M**. If σ_5 is sufficiently small then STOP.

3. Use **m**, **P** and **M** to estimate Λ_i from the camera equations (linearly) $\mathbf{m}_i \Lambda_i = \mathbf{P}_i \mathbf{M}$

4. Goto 2.

In general the algorithm minimizes the *proximity measure* $P(\Lambda,\mathbf{P},\mathbf{M})=\sigma_5$

Note that structure and motion recovered up to an arbitrary projective transformation

N-view geometry
Affine factorization
(HZ Ch 17, 18)

[Tomasi &Kanade '92]

- Affine camera

$P_\infty = [M \mid \mathbf{t}]$ M 2x3 matrix; t 2D vector

- Projection $\begin{pmatrix} x \\ y \end{pmatrix} = M \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \mathbf{t}$

- *n* points, *m* views: measurement matrix $\tilde{\mathbf{x}} = \mathbf{x} - \mathbf{t}$

$$W = \begin{bmatrix} \tilde{\mathbf{x}}_1^1 & \dots & \tilde{\mathbf{x}}_n^1 \\ \vdots & \ddots & \vdots \\ \tilde{\mathbf{x}}_1^m & \dots & \tilde{\mathbf{x}}_n^m \end{bmatrix} = \begin{bmatrix} M^1 \\ \vdots \\ M^m \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 & \cdots & \mathbf{X}_n \end{bmatrix}$$

W: Rank 3

$$W = UDV^T$$

$$\hat{W} = U_{2m\times 3} D_{3\times 3} V_{n\times 3}^T = \hat{M}\hat{X}$$

Assuming isotropic zero-mean Gaussian noise, factorization achieves ML affine reconstruction.

Projective factorization
Homogeneous coord &scale factors

[Sturm & Triggs'96][Heyden '97]

- Measurement matrix

$$W = \begin{bmatrix} \lambda_1^1 \mathbf{x}_1^1 & \cdots & \lambda_n^1 \mathbf{x}_n^1 \\ \vdots & \ddots & \vdots \\ \lambda_1^m \mathbf{x}_1^m & \cdots & \lambda_n^m \mathbf{x}_n^m \end{bmatrix} = \begin{bmatrix} P^1 \\ \vdots \\ P^m \end{bmatrix} [\mathbf{X}_1 \quad \cdots \quad \mathbf{X}_n]$$

3mxn matrix

Rank 4

- Known projective depth λ_j^i

$$W = UDV^T$$

$$\hat{W} = U_{2m\times 4} D_{4\times 4} V_{n\times 4}^T = \hat{P}\hat{X}$$

  – Projective ambiguity

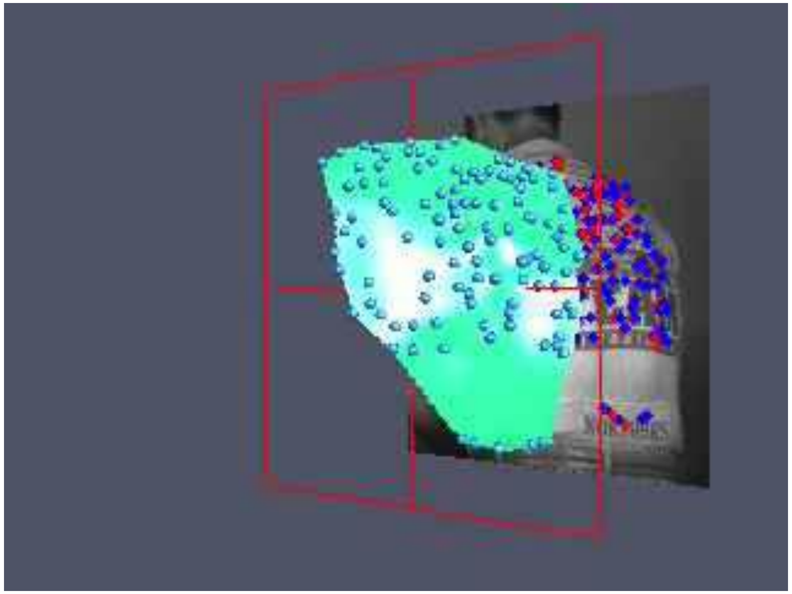
- Iterative algorithm
  – Reconstruct with $\lambda_j^i = 1$
  – Reestimate depth λ_j^i and iterate

Further Factorization work

Factorization with uncertainty

(Irani & Anandan, IJCV'02)

Factorization for dynamic scenes



(Costeira and Kanade '94)

(Bregler et al. 2000,
Brand 2001)

Summary: 2view Reconstuction

<u>Objective</u>

Given two uncalibrated images compute $(P_M, P'_M, \{X_{Mi}\})$
(i.e. within similarity of original scene and cameras)

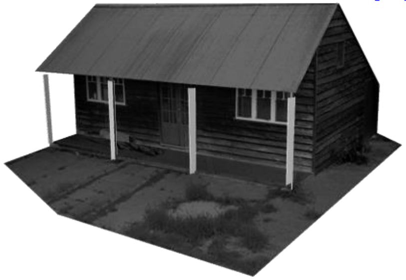
<u>Algorithm</u>

(i) Compute projective reconstruction $(P, P', \{X_i\})$
 (a) Compute F from $x_i \leftrightarrow x'_i$
 (b) Compute P,P' from F
 (c) Triangulate X_i from $x_i \leftrightarrow x'_i$

(ii) Rectify reconstruction from projective to metric

Direct method: compute H from control points $X_{Ei} = H X_i$

$$P_M = P H^{-1} \quad P'_M = P' H^{-1} \quad X_{Mi} = H X_i$$



Stratified method:

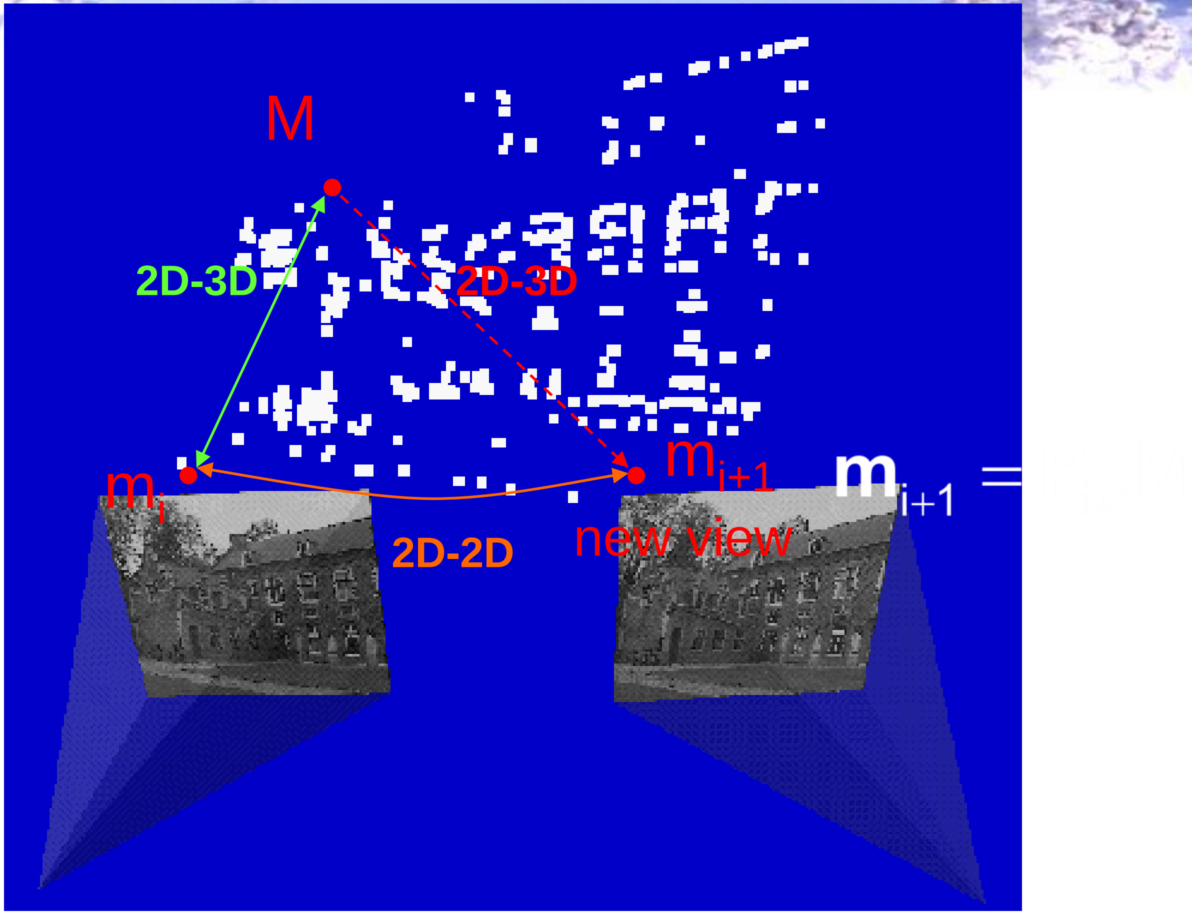
(a) Affine reconstruction: compute π_∞

$$H = \begin{bmatrix} I \mid 0 \\ \pi_\infty \end{bmatrix}$$

(b) Metric reconstruction: compute IAC ω

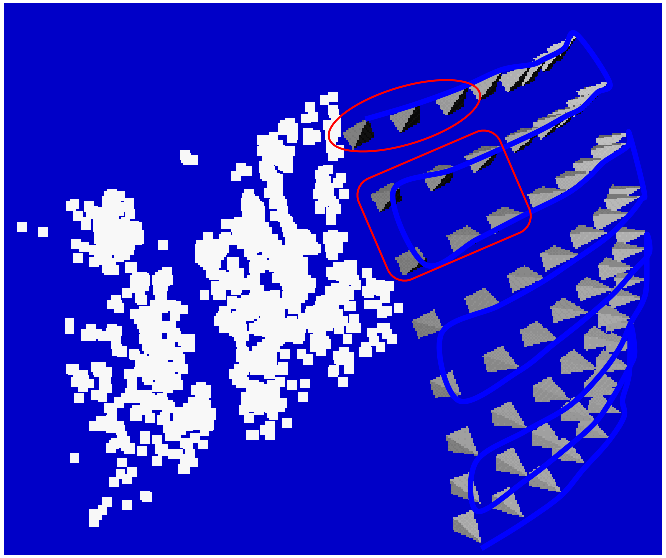
$$H = \begin{bmatrix} A^{-1} & 0 \\ 0 & 1 \end{bmatrix} \quad A A^T = \left(M^T \omega M \right)^{-1}$$

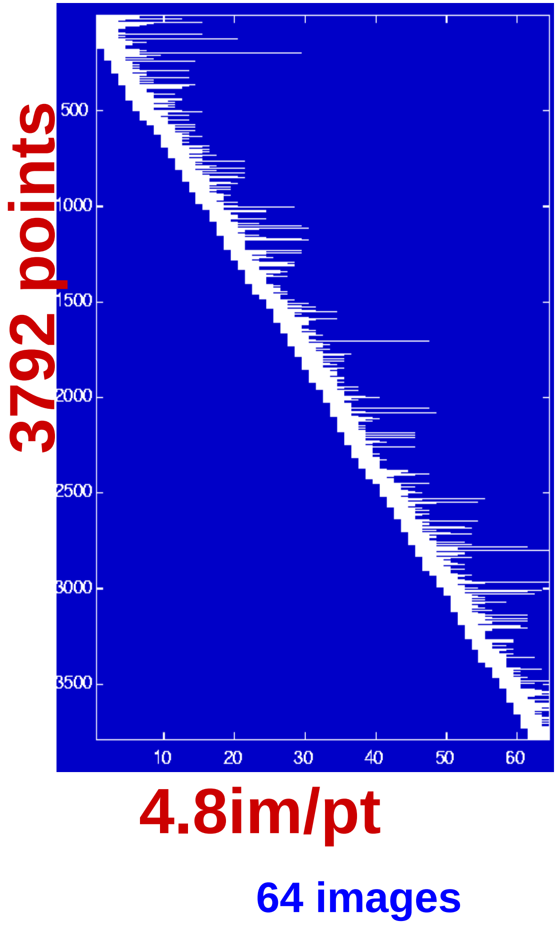
Determine pose towards existing structure



Compute $\mathbf{P}_{i+1}$ using robust approach
Find additional matches using predicted projection
Extend, correct and refine reconstruction

Non-sequential image collections





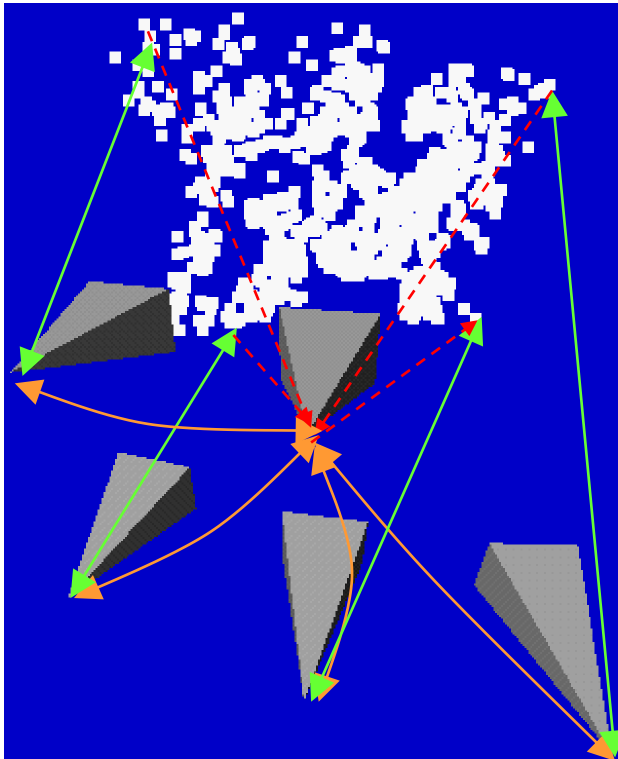
Problem:
Features are lost and reinitialized as new features

Solution:
Match with other *close* views

Relating to more views

For every view *i*

- Extract features
- Compute two view geometry *i*-1/*i* and matches
- Compute pose using robust algorithm
- For all *close* views *k*
  - Compute two view geometry *k*/*i* and matches
  - Infer new 2D-3D matches and add to list
- Refine pose using all 2D-3D matches
- Refine existing structure
- Initialize new structure



Problem:
find *close* views in projective frame

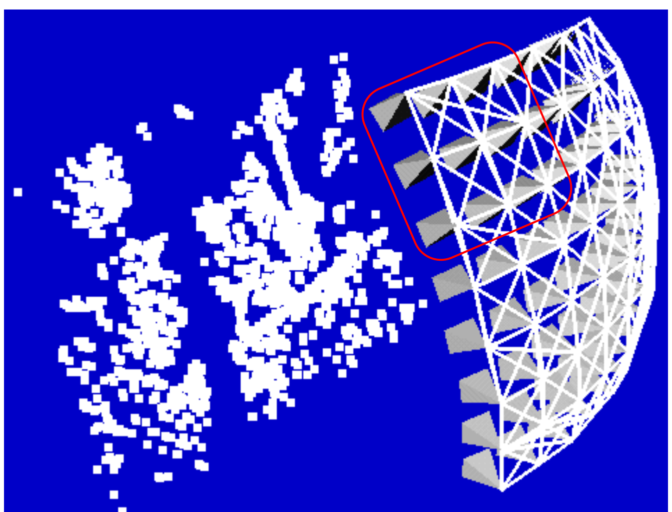
Determining *close* views

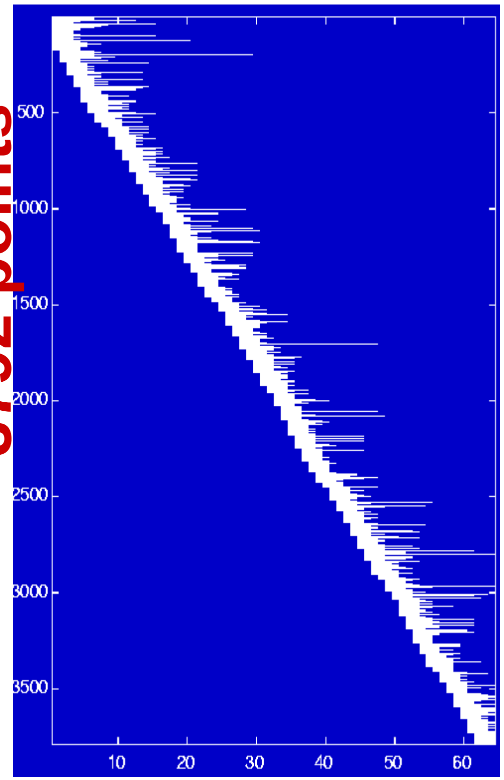
- If viewpoints are *close* then most image changes can be modelled through a *planar homography*

- *Qualitative distance measure* is obtained by looking at the *residual error* on the *best possible planar homography*

Distance = $\min \operatorname{median} D(\mathbf{H}m, m')$

Non-sequential image collections (2)

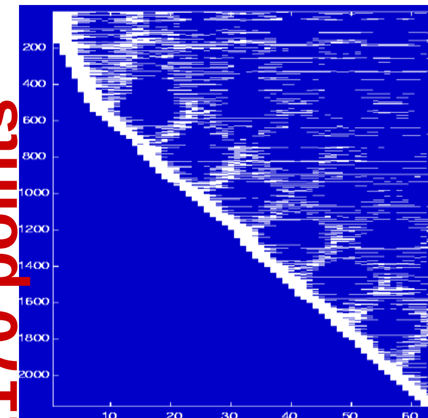




3792 points

4.8im/pt

64 images



2170 points

9.8im/pt

64 images

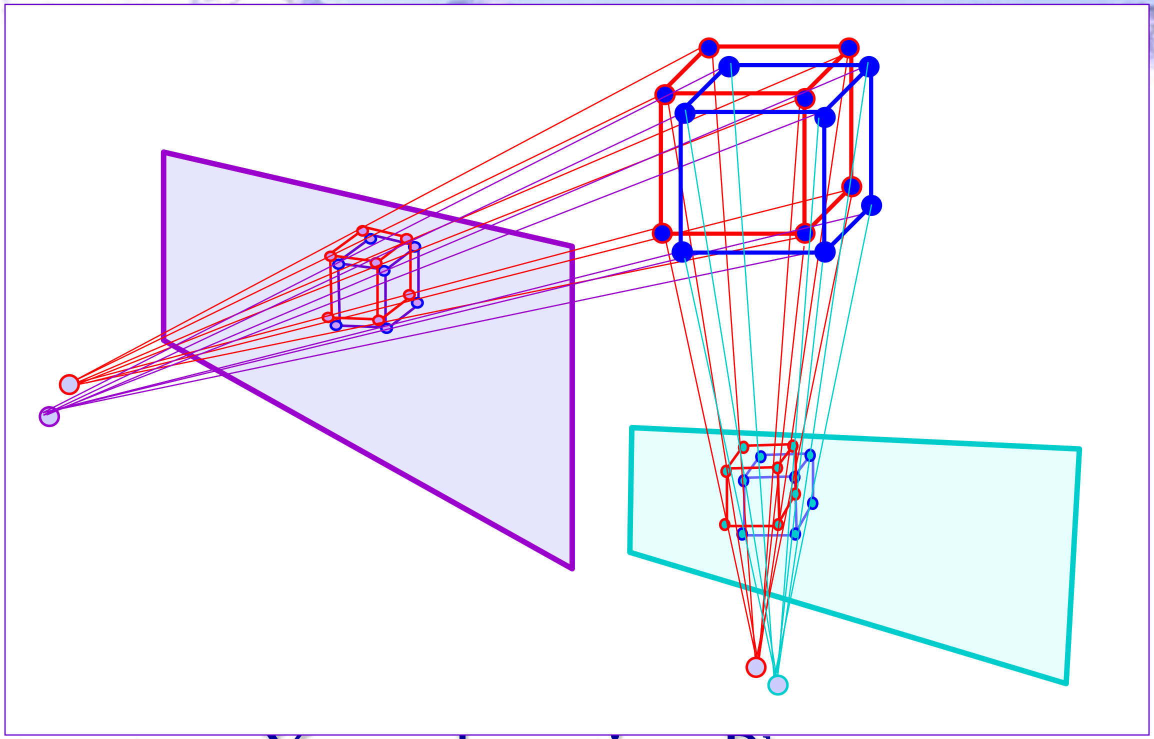
Refining structure and motion

- Minimize reprojection error

$$\min_{\hat{P}_k, \hat{M}_i} \sum_{k=1}^{m} \sum_{i=1}^{n} D\left(m_{ki}, \hat{P}_k \hat{M}_i\right)^2$$

  - Maximum Likelyhood Estimation
    (if error zero-mean Gaussian noise)
  - Huge problem but can be solved efficiently
    (Bundle adjustment)

Refining a captured model: Bundle adjustment



- Refine structure X_j and motion P^i
- Minimize geometric error
- ML solution, assuming noise is Gaussian
- Tolerant to missing data

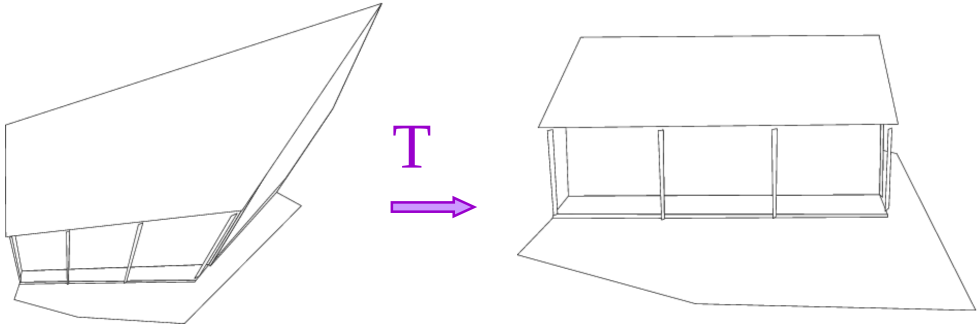
$$\min \sum_{i,j} d(\hat{P}^i \hat{\mathbf{X}}_j, \mathbf{x}^i_j)^2$$

Projective ambiguity and self-calibration

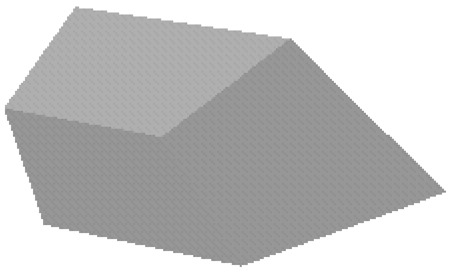
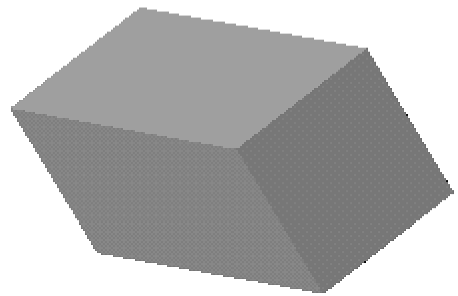
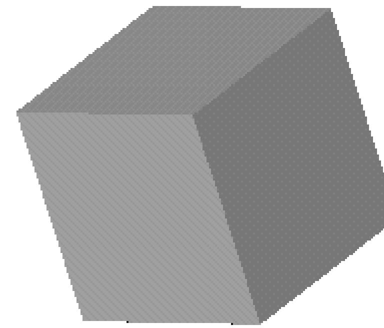
Given an uncalibrated image sequence with corresponding point it is possible to reconstruct the object up to an unknown projective transformation

- Autocalibration (self-calibration): Determine a projective transformation T that upgrades the projective reconstruction to a metric one.

$$m = \mathbf{P}M = (\mathbf{P}\mathbf{T}^{-1})(\mathbf{T}M) = \mathbf{P}'M'$$



Remember: Stratification of geometry

Projective	Affine	Metric
		
15 DOF	**12 DOF**	**7 DOF**
	plane at infinity	***absolute conic***
	parallelism	**angles, rel.dist.**

More general ←

More structure →

Goto slide 78

Constraints ?

- Scene constraints
  - Parallellism, vanishing points, horizon, ...
  - Distances, positions, angles, ...

  Unknown scene → no constraints

- Camera extrinsics constraints
  - Pose, orientation, ...

  Unknown camera motion → no constraints

- Camera intrinsics constraints
  - Focal length, principal point, aspect ratio & skew

  Perspective camera model too general
  → some constraints

- Goto slide 91

Euclidean projection matrix

Factorization of Euclidean projection matrix

$$\mathbf{P} = \mathbf{K}\begin{bmatrix}\mathbf{R}^{\top} & -\mathbf{R}^{\top}\mathbf{t}\end{bmatrix}$$

Intrinsics: $\mathbf{K} = \begin{bmatrix} f_x & s & u_x \\ & f_y & u_y \\ & & 1 \end{bmatrix}$ (camera geometry)

Extrinsics: $(\mathbf{R}, \mathrm{t})$ (camera motion)

Note: every projection matrix can be factorized,

but only meaningful for euclidean projection matrices

Constraints on intrinsic parameters

$$\mathbf{K} = \begin{bmatrix} f_x & s & u_x \\ & f_y & u_y \\ & & 1 \end{bmatrix}$$

Constant

e.g. fixed camera: $\mathbf{K}_1 = \mathbf{K}_2 = \cdots$

Known

e.g. rectangular pixels: $s = 0$

square pixels: $f_x = f_y, s = 0$

principal point known: $(u_x, u_y) = \left(\frac{w}{2}, \frac{h}{2}\right)$

Self-calibration

Upgrade from *projective* structure to *metric* structure using *constraints on intrinsic* camera parameters

- Constant intrinsics
  (Faugeras et al. ECCV´92, Hartley´93, Triggs´97, Pollefeys et al. PAMI´98, ...)
- Some known intrinsics, others varying
  (Heyden&Astrom CVPR´97, Pollefeys et al. ICCV´98,...)
- Constraints on intrincs and restricted motion
  (e.g. pure translation, pure rotation, planar motion)
  (Moons et al.´94, Hartley ´94, Armstrong ECCV´96, ...)

A counting argument

- To go from projective (15DOF) to metric (7DOF) at least 8 constraints are needed
- Minimal sequence length should satisfy

$$n \times (\#known) + (n-1) \times (\#fixed) \geq 8$$

- Independent of algorithm
- Assumes general motion (i.e. not critical)

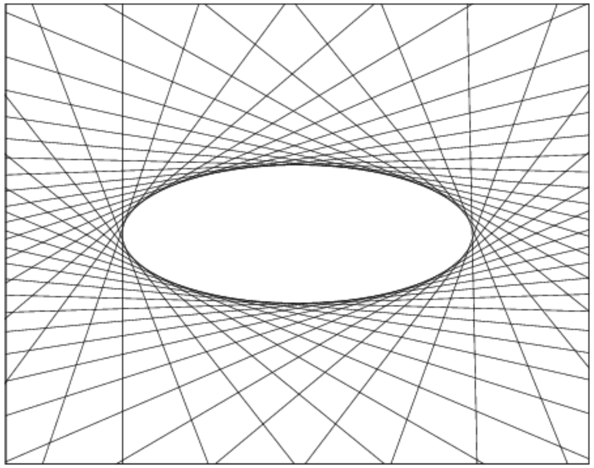
Conics

- Conic:
  - Euclidean geometry: hyperbola, ellipse, parabola & degenerate
  - Projective geometry: equivalent under projective transform
  - Defined by 5 points

$$ax^2 + bxy + cy^2 + dx + ey + f = 0$$

$$\mathbf{x}^T C \mathbf{x} = 0$$

$$C = \begin{bmatrix} a & b/2 & d/2 \\ b/2 & c & e/2 \\ d/2 & e/2 & f \end{bmatrix}$$



- Tangent
- Dual conic C*

$$\mathbf{l} = C\mathbf{x}$$

$$\mathbf{l}^T C^* \mathbf{l} = 0$$

Quadrics

Quadrics: Q

4x4 symmetric matrix

9 DOF (defined by 9 points in general pose)

$$\mathbf{X}^T Q \mathbf{X} = 0$$

- Dual: Q*

Planes tangent to the quadric

$$\boldsymbol{\pi}^T Q^* \boldsymbol{\pi} = 0$$

Summary:
Conics & Quadrics

conics

$$m^T C m = 0 \qquad l^T C^* l = 0$$

$$C^* = C^{-1}$$

quadrics

$$M^T Q M = 0 \qquad \Pi^T Q^* \Pi = 0$$

$$Q^* = Q^{-1}$$

transformations

$$C \mapsto C' \sim H^{-T} C H^{-1}$$

$$C^* \mapsto C^{*\prime} \sim H C^* H^T$$

$$Q \mapsto Q' \sim T^{-T} Q T^{-1}$$

$$Q^* \mapsto Q^{*\prime} \sim T Q^* T^T$$

projection

$$C^* \sim P Q^* P^T$$

The absolute conic

- Absolute conic Ω_∞ is a imaginary circle on $\boldsymbol{\pi}_\infty$
- The absolute dual quadric (rim quadric) Ω^*_∞
- In a metric frame

$$\boldsymbol{\pi}_\infty = (0,0,0,1)$$

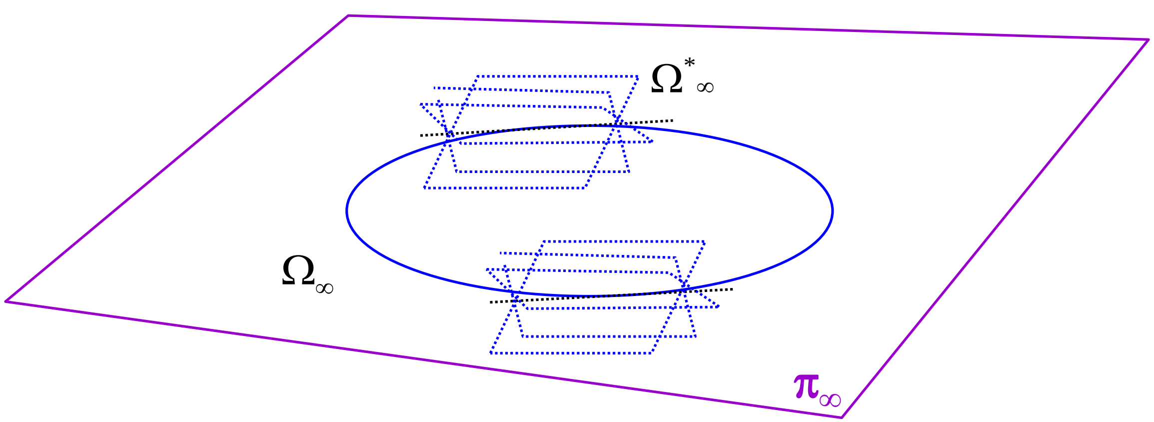
$$\Omega_\infty \quad \left.\begin{array}{r} x_1^2 + x_2^2 + x_3^2 \\ x_4 \end{array}\right\} = 0$$

$$\text{On } \boldsymbol{\pi}_\infty : \; (x_1, x_2, x_3) I (x_1, x_2, x_3)^T = 0$$

$$\Omega^*_\infty = \begin{bmatrix} I & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix}$$

$$\pi^T \Omega^*_\infty \pi = 0$$

Note: is the nullspace of Ω^*_∞



Ω_∞ Fixed under similarity transf.

Self-calibration

- Theoretically formulated by [Faugeras '92]
- 2 basic approaches
  - Stratified: recover $\boldsymbol{\pi}_\infty$ Ω_∞
  - Direct: recover Ω^*_∞ [Triggs'97]
- Constraints:
  - Camera internal constraints
    - Constant parameters [Hartley'94][Mohr'93]
    - Known skew and aspect ratio [Hayden&Åström'98][Pollefeys'98]
  - Scene constraints (angles, ratios of length)
- Choice of H:

Knowing camera K and $\boldsymbol{\pi}_\infty$

$$H = \begin{bmatrix} K & \mathbf{0} \\ -\mathbf{p}^T K & 1 \end{bmatrix}, \quad \pi_\infty = (\mathbf{p}^T, 1)^T$$

Absolute Dual Quadric and Self-calibration

Eliminate extrinsics from equation

$$\mathbf{P} = \mathbf{K}\left[\mathbf{R}^{\mathsf{T}} \quad -\mathbf{R}^{\mathsf{T}}\mathbf{t}\right] \longrightarrow \mathbf{K}\mathbf{R}^{\mathsf{T}}\mathbf{R}\mathbf{K}^{\mathsf{T}} \longrightarrow \mathbf{K}\mathbf{K}^{\mathsf{T}}$$

Equivalent to projection of dual quadric

$$\mathbf{P}\Omega^*_\infty\mathbf{P}^{\mathsf{T}} \propto \mathbf{K}\mathbf{K}^{\mathsf{T}} \qquad \Omega^*_\infty = \mathrm{diag}(1\,1\,1\,0)$$

Abs.Dual Quadric also exists in projective world

$$\mathbf{K}\mathbf{K}^{\mathsf{T}} \propto \mathbf{P}\Omega^*_\infty\mathbf{P}^{\mathsf{T}} \propto (\mathbf{P}\mathbf{T}^{-1})(\mathbf{T}\Omega^*_\infty\mathbf{T}^{\mathsf{T}})(\mathbf{T}^{-\mathsf{T}}\mathbf{P}^{\mathsf{T}})$$
$$\propto \mathbf{P}'\Omega'^*_\infty\mathbf{P}'^{\mathsf{T}}$$

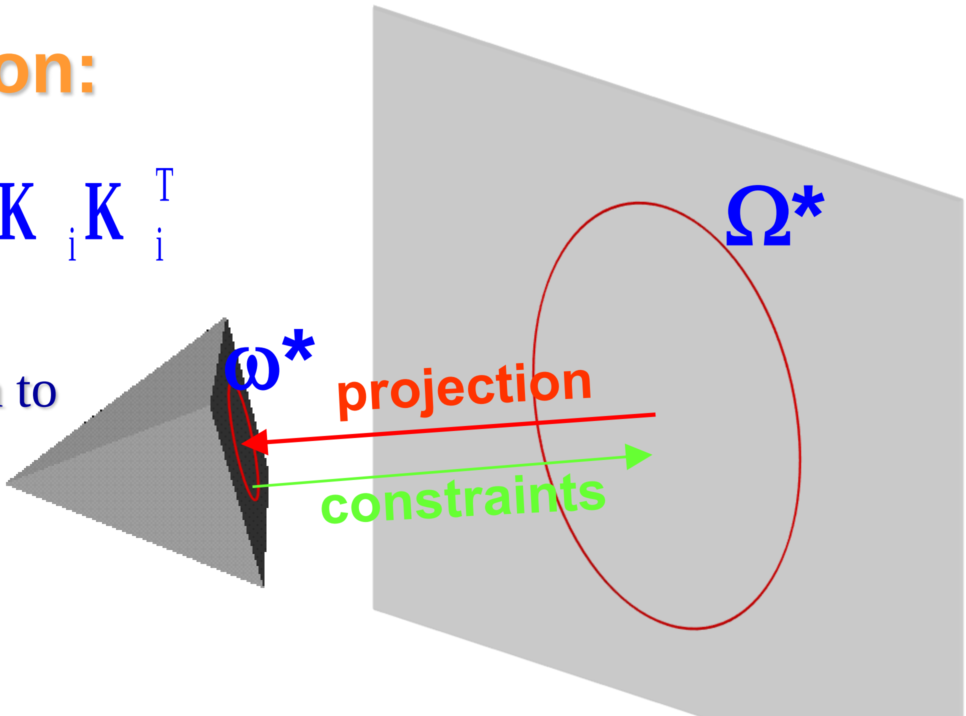
Transforming world so that $\Omega'^*_\infty \rightarrow \Omega^*_\infty$ reduces ambiguity to metric

Absolute Dual Quadric and Self-calibration

Projection equation:

$$\omega_i^* \propto P_i \Omega^* P_i^T \propto K_i K_i^T$$

Translate constraints on **K** through projection equation to constraints on **Ω***



Absolute conic = calibration object which is always present but can only be observed through constraints on the intrinsics

Image of the absolute conic

HZ 7.5.1:

$$x = PX_\infty = KR[I \mid -\tilde{C}]\begin{pmatrix} d \\ 0 \end{pmatrix} = KRd$$

mapping between π_∞ to an image is given by the planar homogaphy x=Hd, with H=KR

image of the absolute conic (IAC) = I

$$\omega = \left(KK^T\right)^{-1} = K^{-T}K^{-1} \qquad \left(C \mapsto H^{-T}CH^{-1}\right)$$

(i) IAC depends only on intrinsics
(ii) angle between two rays $\cos\theta = \frac{x_1^T \omega x_2}{\sqrt{\left(x_1^T \omega x_1\right)\left(x_2^T \omega x_2\right)}}$
(iii) DIAC=ω^*=KK^T
(iv) $\omega \Leftrightarrow$ K (cholesky factorisation)
(v) image of circular points

Constraints on ω^*_∞

$$\omega^*_\infty = \begin{bmatrix} f_x^2 + s^2 + c_x^2 & sf_y + c_x c_y & c_x \\ sf_y + c_x c_y & f_y^2 + c_y^2 & c_y \\ c_x & c_y & 1 \end{bmatrix}$$

condition	constraint	type	#constraints
Zero skew	$\omega^*_{12}\omega^*_{33} = \omega^*_{13}\omega^*_{23}$	quadratic	m
Principal point	$\omega^*_{13} = \omega^*_{23} = 0$	linear	$2m$
Zero skew (& p.p.)	$\omega^*_{12} = 0$	linear	m
Fixed aspect ratio (& p.p.& Skew)	$\omega^*_{11}\omega'^*_{22} = \omega^*_{22}\omega'^*_{11}$	quadratic	m-1
Known aspect ratio (& p.p.& Skew)	$\omega^*_{11} = \omega^*_{22}$	linear	m
Focal length (& p.p. & Skew)	$\omega^*_{33} = \omega^*_{11}$	linear	m

Summary: Self calibration based on the IADC

- Calibrated camera
  - Dual absolute quadric (DAC) $\tilde{I} = diag(1,1,1,0)$
  - Dual image of the absolute conic (DIAC) $\omega^* = KK^T$
- Projective camera
  - DAC $Q^*_\infty = H\tilde{I}H^T$
  - DIAC $\omega^{*i} = P^i Q^*_\infty P^{iT} = K_i K_i^T$
- Autocalibration
  - Determine Ω^*_∞ based on constraints on ω^{*i}
  - Decompose $Q^*_\infty = H\tilde{I}H^T$

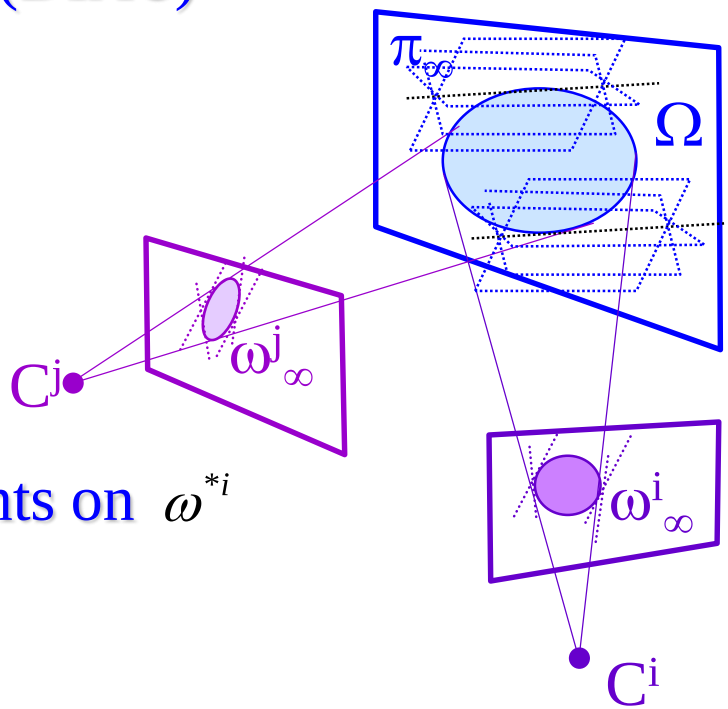
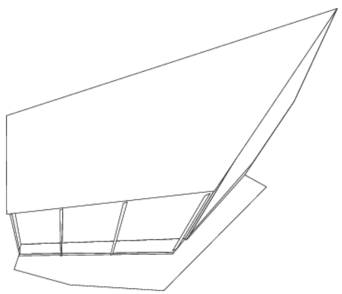
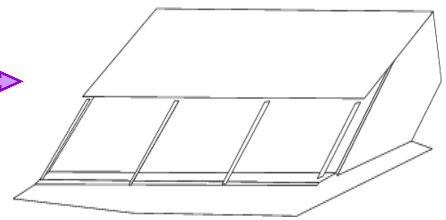
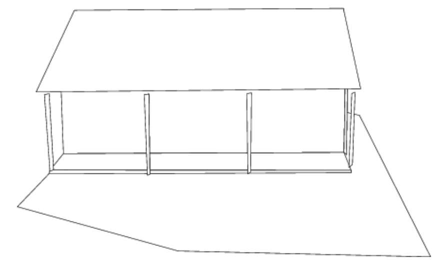


Illustration of self-calibration

Projective Affine Metric







Degenerate configurations

- Pure translation: affine transformation (5 DOF)
- Pure rotation: arbitrary pose for π_∞ (3 DOF)
- Planar motion: scaling axis perpendicular to plane (1DOF)
- Orbital motion: projective distortion along rotation axis (2DOF)

Not unique solution !

A complete modeling system projective

Sequence of frames → scene structure

1. Get corresponding points (tracking).
2. 2,3 view geometry: compute F,T between consecutive frames (recompute correspondences).
3. Initial reconstruction: get an initial structure from a subsequence with big baseline (trilinear tensor, factorization …) and bind more frames/points using resection/intersection.
4. Self-calibration.
5. Bundle adjustment.

A complete modeling system affine

Sequence of frames ⟹ scene structure

1. Get corresponding points (tracking).
2. Affine factorization. (This already computes ML estimate over all frames so no need for bundle adjustment for simple scenes.
3. Self-calibration.
4. If several model segments: Merge, bundle adjust.

Examples – modeling with dynamic texture

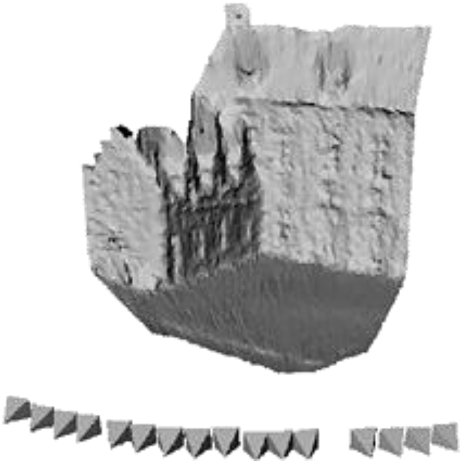
Cobzas,Yerex,Jagersand

Examples: geometric modeling

Debevec and Taylor: Façade

Examples: geometric modeling

Pollefeys: Arenberg Castle



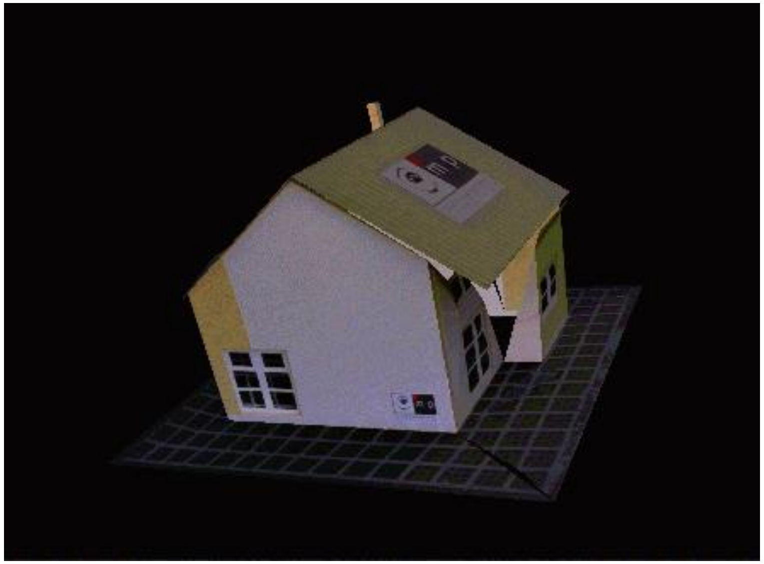
Examples: geometric modeling

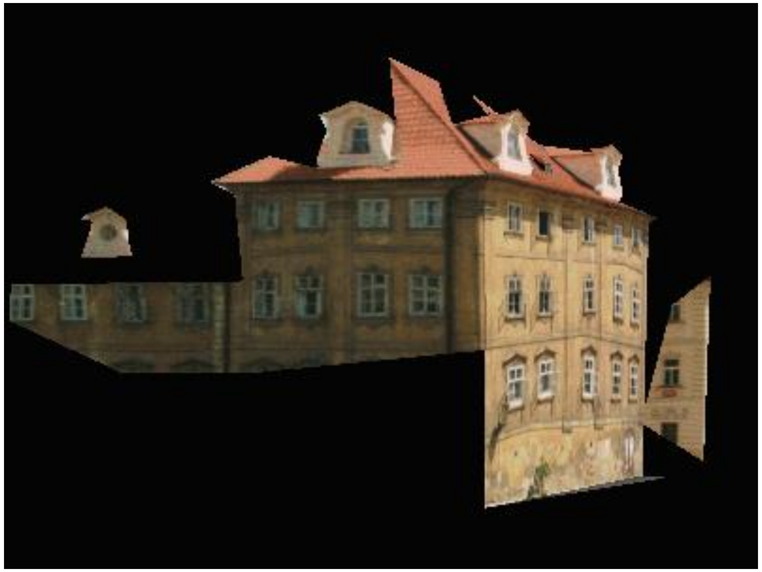
INRIA –VISIRE project

Examples: geometric modeling

CIP Prague –

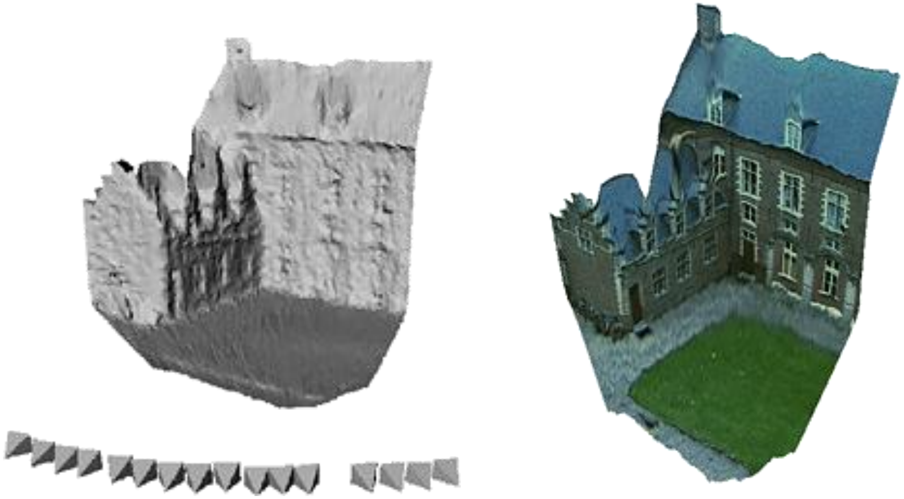
Projective Reconstruction Based on Cake Configuration





Examples: geometric modeling

Pollefeys: Arenberg Castle



Stereo reconstruction

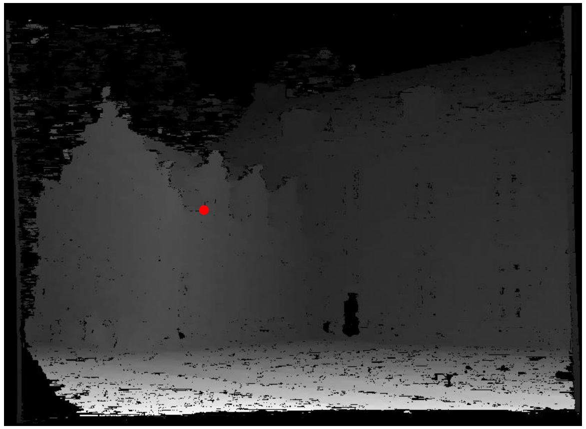
How to go from sparse SFM

…to detailed, model?
Here in the form of
disparity/depth map

Rectified left image I(x,y)

Dense Disparity map D(x,y)

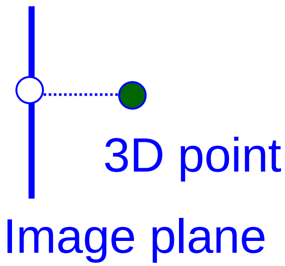
Rectified right image I´(x´,y´)



Many object/surface representation

Image-centered

- Depth/disparity w.r.t. to image plane



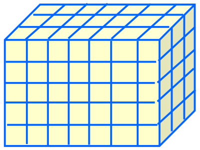
Partial object reconstr.

Limited resolution

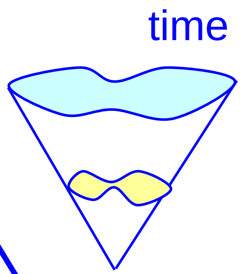
Viewpoint dependent

Object-centered

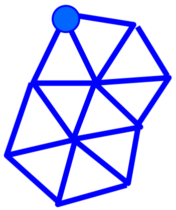
- Voxels



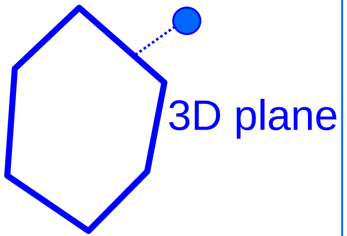
- Level sets (implicit)



- Mesh

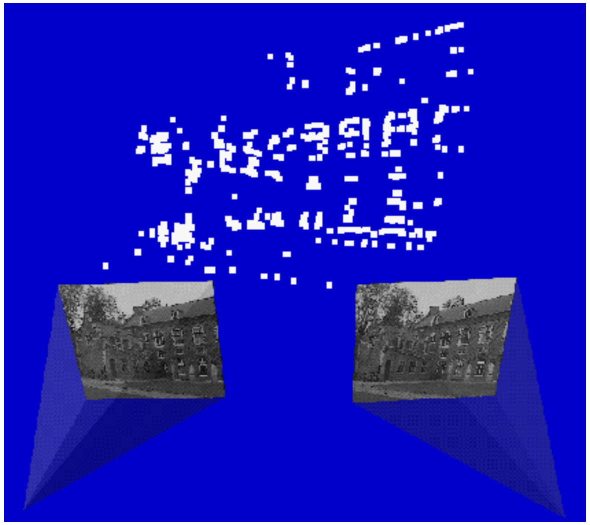


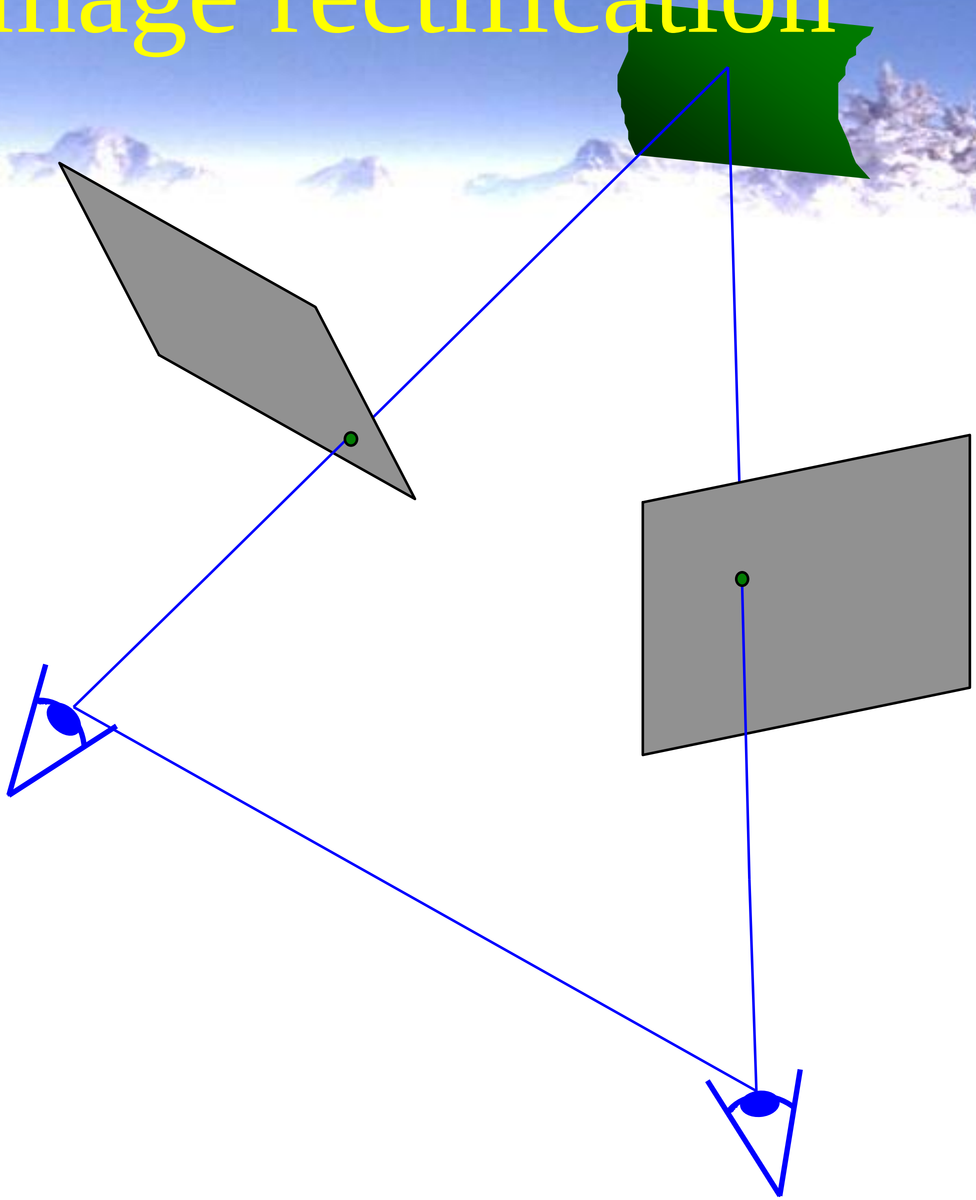
- Depth with respect to a base mesh



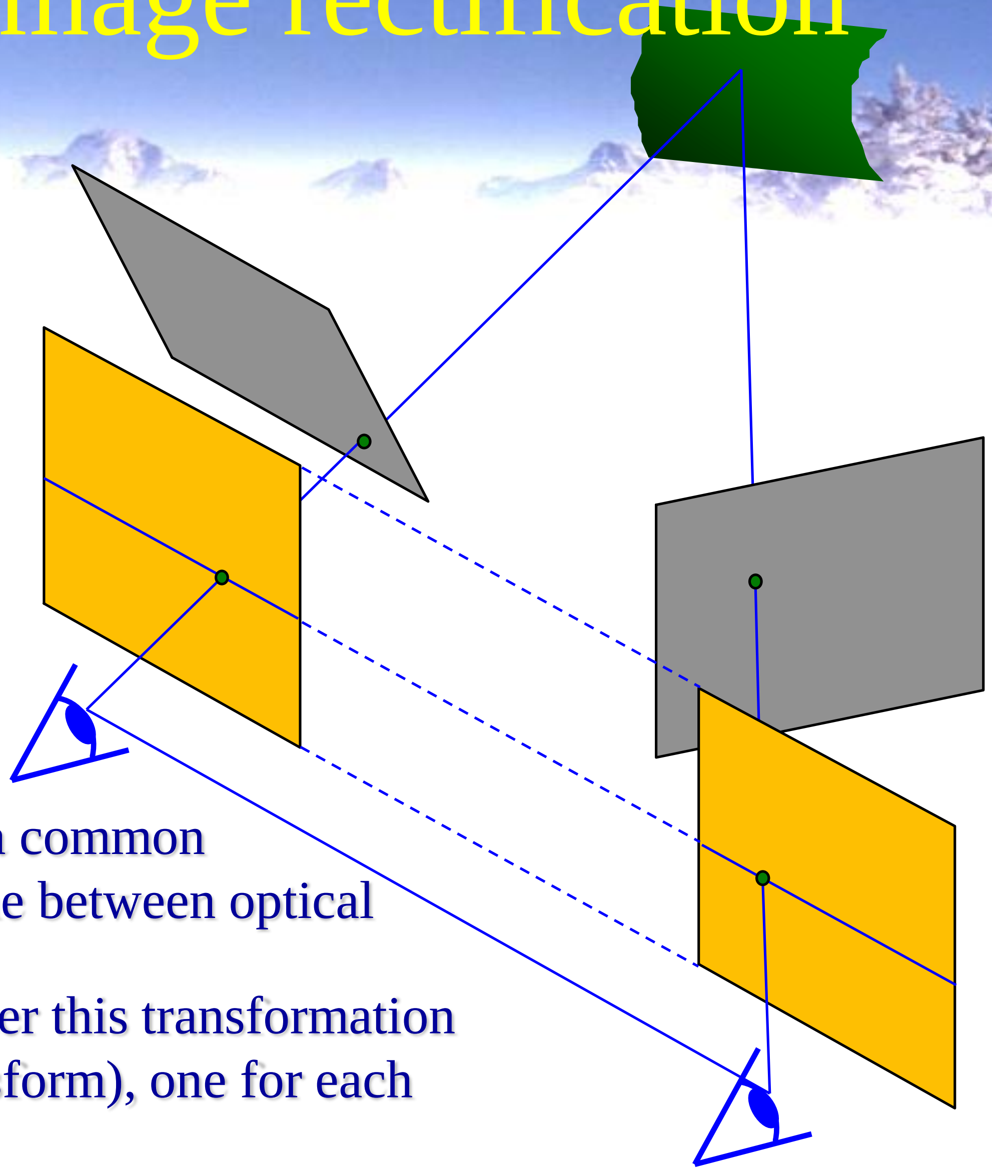
- Local patches

Stereo image rectification





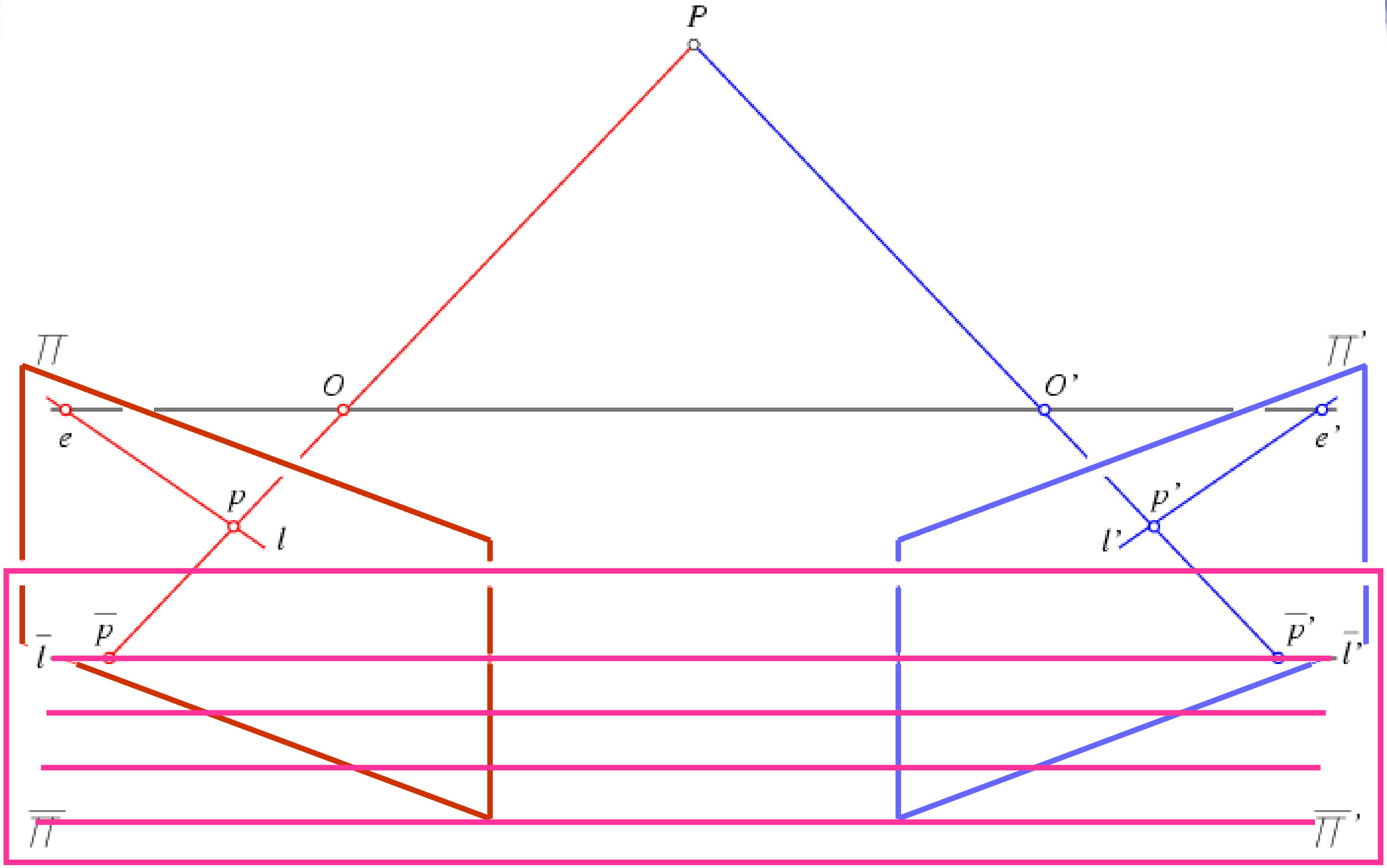
Stereo image rectification



- reproject image planes onto a common
- plane parallel to the line between optical centers
- pixel motion is horizontal after this transformation
- two homographies (3x3 transform), one for each input image reprojection

➢ C. Loop and Z. Zhang. Computing Rectifying Homographies for Stereo Vision. IEEE Conf. Computer Vision and Pattern Recognition, 1999.

Rectification

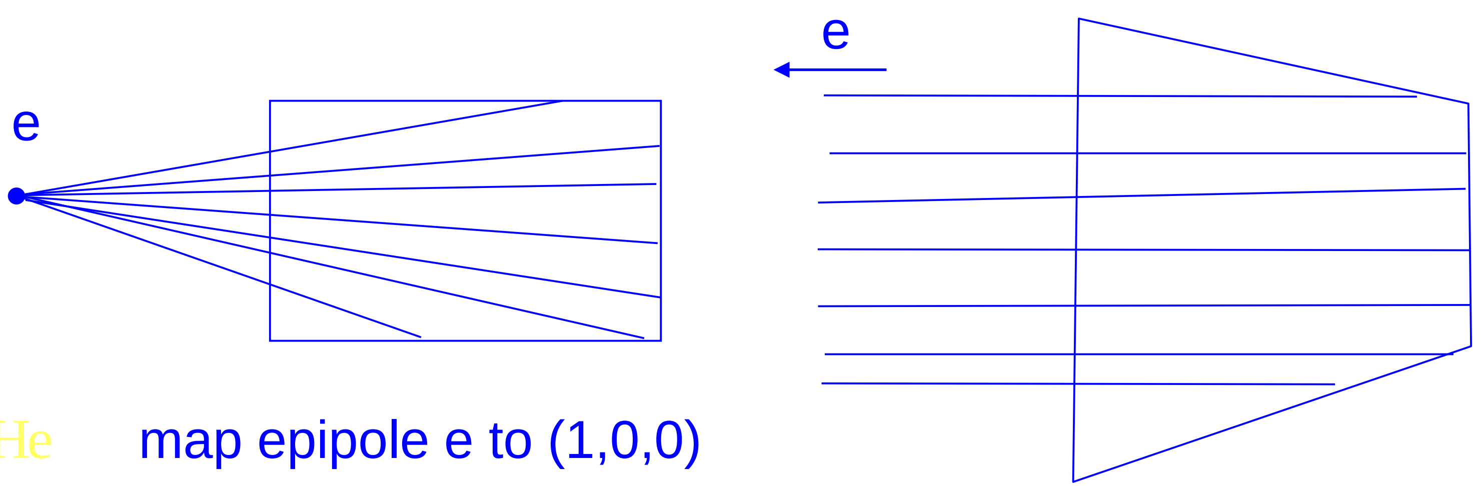


All epipolar lines are parallel in the rectified image plane.

Image rectification through homography warp

simplify stereo matching
by warping the images

Apply projective transformation so that epipolar lines correspond to horizontal scanlines



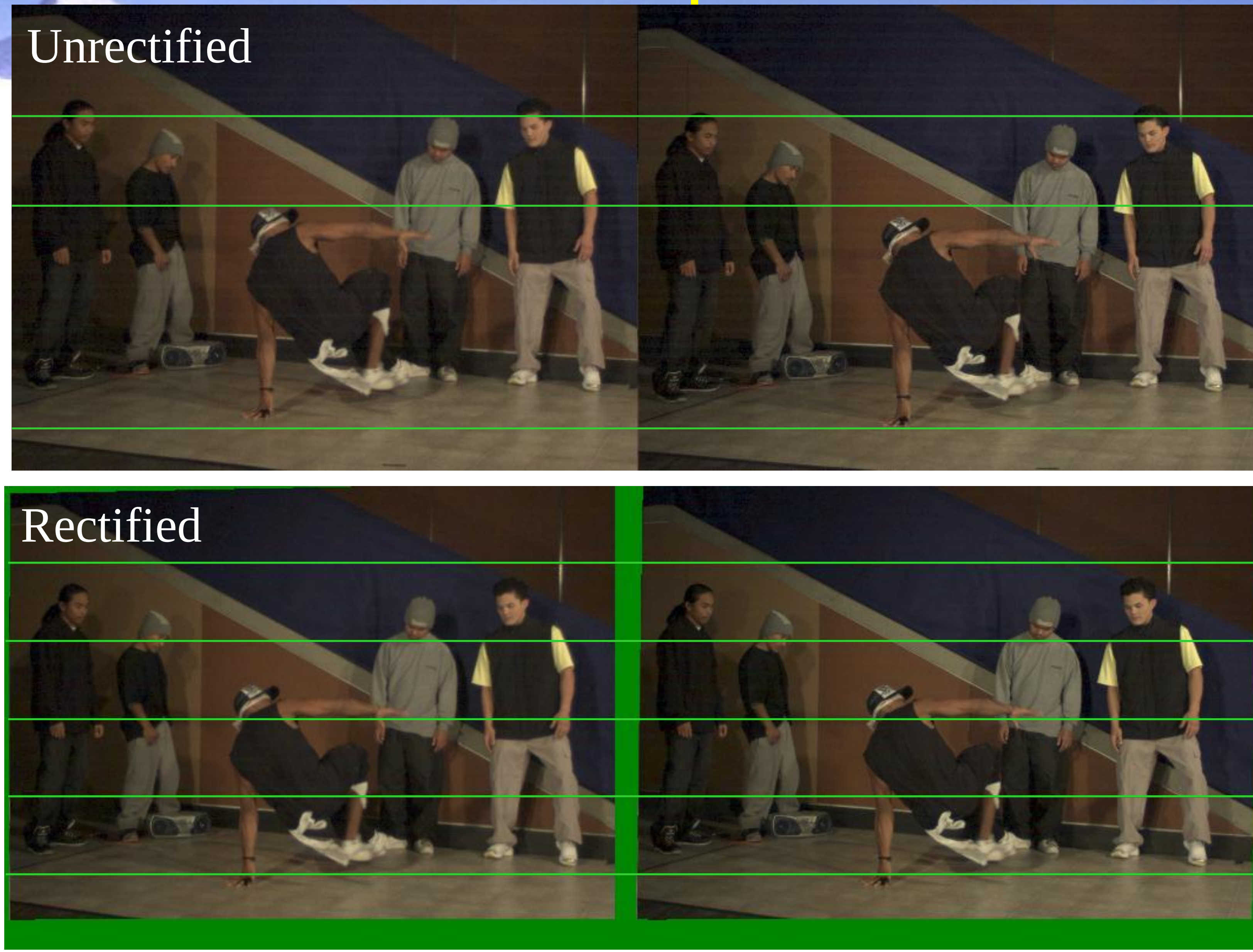
$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = He$$

map epipole e to (1,0,0)

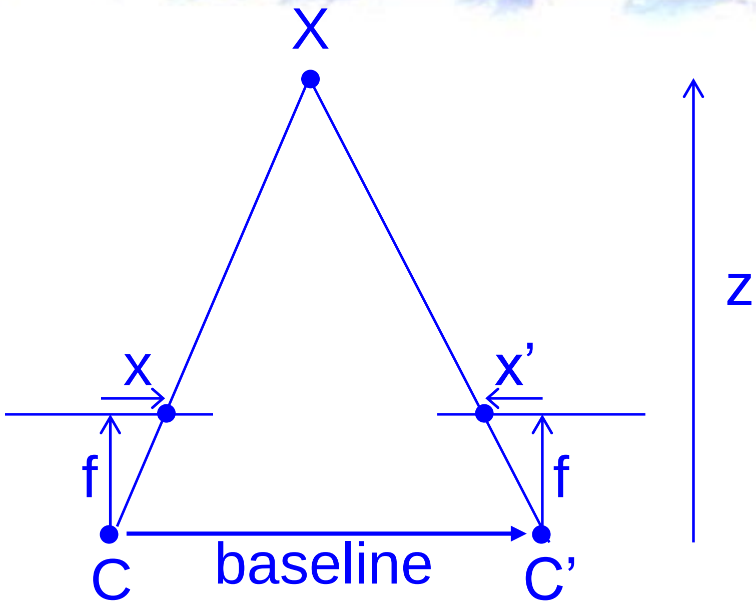
try to minimize image distortion

problem when epipole in (or close to) the image

Example



Depth from disparity

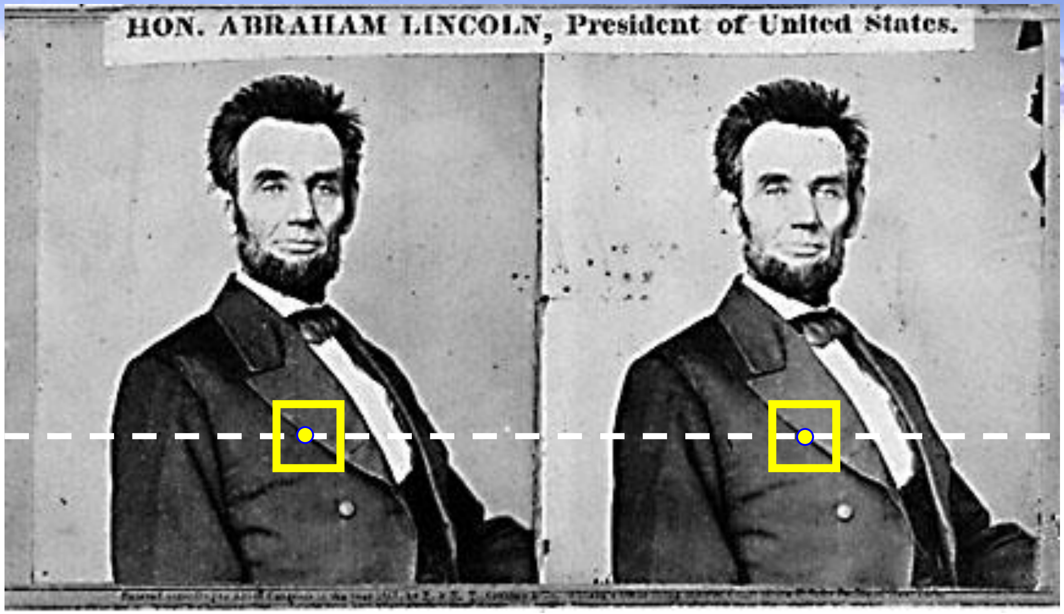


$$disparity = x - x' = \frac{baseline * f}{z}$$

Stereo matching algorithms

- Match Pixels in Conjugate Epipolar Lines
  - Assume brightness constancy
  - This is a tough problem
  - Numerous approaches
    - A good survey and evaluation: http://www.middlebury.edu/stereo/

Your basic stereo algorithm



For each epipolar line

- For each pixel in the left image
  - compare with every pixel on same epipolar line in right image
  - pick pixel with minimum match cost

Improvement: match ***windows***

- This should look familar...

Stereo as energy minimization

- Find disparities *d* that minimize an energy function $E(d)$

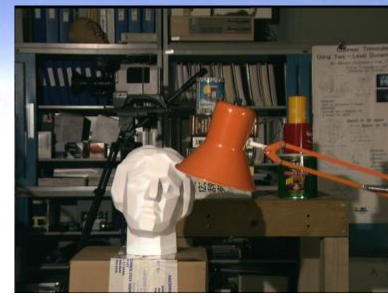
- Simple pixel / window matching

$$E(d) = \sum_{(x,y)\in I} C(x, y, d(x, y))$$

$C(x, y, d(x, y))$ = SSD distance between windows *I*(*x*, *y*) and *J*(*x*, *y* + *d*(*x*,*y*))

Stereo as energy minimization

$I(x, y)$



$J(x, y)$

$C(x, y, d)$; the *disparity space image* (DSI)

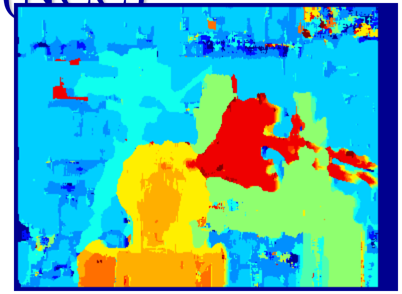
Stereo as energy minimization

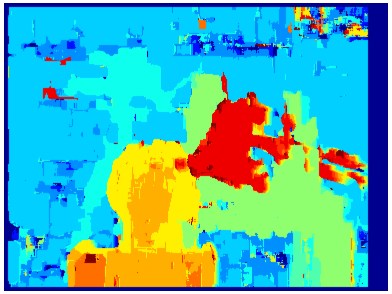
Simple pixel / window matching: choose the minimum of each column in the DSI independently:

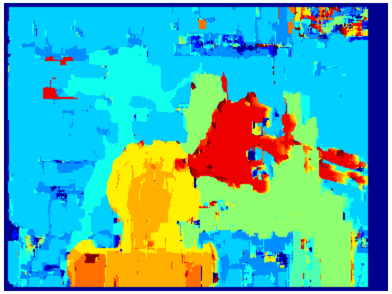
$$d(x, y) = \arg\min_{d'} C(x, y, d')$$

Matching windows

Similarity Measure	Formula
Sum of Absolute Differences (SAD)	$\sum_{(i,j)\in W} \lvert I_1(i,j) - I_2(x+i,y+j) \rvert$
Sum of Squared Differences (SSD)	$\sum_{(i,j)\in W} \left(I_1(i,j) - I_2(x+i,y+j)\right)^2$
Zero-mean SAD	$\sum_{(i,j)\in W} \lvert I_1(i,j) - \bar{I}_1(i,j) - I_2(x+i,y+j) + \bar{I}_2(x+i,y+j) \rvert$
Locally scaled SAD	$\sum_{(i,j)\in W} \lvert I_1(i,j) - \frac{\bar{I}_1(i,j)}{\bar{I}_2(x+i,y+j)} I_2(x+i,y+j) \rvert$
Normalized Cross Correlation (NCC)	$\frac{\sum_{(i,j)\in W} I_1(i,j).I_2(x+i,y+j)}{\sqrt[2]{\sum_{(i,j)\in W} I_1^2(i,j).\sum_{(i,j)\in W} I_2^2(x+i,y+j)}}$


SAD

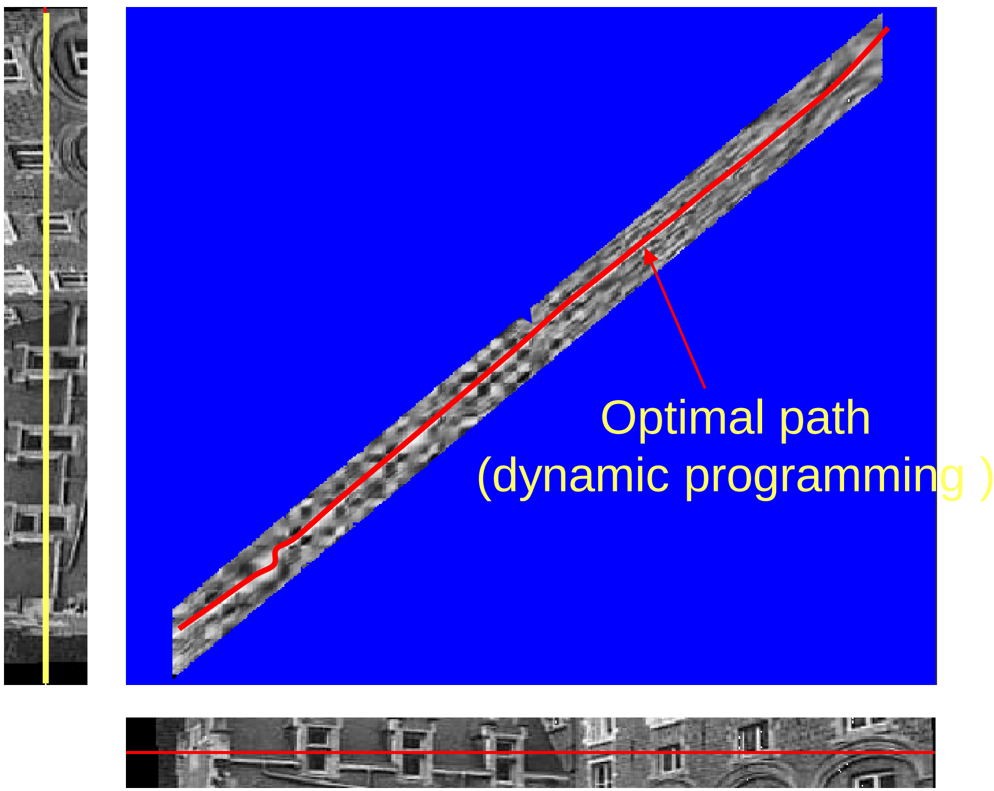

SSD


NCC


Ground truth

http://siddhantahuja.wordpress.com/category/stereo-vision/

Stereo matching



Constraints
- epipolar
- ordering
- uniqueness
- disparity limit
- disparity gradient limit

Trade-off
- Matching cost (data)
- Discontinuities (prior)

(Cox et al. CVGIP’96; Koch’96; Falkenhagen´97;
Van Meerbergen,Vergauwen,Pollefeys,VanGool IJCV‘02)

Disparity map

image I(x,y)

Disparity map D(x,y)

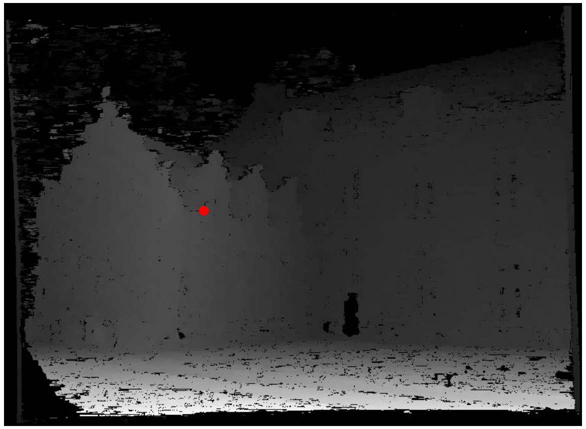


image I´(x´,y´)

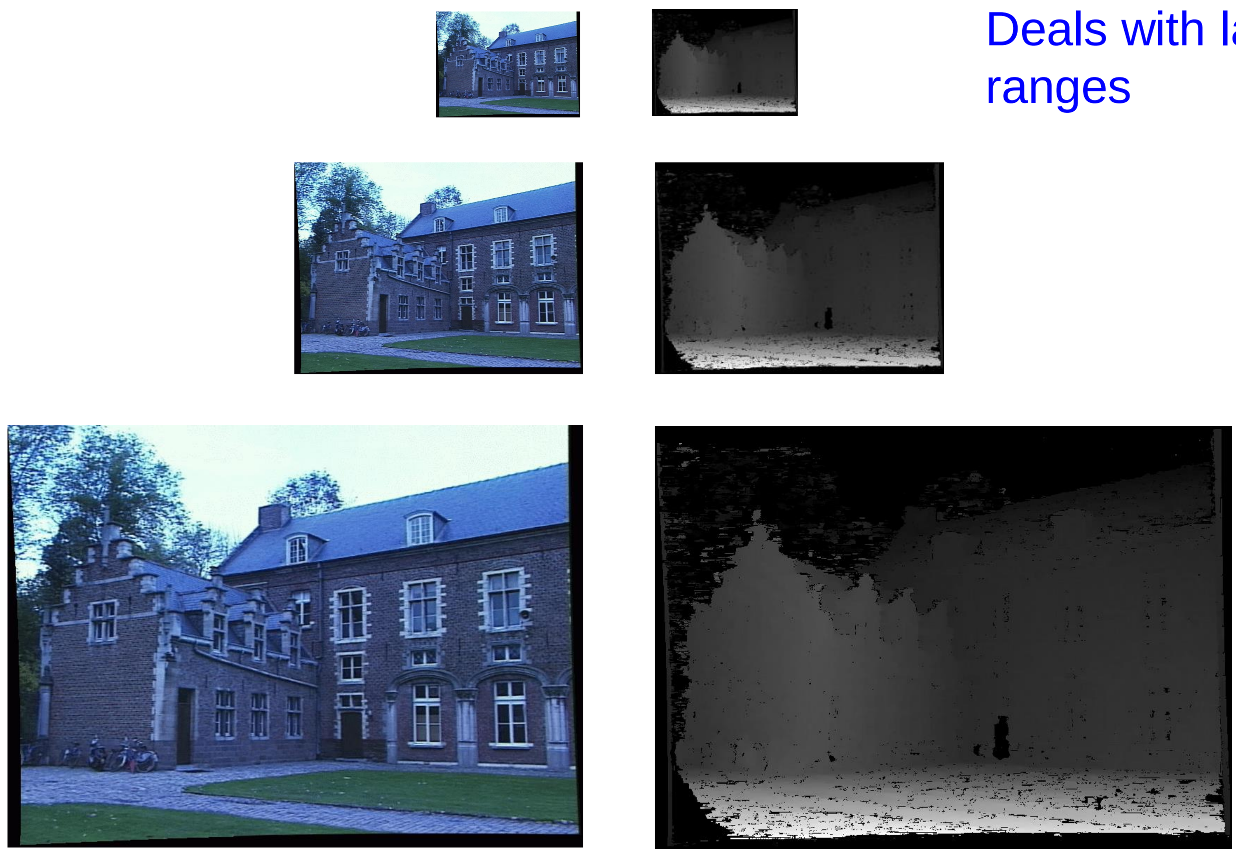
$$(x',y')=(x+D(x,y),y)$$

Hierarchical stereo matching

Allows faster computation

Deals with large disparity ranges

Downsampling (Gaussian pyramid)



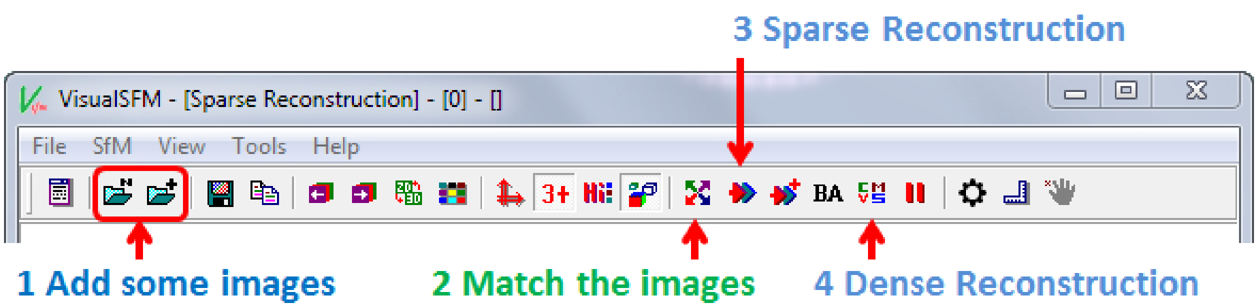
Disparity propagation

(Falkenhagen´97;Van Meerbergen,Vergauwen,Pollefeys,VanGool IJCV‘02)

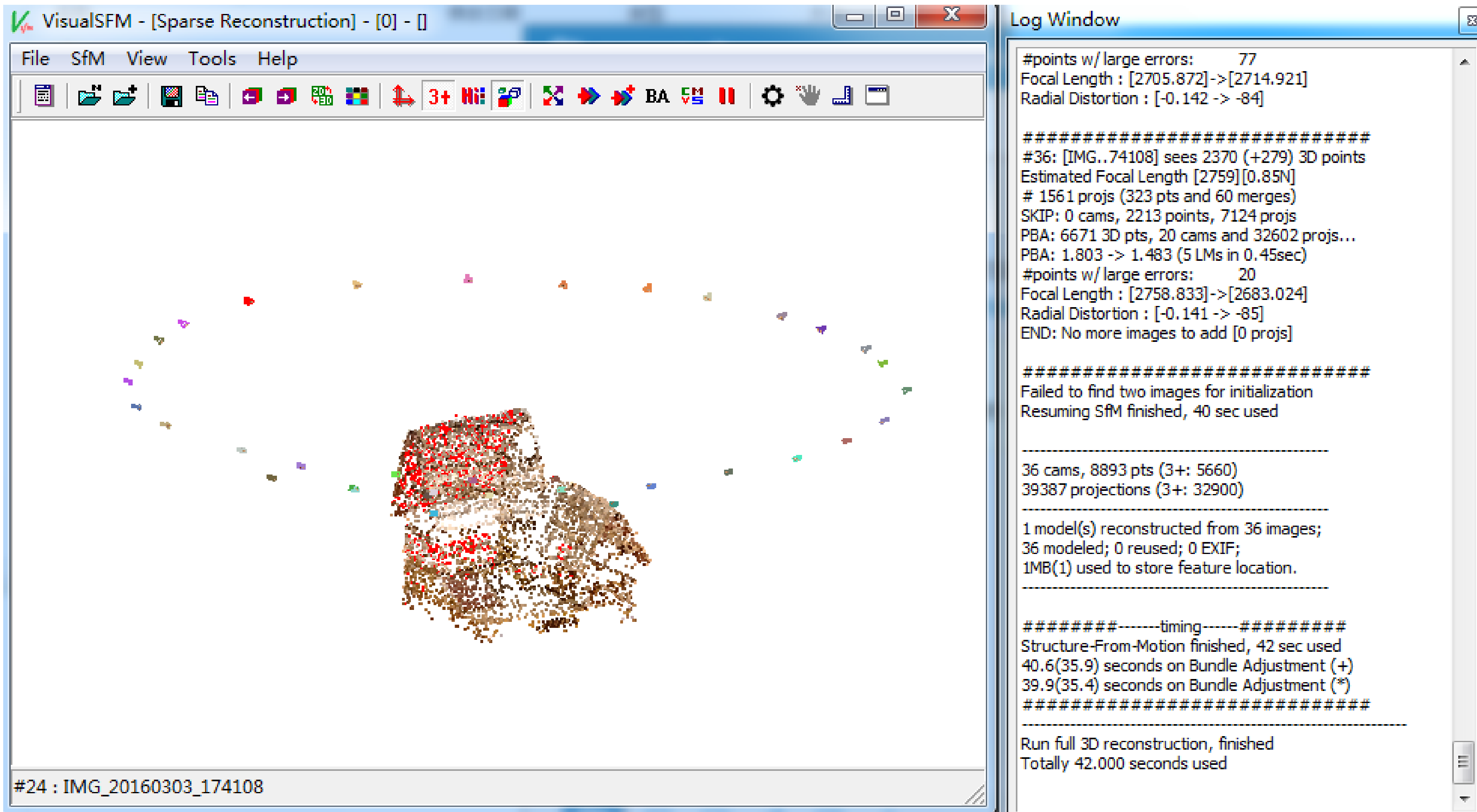
Example: reconstruct image from neighboring images

Many SFM and stereo systems you can try

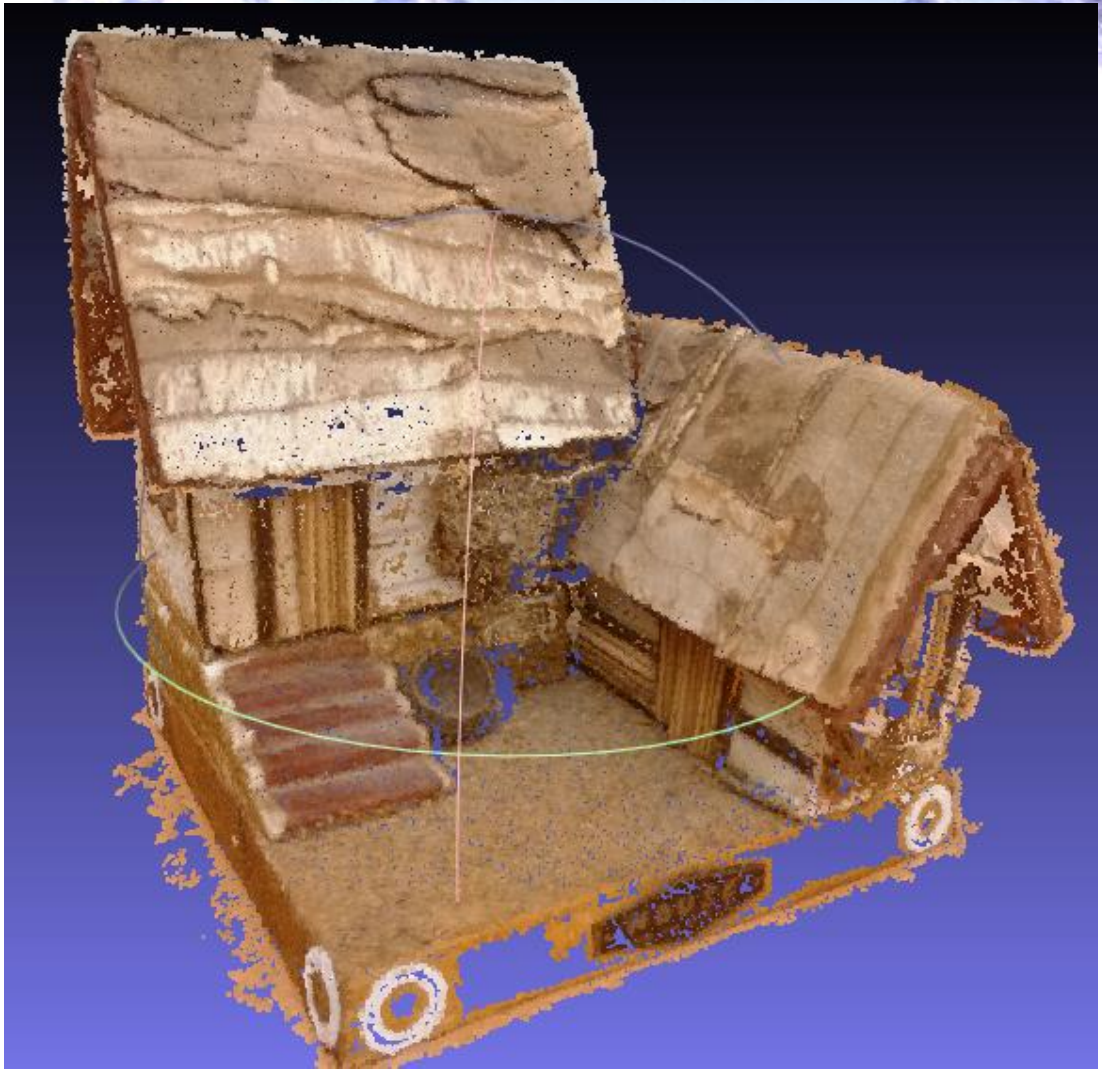
- Microsoft Photosynth: SFM only, on-line
- Arc3D: SFM + Stereo, on-line
- VisualSFM SFM + Stereo, download and install



Visual SFM, House by Bin



Visual SFM, House by Bin



Reconstructing scenes

'Small' scenes (one, few buildings)

- SFM + multi view stereo
- man made scenes: prior on architectural elements
- interactive systems

City scenes (several streets, large area)

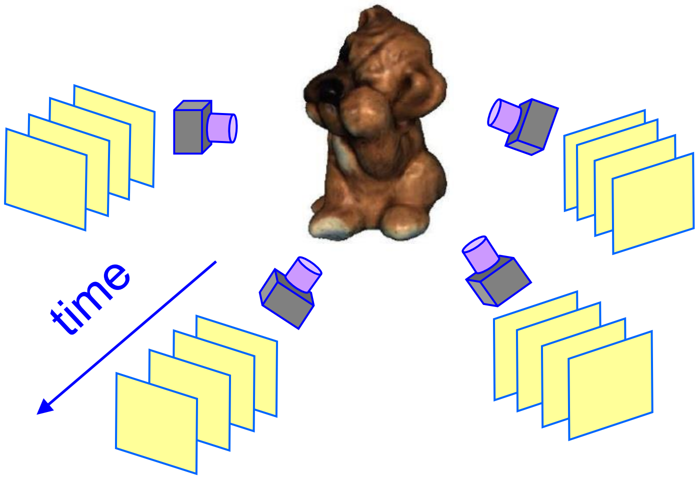
- aerial images
- ground plane, multi cameras

SFM + stereo [+ GPS]

depth map fusions

Large scale (city) modeling

Modeling dynamic scenes



Modeling (large scale) scenes

[Adam Rachmielowski]

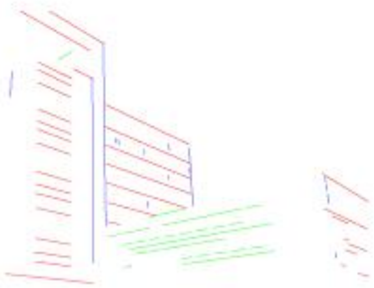
SFM + stereo

Man-made environments :

- straight edges
- family of lines
- vanishing points

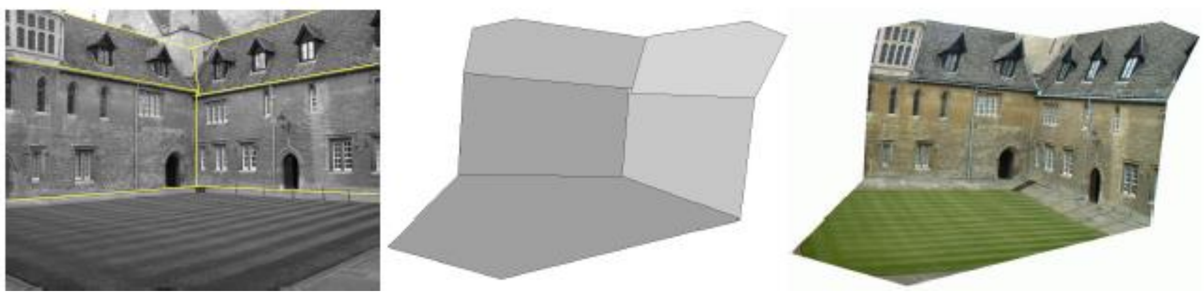
[Dellaert et al 3DPVT06]

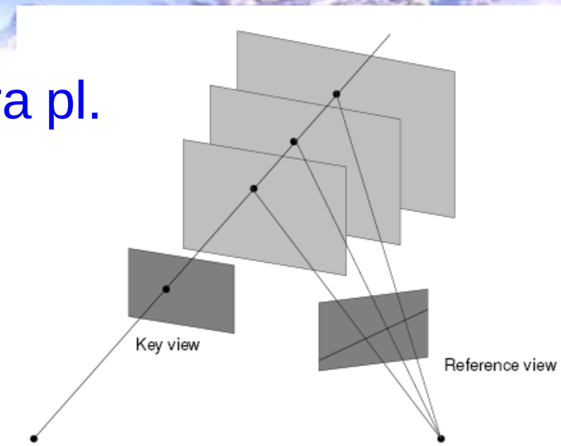
[Zisserman, Werner ECCV02]

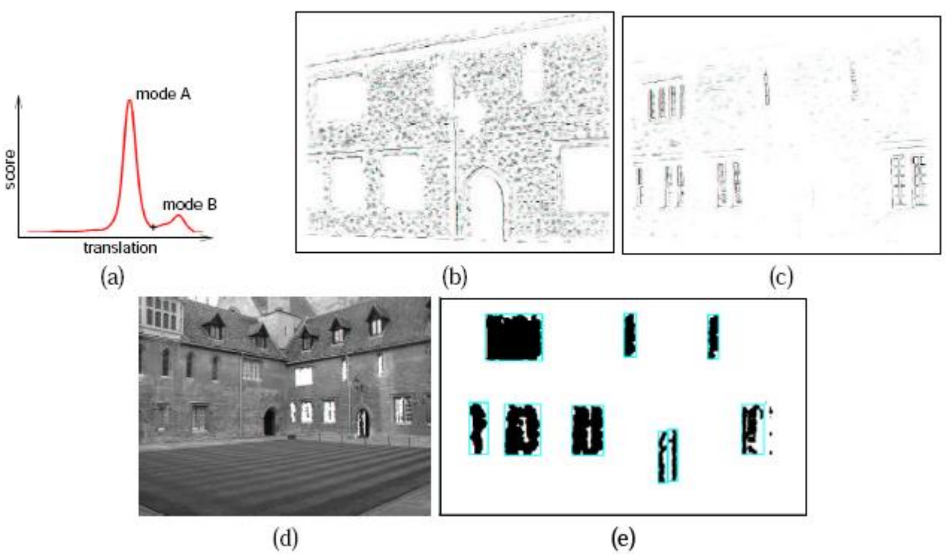


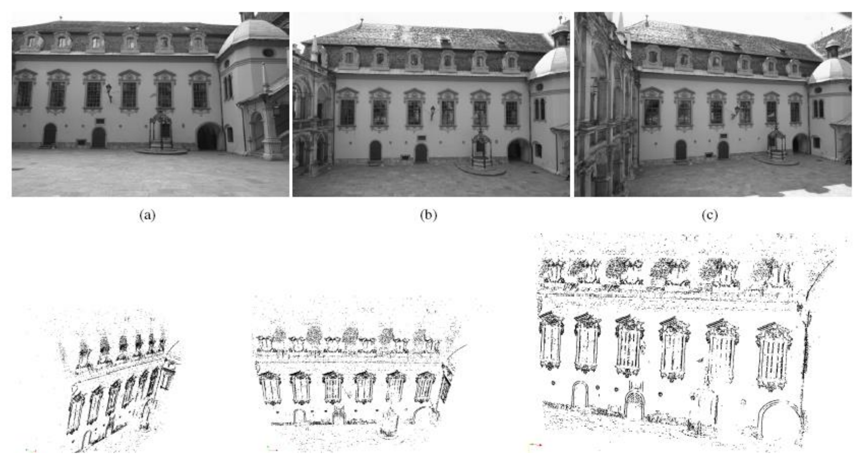
SFM + stereo

- dominant planes
- plane sweep – homog between 3D pl. and camera pl.







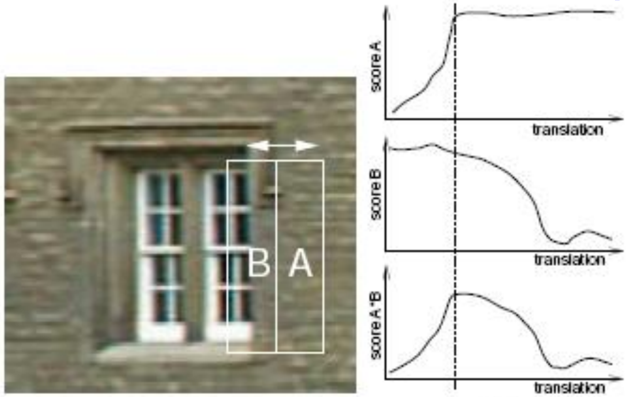


[Zisserman, Werner ECCV02]

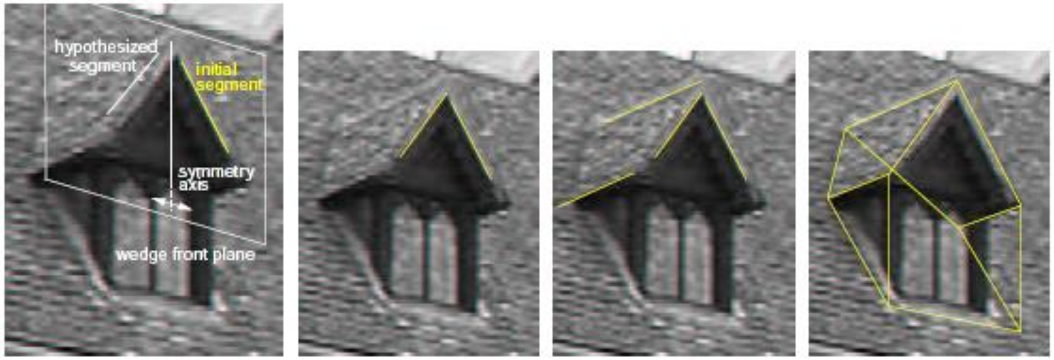
[Bischof et al 3DPVT06]

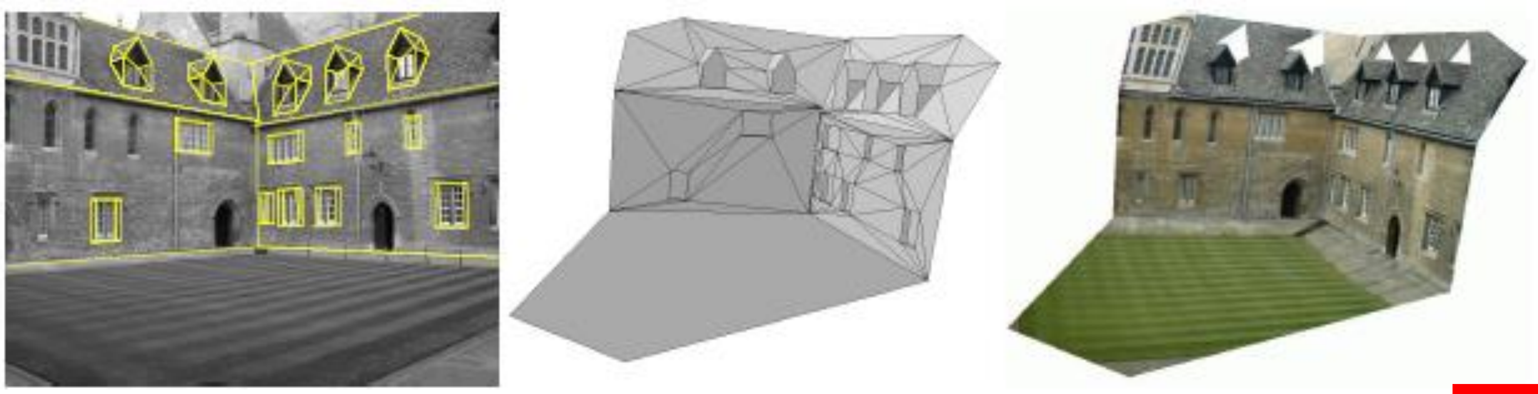
SFM + stereo

- refinement – architectural primitives



(a) (b)

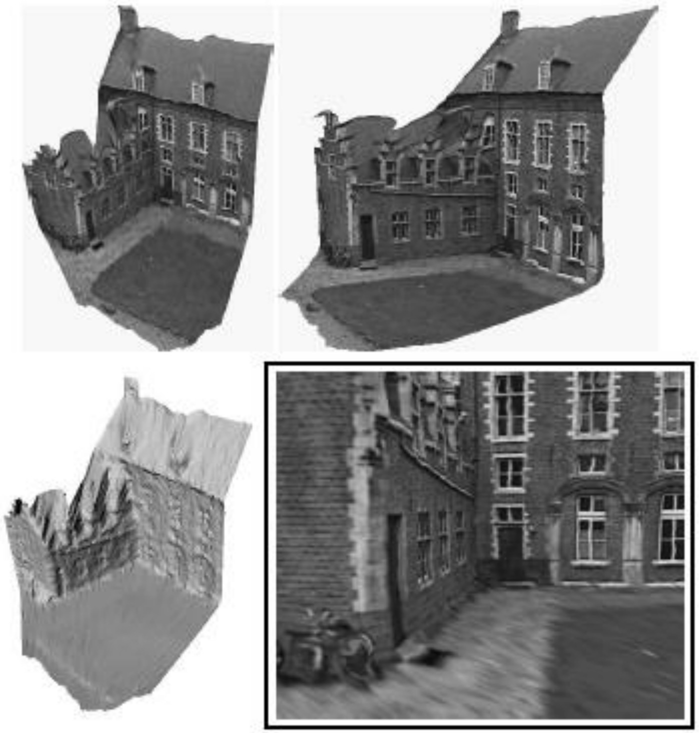




[Zisserman, Werner ECCV02 …]

SFM+stereo

- Refinement – dense stereo



[Pollefeys, Van Gool 98,00,01]

ARC 3D Webservice

A Family of Web Tools for Remote 3D Reconstruction

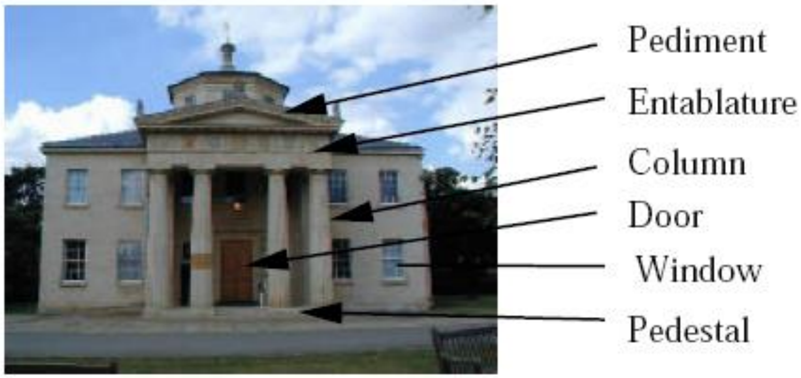
www.arc3d.be

Façade – first system

Based on SFM
(points, lines, stereo)
Some manual modeling
View dependent texture

[Debevec, Taylor et al. Siggraph 96]

Priors on architectural primitives



$$\Pr(\mathbf{M}\boldsymbol{\theta}|\mathbf{DI}) \; \alpha \; \Pr(\mathbf{D}|\mathbf{M}\boldsymbol{\theta}\mathbf{I})\Pr(\mathbf{M}\boldsymbol{\theta}\mathbf{I})$$

$$= \Pr(\mathbf{D}|\mathbf{M}\boldsymbol{\theta}_L\boldsymbol{\theta}_S\boldsymbol{\theta}_T\mathbf{I})\Pr(\boldsymbol{\theta}_T|\boldsymbol{\theta}_L\mathbf{MI})$$
$$\Pr(\boldsymbol{\theta}_S|\boldsymbol{\theta}_L\mathbf{MI})\Pr(\boldsymbol{\theta}_L|\mathbf{MI})$$

prior

θ – parameters for architectural priors

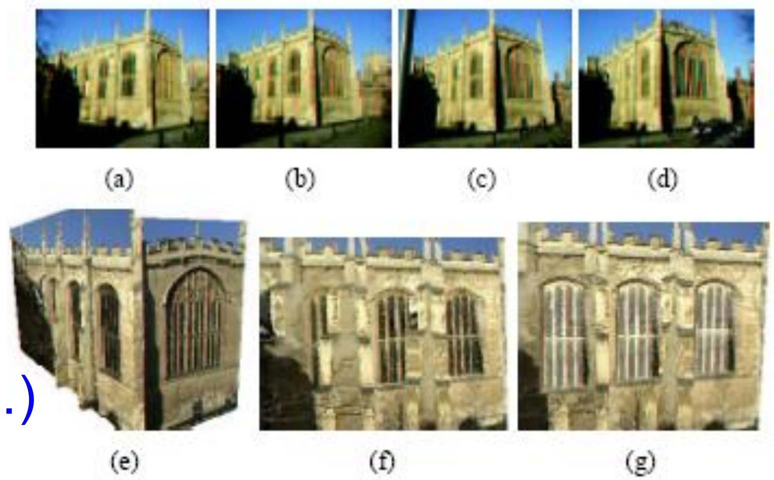
type, shape, texture

M – model

D – data (images)

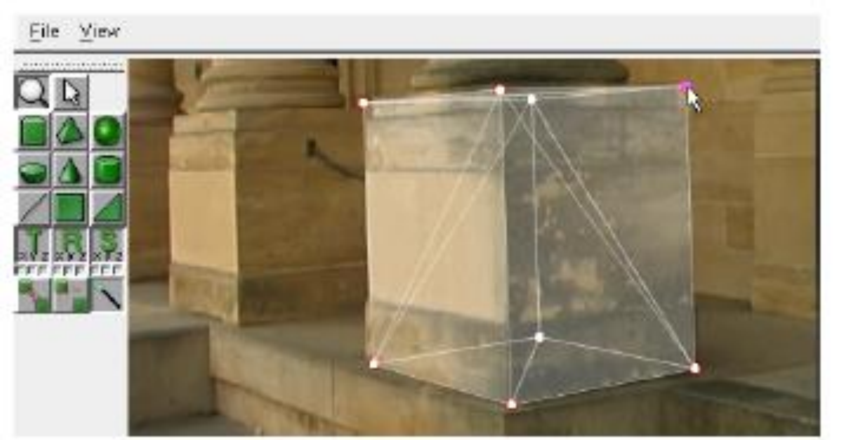
I – reconstructed structures (planes, lines …)

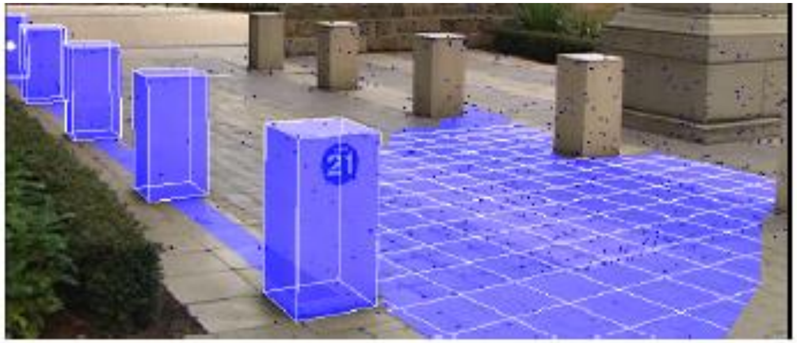
[Cipolla, Torr, … ICCV01]



Occluded windows

Interactive systems





Video, sparse 3D points, user input

$$\Pr(\mathcal{M}|\mathrm{DI}) \propto \Pr(\mathrm{D}|\mathcal{M}\mathrm{I})\Pr(\mathcal{M}|\mathrm{I}).$$

M – model primitives

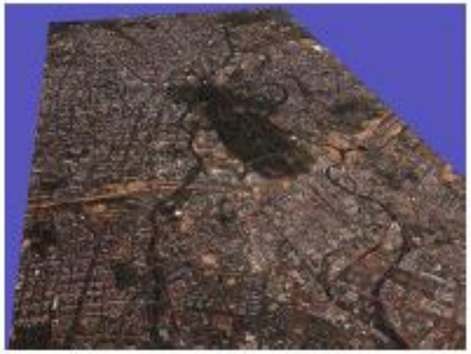
D- data

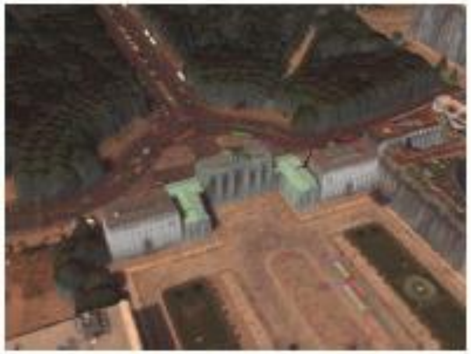
I – reconstructed geometry

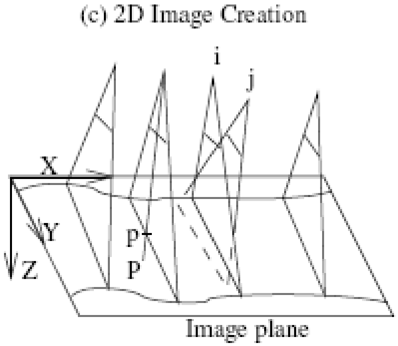
Solved with graph cut

[Torr et al. Eurogr.06, Siggraph07]

City modeling – aerial images







Airborne pushbroom camera

[Heiko Hirschmuller et al - DLR]

Semi-global stereo matching
(based on mutual information)

City modeling – ground plane

Camera cluster

car + GPS

Video: Cannot do frame-frame correspondences

Calibrated cameras – relative pose
GPS – car position - 3D tracking

2D feature tracker

SFM

3D points
Dense stereo+fusion
Texture

3D MODEL

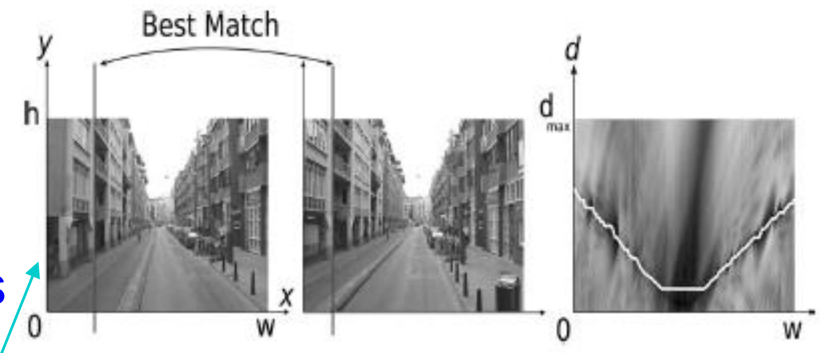
[Nister, Pollefeys et al 3DPVT06, ICCV07]

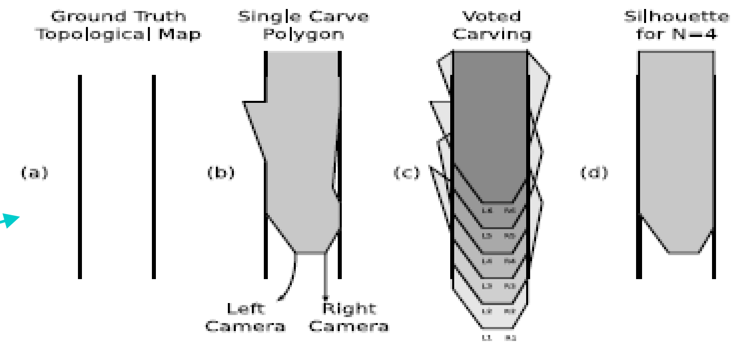
[Cornelis, Van Gool CVPR06…]

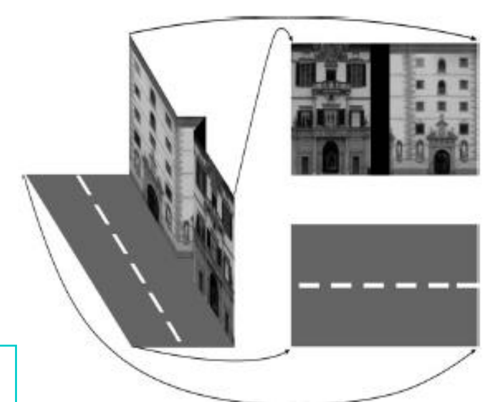
City modeling - example

[Cornelis, Van Gool CVPR06…]

1. feature matching = tracking
2. SFM – camera pose + sparse 3D points
3. Façade reconstruction
   - – rectification of the stereo images
   - - vertical line correlation
4. Topological map generation
   - - orthogonal proj. in the horiz. plane
   - - voting based carving
5. Texture generation
   - - each line segment – column in texture space



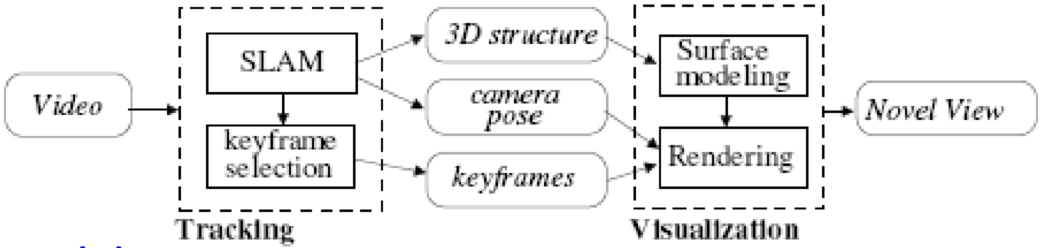




VIDEO

On-line scene modeling : Adam's project

On-line modeling from video
Model not perfect but enough for scene visualization
Application predictive display

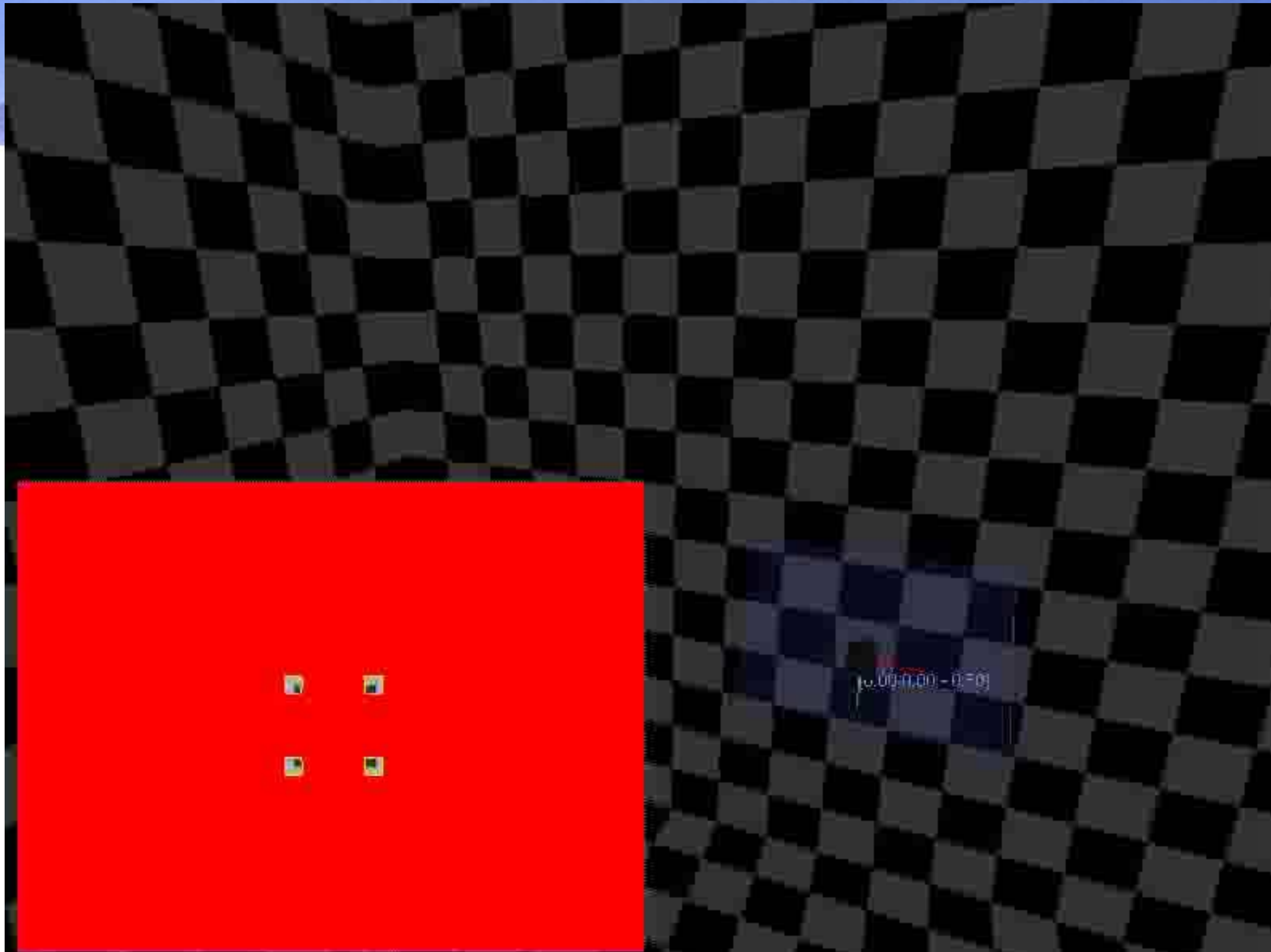


Tracking and Modeling
- New image
- Detect fast corners (similar to Harris)
- SLAM (mono SLAM [Davison ICCV03])
  - Estimate camera pose
  - Update visible structure
- Partial bundle adjustment – update all points
- Save image if keyframe (new view – for texture)

Visualization
- New visual pose
  - Compute closet view
  - Triangulate
  - Project images from closest views onto surface

SLAM
Camera pose
3D structure
Noise model
Extended Kalman Filter

Model refinement



MTF – Modular Tracking Framework

Modular Tracking Framework

A Unified Approach to Registration based Tracking

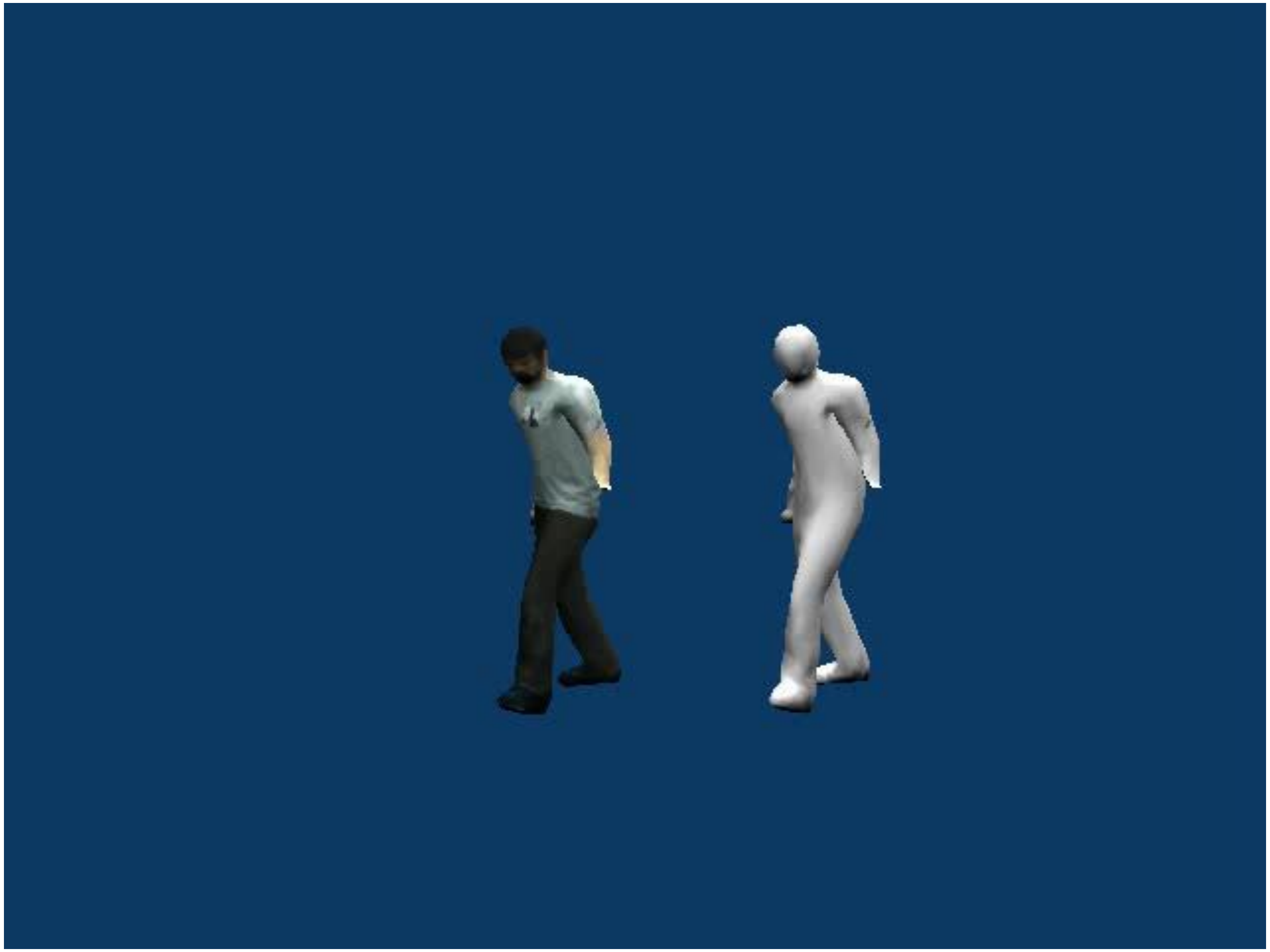
Abhineet Singh and Martin Jagersand

UNIVERSITY OF ALBERTA

- Open source
- C++ implementation
- ROS interface
- Matlab/Pyhton
- Cross platform

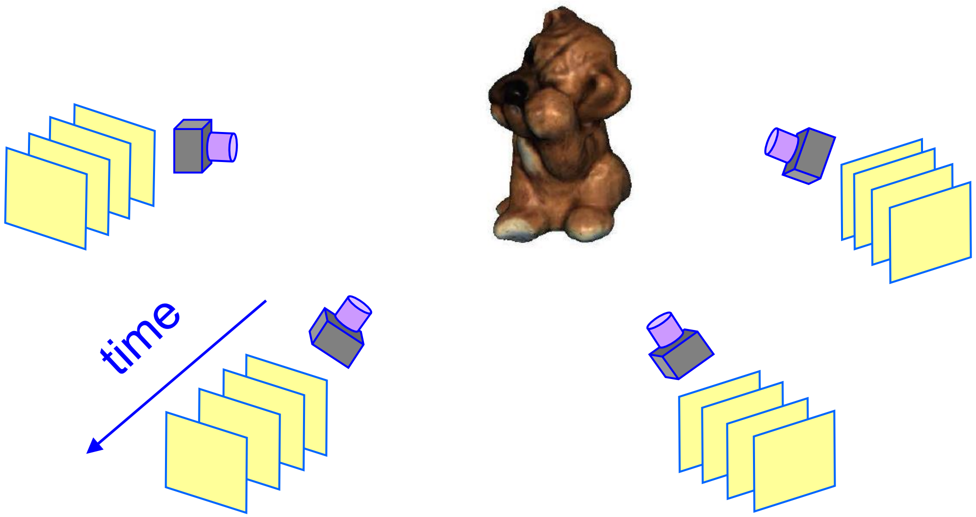
3D tracking and modeling application: Augmenter Reality

Modeling dynamic scenes



[Neil Birkbeck]

Multi-camera systems



Several cameras mutually registered (precalibrated)

Video sequence in each camera

Moving object

Techniques

- Naïve : reconstruct shape every frame
- Integrate stereo and image motion cues
- Extend stereo in temporal domain
- Estimate scene flow in 3D from optic flow and stereo

Representations :

- Disparity/depth
- Voxels / level sets
- Deformable mesh – hard to keep time consistency

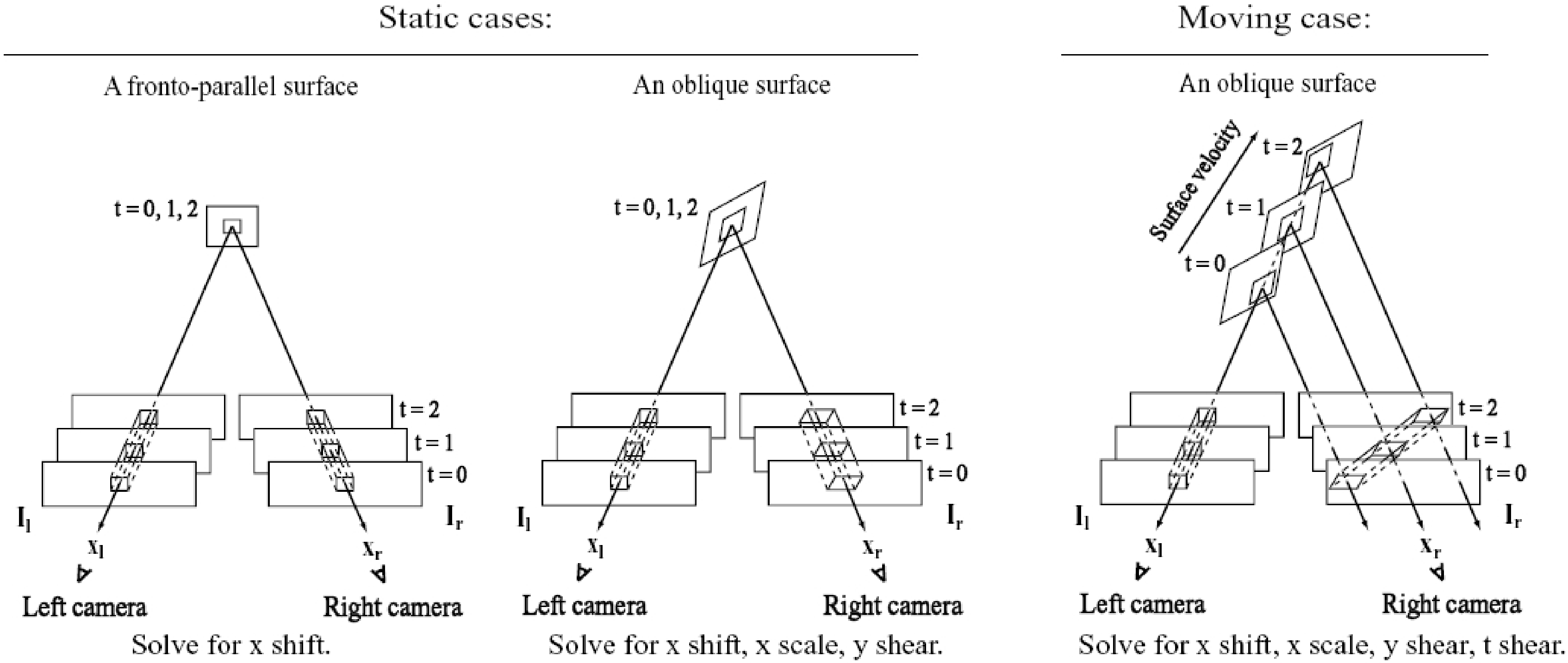
Knowledge:

- Camera positions
- Scene correspondences (structured light)

Spacetime stereo

[Zhang, Curless, Seitz: Spacetime stereo, CVPR 2003]

Extends stereo in time domain: assumes intra-frame correspondences



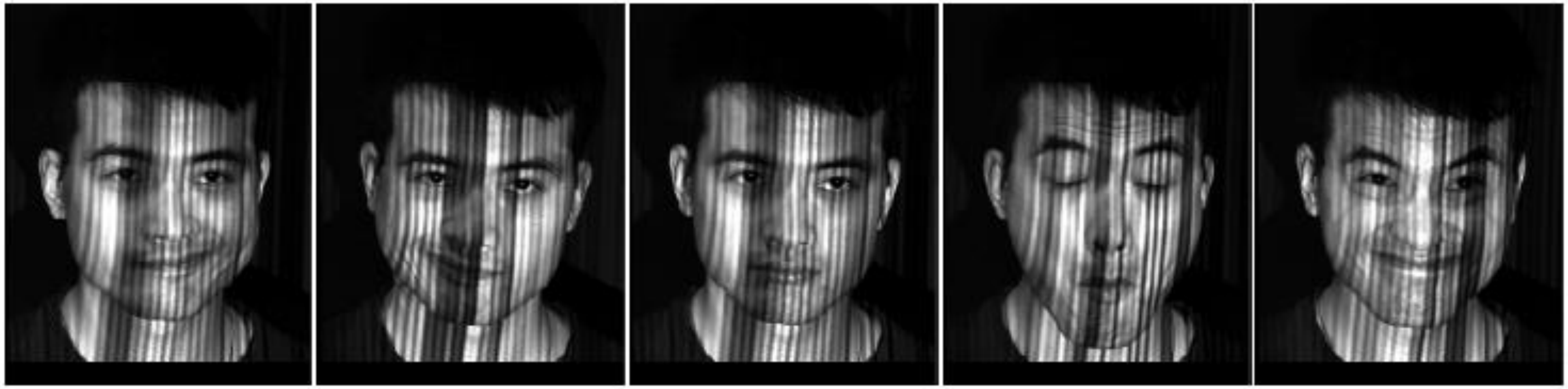
Static scene: disparity

$$d(x,y,t) \approx d_0 + d_{x_0}(x - x_0) + d_{y_0}(y - y_0)$$

Dynamic scene:

$$d(x,y,t) \approx d_0 + d_{x_0}(x - x_0) + d_{y_0}(y - y_0) + \boxed{d_{t_0}(t - t_0)}$$

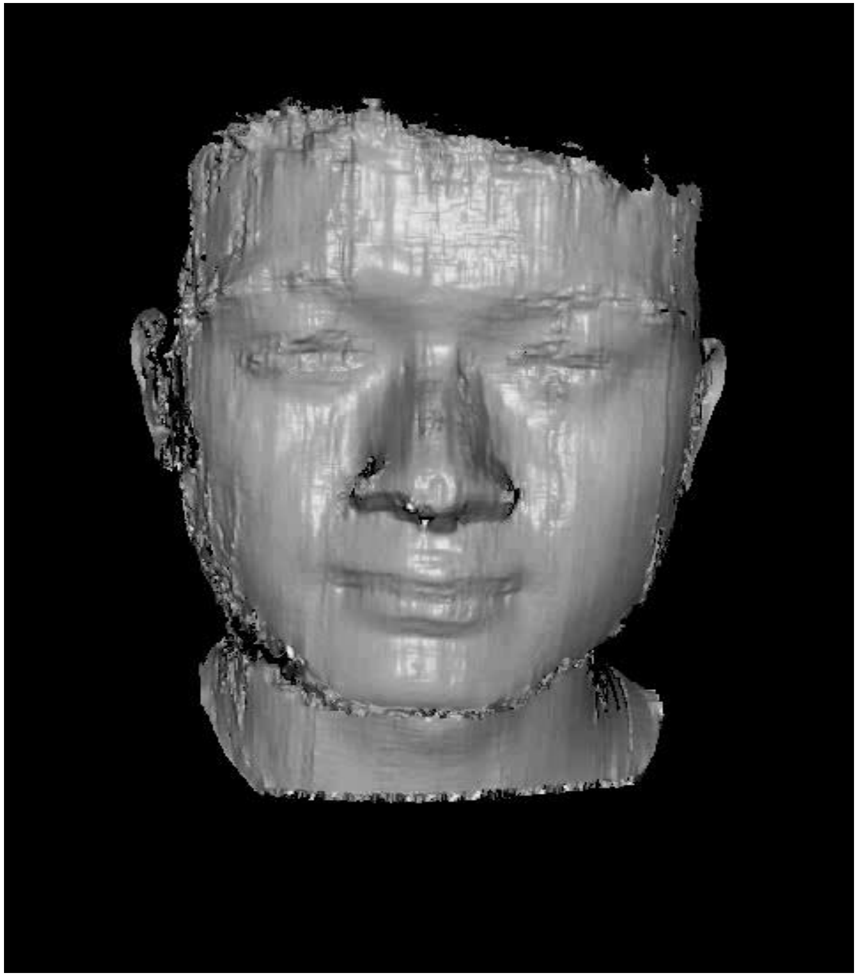
Spacetime stereo: Results



Input: 400 stereo pairs (5 left camera imagest shown here)

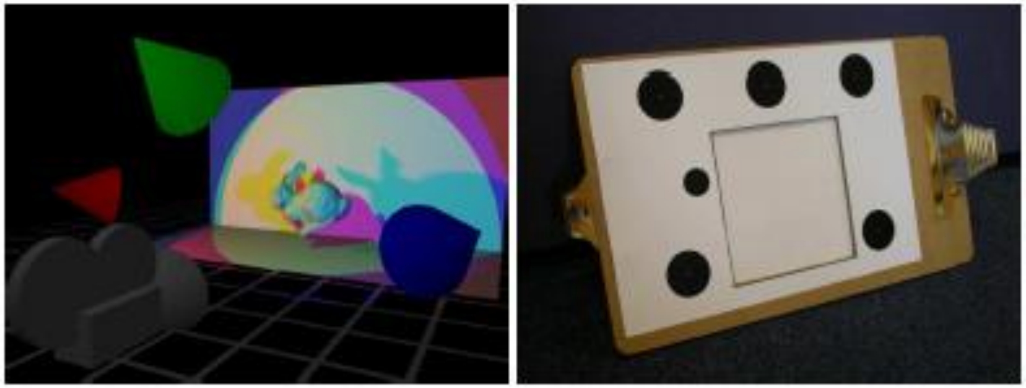
Spacetime stereo reconstruction with 9x5x5 window

Spacetime stereo:video



Spacetime photometric stereo

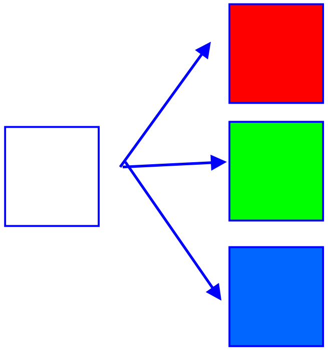
[Hernandez et al. ICCV 2007]



One color camera

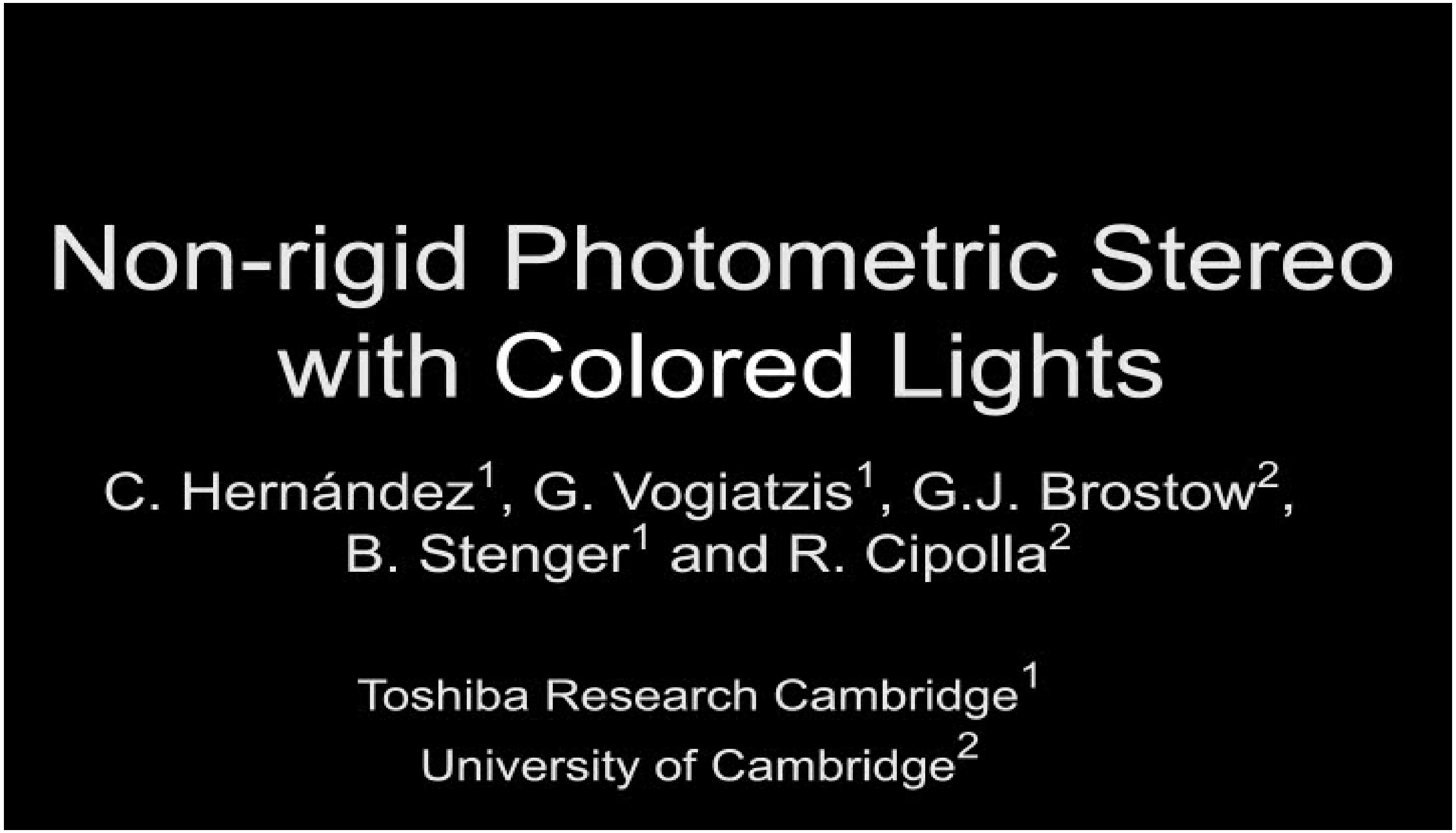
projectors – 3 different positions

Calibrated w.r. camera



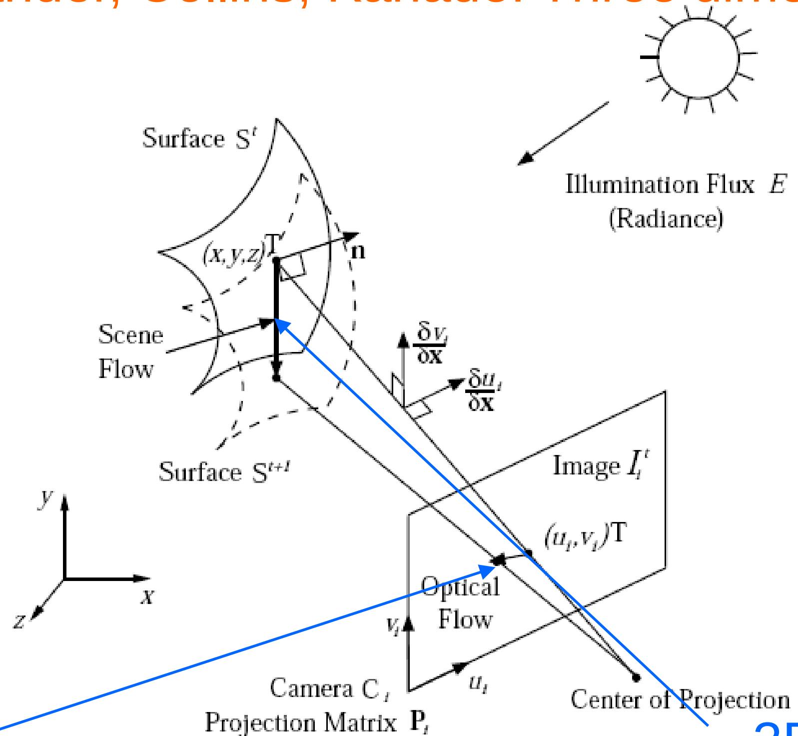
Each channel (R,G,B) – one colored light pose
Photometric stereo

Spacetime PS - Results



3. Scene flow

[Vedula, Baker, Rander, Collins, Kanade: Three dimensional scene flow, ICCV 99]



2D Optic flow

$$\frac{d\mathbf{u}}{dt} : \quad \nabla I_i \frac{d\mathbf{u}_i}{dt} + \frac{\partial I_i}{\partial t}$$

3D Scene flow

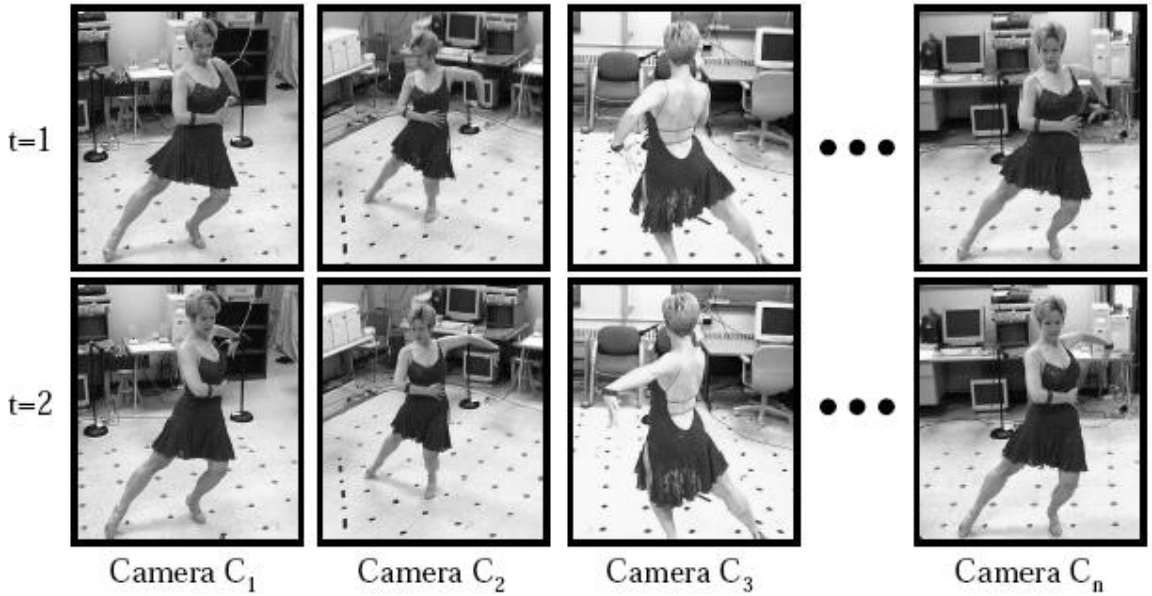
$$\mathbf{x} = \mathbf{x}(\mathbf{u}_i(t);t)$$

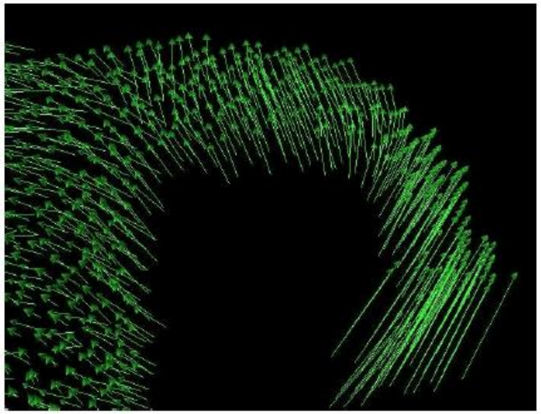
$$\frac{d\mathbf{x}}{dt} = \underbrace{\frac{\partial \mathbf{x}}{\partial \mathbf{u}_i}\frac{d\mathbf{u}_i}{dt}} + \underbrace{\left.\frac{\partial \mathbf{x}}{\partial t}\right|_{u_i}}$$

Scene flow on tangent plane

Motion of x along a ray

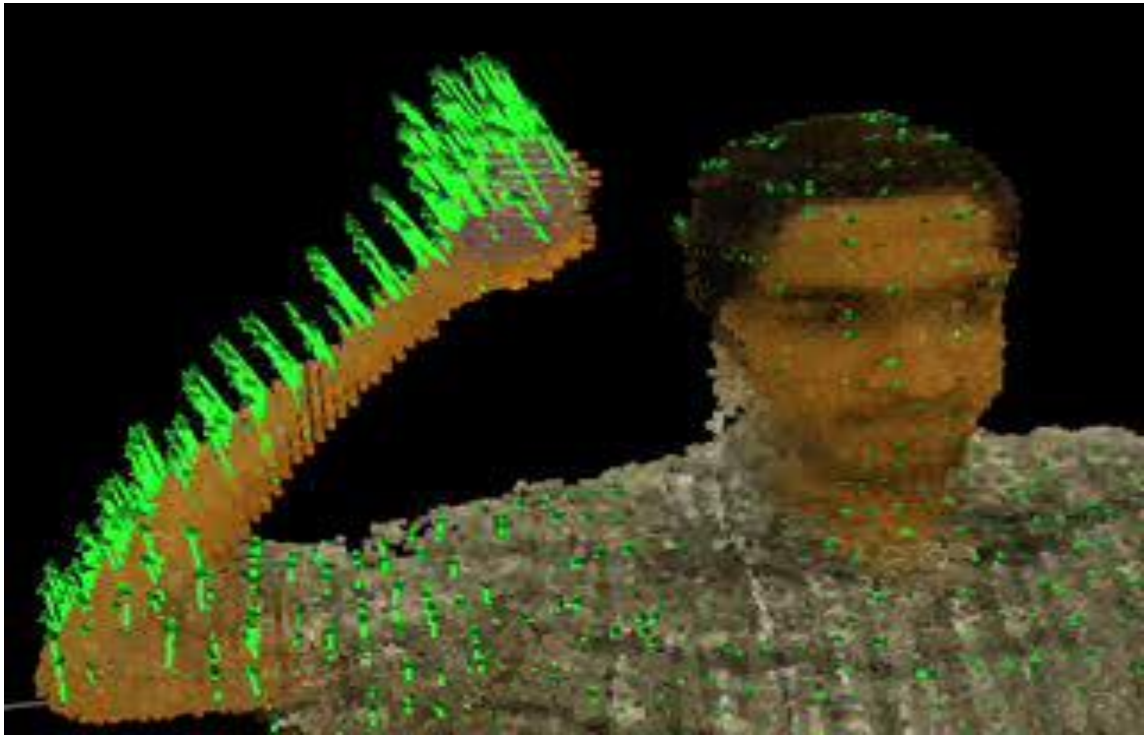
Scene flow: results





[Vedula, et al. ICCV 99]

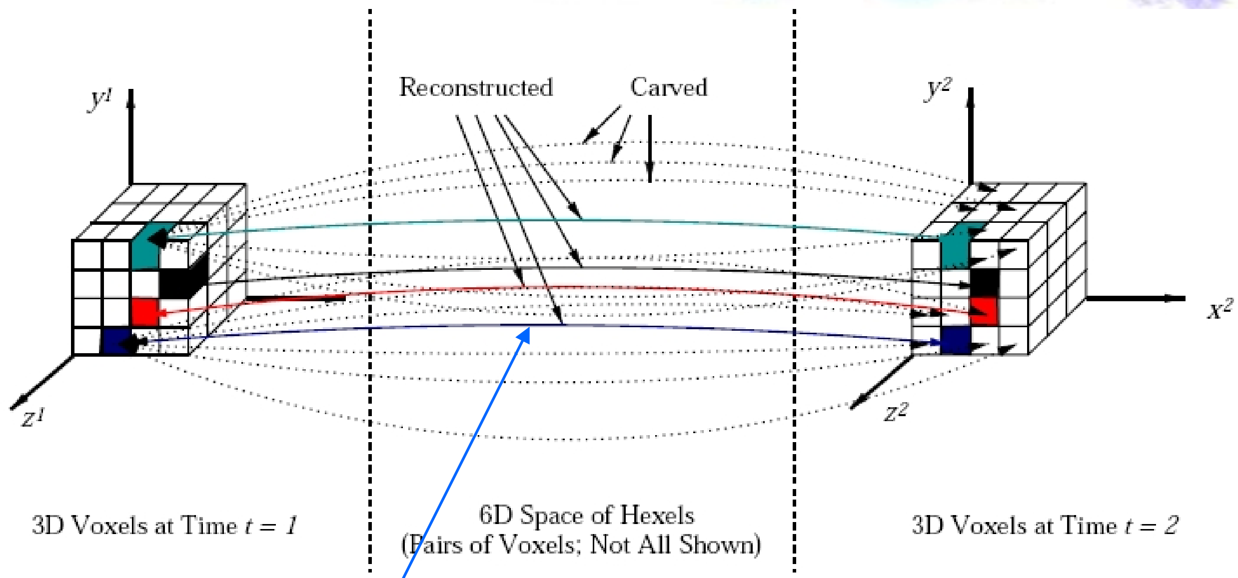
Scene flow: video



[Vedula, et al. ICCV 99]

4. Carving in 6D

[Vedula, Baker, Seitz, Kanade: Shape and motion carving in 6D]



Hexel:

$$(x_1, y_1, z_1, \Delta x, \Delta y, \Delta z)$$

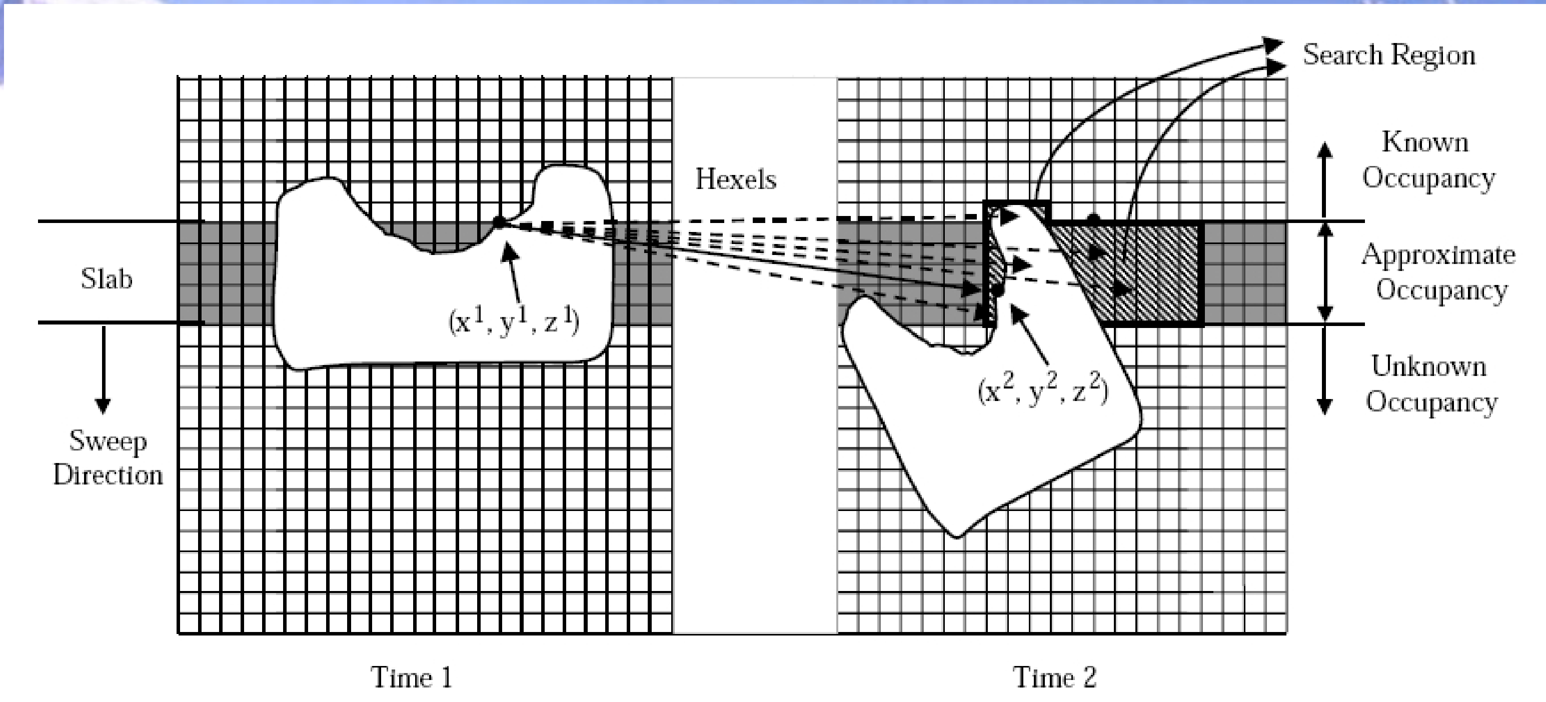
$$(x_2, y_2, z_2) = (x_1, y_1, z_1) + (\Delta x, \Delta y, \Delta z)$$

6D photo-consistency:

$$S^t = \sum_i I_i^t(P_i(\mathbf{x}^t)); \quad SS^t = \sum_i \left(I_i^t(P_i(\mathbf{x}^t))\right)^2$$

$$\frac{SS^1 + SS^2 - (S^1 + S^2) * (S^1 + S^2)}{n^1 + n^2}$$

6D slab sweeping

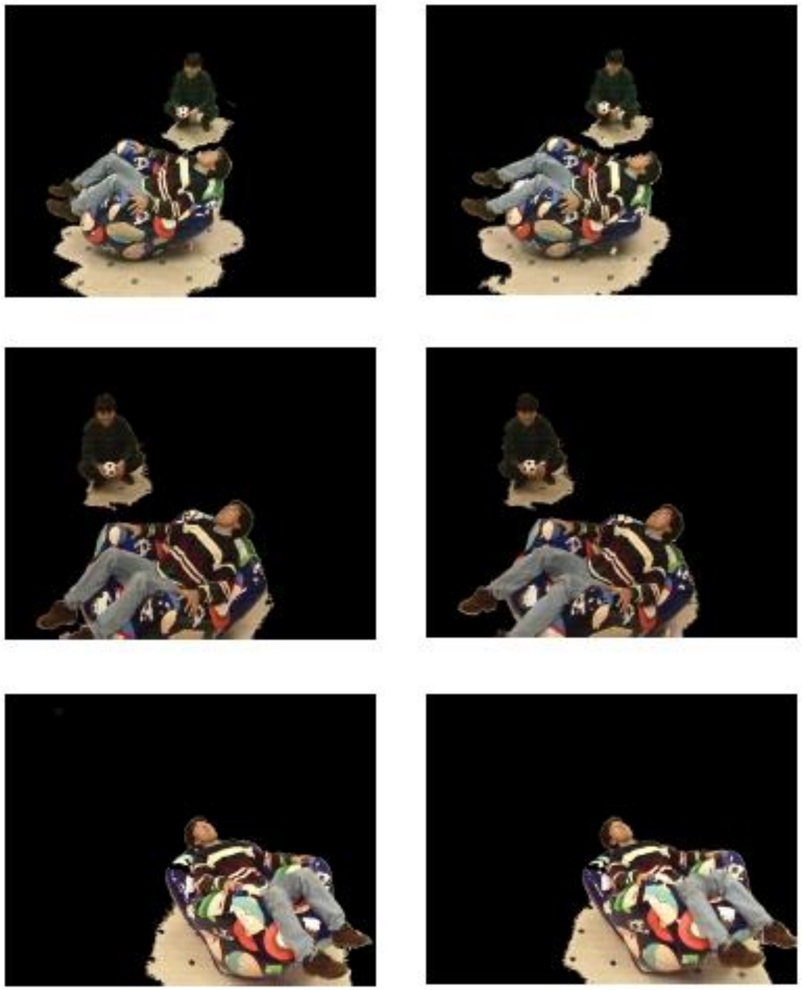


Slab = thickened plane (thikness = upper bound on the flow magnitude)

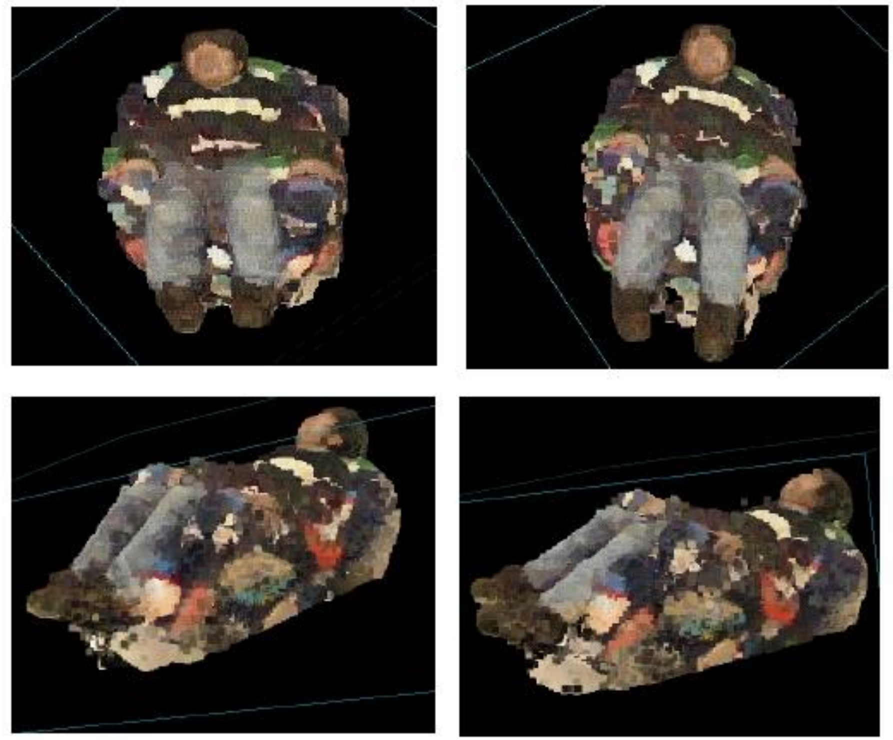
- compute visibility for x^1
- determine search region
- compute all hexel photo-consistency
- carving hexels
- update visibility

(Problem: visibility below the top layer in the slab before carving)

Carving in 6D: results



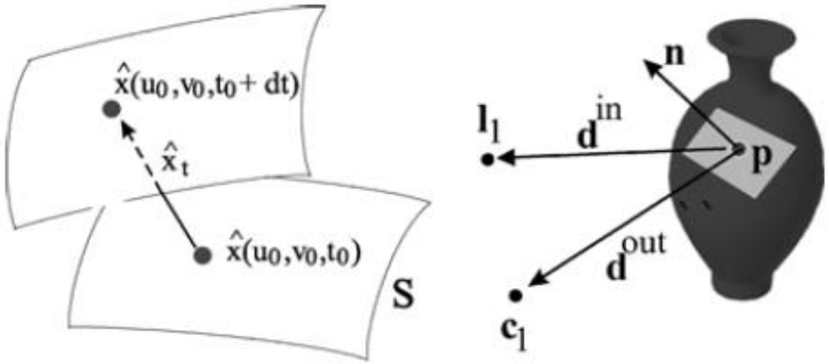
Time 1 Time 2



Time 1 Time 2

7. Surfel sampling

[Carceroni, Kutulakos: Multi-view scene capture by surfel samplig, ICCV01]



Surfel: dynamic surface element

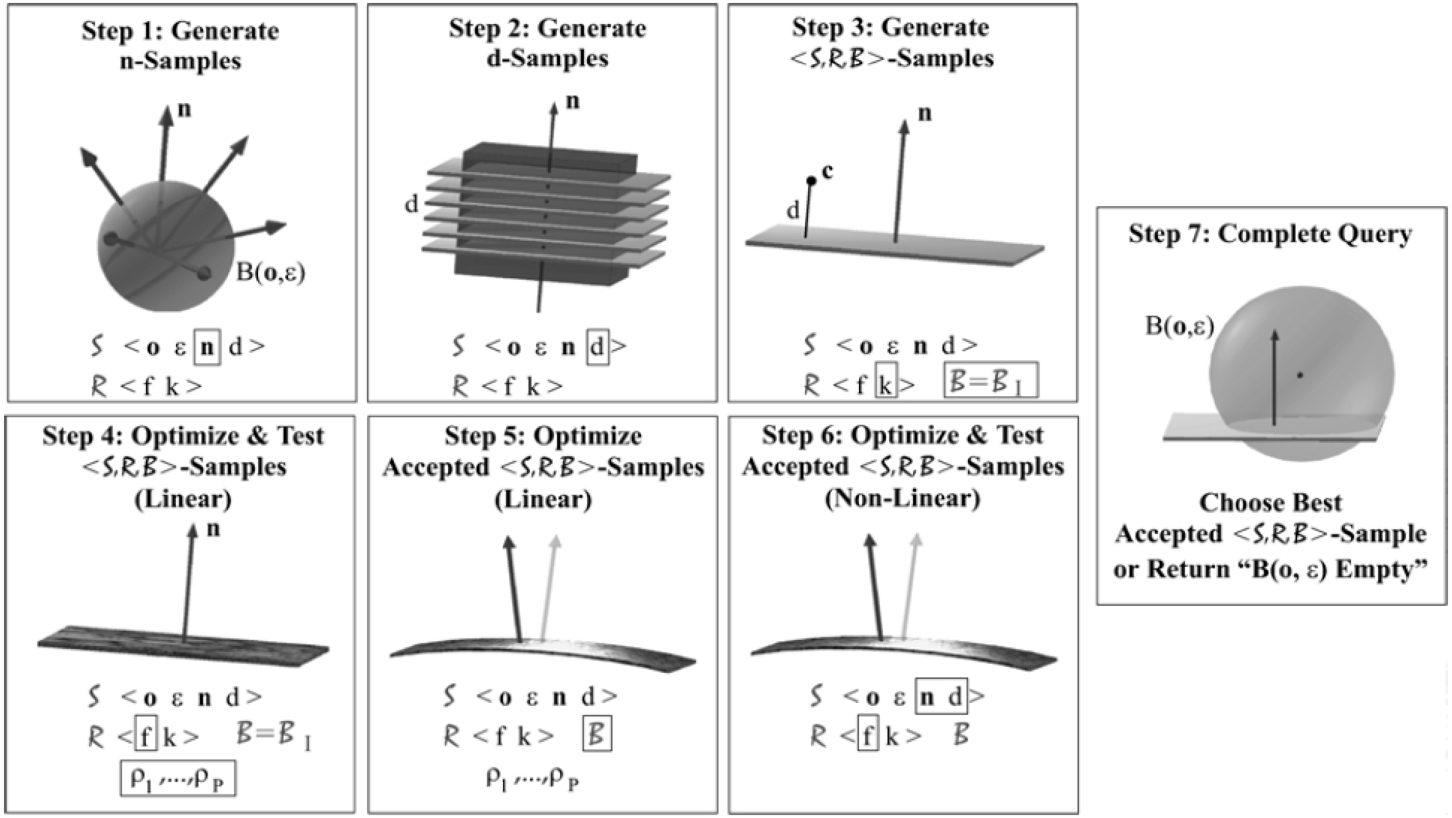
- shape component : center, normal, curvature
- motion component:
- reflectance component: Phong parameters

$$S = \langle \mathbf{x}_0, \mathbf{n}_0, \mathbf{k} \rangle$$

$$M = \langle \tilde{\mathbf{x}}_t, \tilde{\mathbf{x}}_{ut}, \tilde{\mathbf{x}}_{vt} \rangle$$

$$R = \langle f, k, \{\rho_1, \ldots, \rho_P\} \rangle$$

Reconstruction algorithm



$$E(S,R) = \sum_i \sum_j v_{c_i}(\mathbf{p}_i)\left[I_i^{pred}(\mathbf{p}_i) - I_i(\mathbf{p}_i)\right]$$

visibility

$$\langle S^*, R^* \rangle = \arg\min_S (\min_R E(S,R))$$

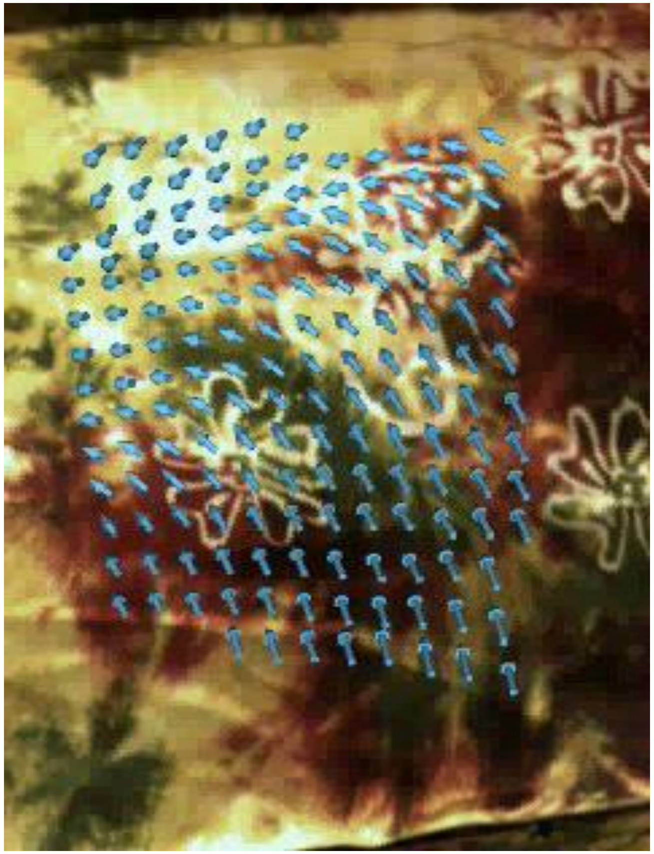
$$I_i^{pred}(\mathbf{p}_i) = \sum_l r(\mathbf{p},\mathbf{n},\mathbf{c}_i - \mathbf{p},\mathbf{l}_l - \mathbf{p})L_l(\mathbf{p})$$

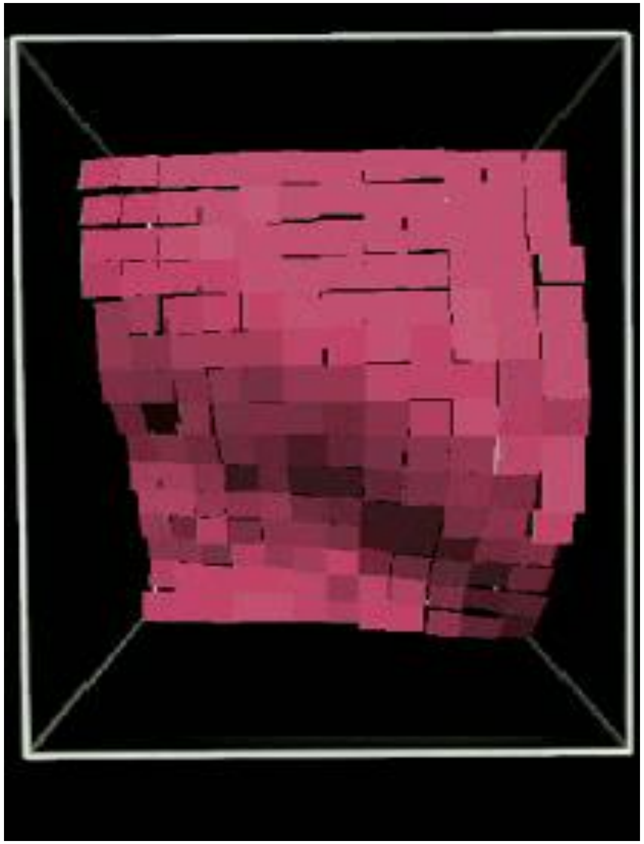
Phong reflectance   shadow

c_i – camera i

$\mathbf{l}_l$ - light l

Surfel sampling : results

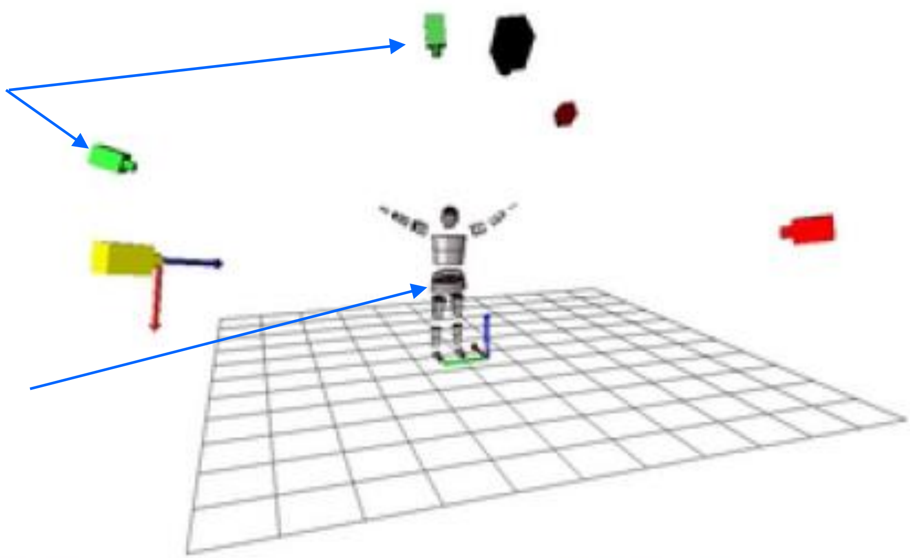




Modeling humans in motion

Multiple calibrated cameras

Human in motion



Goal: 3D model of the human

Instantaneous model that can be viewed from different poses (‘Matrix’) and inserted in an artificial scene (tele-conferences)

GRIMAGE platform- INRIA Grenoble

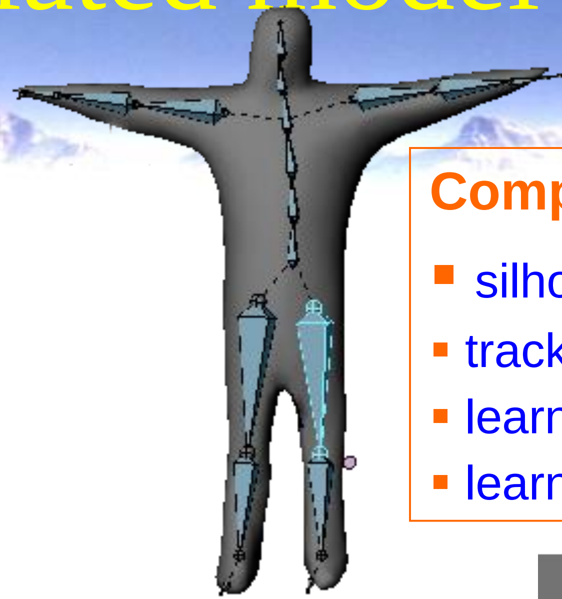
Our goal: 3D animated human model

- capture model deformations and appearance change in motion
- animated in a video game

[Neil Birkbeck]

Articulated model

Model based approach



Geometric Model

Skeleton + skinned mesh
(bone weights)

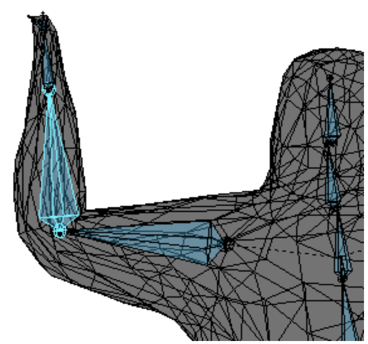
50+ DOF (CMU mocap data)

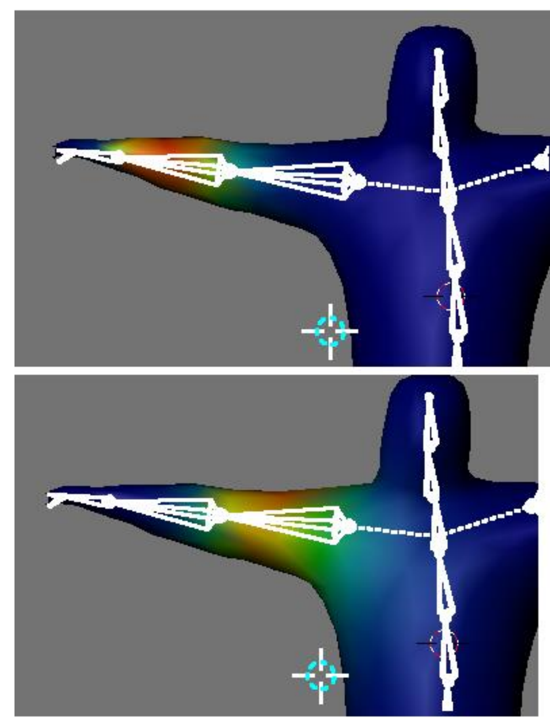
Tracking

- visual hull – bone weights by diffusion
- refine mesh/weights

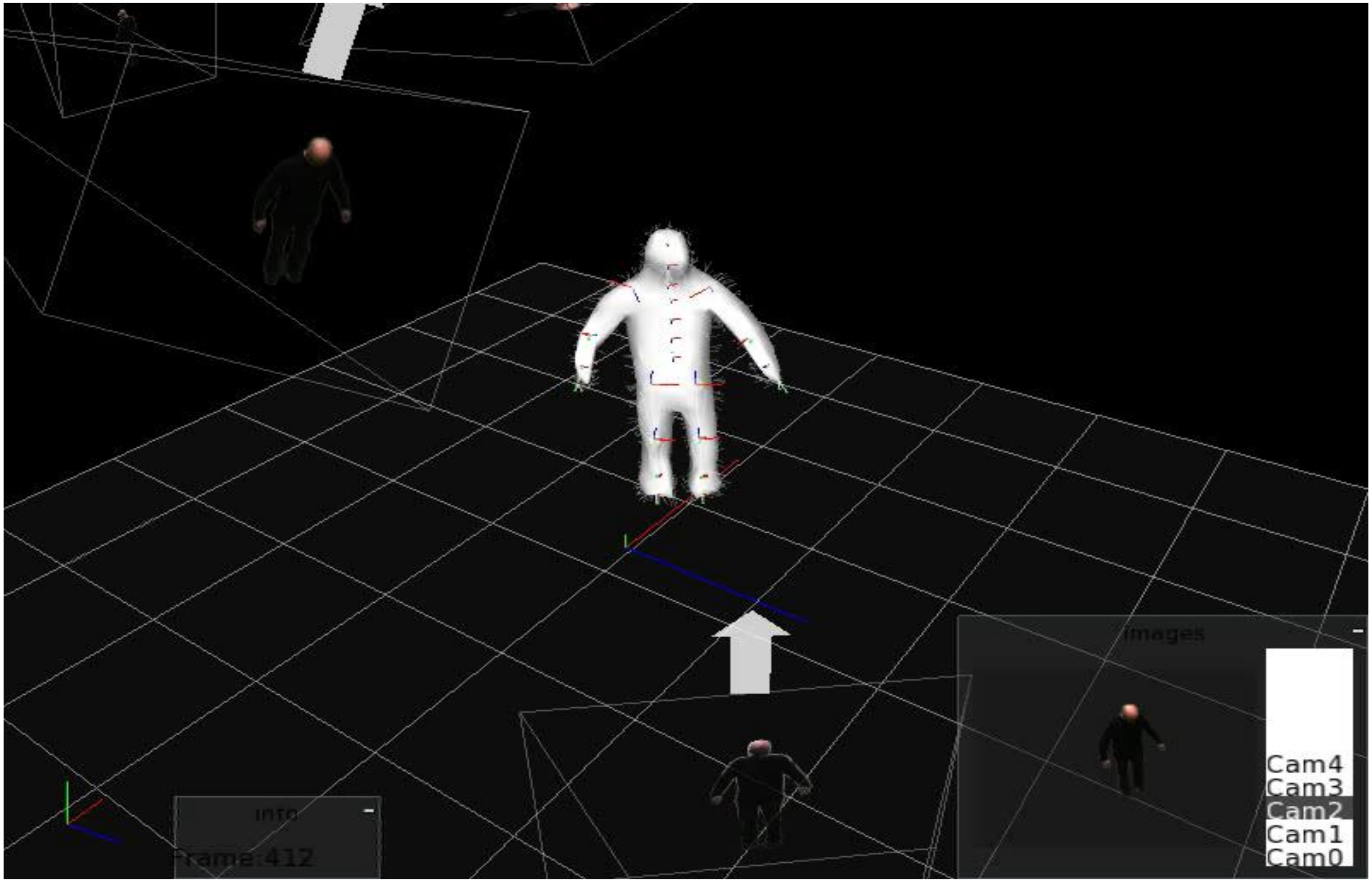
Components

- silhouette extraction
- tracking the course model
- learn deformations
- learn appearance change





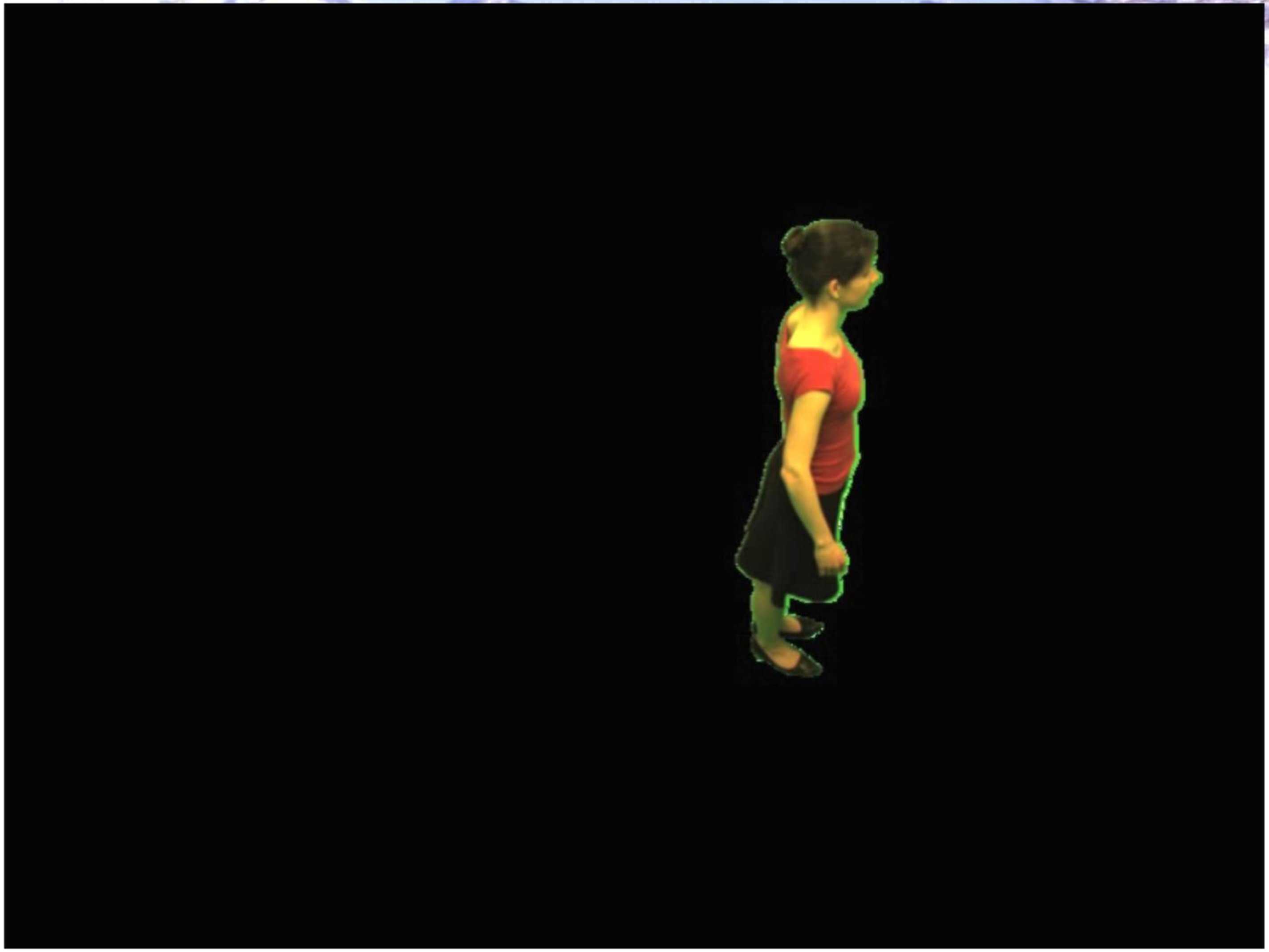
Neil- tracking results



Beyond 3D
Non-rigid and articulated motion

- Humans ubiquitous in graphics applications
- A practical, realistic model requires
  - Skeleton
  - Geometry (manually modeled, laser scanned)
    - Physical simulation for clothes, muscle
  - Texture/appearance (from images)
  - Animation (mocap, simulation, artist)

Beyond 3D:
Non-rigid and articulated motion



PhD work of Neil Birkbeck, Best thesis prize winner

Computer Vision

Questions?