UNIVERSITÄT BONN

Juergen Gall

Structure from motion

MA-INF 2201 - Computer Vision

WS24/25

Exam

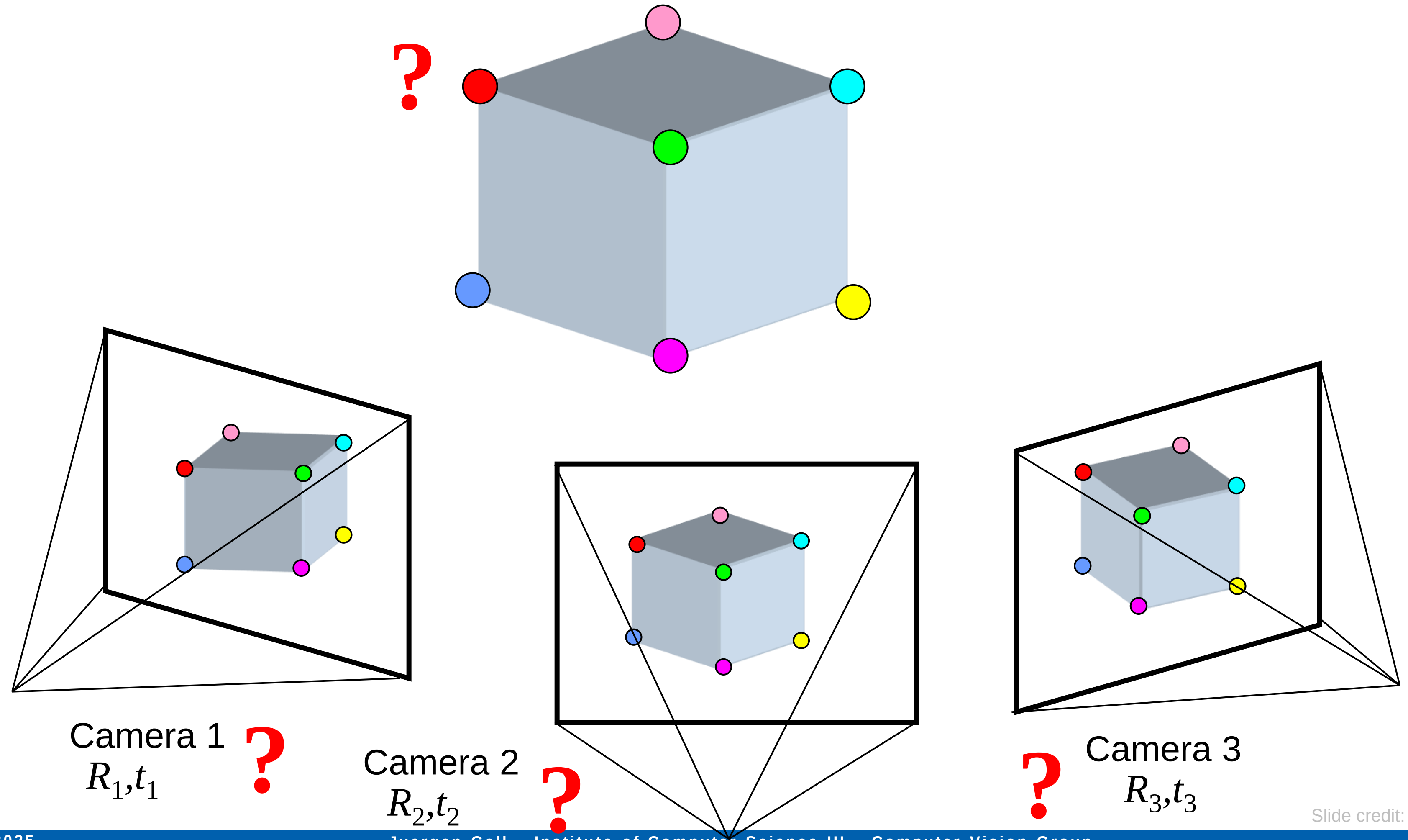
- Recap lecture: 31.1.
- Exam: Thursday, 6.2., 10:00-12:00
  Meckenheimer Allee 176, Hörsaal IV
- 2nd Exam: Monday, 24.3., 12:00-14:00
  Meckenheimer Allee 176, Hörsaal IV

Seminar

- MA-INF 2206 - Seminar Vision
- Meetings:
    - Today, 24.1., 11:45, HSZ / HS 3
    - Friday, 31.1., 11:45, HSZ / HS 3

Structure from motion

Given a set of corresponding points in two or more images, compute the camera parameters and the 3D point coordinates



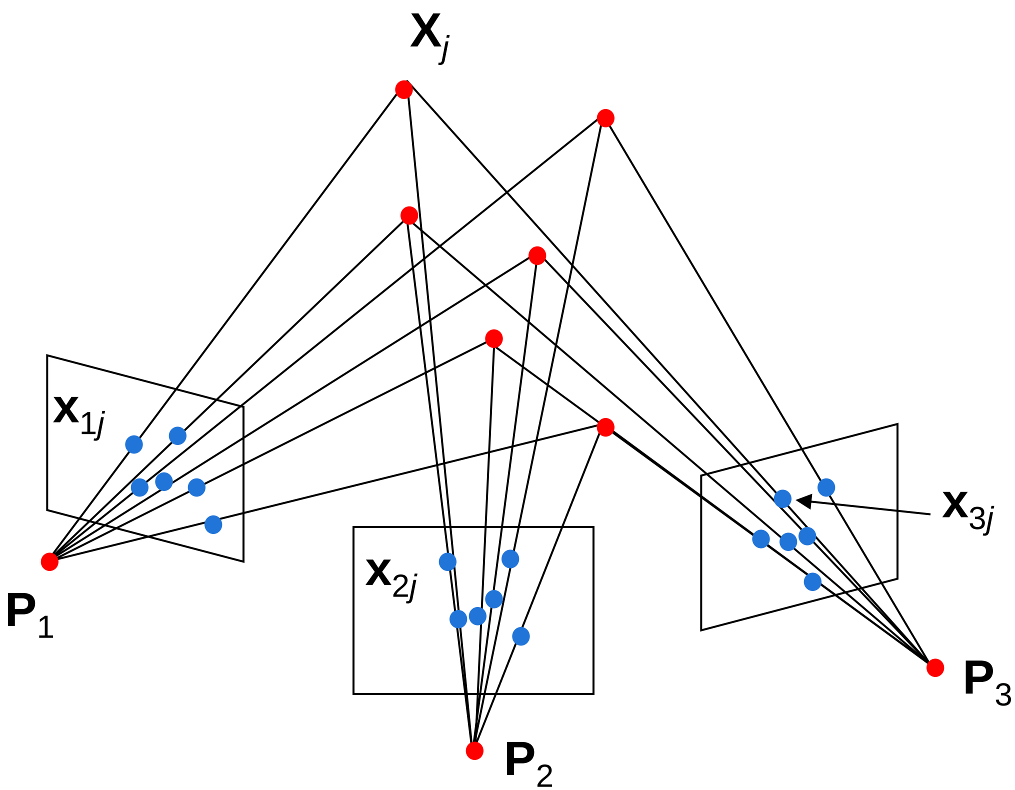
Slide credit: Noah Snavely

Structure from motion

Given: *m* images of *n* fixed 3D points

$$\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j , \qquad i = 1, \dots , m, \quad j = 1, \dots , n$$

Problem: estimate *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the *mn* correspondences $\mathbf{x}_{ij}$



Lazebnik

Structure from motion ambiguity

UNIVERSITÄT BONN

- If we scale the entire scene by some factor *k* and, at the same time, scale the camera matrices by the factor of 1/*k*, the projections of the scene points in the image remain exactly the same:

$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\frac{1}{k}\mathbf{P}\right)(k\,\mathbf{X})$$

It is impossible to recover the absolute scale of the scene!

Lazebnik

Structure from motion ambiguity

If we scale the entire scene by some factor *k* and, at the same time, scale the camera matrices by the factor of 1/*k*, the projections of the scene points in the image remain exactly the same

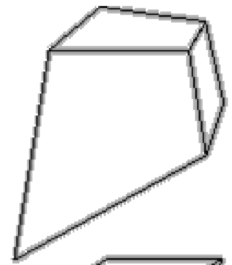
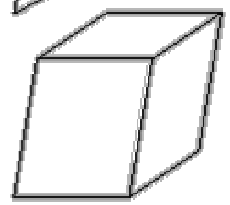
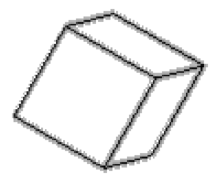
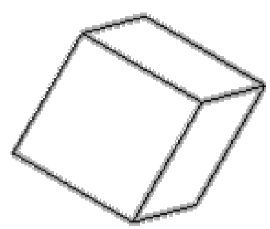
More generally: if we transform the scene using a transformation **Q** and apply the inverse transformation to the camera matrices, then the images do not change

$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\mathbf{P}\mathbf{Q}^{-1}\right)\left(\mathbf{Q}\mathbf{X}\right)$$

Lazebnik

Types of ambiguity

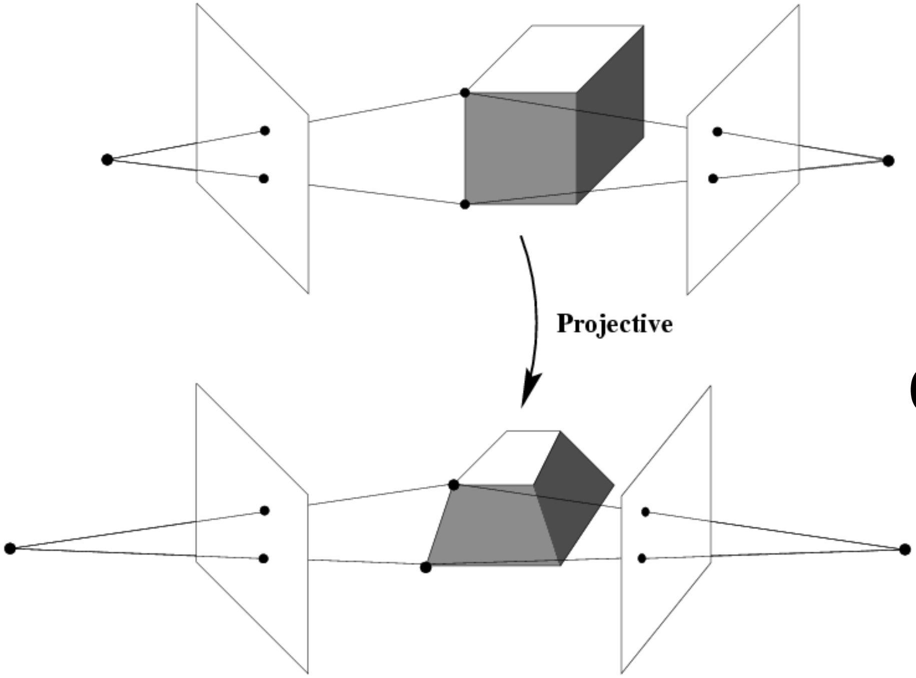
UNIVERSITÄT BONN

Projective 15dof	$\begin{bmatrix} A & t \\ v^T & v \end{bmatrix}$		Preserves intersection and tangency
Affine 12dof	$\begin{bmatrix} A & t \\ 0^T & 1 \end{bmatrix}$		Preserves parallellism, volume ratios
Similarity 7dof	$\begin{bmatrix} s\,R & t \\ 0^T & 1 \end{bmatrix}$		Preserves angles, ratios of length
Euclidean 6dof	$\begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix}$		Preserves angles, lengths

- With no constraints on the camera calibration matrix or on the scene, we get a *projective* reconstruction
- Need additional information to *upgrade* the reconstruction to affine, similarity, or Euclidean

Lazebnik

Projective ambiguity

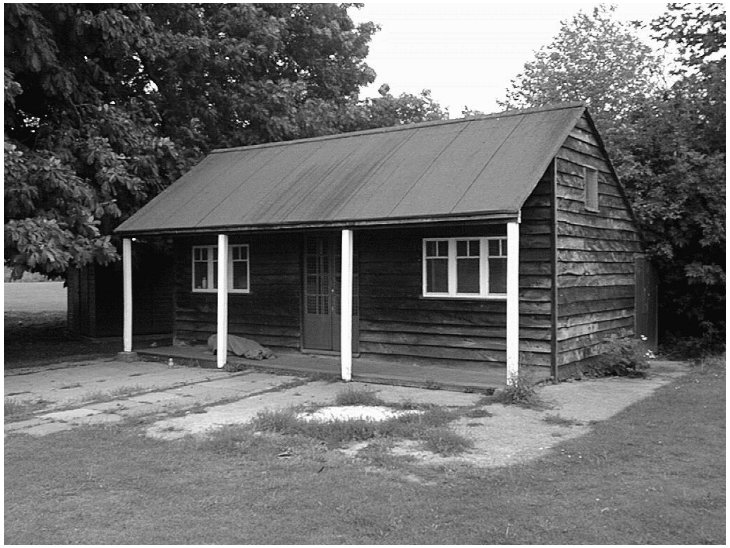


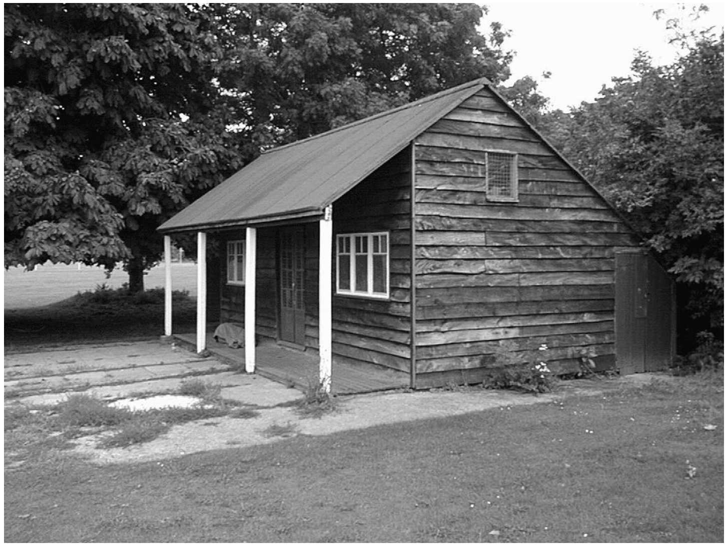
$$\mathbf{Q_p} = \begin{bmatrix} A & t \\ v^{\mathsf{T}} & v \end{bmatrix}$$

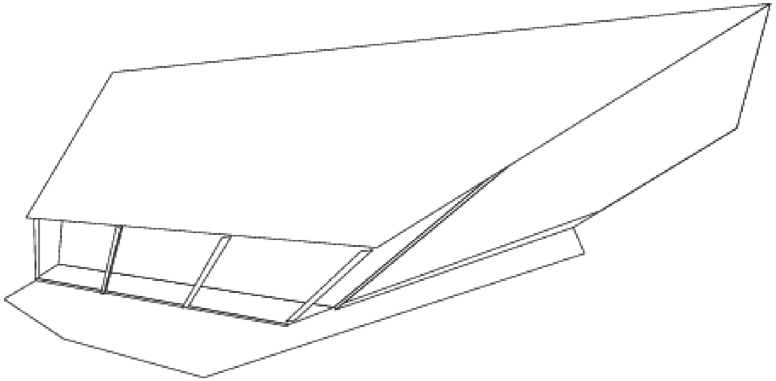
$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\mathbf{P}\mathbf{Q}_\mathbf{P}^{-1}\right)\left(\mathbf{Q}_\mathbf{P}\,\mathbf{X}\right)$$

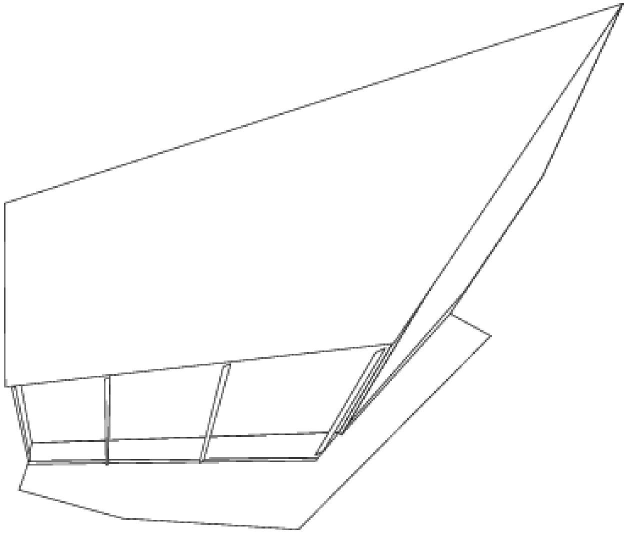
Lazebnik

Projective ambiguity



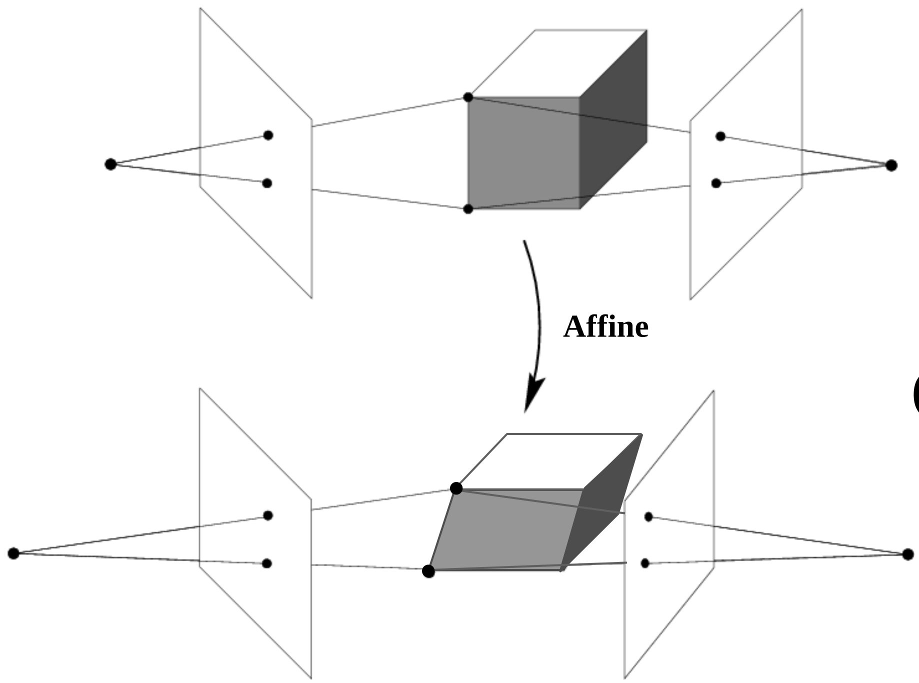






Lazebnik

Affine ambiguity

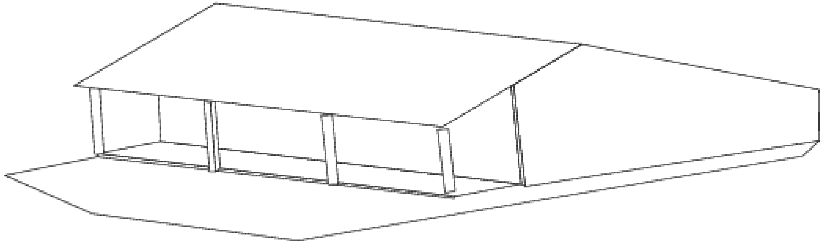


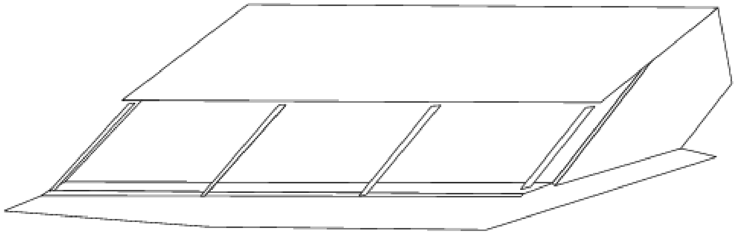
$$\mathbf{Q}_A = \begin{bmatrix} \mathrm{A} & \mathrm{t} \\ 0^{\mathsf{T}} & 1 \end{bmatrix}$$

$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\mathbf{P}\mathbf{Q}_A^{-1}\right)\left(\mathbf{Q}_A\mathbf{X}\right)$$

Lazebnik

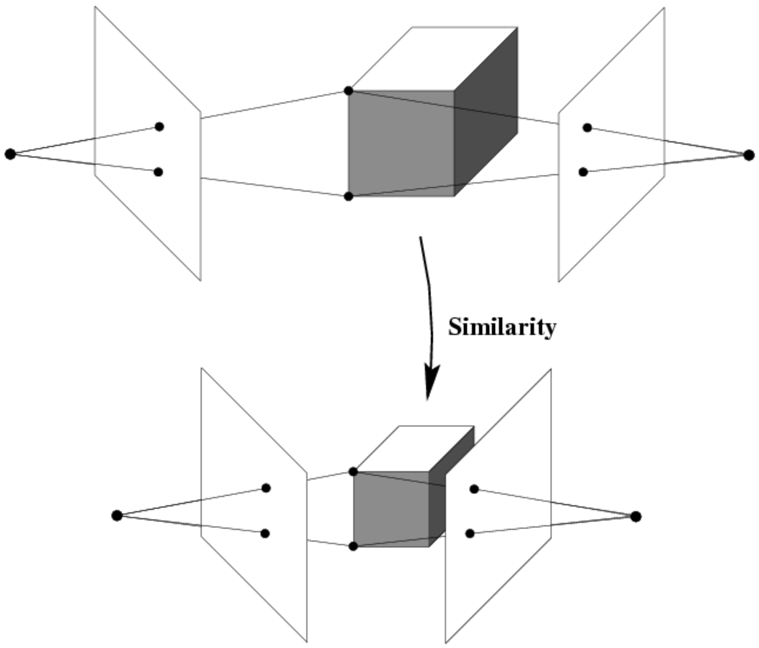
Affine ambiguity





Lazebnik

Similarity ambiguity

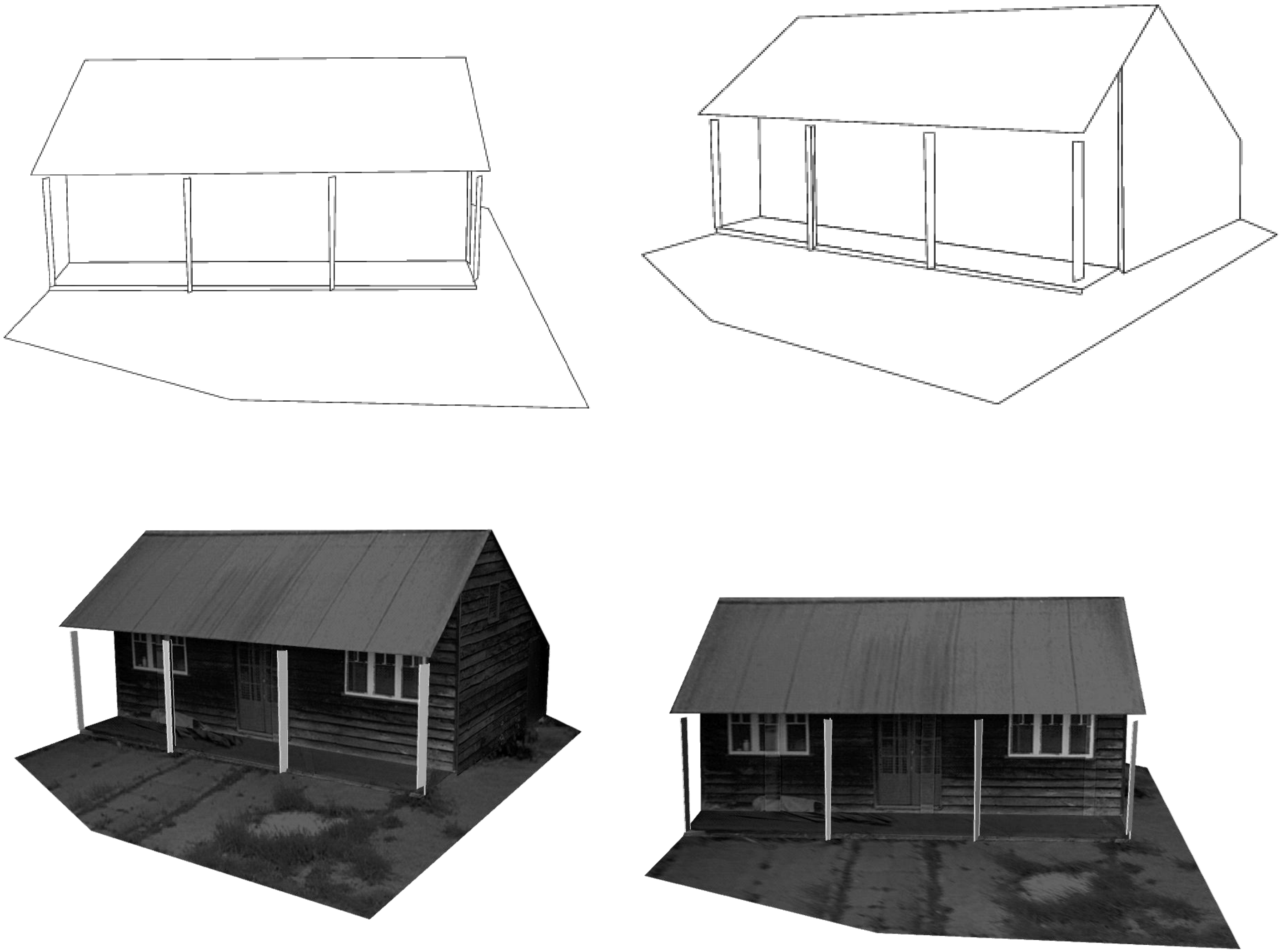


$$\mathbf{Q}_s = \begin{bmatrix} s\mathrm{R} & \mathrm{t} \\ 0^{\mathsf{T}} & 1 \end{bmatrix}$$

$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\mathbf{P}\mathbf{Q}_S^{-1}\right)\left(\mathbf{Q}_S\mathbf{X}\right)$$

Lazebnik

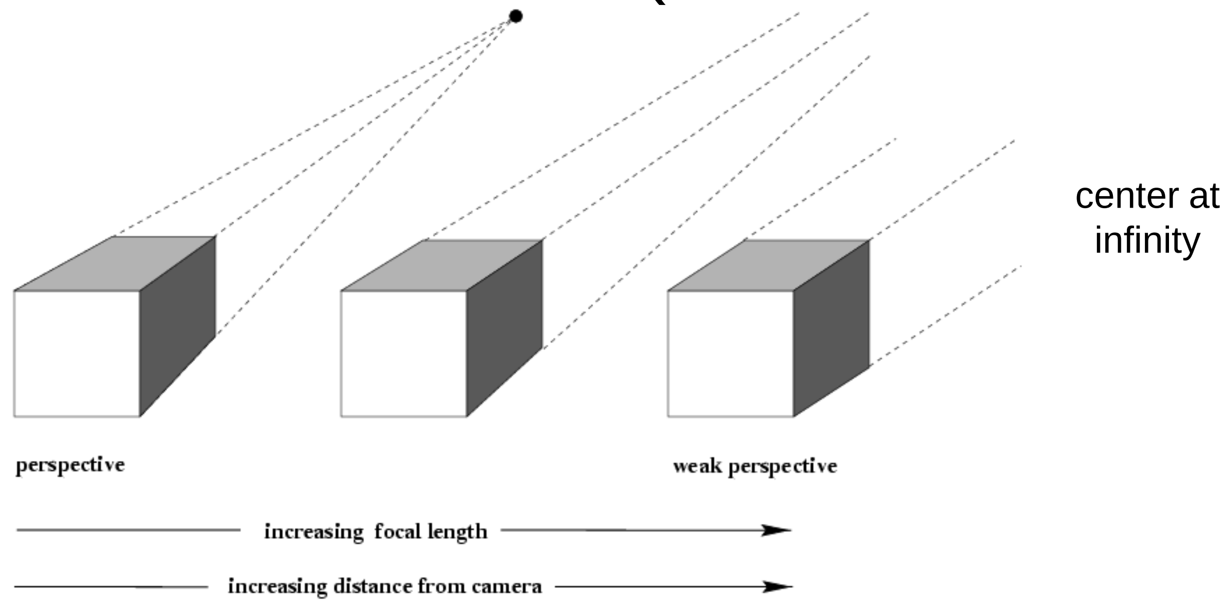
Similarity ambiguity



Lazebnik

Structure from motion

- Let’s start with *affine cameras* (the math is easier)

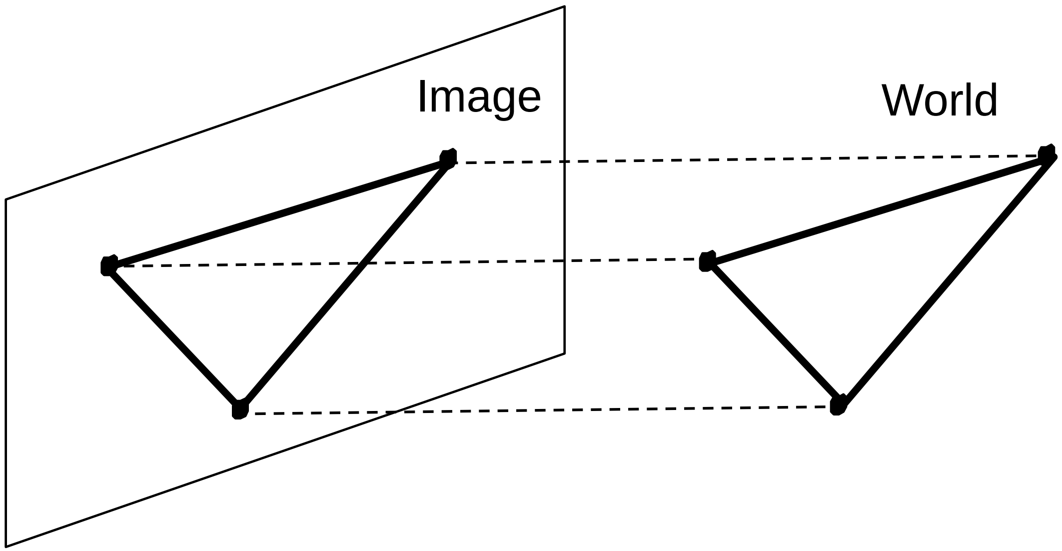


Lazebnik

Recall: Orthographic Projection

Special case of perspective projection

– Distance from center of projection to image plane is infinite



– Projection matrix:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow (x, y)$$

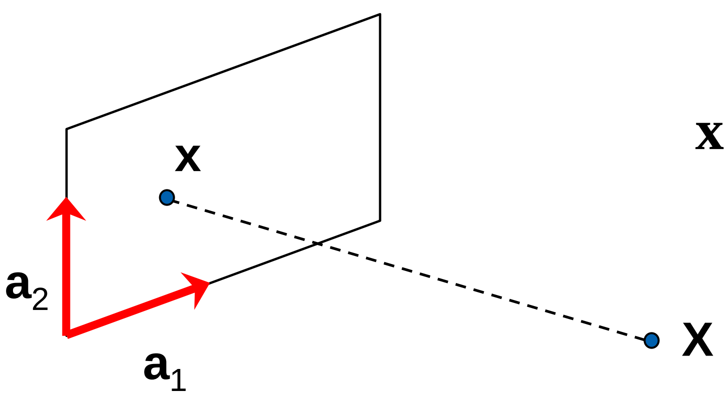
Slide by Steve Seitz

Affine cameras

- A general affine camera combines the effects of an affine transformation of the 3D space, orthographic projection, and an affine transformation of the image:

$$\mathbf{P} = [3\times 3\,\text{affine}]\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}[4\times 4\,\text{affine}] = \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$$

- Affine projection is a linear mapping + translation in inhomogeneous coordinates



$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \mathbf{AX} + \mathbf{b}$$

Projection of world origin

Lazebnik

Affine structure from motion

- Given: *m* images of *n* fixed 3D points:

$$\mathbf{x}_{ij} = \mathbf{A}_i \mathbf{X}_j + \mathbf{b}_i, \quad i = 1, \dots, m, \; j = 1, \dots, n$$

- Problem: use the *mn* correspondences $\mathbf{x}_{ij}$ to estimate *m* projection matrices $\mathbf{A}_i$ and translation vectors $\mathbf{b}_i$, and *n* points $\mathbf{X}_j$
- The reconstruction is defined up to an arbitrary *affine* transformation **Q** (12 degrees of freedom):

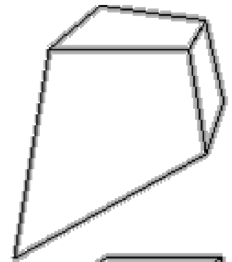
$$\begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & 1 \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{Q}^{-1}, \qquad \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix} \rightarrow \mathbf{Q} \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}$$

Lazebnik

Recall: Types of ambiguity

Projective
15dof

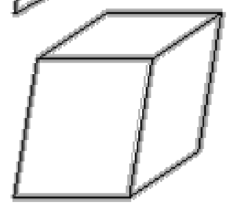
$$\begin{bmatrix} A & t \\ v^T & v \end{bmatrix}$$



Preserves intersection and tangency

Affine
12dof

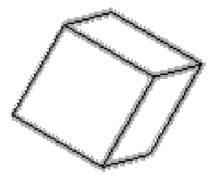
$$\begin{bmatrix} A & t \\ 0^T & 1 \end{bmatrix}$$



Preserves parallellism, volume ratios

Similarity
7dof

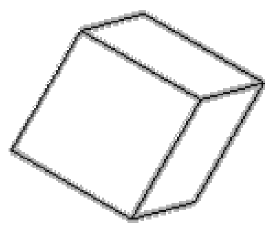
$$\begin{bmatrix} s\,R & t \\ 0^T & 1 \end{bmatrix}$$



Preserves angles, ratios of length

Euclidean
6dof

$$\begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix}$$



Preserves angles, lengths

- With no constraints on the camera calibration matrix or on the scene, we get a *projective* reconstruction
- Need additional information to *upgrade* the reconstruction to affine, similarity, or Euclidean

Lazebnik

Affine structure from motion

- Given: *m* images of *n* fixed 3D points:

$$\mathbf{x}_{ij} = \mathbf{A}_i \mathbf{X}_j + \mathbf{b}_i , \quad i = 1,\dots,m, \; j = 1, \dots, n$$

- Problem: use the *mn* correspondences $\mathbf{x}_{ij}$ to estimate *m* projection matrices $\mathbf{A}_i$ and translation vectors $\mathbf{b}_i$, and *n* points $\mathbf{X}_j$
- The reconstruction is defined up to an arbitrary *affine* transformation **Q** (12 degrees of freedom):

$$\begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & 1 \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{Q}^{-1}, \qquad \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix} \rightarrow \mathbf{Q} \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}$$

- We have 2*mn* knowns and 8*m* + 3*n* unknowns (minus 12 dof for affine ambiguity)
- Thus, we must have 2*mn* >= 8*m* + 3*n* – 12
- For two views, we need four point correspondences

Lazebnik

Affine structure from motion

UNIVERSITÄT BONN

- Centering: subtract the centroid of the image points

$$\hat{\mathbf{x}}_{ij} = \mathbf{x}_{ij} - \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_{ik} = \mathbf{A}_i\mathbf{X}_j + \mathbf{b}_i - \frac{1}{n}\sum_{k=1}^{n}\left(\mathbf{A}_i\mathbf{X}_k + \mathbf{b}_i\right)$$

$$= \mathbf{A}_i\left(\mathbf{X}_j - \frac{1}{n}\sum_{k=1}^{n}\mathbf{X}_k\right) = \mathbf{A}_i\hat{\mathbf{X}}_j$$

- For simplicity, assume that the origin of the world coordinate system is at the centroid of the 3D points
- After centering, each normalized point $\mathbf{x}_{ij}$ is related to the 3D point $\mathbf{X}_i$ by

$$\hat{\mathbf{x}}_{ij} = \mathbf{A}_i\mathbf{X}_j$$

Lazebnik

Affine structure from motion

- Let's create a $2m \times n$ data (measurement) matrix:

$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix}$$

cameras ($2m$)

points (n)

C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

Lazebnik

Affine structure from motion

UNIVERSITÄT BONN

- Let's create a $2m \times n$ data (measurement) matrix:

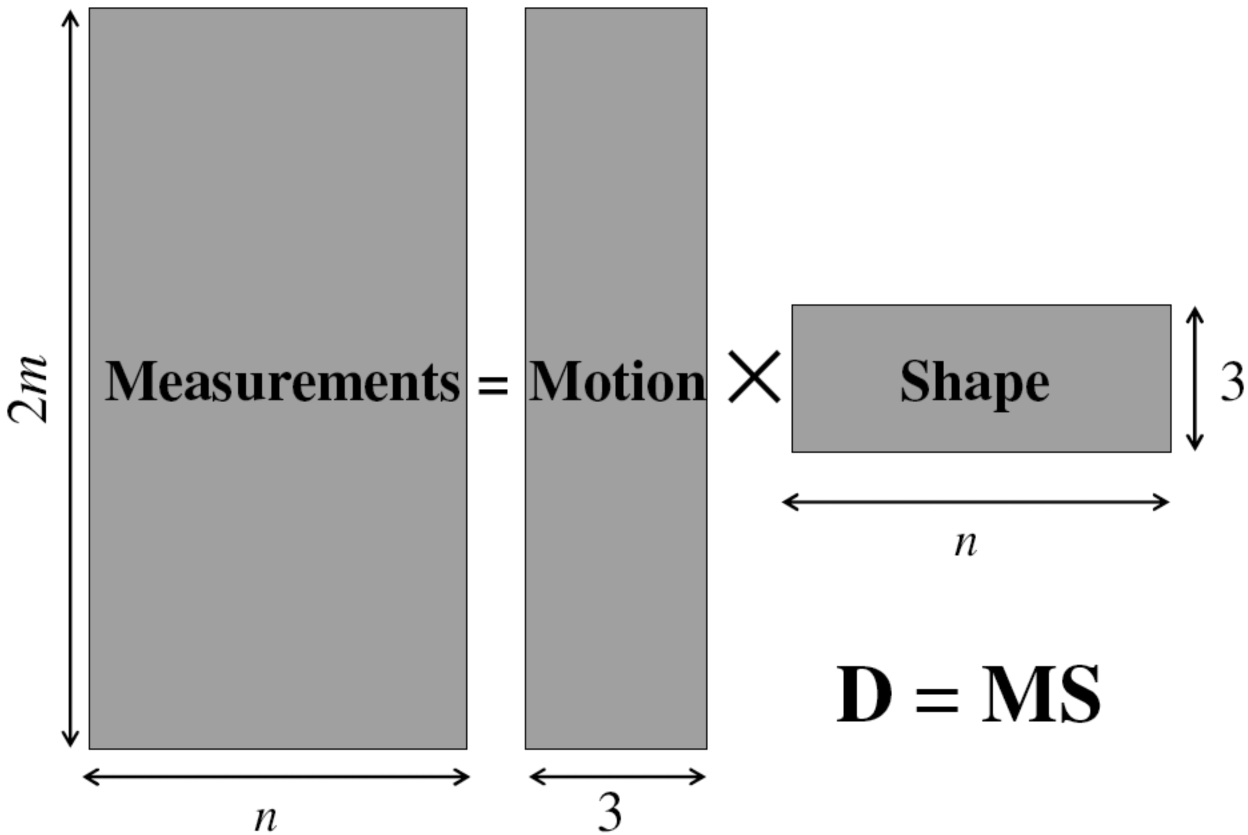
$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_m \end{bmatrix}}_{\text{cameras } (2m \times 3)} \underbrace{\begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}}_{\text{points } (3 \times n)}$$

The measurement matrix $\mathbf{D} = \mathbf{MS}$ must have rank 3!

C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

Lazebnik

Factorizing the measurement matrix

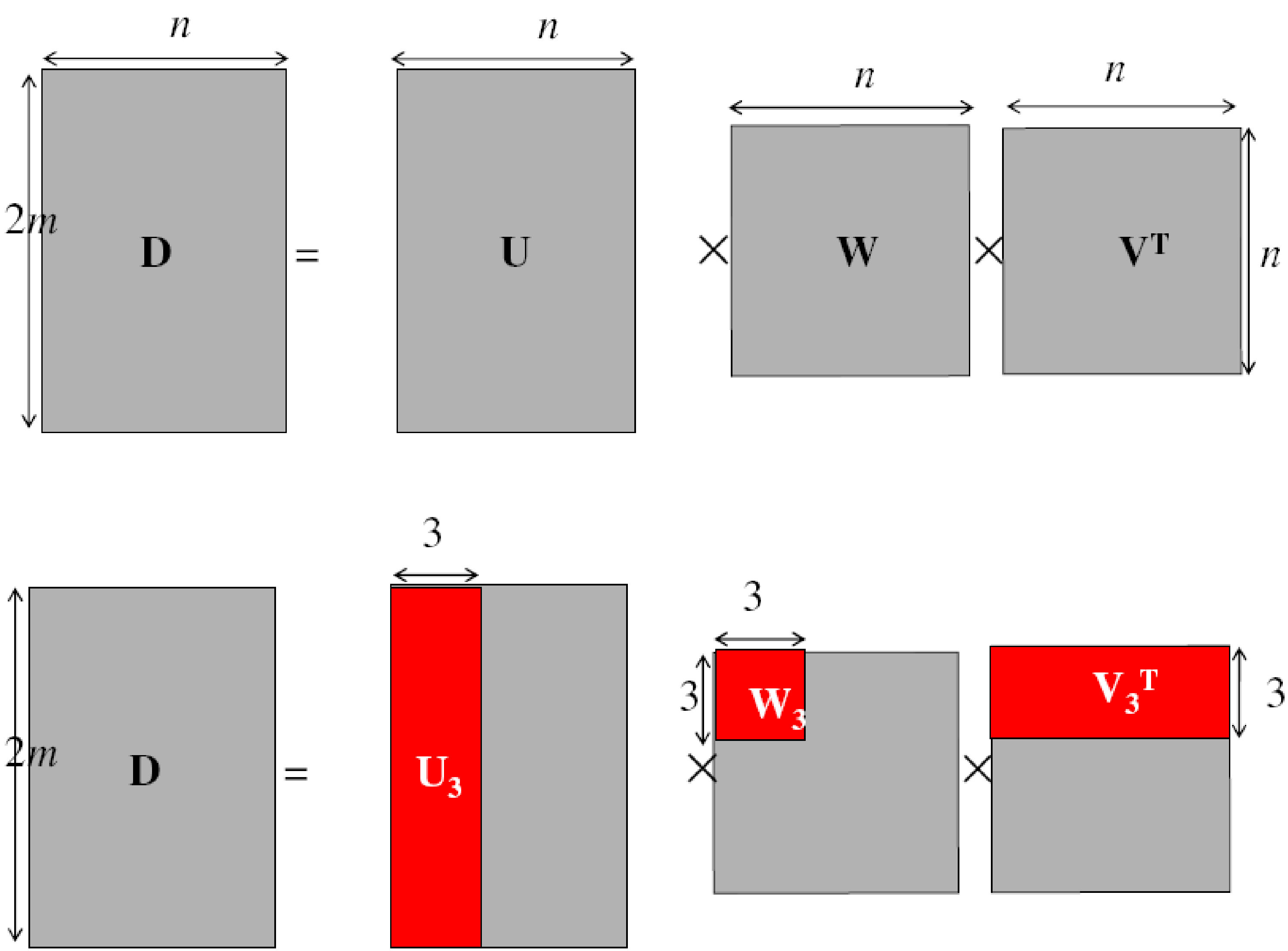


$$\mathbf{D} = \mathbf{MS}$$

Source: M. Hebert

Factorizing the measurement matrix

- Singular value decomposition of D:

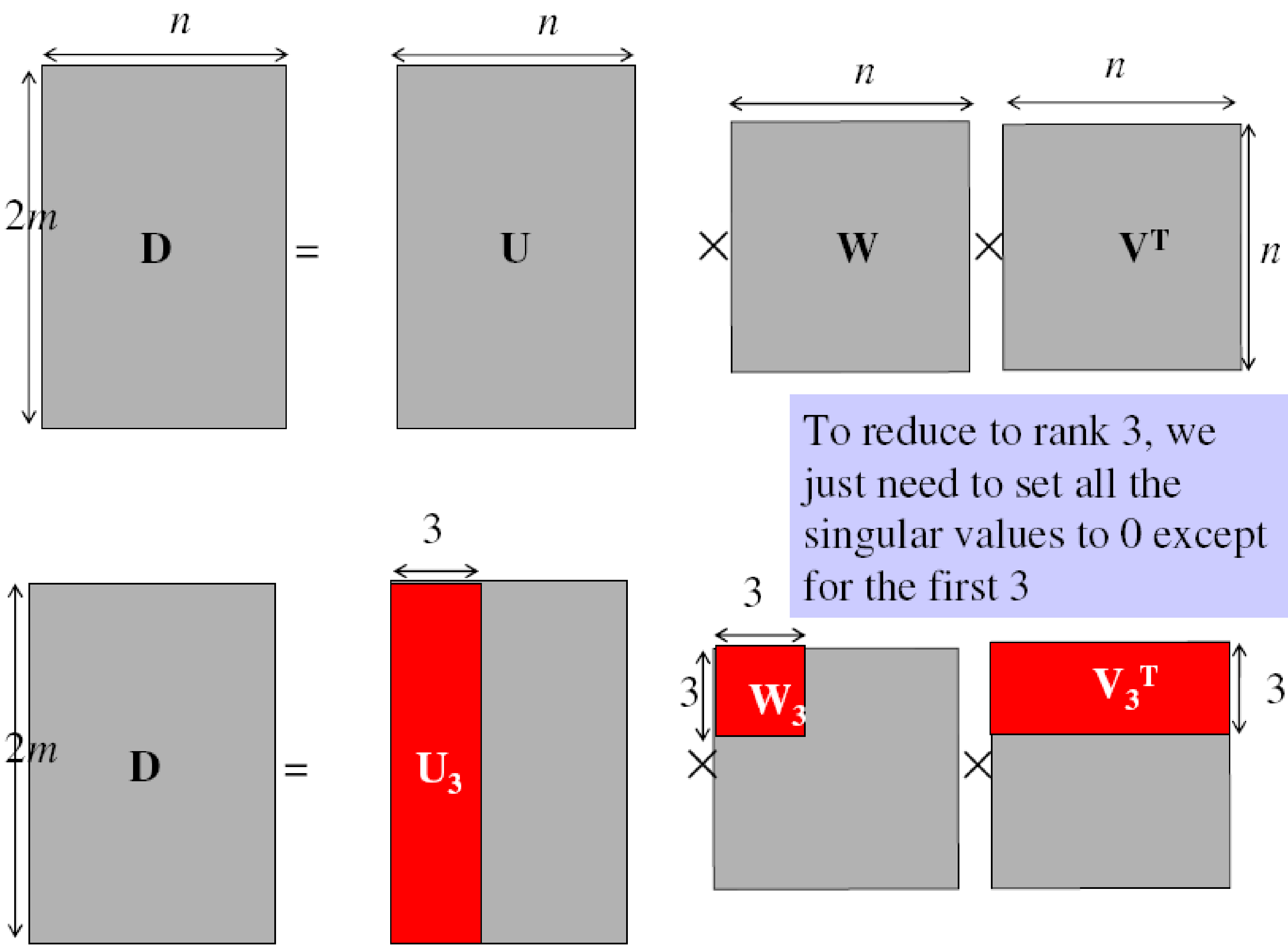


Source: M. Hebert

Factorizing the measurement matrix

UNIVERSITÄT BONN

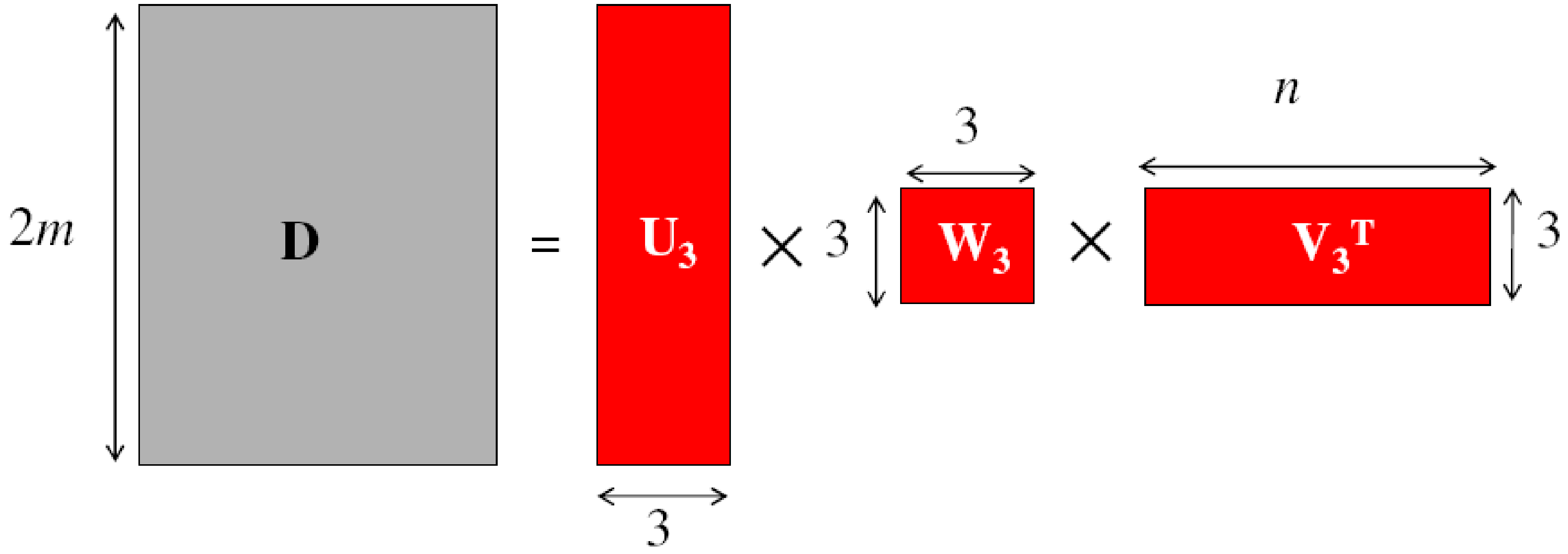
- Singular value decomposition of D:



Source: M. Hebert

Factorizing the measurement matrix

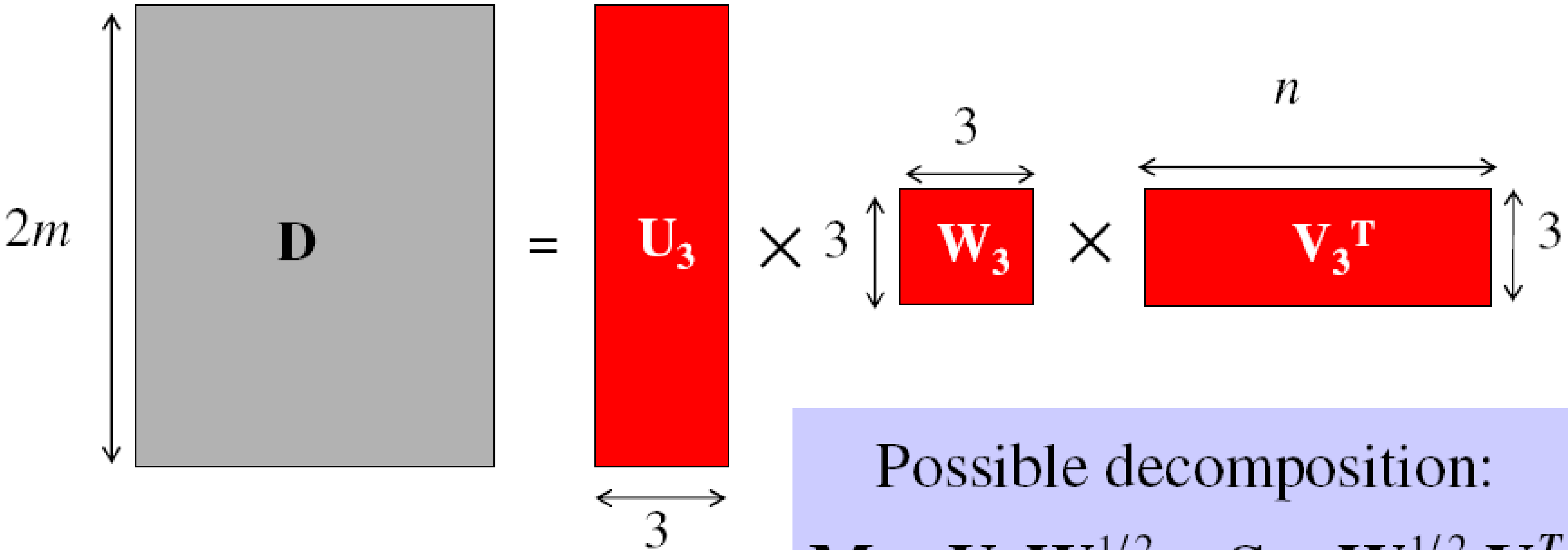
- Obtaining a factorization from SVD:



Source: M. Hebert

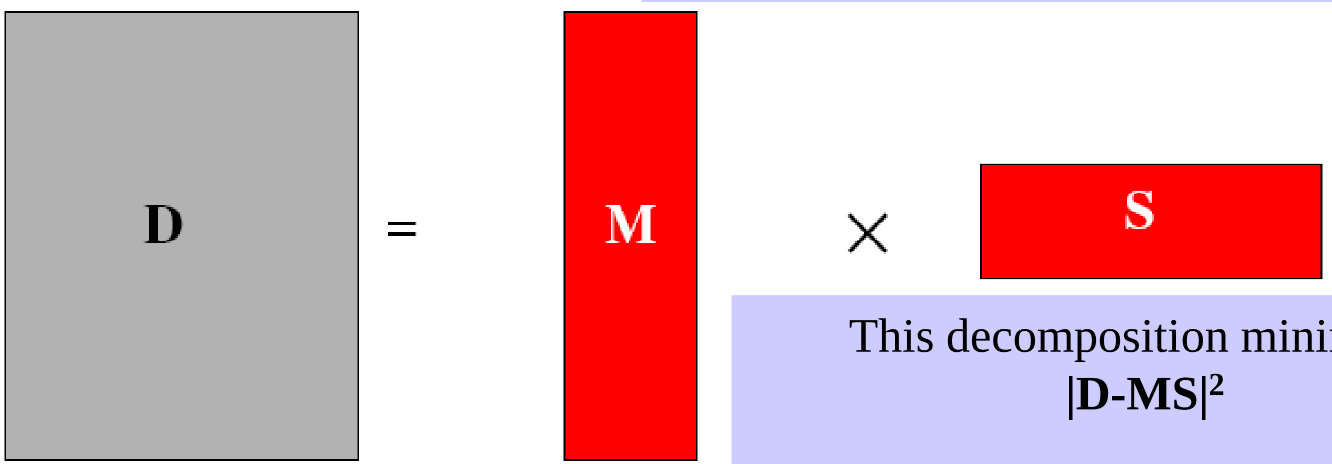
Factorizing the measurement matrix

- Obtaining a factorization from SVD:



Possible decomposition:

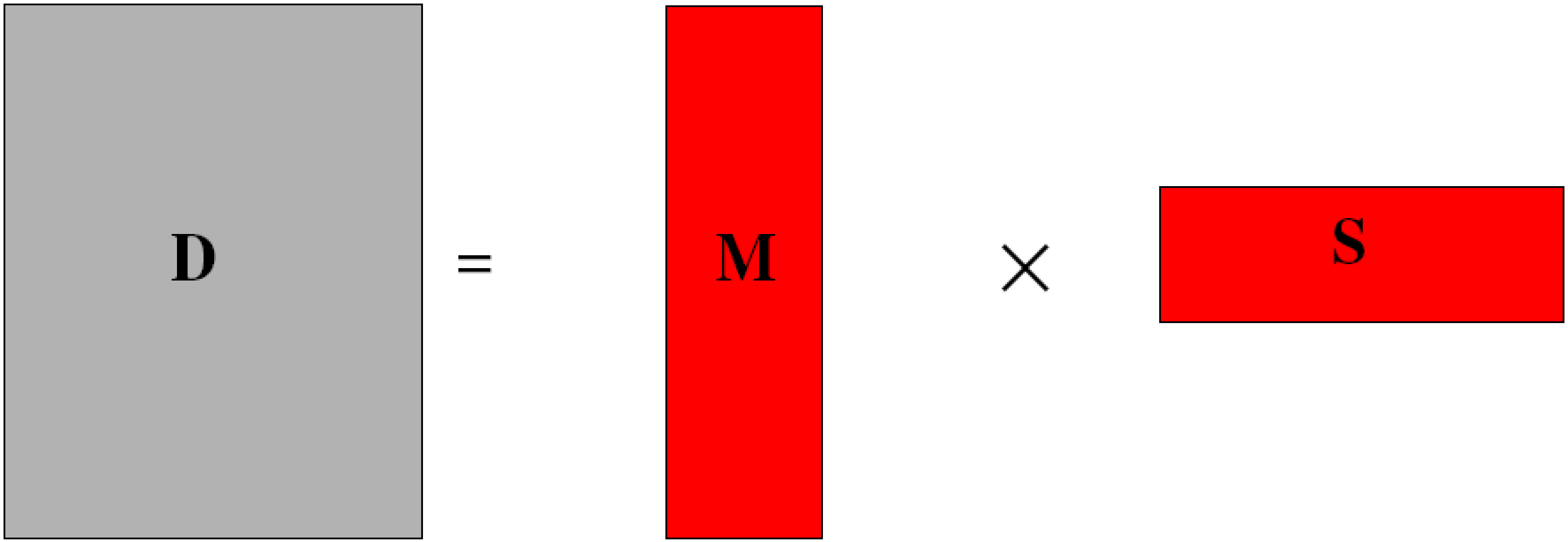
$$\mathbf{M} = \mathbf{U}_3 \mathbf{W}_3^{1/2} \qquad \mathbf{S} = \mathbf{W}_3^{1/2}\, \mathbf{V}_3^T$$



This decomposition minimizes
$|\mathbf{D}\text{-}\mathbf{MS}|^2$

Source: M. Hebert

Affine ambiguity

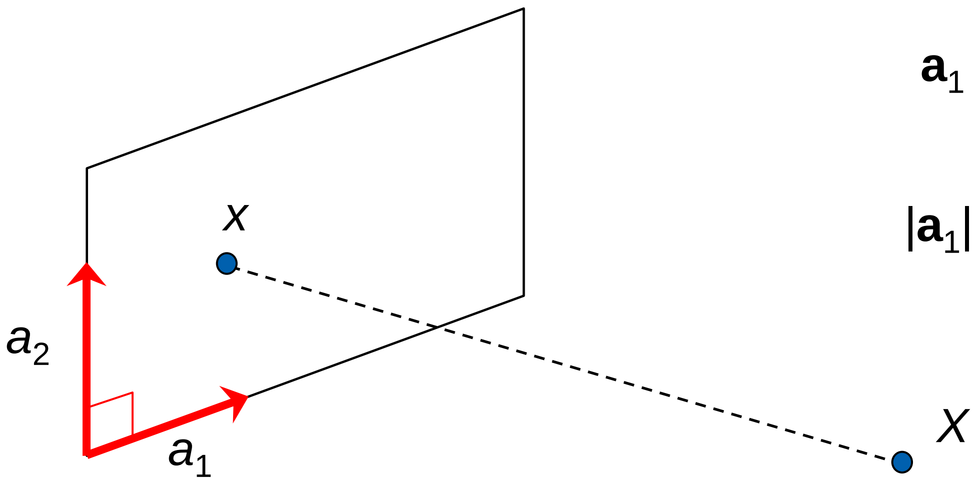


- The decomposition is not unique. We get the same **D** by using any 3×3 matrix **C** and applying the transformations **M** → **MC, S** →**C^{-1}S**
- That is because we have only an affine transformation and we have not enforced any Euclidean constraints (like forcing the image axes to be perpendicular, for example)

Source: M. Hebert

Eliminating the affine ambiguity

- Orthographic: image axes are perpendicular and of unit length



$$\mathbf{a}_1 \cdot \mathbf{a}_2 = 0$$

$$|\mathbf{a}_1|^2 = |\mathbf{a}_2|^2 = 1$$

Source: M. Hebert

Solve for orthographic constraints

UNIVERSITÄT BONN

- Three equations for each image i

$$\begin{aligned} \mathbf{m}_{i1}^T \mathbf{C}\mathbf{C}^T \mathbf{m}_{i1} &= 1 \\ \mathbf{m}_{i2}^T \mathbf{C}\mathbf{C}^T \mathbf{m}_{i2} &= 1 \quad \text{where} \quad \mathbf{M}_i = \begin{bmatrix} \mathbf{m}_{i1}^T \\ \mathbf{m}_{i2}^T \end{bmatrix} \\ \mathbf{m}_{i1}^T \mathbf{C}\mathbf{C}^T \mathbf{m}_{i2} &= 0 \end{aligned}$$

- Two options:
  - Solve for **C** (Newton's method, quadratic)
  - Solve linearly **L = CC^T**
  - Recover **C** from **L** by SVD or Cholesky decomposition: **L = CC^T**
- Update **M** and **S**: **M' = MC, S' = C^{-1}S**

Algorithm summary

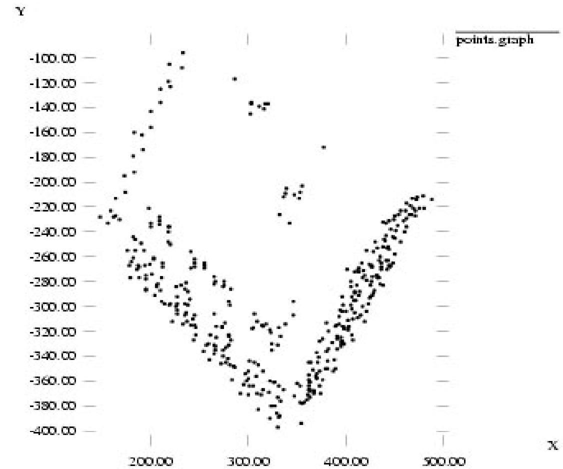
- Given: m images and n features $\mathbf{x}_{ij}$
- For each image i, center the feature coordinates
- Construct a $2m \times n$ measurement matrix $\mathbf{D}$:
  - Column j contains the projection of point j in all views
  - Row i contains one coordinate of the projections of all the n points in image i
- Factorize $\mathbf{D}$:
  - Compute SVD: $\mathbf{D} = \mathbf{U}\ \mathbf{W}\ \mathbf{V}^{\mathsf{T}}$
  - Create $\mathbf{U}_3$ by taking the first 3 columns of $\mathbf{U}$
  - Create $\mathbf{V}_3$ by taking the first 3 columns of $\mathbf{V}$
  - Create $\mathbf{W}_3$ by taking the upper left 3×3 block of $\mathbf{W}$
- Create the motion and shape matrices:
  - $\mathbf{M} = \mathbf{U}_3\mathbf{W}_3^{1/2}$ and $\mathbf{S} = \mathbf{W}_3^{1/2}\ \mathbf{V}_3^{\mathsf{T}}$ (**or** $\mathbf{M} = \mathbf{U}_3$ and $\mathbf{S} = \mathbf{W}_3\mathbf{V}_3^{\mathsf{T}}$)
- Eliminate affine ambiguity

Source: M. Hebert

Reconstruction results

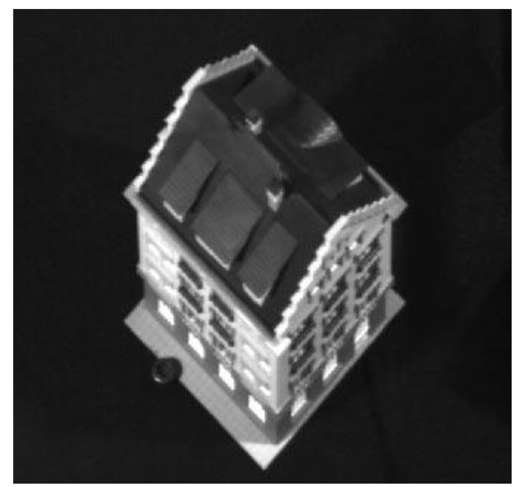
1

60



120

150



C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

Lazebnik

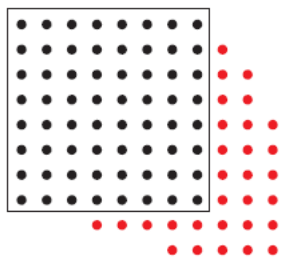
Dealing with missing data

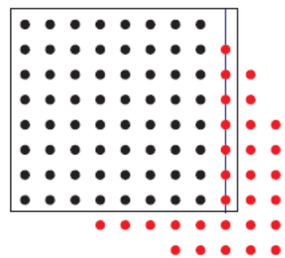
- So far, we have assumed that all points are visible in all views
- In reality, the measurement matrix typically looks something like this:

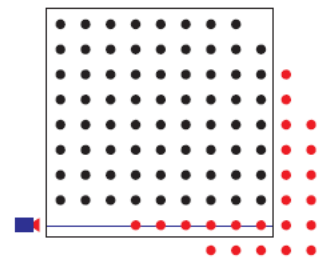
Lazebnik

Dealing with missing data

- Possible solution: decompose matrix into dense sub-blocks, factorize each sub-block, and fuse the results
  - Finding dense maximal sub-blocks of the matrix is NP-complete (equivalent to finding maximal cliques in a graph)
- Incremental bilinear refinement







(1) Perform factorization on a dense sub-block

(2) Solve for a new 3D point visible by at least two known cameras (linear least squares)

(3) Solve for a new camera that sees at least three known 3D points (linear least squares)

F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. Segmenting, Modeling, and Matching Video Clips Containing Multiple Moving Objects. PAMI 2007.

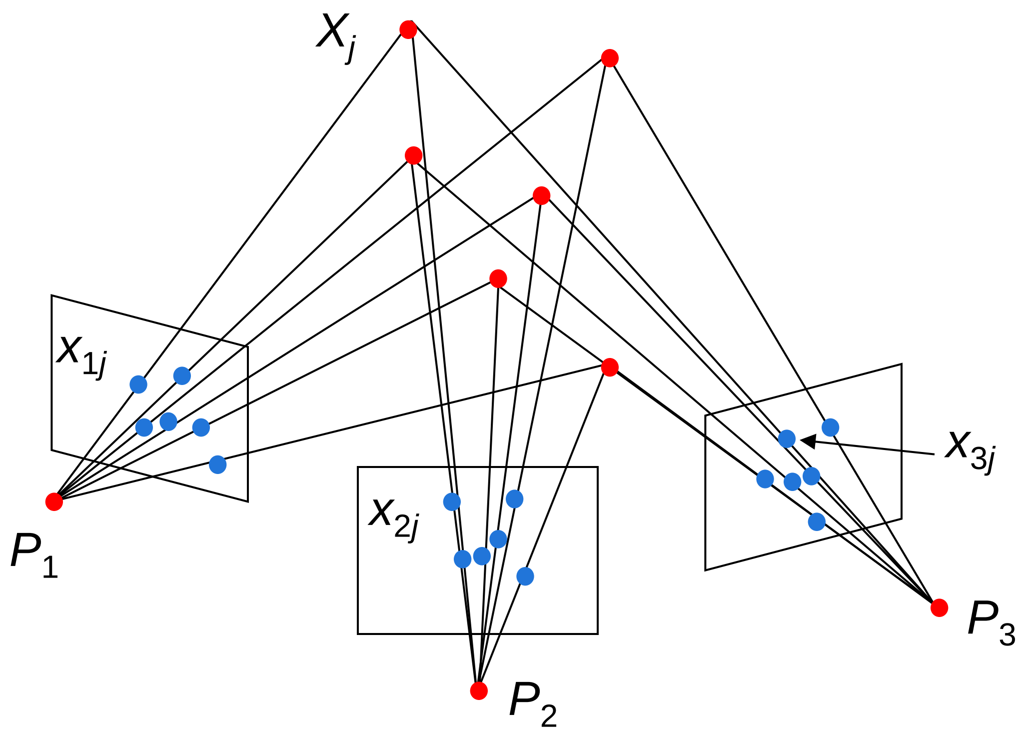
Lazebnik

Projective structure from motion

- Given: *m* images of *n* fixed 3D points

$$\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j, \quad i = 1, \dots, m, \quad j = 1, \dots, n$$

- Problem: estimate *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the *mn* correspondences $\mathbf{x}_{ij}$



Lazebnik

Projective structure from motion

- Given: *m* images of *n* fixed 3D points

$$\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j, \quad i = 1, \dots, m, \quad j = 1, \dots, n$$

- Problem: estimate *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the *mn* correspondences $\mathbf{x}_{ij}$
- With no calibration info, cameras and points can only be recovered up to a 4x4 projective transformation **Q**:

$$\mathbf{X} \rightarrow \mathbf{QX}, \mathbf{P} \rightarrow \mathbf{PQ}^{-1}$$

- We can solve for structure and motion when

$$2mn >= 11m + 3n - 15$$

- For two cameras, at least 7 points are needed

Lazebnik

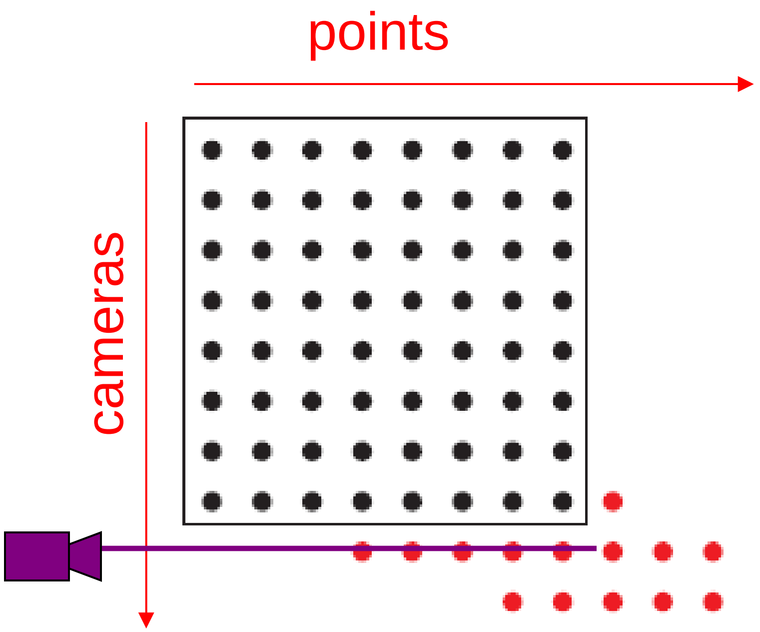
Projective SFM: Two-camera case

- Compute fundamental matrix $\mathbf{F}$ between the two views
- First camera matrix: $[\mathbf{I}|\mathbf{0}]$
- Second camera matrix: $[\mathbf{A}|\mathbf{b}]$
- Then $\mathbf{b}$ is the epipole ($\mathbf{F}^{T}\mathbf{b} = 0$), $\mathbf{A} = -[\mathbf{b}_{\times}]\mathbf{F}$

Lazebnik

Sequential structure from motion

- Initialize motion from two images using fundamental matrix
- Initialize structure by triangulation
- For each additional view:
  - Determine projection matrix of new camera using all the known 3D points that are visible in its image – *calibration*



Lazebnik

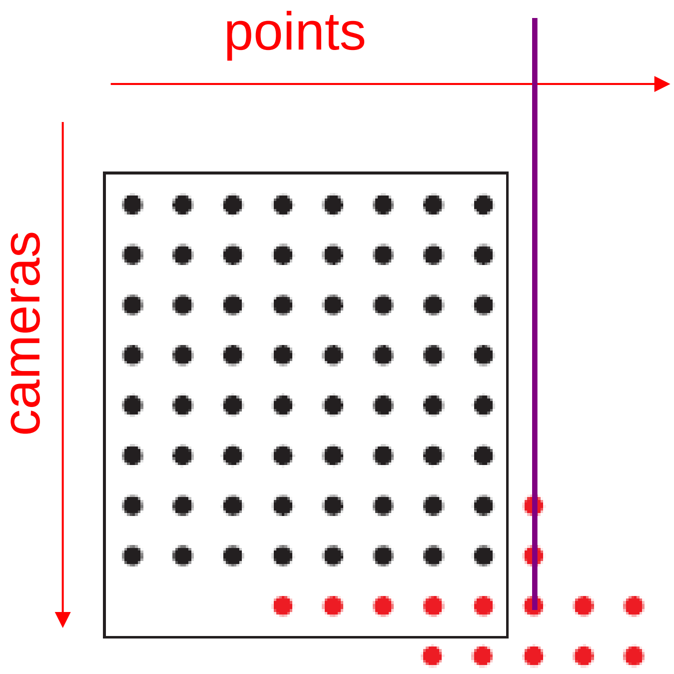
Sequential structure from motion

- Initialize motion from two images using fundamental matrix

- Initialize structure by triangulation

- For each additional view:
  - Determine projection matrix of new camera using all the known 3D points that are visible in its image – *calibration*

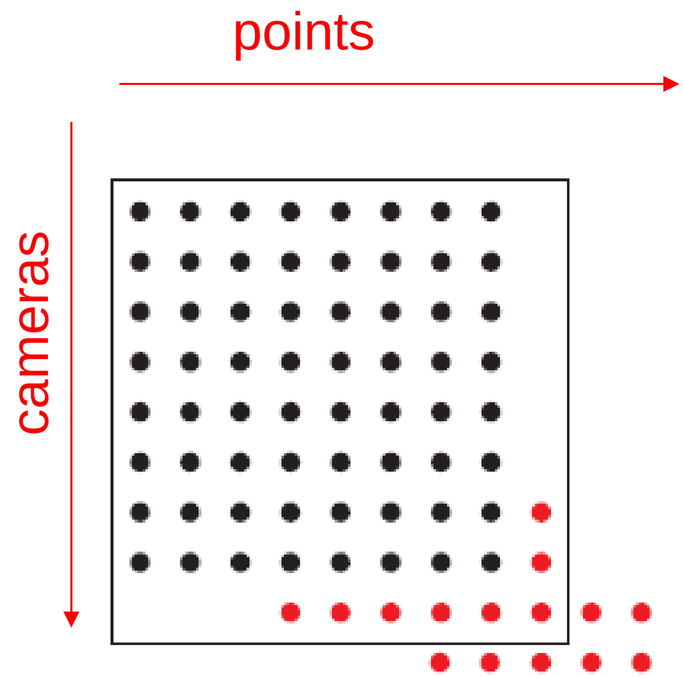
  - Refine and extend structure: compute new 3D points, re-optimize existing points that are also seen by this camera – *triangulation*



Lazebnik

Sequential structure from motion

- Initialize motion from two images using fundamental matrix
- Initialize structure by triangulation
- For each additional view:
  - Determine projection matrix of new camera using all the known 3D points that are visible in its image – *calibration*
  - Refine and extend structure: compute new 3D points, re-optimize existing points that are also seen by this camera – *triangulation*
- Refine structure and motion: bundle adjustment

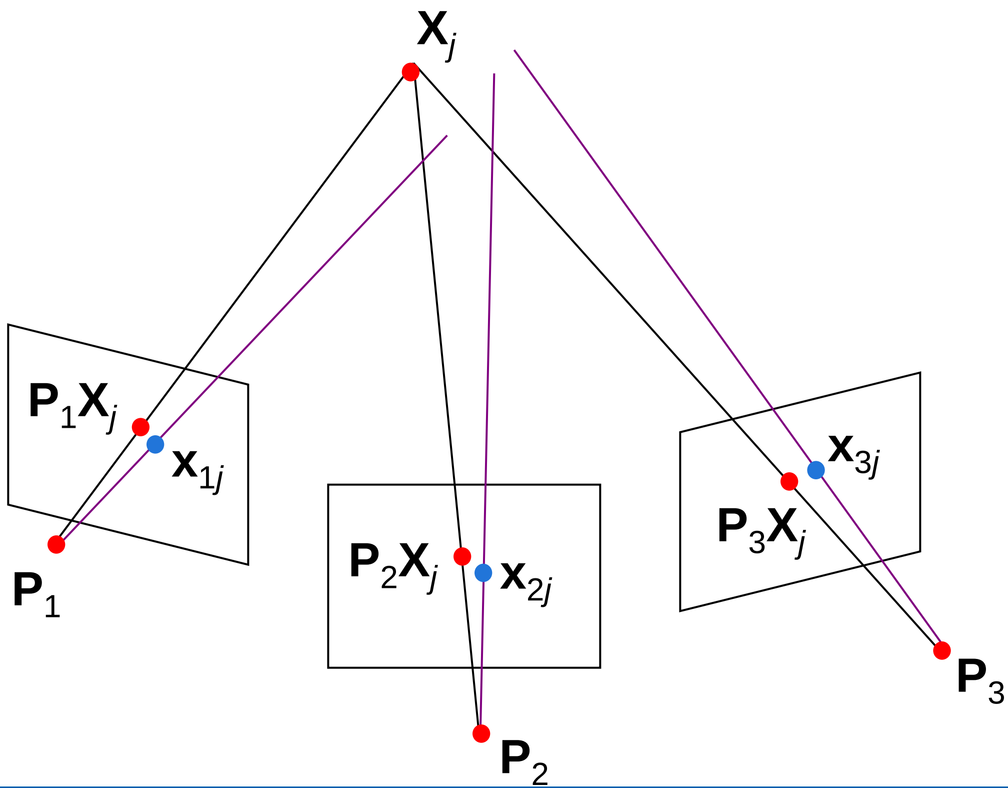


Lazebnik

Bundle adjustment

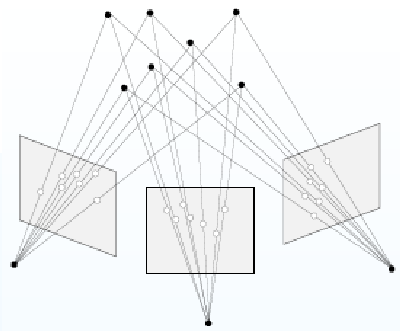
- Non-linear method for refining structure and motion
- Minimizing reprojection error

$$E(\mathbf{P}, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} D\left(\mathbf{x}_{ij}, \mathbf{P}_i \mathbf{X}_j\right)^2$$



Lazebnik

Bundle adjustment



- Assume n 3D points are seen in m views. Illustration with $n = 7, m = 3$
- Let $\mathbf{x}_{ij}$ be the projection of the i-th point on image j, $\mathbf{a}_j$ the vector of parameters for camera j and $\mathbf{b}_i$ the vector of parameters for point i
- BA minimizes the *reprojection error* over all point and camera parameters: $\min_{\mathbf{a}_j, \mathbf{b}_i} \sum_{i=1}^{n} \sum_{j=1}^{m} v_{ij} d(\mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i),\ \mathbf{x}_{ij})^2$, $\mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i)$ being the predicted projection of point i on image j, $d(.,.)$ the Euclidean distance between image points and $v_{ij} = 1$ iff point i is visible in image j
- This is a large problem: if κ, λ are the dimensions of the $\mathbf{a}_j$ & $\mathbf{b}_i$, the total number of parameters involved in BA is $m\kappa + n\lambda$

Lourakis

Bundle adjustment

UNIVERSITÄT BONN

- A parameter vector $\mathbf{P}$ is defined by partitioning parameters as $\mathbf{P} = (\mathbf{a}_1{}^T, \ldots, \mathbf{a}_m{}^T, \ldots, \mathbf{b}_1{}^T, \ldots, \mathbf{b}_n{}^T)^T$
- A measurement vector $\mathbf{X}$ is defined as $(\mathbf{x}_{11}{}^T, \ldots, \mathbf{x}_{1m}{}^T, \; \mathbf{x}_{21}{}^T, \ldots, \mathbf{x}_{2m}{}^T, \; \ldots, \; \mathbf{x}_{n1}{}^T, \ldots, \mathbf{x}_{nm}{}^T)^T$
- For each parameter vector, an estimated measurement $\hat{\mathbf{X}}$ is $(\hat{\mathbf{x}}_{11}{}^T, \ldots, \hat{\mathbf{x}}_{1m}{}^T, \; \hat{\mathbf{x}}_{21}{}^T, \ldots, \hat{\mathbf{x}}_{2m}{}^T, \; \ldots, \; \hat{\mathbf{x}}_{n1}{}^T, \ldots, \hat{\mathbf{x}}_{nm}{}^T)^T$ and the corresponding error $(\epsilon_{11}{}^T, \ldots, \epsilon_{1m}{}^T, \; \epsilon_{21}{}^T, \ldots, \epsilon_{2m}{}^T, \; \ldots, \; \epsilon_{n1}{}^T, \ldots, \epsilon_{nm}{}^T)^T$, where $\hat{\mathbf{x}}_{ij} \equiv \mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i)$ and $\epsilon_{ij} \equiv \mathbf{x}_{ij} - \hat{\mathbf{x}}_{ij} \; \forall \; i, j$
- With the above definitions, BA corresponds to minimizing $\sum_{i=1}^{n} \sum_{j=1}^{m} ||\epsilon_{ij}||^2 = ||\mathbf{X} - \hat{\mathbf{X}}||^2$ over $\mathbf{P}$, which is a nonlinear least squares problem

Lourakis

Levenberg-Marquardt

UNIVERSITÄT BONN

- Let $f : \mathcal{R}^m \to \mathcal{R}^n$. Given an initial estimate $\mathbf{p}_0 \in \mathcal{R}^m$ and a measurement vector $\mathbf{x} \in \mathcal{R}^n$, LM seeks to find $\mathbf{p}^+$ minimizing $\epsilon^T\epsilon$, $\epsilon = \mathbf{x} - f(\mathbf{p})$

- Note that this is a (nonlinear) least squares problem since $\epsilon^T\epsilon = ||\mathbf{x} - f(\mathbf{p})||^2$, $||.||$ being the L2 norm

- The minimizer can be found by the Gauss-Newton method, which iteratively linearizes f at $\mathbf{p}$ and determines incremental update steps $\delta_{\mathbf{p}}$ by solving the *normal equations* $\mathbf{J}^T\mathbf{J}\delta_{\mathbf{p}} = \mathbf{J}^T\epsilon$, $\mathbf{J}$ being the Jacobian of f at $\mathbf{p}$ and $\mathbf{J}^T\mathbf{J}$ the approximate Hessian of $||\epsilon||^2$

- To ensure convergence, LM uses *damping*, i.e. adaptively alters the diagonal elements of $\mathbf{J}^T\mathbf{J}$ and solves the *augmented normal equations* $(\mathbf{J}^T\mathbf{J} + \mu\mathbf{I})\delta_{\mathbf{p}} = \mathbf{J}^T\epsilon, \ \mu > 0$

Lourakis

Levenberg-Marquardt

UNIVERSITÄT BONN

- The Jacobian $\mathbf{J} = \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{P}}$ has a block structure $[\,\mathbf{A}|\mathbf{B}\,]$, where $\mathbf{A} = \left[\, \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{a}} \,\right]$ and $\mathbf{B} = \left[\, \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{b}} \,\right]$
- The LM updating vector δ is partitioned as $({\delta_{\mathbf{a}}}^T,\ {\delta_{\mathbf{b}}}^T)^T$
- The normal equations become

$$\left[\begin{array}{c|c} \mathbf{A}^T\mathbf{A} & \mathbf{A}^T\mathbf{B} \\ \hline \mathbf{B}^T\mathbf{A} & \mathbf{B}^T\mathbf{B} \end{array}\right] \left(\begin{array}{c} \delta_{\mathbf{a}} \\ \hline \delta_{\mathbf{b}} \end{array}\right) = \left(\begin{array}{c} \mathbf{A}^T\epsilon \\ \hline \mathbf{B}^T\epsilon \end{array}\right)$$

- The lhs matrix above is sparse due to $\mathbf{A}$ and $\mathbf{B}$ being sparse: $\frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_k} = \mathbf{0},\ \forall\ j \neq k$ and $\frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_k} = \mathbf{0},\ \forall\ i \neq k$
- This is the so-called *primary structure* of BA

Lourakis

Example (n=4,m=3)

- Assume all points are seen in all images

- The measurement vector $\mathbf{x} =$
$(\mathbf{x}_{11}{}^T, \mathbf{x}_{12}{}^T, \mathbf{x}_{13}{}^T, \mathbf{x}_{21}{}^T, \mathbf{x}_{22}{}^T, \mathbf{x}_{23}{}^T, \mathbf{x}_{31}{}^T, \mathbf{x}_{32}{}^T, \mathbf{x}_{33}{}^T, \mathbf{x}_{41}{}^T, \mathbf{x}_{42}{}^T, \mathbf{x}_{43}{}^T)^T$

- The parameter vector
$\mathbf{P} = (\mathbf{a}_1{}^T, \mathbf{a}_2{}^T, \mathbf{a}_3{}^T, \mathbf{b}_1{}^T, \mathbf{b}_2{}^T, \mathbf{b}_3{}^T, \mathbf{b}_4{}^T)^T$

- The LM updating vector
$\delta = (\delta_{\mathbf{a}_1}{}^T, \delta_{\mathbf{a}_2}{}^T, \delta_{\mathbf{a}_3}{}^T, \delta_{\mathbf{b}_1}{}^T, \delta_{\mathbf{b}_2}{}^T, \delta_{\mathbf{b}_3}{}^T, \delta_{\mathbf{b}_4}{}^T)^T$

- Let $\mathbf{A}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j}$ and $\mathbf{B}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$

Lourakis

Jacobian (n=4,m=3)

$$
\frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{P}} =
\begin{array}{c}
\\
\mathbf{x}_{11} \\ \mathbf{x}_{12} \\ \mathbf{x}_{13} \\ \mathbf{x}_{21} \\ \mathbf{x}_{22} \\ \mathbf{x}_{23} \\ \mathbf{x}_{31} \\ \mathbf{x}_{32} \\ \mathbf{x}_{33} \\ \mathbf{x}_{41} \\ \mathbf{x}_{42} \\ \mathbf{x}_{43}
\end{array}
\begin{array}{c}
\begin{array}{ccccccc}
\mathbf{a}_1{}^T & \mathbf{a}_2{}^T & \mathbf{a}_3{}^T & \mathbf{b}_1{}^T & \mathbf{b}_2{}^T & \mathbf{b}_3{}^T & \mathbf{b}_4{}^T
\end{array} \\
\begin{pmatrix}
\mathbf{A}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{A}_{12} & \mathbf{0} & \mathbf{B}_{12} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{A}_{13} & \mathbf{B}_{13} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{A}_{21} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{21} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{A}_{22} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{22} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{A}_{23} & \mathbf{0} & \mathbf{B}_{23} & \mathbf{0} & \mathbf{0} \\
\mathbf{A}_{31} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{31} & \mathbf{0} \\
\mathbf{0} & \mathbf{A}_{32} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{32} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{A}_{33} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{33} & \mathbf{0} \\
\mathbf{A}_{41} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{41} \\
\mathbf{0} & \mathbf{A}_{42} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{42} \\
\mathbf{0} & \mathbf{0} & \mathbf{A}_{43} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{43}
\end{pmatrix}
\end{array}
$$

Lourakis

Hessian (n=4,m=3)

- Approximate Hessian in block form:

$$
\mathbf{J}^T\mathbf{J} = \begin{array}{c c}
 & \begin{array}{ccccccc} \mathbf{a}_1{}^T & \mathbf{a}_2{}^T & \mathbf{a}_3{}^T & \mathbf{b}_1{}^T & \mathbf{b}_2{}^T & \mathbf{b}_3{}^T & \mathbf{b}_4{}^T \end{array} \\
\begin{array}{c} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \\ \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \\ \mathbf{b}_4 \end{array} &
\begin{pmatrix}
\mathbf{U}_1 & 0 & 0 & \mathbf{W}_{11} & \mathbf{W}_{21} & \mathbf{W}_{31} & \mathbf{W}_{41} \\
0 & \mathbf{U}_2 & 0 & \mathbf{W}_{12} & \mathbf{W}_{22} & \mathbf{W}_{32} & \mathbf{W}_{42} \\
0 & 0 & \mathbf{U}_3 & \mathbf{W}_{13} & \mathbf{W}_{23} & \mathbf{W}_{33} & \mathbf{W}_{43} \\
\mathbf{W}_{11}^T & \mathbf{W}_{12}^T & \mathbf{W}_{13}^T & \mathbf{V}_1 & 0 & 0 & 0 \\
\mathbf{W}_{21}^T & \mathbf{W}_{22}^T & \mathbf{W}_{23}^T & 0 & \mathbf{V}_2 & 0 & 0 \\
\mathbf{W}_{31}^T & \mathbf{W}_{32}^T & \mathbf{W}_{33}^T & 0 & 0 & \mathbf{V}_3 & 0 \\
\mathbf{W}_{41}^T & \mathbf{W}_{42}^T & \mathbf{W}_{43}^T & 0 & 0 & 0 & \mathbf{V}_4
\end{pmatrix}
\end{array}
\equiv \begin{pmatrix} \mathbf{U} & \mathbf{W} \\ \mathbf{W}^T & \mathbf{V} \end{pmatrix},
$$

(2)

where

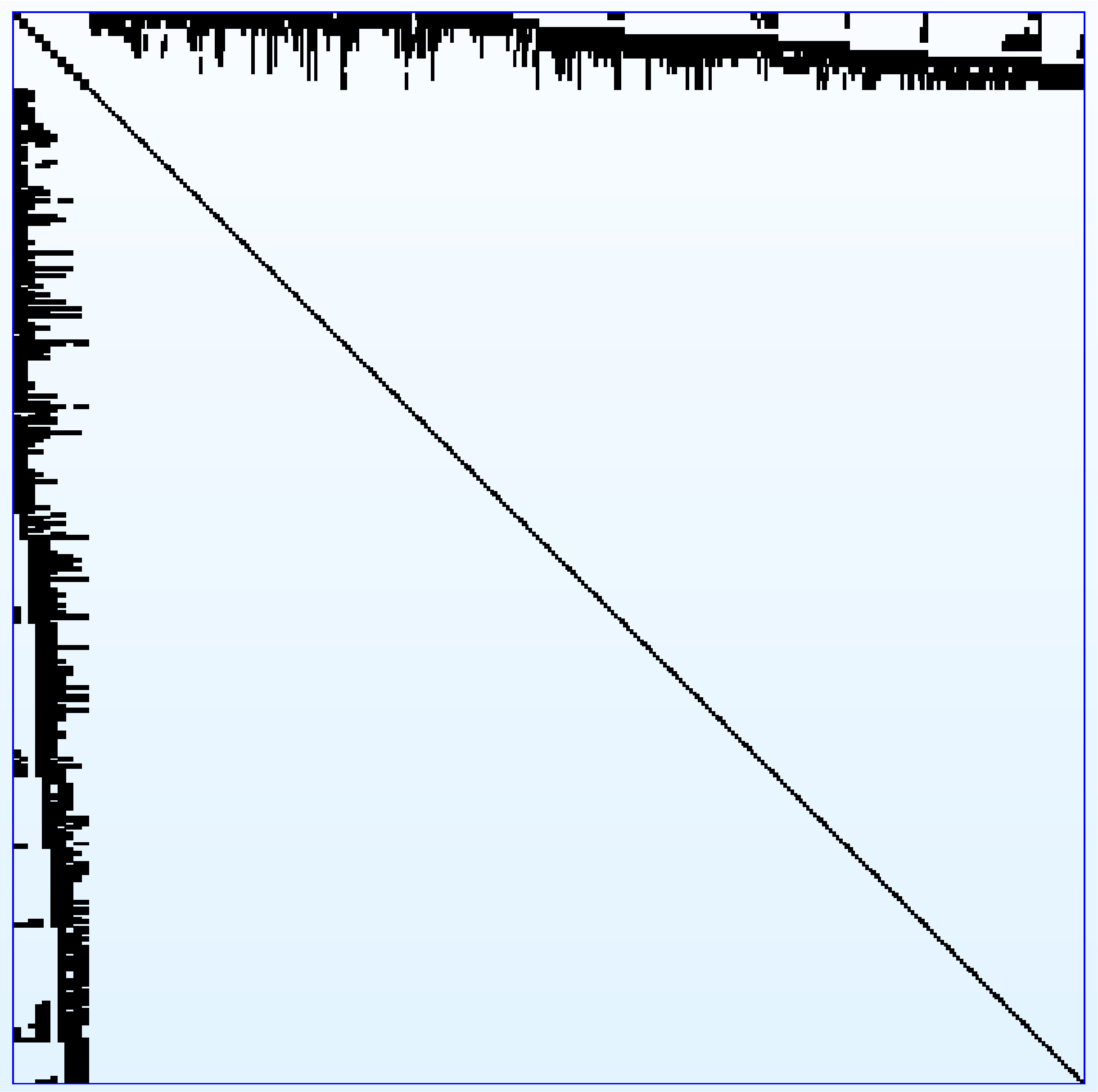
$$\mathbf{U}_j \equiv \textstyle\sum_{i=1}^{4} \mathbf{A}_{ij}^T \mathbf{A}_{ij},$$
$$\mathbf{V}_i \equiv \textstyle\sum_{j=1}^{3} \mathbf{B}_{ij}^T \mathbf{B}_{ij},$$
$$\mathbf{W}_{ij} = \mathbf{A}_{ij}^T \mathbf{B}_{ij}$$

- The above generalize directly to arbitrary n and m

- $\mathbf{U}$ and $\mathbf{V}$ are block diagonal, $\mathbf{W}$ arbitrarily sparse

Lourakis

Hessian (real problem)



Black: non-zero

Lourakis

Bundle adjustment

- Can be solved efficiently due to sparseness
- Good news: open source libraries exist, e.g.,
  sba : A Generic Sparse Bundle Adjustment C/C++ Package (http://users.ics.forth.gr/~lourakis/sba/)

Self-calibration

- Self-calibration (auto-calibration) is the process of determining intrinsic camera parameters directly from uncalibrated images
- For example, when the images are acquired by a single moving camera, we can use the constraint that the intrinsic parameter matrix remains fixed for all the images
  - Compute initial projective reconstruction and find 3D projective transformation matrix $\mathbf{Q}$ such that all camera matrices are in the form $\mathbf{P}_i = \mathbf{K}\,[\mathbf{R}_i \mid \mathbf{t}_i]$
- Can use constraints on the form of the calibration matrix: zero skew

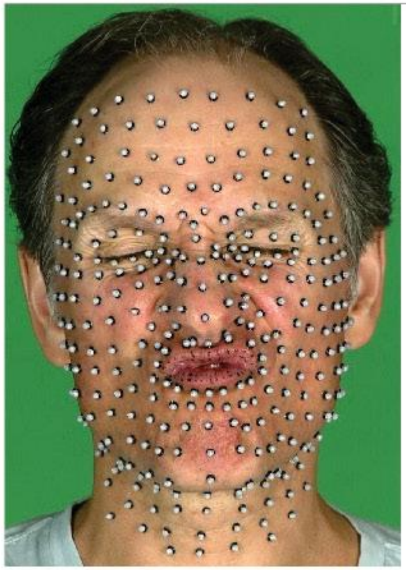
Lazebnik

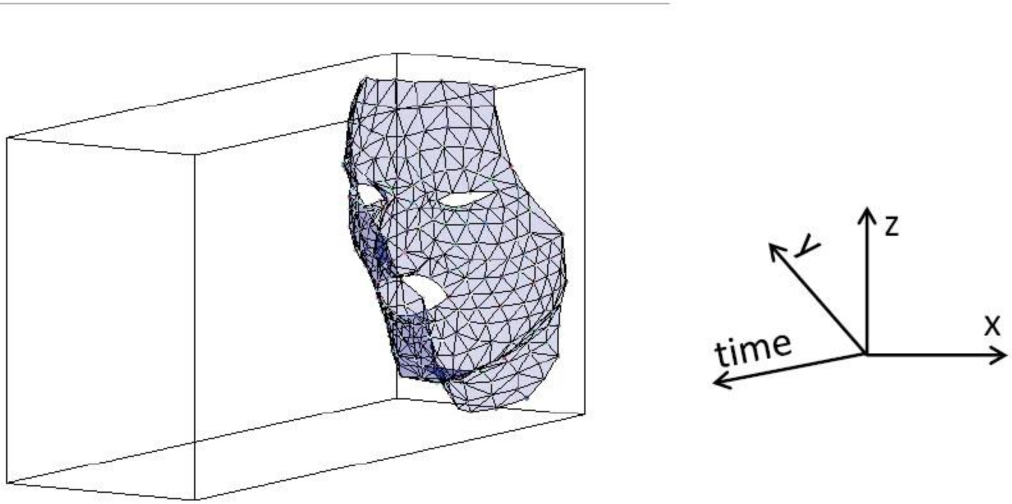
Example

N. Hasler et al. Markerless Motion Capture with Unsynchronized Moving Cameras. CVPR'09.

NONRIGID STRUCTURE

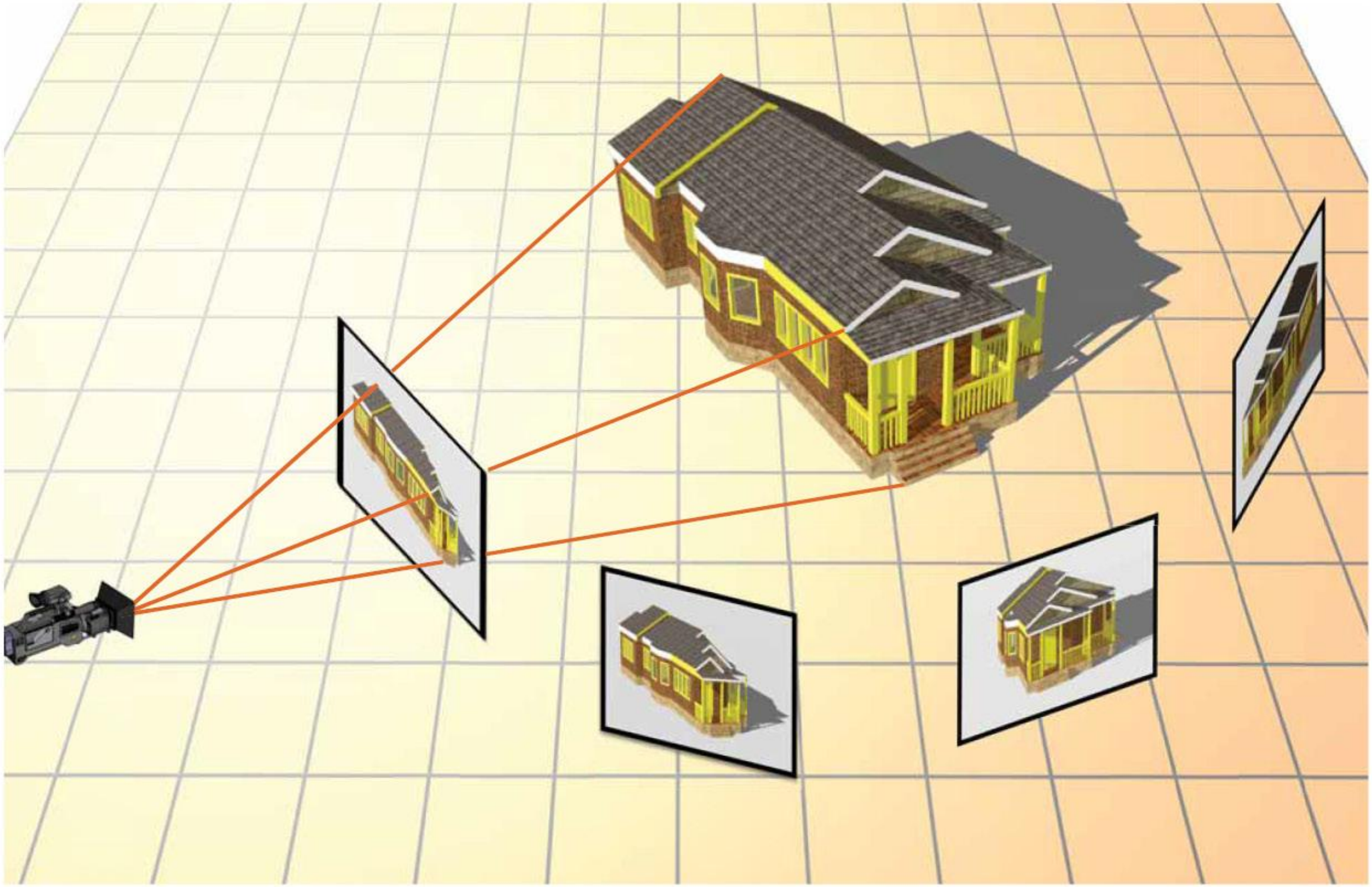
3D Structure That Deforms Over Time



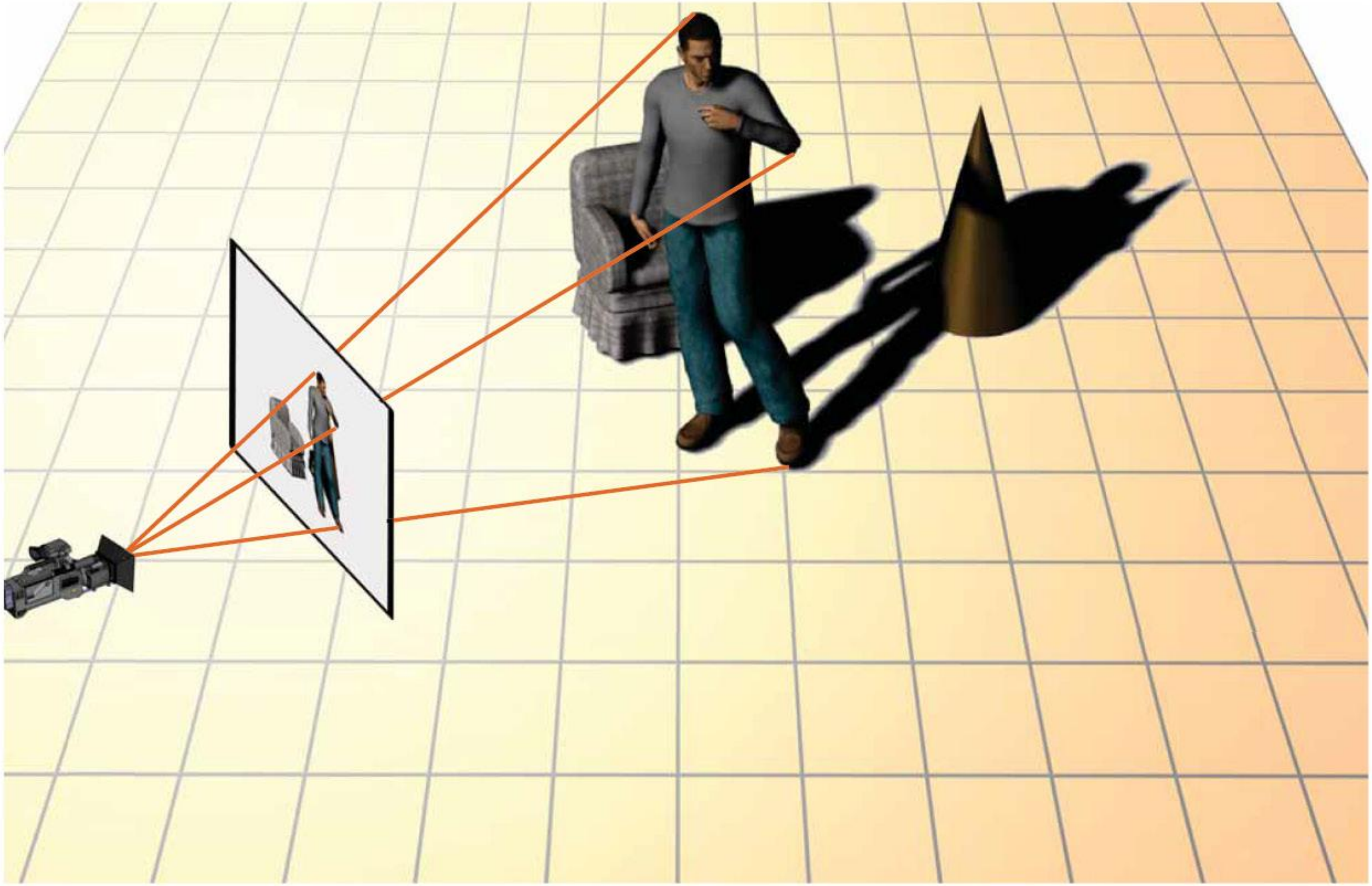


4D DYNAMIC STRUCTURE

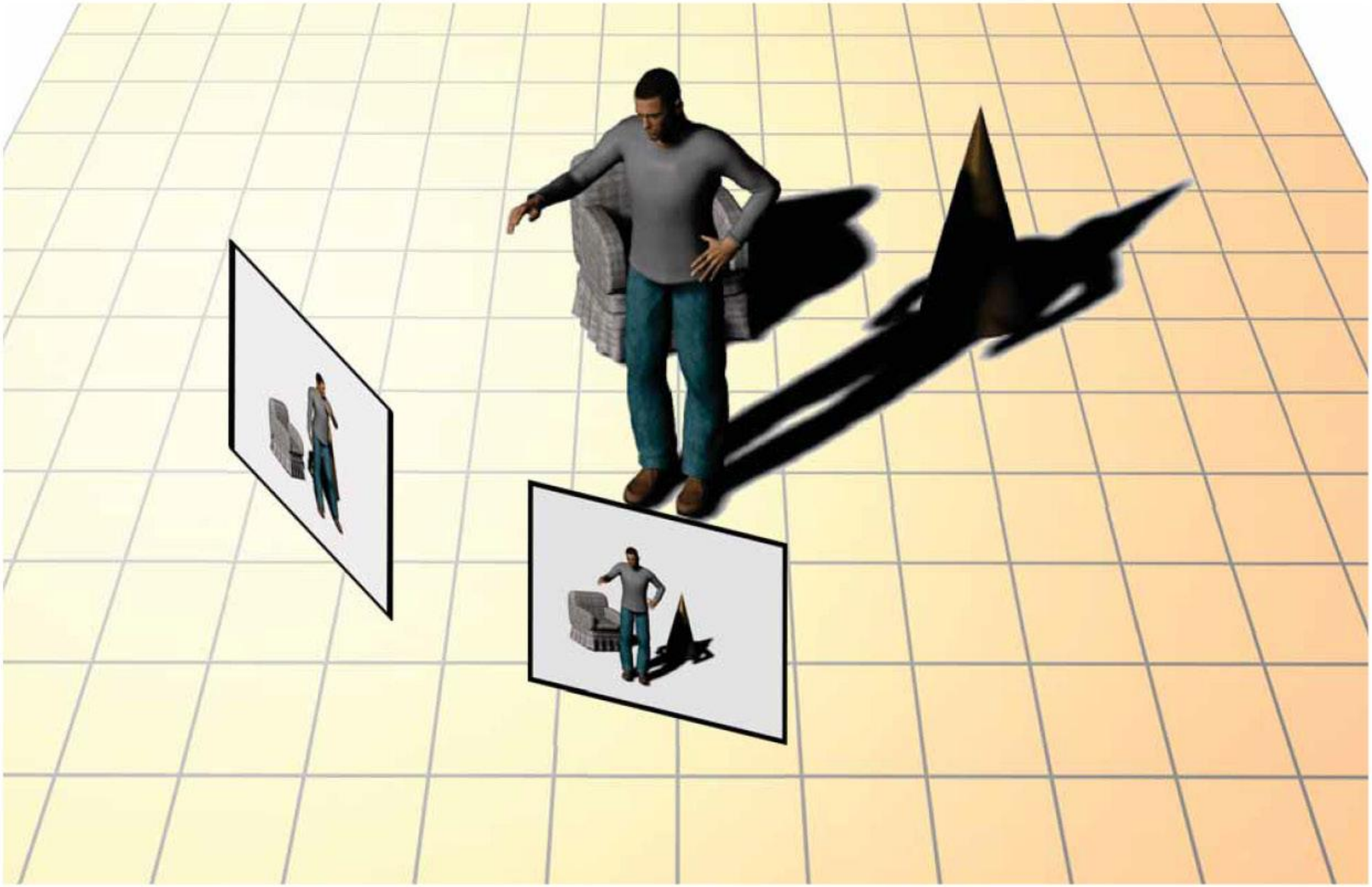
Rigid Structure from Motion



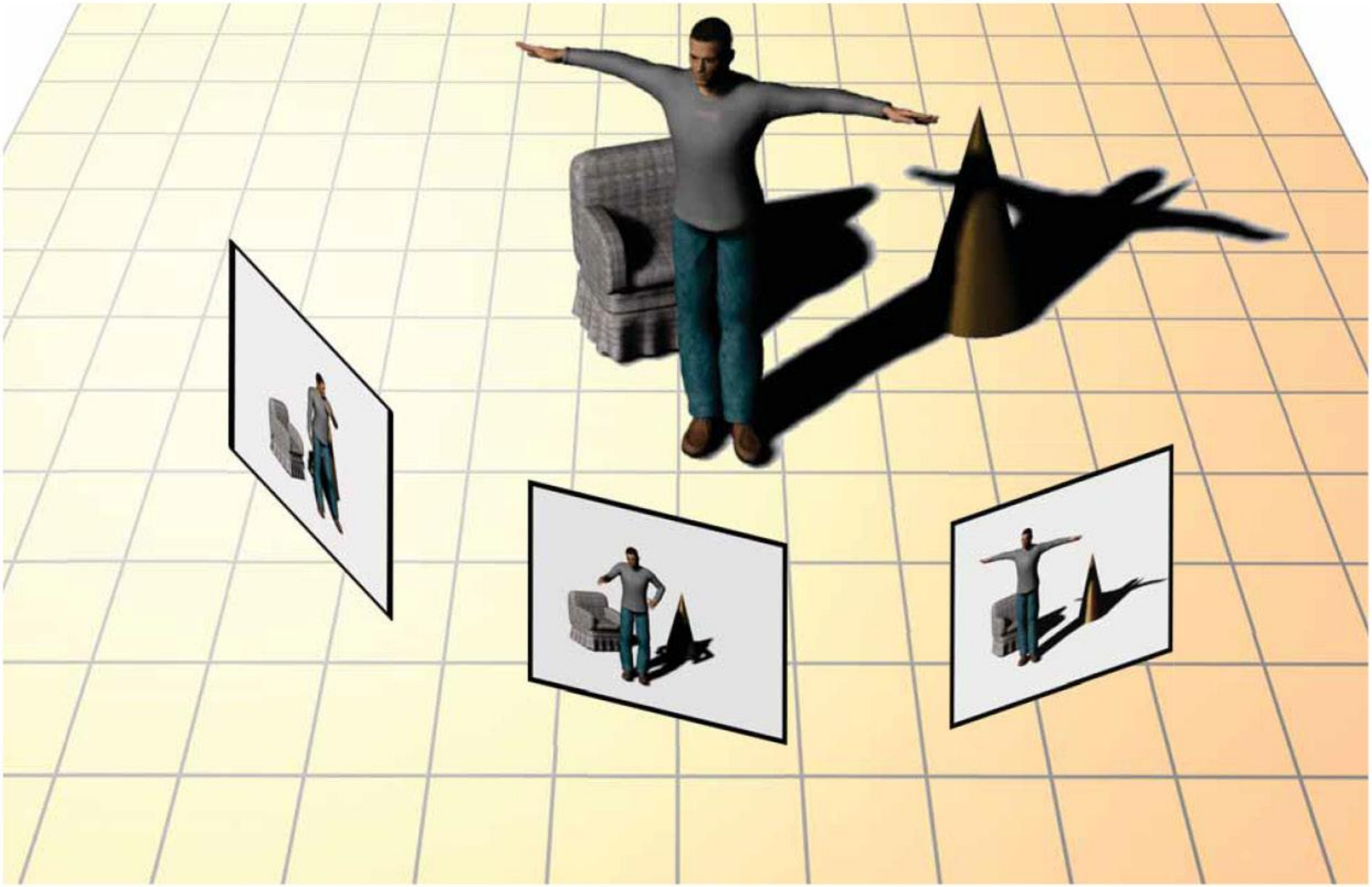
Nonrigid Structure from Motion



Nonrigid Structure from Motion



Nonrigid Structure from Motion



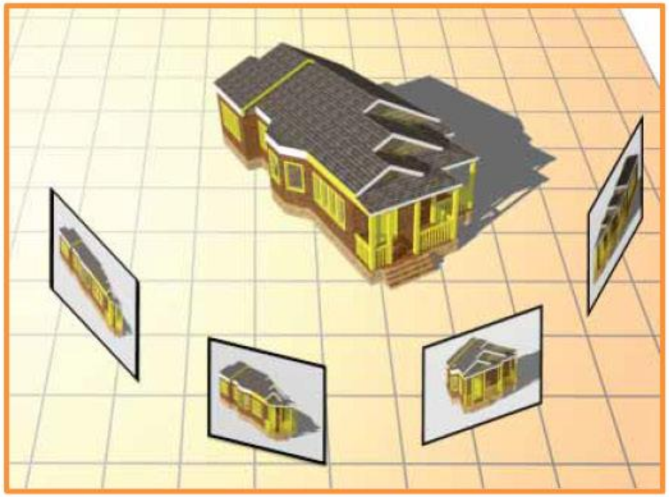
NONRIGID STRUCTURE FROM MOTION

Factorization Method for Rigid SFM

Kontsevich *et al.* 1987, Tomasi and Kanade 1992

ASSUMPTIONS

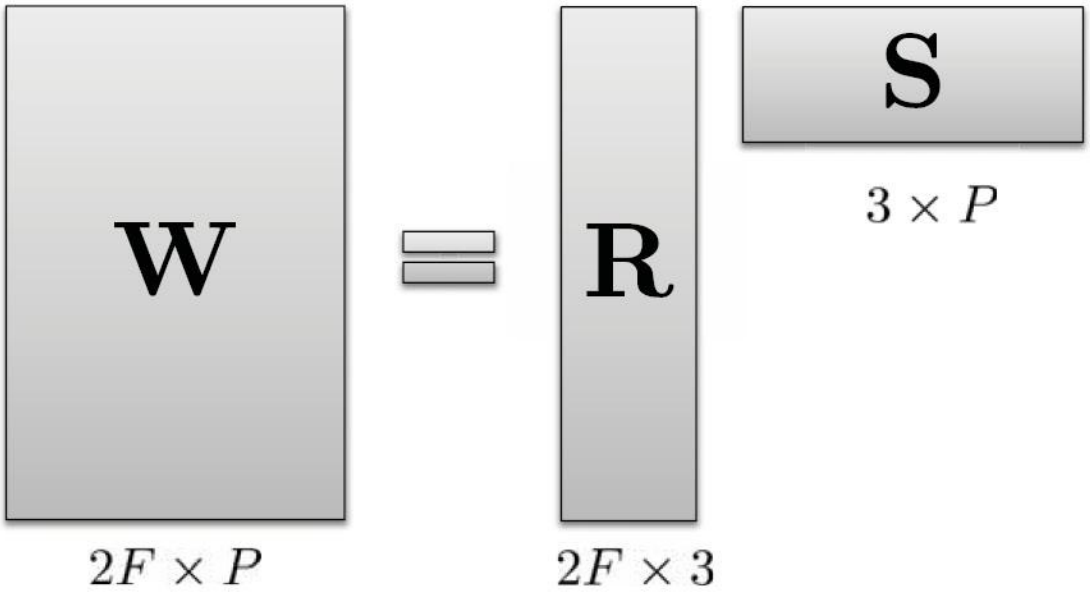
- Orthographic Camera
- At least 3 images
- Rigid Scene
- Camera Motion
- Corresponding points available



FACTORIZATION METHOD FOR RIGID SFM

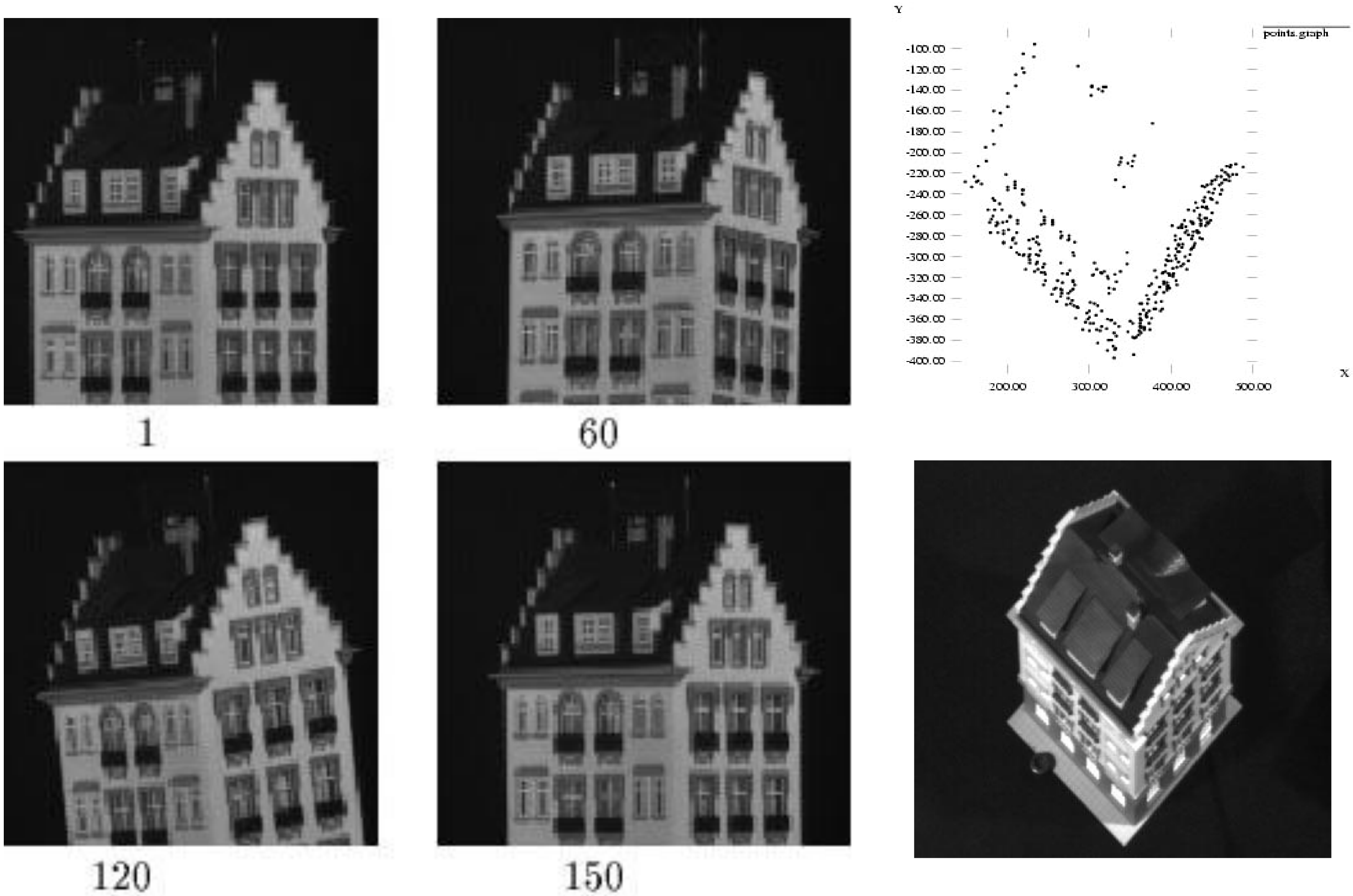
Kontsevich *et al.* 1987, Tomasi and Kanade 1992

PROJECTION OF P 3D POINTS IN F IMAGES



$$\mathbf{W}_{\text{measurement}} = \mathbf{R}_{\text{motion}} \times \mathbf{S}_{\text{shape}}$$

Recall: Reconstruction results



C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

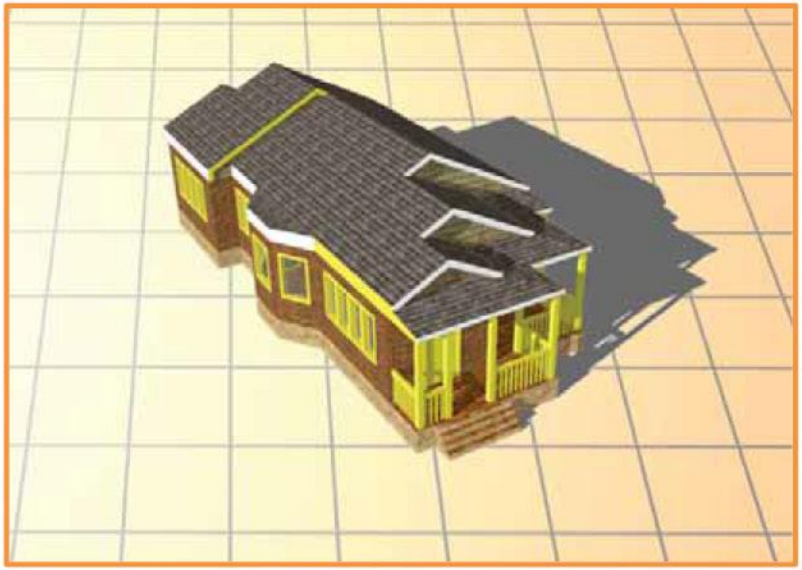
Lazebnik

NonRigid Structure

3D Structure That Deforms Over Time

RIGID STRUCTURE

$$\mathbf{S}_{3\times P} = \begin{bmatrix} X_1 & X_2 & \dots & X_P \\ Y_1 & Y_2 & \dots & Y_P \\ Z_1 & Z_2 & \dots & Z_P \end{bmatrix}$$

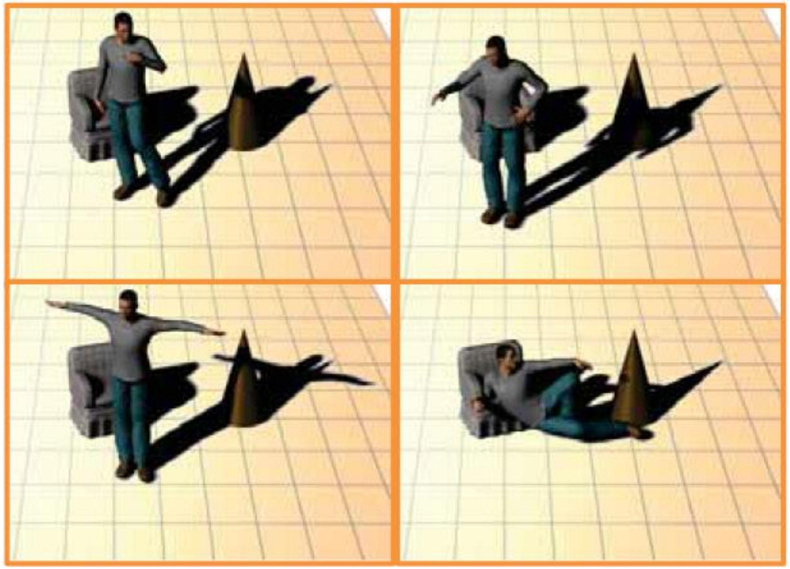


NonRigid Structure

3D Structure That Deforms Over Time

RIGID STRUCTURE

$$\mathbf{S}_{3\times P} = \begin{bmatrix} X_1 & X_2 & \dots & X_P \\ Y_1 & Y_2 & \dots & Y_P \\ Z_1 & Z_2 & \dots & Z_P \end{bmatrix}$$



NONRIGID STRUCTURE

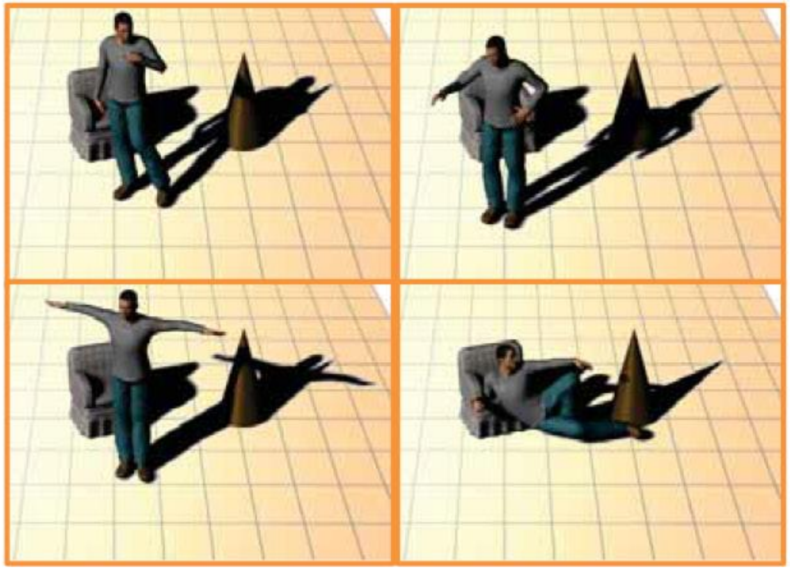
$$\mathbf{S}_{3F\times P} = \begin{bmatrix} \begin{bmatrix} X_{11} & X_{12} & \dots & X_{1P} \\ Y_{11} & Y_{12} & \dots & Y_{1P} \\ Z_{11} & Z_{12} & \dots & Z_{1P} \end{bmatrix}_{3\times P} \\ \\ \begin{bmatrix} X_{21} & X_{22} & \dots & X_{2P} \\ Y_{21} & Y_{22} & \dots & Y_{2P} \\ Z_{21} & Z_{22} & \dots & Z_{2P} \end{bmatrix}_{3\times P} \\ \vdots \\ \begin{bmatrix} X_{F1} & X_{F2} & \dots & X_{FP} \\ Y_{F1} & Y_{F2} & \dots & Y_{FP} \\ Z_{F1} & Z_{F2} & \dots & Z_{FP} \end{bmatrix}_{3\times P} \end{bmatrix}$$

NonRigid Structure

3D Structure That Deforms Over Time

RIGID STRUCTURE

$$\mathbf{S}_{3\times P} = \begin{bmatrix} X_1 & X_2 & \dots & X_P \\ Y_1 & Y_2 & \dots & Y_P \\ Z_1 & Z_2 & \dots & Z_P \end{bmatrix}$$



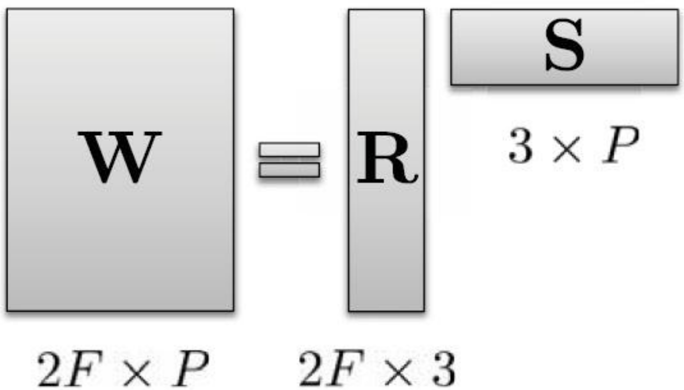
NONRIGID STRUCTURE

$$\mathbf{S}_{3F\times P} = \begin{bmatrix} \mathbf{X}_{11} & \mathbf{X}_{12} & \cdots & \mathbf{X}_{1P} \\ \mathbf{X}_{21} & \mathbf{X}_{22} & \cdots & \mathbf{X}_{2P} \\ \vdots & \vdots & & \vdots \\ \mathbf{X}_{F1} & \mathbf{X}_{F2} & \cdots & \mathbf{X}_{FP} \end{bmatrix}$$

NonRigid Structure From Motion

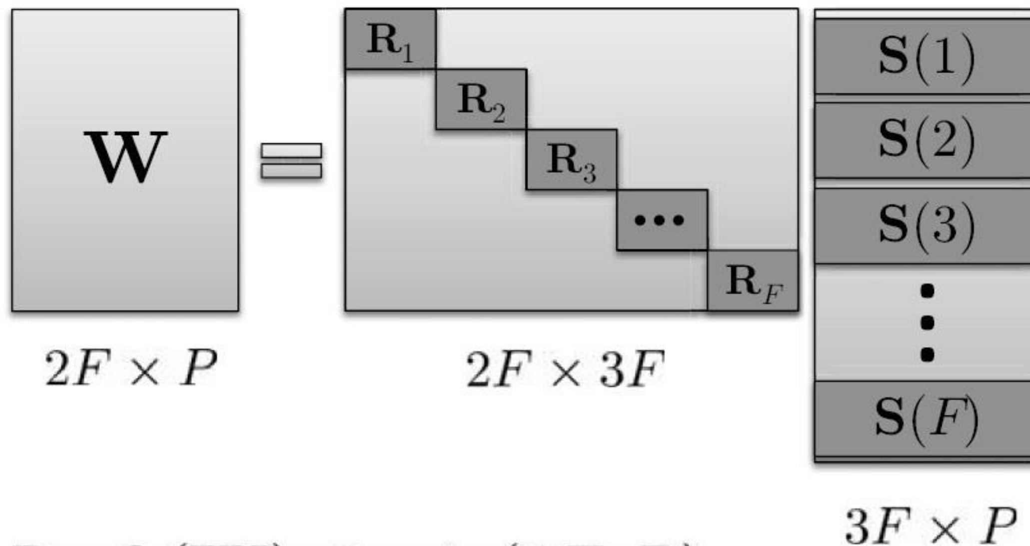
Comparison with Rigid Structure from Motion

RIGID SFM



$$\mathrm{Rank}(\mathbf{W}) \leq 3$$

NONRIGID SFM



$$\mathrm{Rank}(\mathbf{W}) \leq \min(2F, P)$$

NONRIGID STRUCTURE FROM MOTION

Explosion of Unknowns

Example: Given a 40 second video with 100 tracked points

RIGID SFM

- Inputs:

  100 pts x 40 sec x 30 fps x 2 (x, y)

  = 240,000 observations

- Unknowns:

  100 points x 3 (X,Y,Z)

  = **300** unknowns

NONRIGID SFM

- Inputs:

  100 pts x 40 sec x 30 fps x 2

  = 240,000 observations

- Unknowns:

  100 points x 40 sec x 30 fps x 3

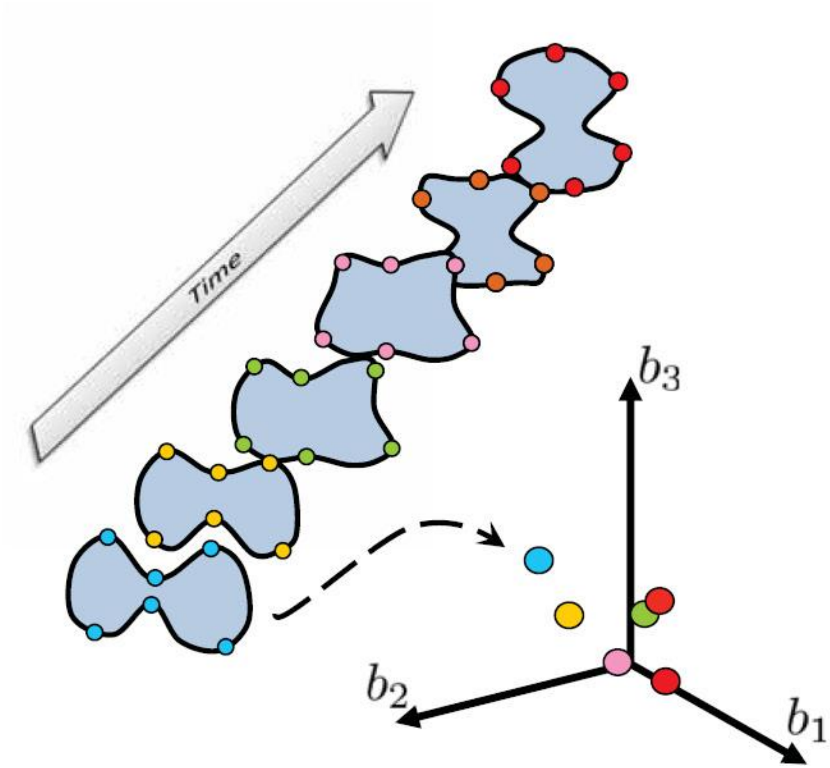
  = **360,000** unknowns

NONRIGID STRUCTURE FROM MOTION

Two Major Approaches

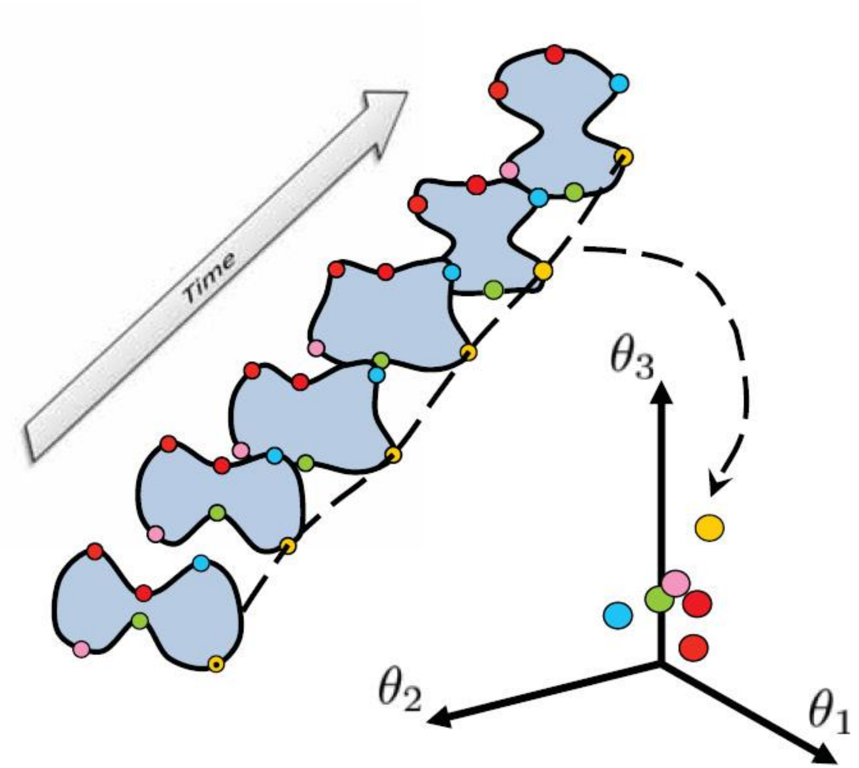
Shape Basis

3D points at each time instant lie in a low dimensional subspace



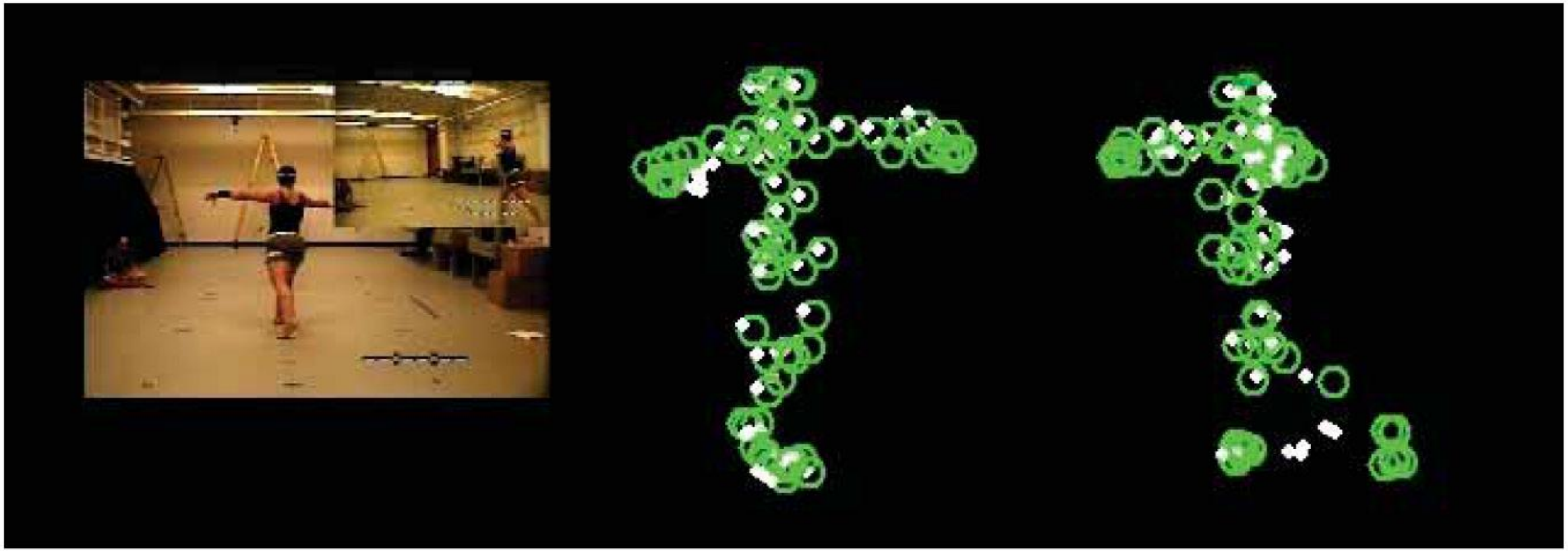
Trajectory Basis

Trajectory of each point over time lies in a low dimensional subspace



Examples of Applications

Motion-Capture



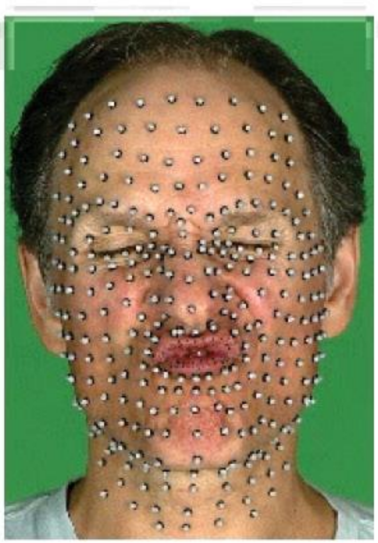
Input Video

Two views of the reconstruction

Akhter *et al.* NIPS 2008

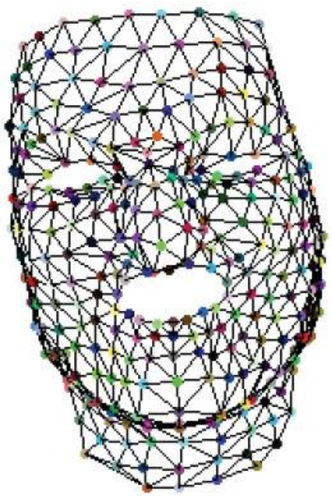
Examples of Applications

Motion-Capture Cleanup



Video

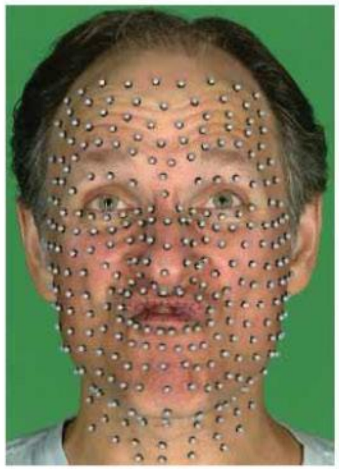
Unlabeled Data
Input

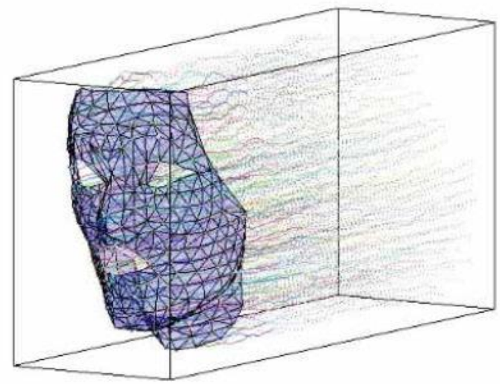


Reconstruction
Output

Disney Research, Pittsburgh

Dynamic Structure





$$\mathbf{S}_{3F\times P} = \begin{bmatrix} \mathbf{X}_{11} & \mathbf{X}_{12} & \cdots & \mathbf{X}_{1P} \\ \mathbf{X}_{21} & \mathbf{X}_{22} & \cdots & \mathbf{X}_{2P} \\ \vdots & \vdots & & \vdots \\ \mathbf{X}_{F1} & \mathbf{X}_{F2} & \cdots & \mathbf{X}_{FP} \end{bmatrix}$$

Trajectory

Shape

DYNAMIC STRUCTURE

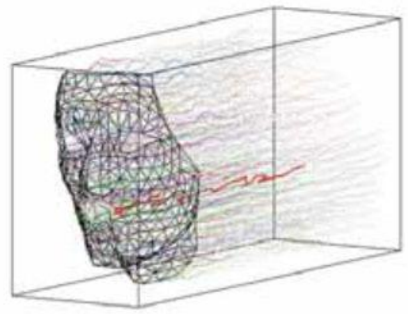
UNDER ORTHOGRAPHIC PROJECTION

$$\begin{bmatrix} \mathbf{x}_{11} & \cdots & \mathbf{x}_{1P} \\ \mathbf{x}_{21} & & \mathbf{x}_{2P} \\ \vdots & & \vdots \\ \mathbf{x}_{F1} & \cdots & \mathbf{x}_{FP} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_1 & & & \\ & \mathbf{R}_2 & & \\ & & \ddots & \\ & & & \mathbf{R}_F \end{bmatrix} \begin{bmatrix} \mathbf{X}_{11} & \cdots & \mathbf{X}_{1P} \\ \mathbf{X}_{21} & & \mathbf{X}_{2P} \\ \vdots & & \vdots \\ \mathbf{X}_{F1} & \cdots & \mathbf{X}_{FP} \end{bmatrix}$$

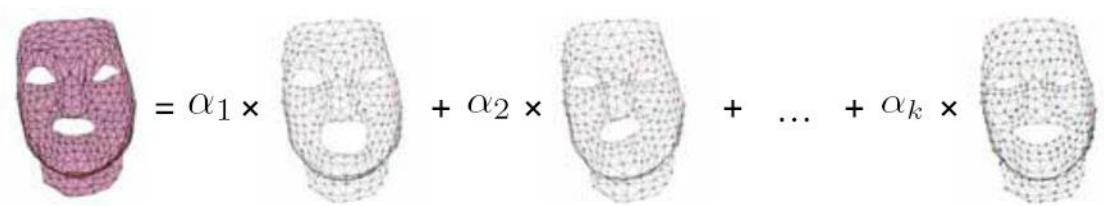
$$\mathbf{W} = \mathbf{R}\mathbf{X}$$

LINEAR SHAPE MODEL

[T. Cootes et al. 91, Bregler et al. 97]



$$\begin{bmatrix} \mathbf{X}_{11} & \cdots & \mathbf{X}_{1P} \\ \mathbf{X}_{21} & & \mathbf{X}_{2P} \\ \vdots & & \vdots \\ \mathbf{X}_{F1} & \cdots & \mathbf{X}_{FP} \end{bmatrix}$$



LINEAR SHAPE MODEL

$$\begin{bmatrix} \mathbf{X}_{11} & \cdots & \mathbf{X}_{1P} \\ \mathbf{X}_{21} & & \mathbf{X}_{2P} \\ \vdots & & \vdots \\ \mathbf{X}_{F1} & \cdots & \mathbf{X}_{FP} \end{bmatrix} = \begin{bmatrix} \omega_{11} & \cdots & \omega_{1k} \\ \omega_{21} & & \omega_{2k} \\ \vdots & & \vdots \\ \omega_{F1} & \cdots & \omega_{Fk} \end{bmatrix} \begin{bmatrix} -\mathbf{b_1}- \\ -\mathbf{b_2}- \\ \vdots \\ -\mathbf{b_k}- \end{bmatrix}$$

LINEAR SHAPE MODEL

RECONSTRUCTION

5 Basis 15 Basis 25 Basis

LINEAR SHAPE MODEL

UNDER ORTHOGRAPHIC PROJECTION

$$\underbrace{\begin{bmatrix} \mathbf{x}_{11} & \cdots & \mathbf{x}_{1P} \\ \mathbf{x}_{21} & & \mathbf{x}_{2P} \\ \vdots & & \vdots \\ \mathbf{x}_{F1} & \cdots & \mathbf{x}_{FP} \end{bmatrix}}_{2F \times P} = \underbrace{\begin{bmatrix} \mathbf{R}_1 & & & \\ & \mathbf{R}_2 & & \\ & & \ddots & \\ & & & \mathbf{R}_F \end{bmatrix}}_{2F \times 3F \ (6F)} \underbrace{\begin{bmatrix} \mathbf{X}_{11} & \cdots & \mathbf{X}_{1P} \\ \mathbf{X}_{21} & & \mathbf{X}_{2P} \\ \vdots & & \vdots \\ \mathbf{X}_{F1} & \cdots & \mathbf{X}_{FP} \end{bmatrix}}_{3F \times P}$$

$$= \underbrace{\begin{bmatrix} \mathbf{R}_1 & & & \\ & \mathbf{R}_2 & & \\ & & \ddots & \\ & & & \mathbf{R}_F \end{bmatrix}}_{2F \times 3F \ (6F)} \underbrace{\begin{bmatrix} \omega_{11} & \cdots & \omega_{1k} \\ \omega_{21} & & \omega_{2k} \\ \vdots & & \vdots \\ \omega_{F1} & \cdots & \omega_{Fk} \end{bmatrix}}_{3F \times 3k} \underbrace{\begin{bmatrix} -\mathbf{b_1}- \\ -\mathbf{b_2}- \\ \vdots \\ -\mathbf{b_k}- \end{bmatrix}}_{3k \times P}$$

CHALLENGE

TRILINEAR ESTIMATION

$$\begin{bmatrix} \mathbf{R}_1 & & & \\ & \mathbf{R}_2 & & \\ & & \ddots & \\ & & & \mathbf{R}_F \end{bmatrix} \begin{bmatrix} \omega_{11} & \cdots & \omega_{1k} \\ \omega_{21} & & \omega_{2k} \\ \vdots & & \vdots \\ \omega_{F1} & \cdots & \omega_{Fk} \end{bmatrix} \begin{bmatrix} -\mathbf{b_1}- \\ -\mathbf{b_2}- \\ \vdots \\ -\mathbf{b_k}- \end{bmatrix}$$

$$\mathbf{W} = \mathbf{R}\boldsymbol{\Omega}\mathbf{B}$$

BREGLER *et al.* 2000

Nested SVD

$$\begin{bmatrix} \mathbf{x}_{11} & \cdots & \mathbf{x}_{1P} \\ \mathbf{x}_{21} & & \mathbf{x}_{2P} \\ \vdots & & \vdots \\ \mathbf{x}_{F1} & \cdots & \mathbf{x}_{FP} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_1 & & & \\ & \mathbf{R}_2 & & \\ & & \ddots & \\ & & & \mathbf{R}_F \end{bmatrix} \begin{bmatrix} \omega_{11} & \cdots & \omega_{1k} \\ \omega_{21} & & \omega_{2k} \\ \vdots & & \vdots \\ \omega_{F1} & \cdots & \omega_{Fk} \end{bmatrix} \begin{bmatrix} -\mathbf{b_1}- \\ -\mathbf{b_2}- \\ \vdots \\ -\mathbf{b_k}- \end{bmatrix}$$

$$= \underbrace{\begin{bmatrix} \omega_{11}\mathbf{R}_1 & \cdots & \omega_{1k}\mathbf{R}_1 \\ \omega_{21}\mathbf{R}_2 & & \omega_{2k}\mathbf{R}_2 \\ \vdots & & \vdots \\ \omega_{F1}\mathbf{R}_F & \cdots & \omega_{Fk}\mathbf{R}_F \end{bmatrix}}_{2F \times 3k} \underbrace{\begin{bmatrix} -\mathbf{b}_1- \\ -\mathbf{b}_2- \\ \vdots \\ -\mathbf{b}_k- \end{bmatrix}}_{3k \times P}$$

BREGLER *et al.* 2000

Outer SVD

$$
\begin{array}{ccc}
\mathbf{W} & = & \mathbf{H} \qquad\qquad \mathbf{B} \\
\begin{bmatrix} \mathbf{x}_{11} & \cdots & \mathbf{x}_{1P} \\ \mathbf{x}_{21} & & \mathbf{x}_{2P} \\ \vdots & & \vdots \\ \mathbf{x}_{F1} & \cdots & \mathbf{x}_{FP} \end{bmatrix} & = & \underbrace{\begin{bmatrix} \omega_{11}\mathbf{R}_1 & \cdots & \omega_{1k}\mathbf{R}_1 \\ \omega_{21}\mathbf{R}_2 & & \omega_{2k}\mathbf{R}_2 \\ \vdots & & \vdots \\ \omega_{F1}\mathbf{R}_F & \cdots & \omega_{Fk}\mathbf{R}_F \end{bmatrix}}_{2F \times 3k} \underbrace{\begin{bmatrix} -\mathbf{b}_1- \\ -\mathbf{b}_2- \\ \vdots \\ -\mathbf{b}_k- \end{bmatrix}}_{3k \times P}
\end{array}
$$

SVD

$$\mathbf{W} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

$$\mathbf{W} = (\mathbf{U}\mathbf{D}^{\frac{1}{2}})(\mathbf{D}^{\frac{1}{2}}\mathbf{V}^T)$$

$$\mathbf{W} = \hat{\mathbf{H}}\hat{\mathbf{B}}$$

BREGLER *et al.* 2000

Inner SVD

$$\mathbf{W} = \hat{\mathbf{H}}\hat{\mathbf{B}}$$

$$\mathbf{H} = \begin{bmatrix} \omega_{11}\mathbf{R}_1 & \cdots & \omega_{1k}\mathbf{R}_1 \\ \omega_{21}\mathbf{R}_2 & & \omega_{2k}\mathbf{R}_2 \\ \vdots & & \vdots \\ \omega_{F1}\mathbf{R}_F & \cdots & \omega_{Fk}\mathbf{R}_1 \end{bmatrix}$$

$$\mathbf{h}_1 = \begin{bmatrix} \omega_{11}r_1^1 & \omega_{11}r_1^2 & \omega_{11}r_1^3 & \cdots & \omega_{1k}r_1^1 & \omega_{1k}r_1^2 & \omega_{1k}r_1^3 \\ \omega_{11}r_1^4 & \omega_{11}r_1^5 & \omega_{11}r_1^6 & \cdots & \omega_{1k}r_1^4 & \omega_{1k}r_1^5 & \omega_{1k}r_1^6 \end{bmatrix}$$

$$\mathbf{h}_1' = \underbrace{\begin{bmatrix} \omega_{11}r_1^1 & \omega_{11}r_1^2 & \omega_{11}r_1^3 & \omega_{11}r_1^4 & \omega_{11}r_1^5 & \omega_{11}r_1^6 \\ \omega_{12}r_1^1 & \omega_{12}r_1^2 & \omega_{12}r_1^3 & \omega_{12}r_1^4 & \omega_{12}r_1^5 & \omega_{12}r_1^6 \\ \vdots & & & & & \vdots \\ \omega_{1k}r_1^1 & \omega_{1k}r_1^2 & \omega_{1k}r_1^3 & \omega_{1k}r_1^4 & \omega_{1k}r_1^5 & \omega_{1k}r_1^6 \end{bmatrix}}_{\text{rank 1}} = \begin{bmatrix} \omega_{11} \\ \omega_{12} \\ \vdots \\ \omega_{1k} \end{bmatrix} \begin{bmatrix} r_1^1 & r_1^2 & r_1^3 & r_1^4 & r_1^5 & r_1^6 \end{bmatrix}$$

SVD $\mathbf{h}_1' = \mathbf{u}\mathbf{d}\mathbf{v}^T = \hat{\omega}\hat{\mathbf{r}}$

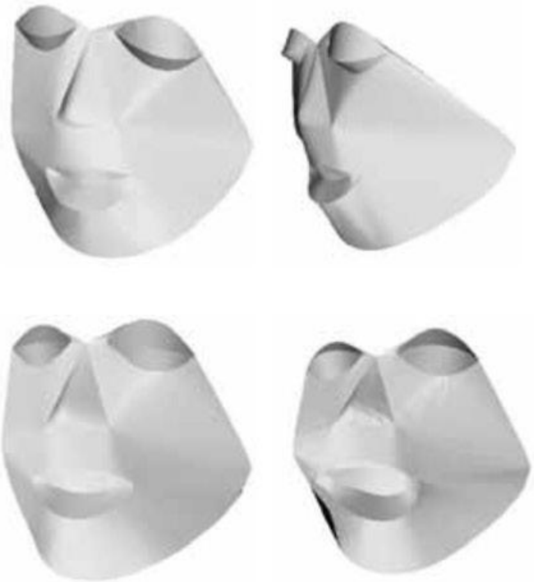
METRIC RECTIFICATION USING ORTHONORMALITY CONSTRAINTS

BREGLER *et al.* 2000

OVERVIEW

- <u>OUTER SVD</u>: PERFORM SVD ON **W** TO GET ESTIMATES OF:
  - **H**: CAMERA PROJECTIONS AND COEFFICIENTS
    - <u>INNER SVD</u>: PERFORM SVD ON **H** TO GET ESTIMATES OF:
      - OMEGA: COEFFICIENTS
      - **R**: CAMERA PROJECTIONS
    - METRIC RECTIFY USING ORTHONORMALITY CONSTRAINTS
  - **B**: THE SHAPE BASIS

RESULTS

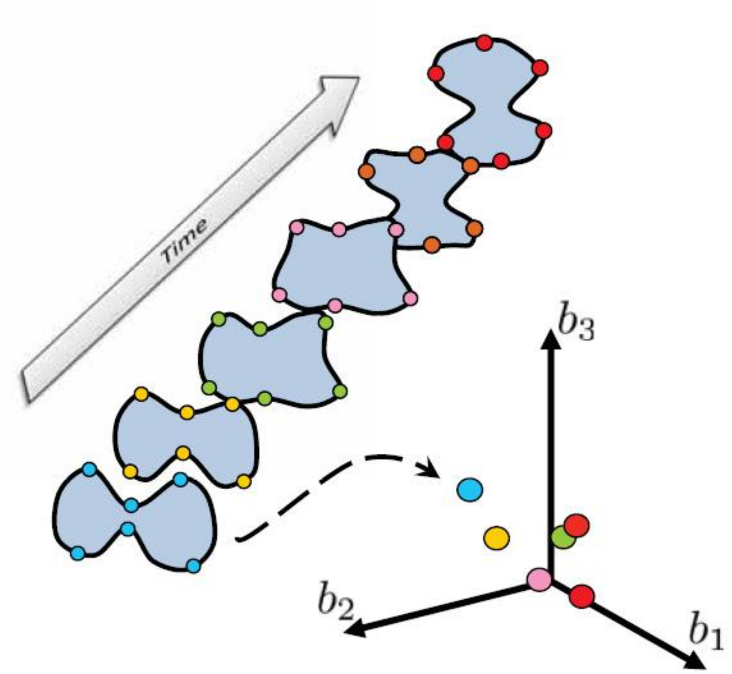


Nonrigid Structure From Motion

Two Major Approaches

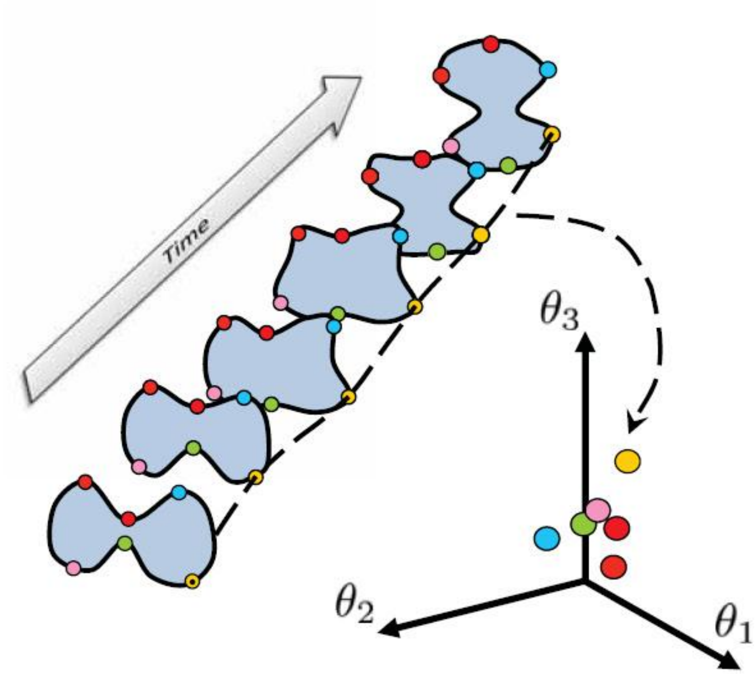
Shape Basis

3D points at each time instant lie in a low dimensional subspace

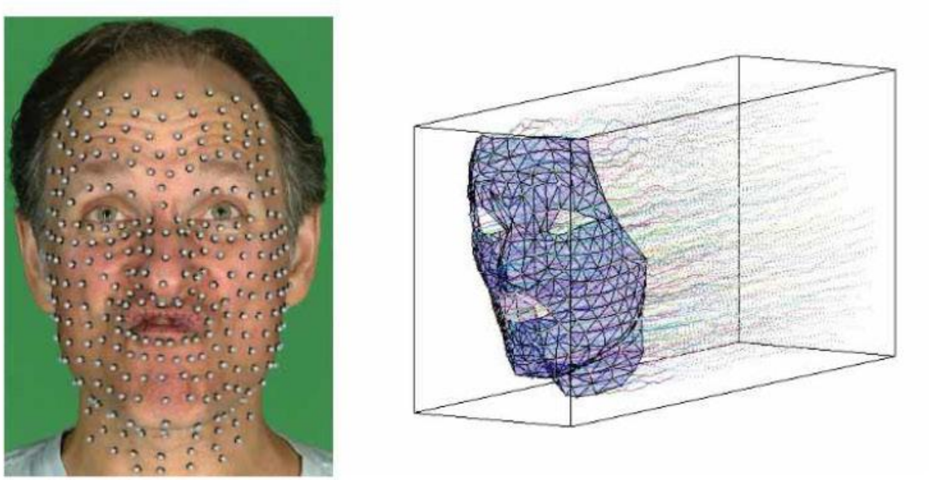


Trajectory Basis

Trajectory of each point over time lies in a low dimensional subspace



Dynamic Structure



$$S_{3F\times P} = \begin{bmatrix} \mathbf{X}_{11} & \mathbf{X}_{12} & \cdots & \mathbf{X}_{1P} \\ \mathbf{X}_{21} & \mathbf{X}_{22} & \cdots & \mathbf{X}_{2P} \\ \vdots & \vdots & & \vdots \\ \mathbf{X}_{F1} & \mathbf{X}_{F2} & \cdots & \mathbf{X}_{FP} \end{bmatrix}$$

Trajectory

Shape

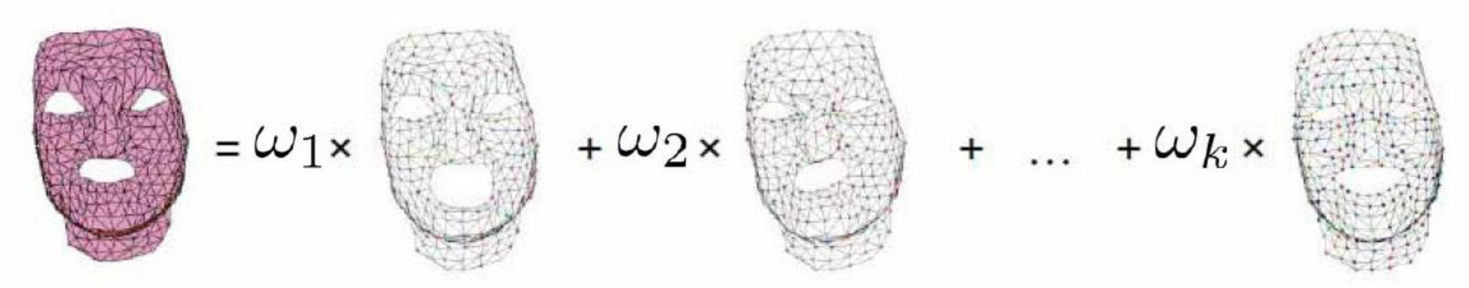
Dynamic Structure

Shape Representation

$$\mathbf{S}_{3F\times P} = \begin{bmatrix} \mathbf{X}_{11} & \mathbf{X}_{12} & \cdots & \mathbf{X}_{1P} \\ \mathbf{X}_{21} & \mathbf{X}_{22} & \cdots & \mathbf{X}_{2P} \\ \vdots & \vdots & & \vdots \\ \mathbf{X}_{F1} & \mathbf{X}_{F2} & \cdots & \mathbf{X}_{FP} \end{bmatrix}$$

Shape

LINEAR SHAPE MODEL

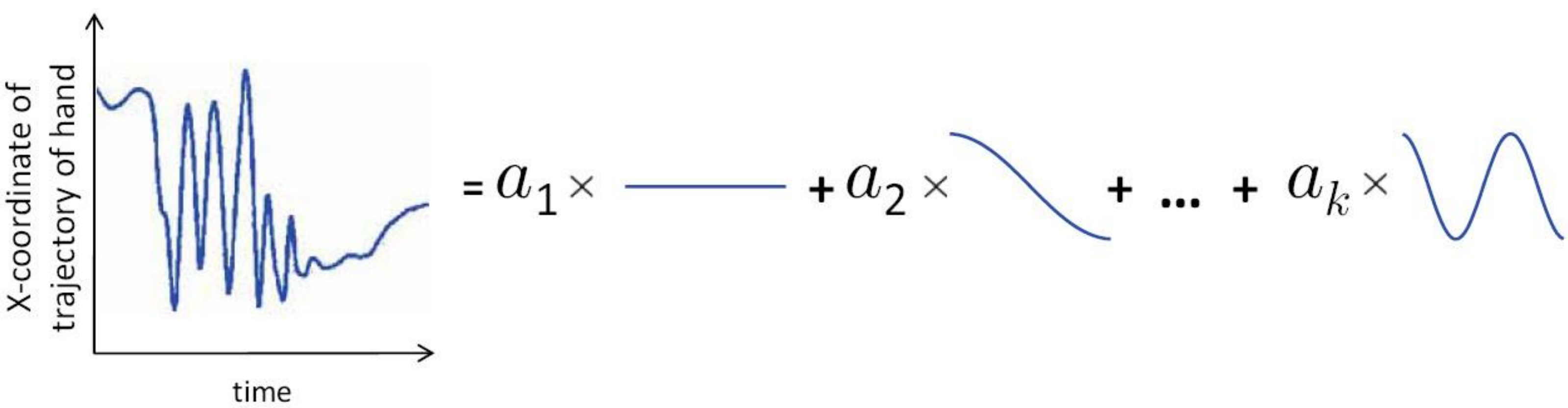


DYNAMIC STRUCTURE

Trajectory Representation

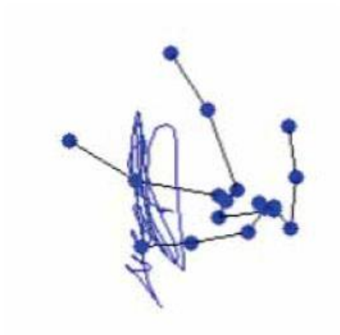
$$\mathbf{S}_{3F\times P} = \begin{bmatrix} \mathbf{X}_{11} & \mathbf{X}_{12} & \cdots & \mathbf{X}_{1P} \\ \mathbf{X}_{21} & \mathbf{X}_{22} & \cdots & \mathbf{X}_{2P} \\ \vdots & \vdots & & \vdots \\ \mathbf{X}_{F1} & \mathbf{X}_{F2} & \cdots & \mathbf{X}_{FP} \end{bmatrix} \downarrow \text{Trajectory}$$

LINEAR TRAJECTORY MODEL



DYNAMIC STRUCTURE

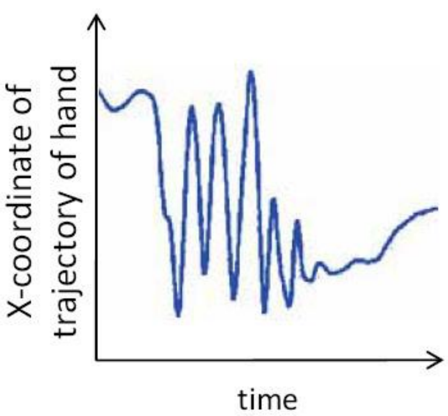
Trajectory Representation



$$\mathbf{S}_{3F\times P} = \begin{bmatrix} \mathbf{X}_{11} & \mathbf{X}_{12} & \cdots & \mathbf{X}_{1P} \\ \mathbf{X}_{21} & \mathbf{X}_{22} & \cdots & \mathbf{X}_{2P} \\ \vdots & \vdots & & \vdots \\ \mathbf{X}_{F1} & \mathbf{X}_{F2} & \cdots & \mathbf{X}_{FP} \end{bmatrix}$$

Trajectory

LINEAR TRAJECTORY MODEL



$$T_j^X = \sum_{k=1}^{K} a_{jk}^X \theta^k$$

Trajectory Coefficient
Contribution of k^{th} basis in the trajectory of j^{th} point

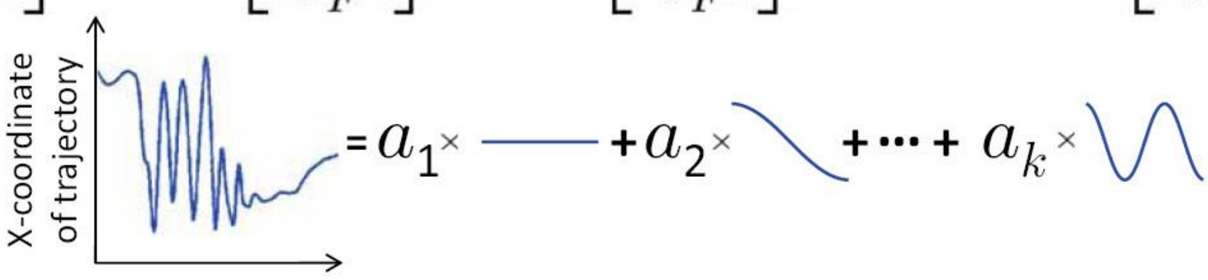
k^{th} trajectory basis vector

Trajectory of j^{th} point (X-component only)

Trajectory Representation of Dynamic Structure

$$T_j^X = \sum_{k=1}^{K} a_{jk}^X \theta^k \qquad T_j^Y = \sum_{k=1}^{K} a_{jk}^Y \theta^k \qquad T_j^Z = \sum_{k=1}^{K} a_{jk}^Z \theta^k$$

$$\begin{bmatrix} X_{1j} \\ X_{2j} \\ \vdots \\ X_{Fj} \end{bmatrix} = a_{j1}^X \begin{bmatrix} \theta_1^1 \\ \theta_2^1 \\ \vdots \\ \theta_F^1 \end{bmatrix} + a_{j2}^X \begin{bmatrix} \theta_1^2 \\ \theta_2^2 \\ \vdots \\ \theta_F^2 \end{bmatrix} + \ldots + a_{jK}^X \begin{bmatrix} \theta_1^K \\ \theta_2^K \\ \vdots \\ \theta_F^K \end{bmatrix}$$



Trajectory Representation of Dynamic Structure

X-component of trajectory of jth point as linear combination of K basis trajectories

$$\begin{bmatrix} X_{1j} \\ X_{2j} \\ \vdots \\ X_{Fj} \end{bmatrix} = a_{j1}^X \begin{bmatrix} \theta_1^1 \\ \theta_2^1 \\ \vdots \\ \theta_F^1 \end{bmatrix} + a_{j2}^X \begin{bmatrix} \theta_1^2 \\ \theta_2^2 \\ \vdots \\ \theta_F^2 \end{bmatrix} + \ldots + a_{jK}^X \begin{bmatrix} \theta_1^K \\ \theta_2^K \\ \vdots \\ \theta_F^K \end{bmatrix}$$

X-component of trajectory of **all** point as linear combination of K basis trajectories

$$\begin{bmatrix} X_{11} & X_{12} & \ldots & X_{1P} \\ X_{21} & X_{22} & \ldots & X_{2P} \\ \vdots & \vdots & \vdots & \vdots \\ X_{F1} & X_{F2} & \ldots & X_{FP} \end{bmatrix} = \begin{bmatrix} \theta_1^1 & \theta_1^2 & \ldots & \theta_1^K \\ \theta_2^1 & \theta_2^2 & \ldots & \theta_2^K \\ \vdots & \vdots & \vdots & \vdots \\ \theta_F^1 & \theta_F^2 & \ldots & \theta_F^K \end{bmatrix} \begin{bmatrix} a_{11}^X & a_{21}^X & \ldots & a_{P1}^X \\ a_{12}^X & a_{22}^X & \ldots & a_{P2}^X \\ \vdots & \vdots & \vdots & \vdots \\ a_{1K}^X & a_{2K}^X & \ldots & a_{PK}^X \end{bmatrix}$$

$$\underset{F \times P}{\mathbf{S}^X} = \underset{F \times K}{\mathbf{\Theta}^X} \times \underset{K \times P}{\mathbf{A}^X}$$

X-component of trajectory of all points

$$
\begin{bmatrix} X_{11} & X_{12} & \dots & X_{1P} \\ X_{21} & X_{22} & \dots & X_{2P} \\ \vdots & \vdots & \vdots & \vdots \\ X_{F1} & X_{F2} & \dots & X_{FP} \end{bmatrix} = \begin{bmatrix} \theta_1^1 & \theta_1^2 & \dots & \theta_1^K \\ \theta_2^1 & \theta_2^2 & \dots & \theta_2^K \\ \vdots & \vdots & \vdots & \vdots \\ \theta_F^1 & \theta_F^2 & \dots & \theta_F^K \end{bmatrix} \begin{bmatrix} a_{11}^X & a_{21}^X & \dots & a_{P1}^X \\ a_{12}^X & a_{22}^X & \dots & a_{P2}^X \\ \vdots & \vdots & \vdots & \vdots \\ a_{1K}^X & a_{2K}^X & \dots & a_{PK}^X \end{bmatrix}
$$

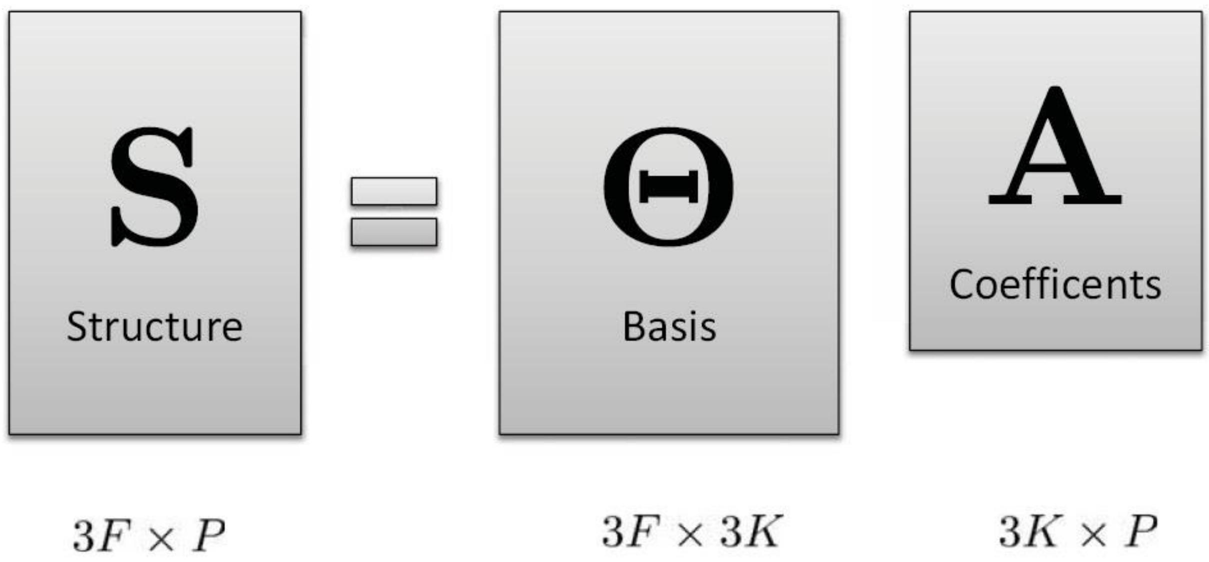
X, Y and Z-components of trajectory of all points

$$
\begin{bmatrix} X_{11} & \dots & X_{1P} \\ Y_{11} & \dots & Y_{1P} \\ Z_{11} & \dots & Z_{1P} \\ X_{21} & \dots & X_{2P} \\ Y_{21} & \dots & Y_{2P} \\ Z_{21} & \dots & Z_{2P} \\ \vdots & & \vdots \\ X_{F1} & \dots & X_{FP} \\ Y_{F1} & \dots & Y_{FP} \\ Z_{F1} & \dots & Z_{FP} \end{bmatrix} = \begin{bmatrix} \theta_1^1 & \dots & \theta_1^K & & & & & & \\ & & & \theta_1^1 & \dots & \theta_1^K & & & \\ & & & & & & \theta_1^1 & \dots & \theta_1^K \\ \theta_2^1 & \dots & \theta_2^K & & & & & & \\ & & & \theta_2^1 & \dots & \theta_2^K & & & \\ & & & & & & \theta_2^1 & \dots & \theta_2^K \\ \vdots & & & & \vdots & & & & \vdots \\ \theta_F^1 & \dots & \theta_F^K & & & & & & \\ & & & \theta_F^1 & \dots & \theta_F^K & & & \\ & & & & & & \theta_F^1 & \dots & \theta_F^K \end{bmatrix} \begin{bmatrix} \begin{bmatrix} a_{11}^X & a_{21}^X & \dots & a_{P1}^X \\ \vdots & \vdots & \vdots & \vdots \\ a_{1K}^X & a_{2K}^X & \dots & a_{PK}^X \end{bmatrix}_{A^X} \\ \begin{bmatrix} a_{11}^Y & a_{21}^Y & \dots & a_{P1}^Y \\ \vdots & \vdots & \vdots & \vdots \\ a_{1K}^Y & a_{2K}^Y & \dots & a_{PK}^Y \end{bmatrix}_{A^Y} \\ \begin{bmatrix} a_{11}^Z & a_{21}^Z & \dots & a_{P1}^Z \\ \vdots & \vdots & \vdots & \vdots \\ a_{1K}^Z & a_{2K}^Z & \dots & a_{PK}^Z \end{bmatrix}_{A^Z} \end{bmatrix}
$$

$$
\mathbf{S}_{3F\times P} = \Theta_{3F\times 3K} \quad \mathbf{A}_{3K\times P}
$$

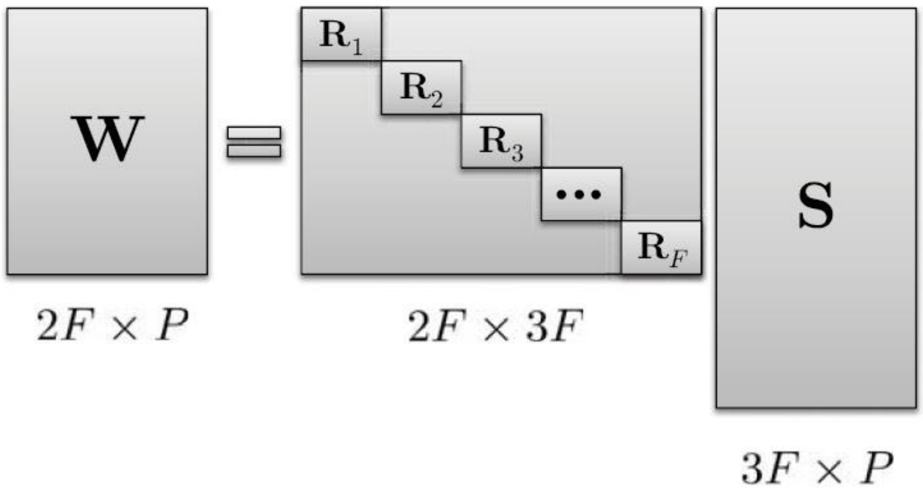
TRAJECTORY REPRESENTATION

of Dynamic Structure



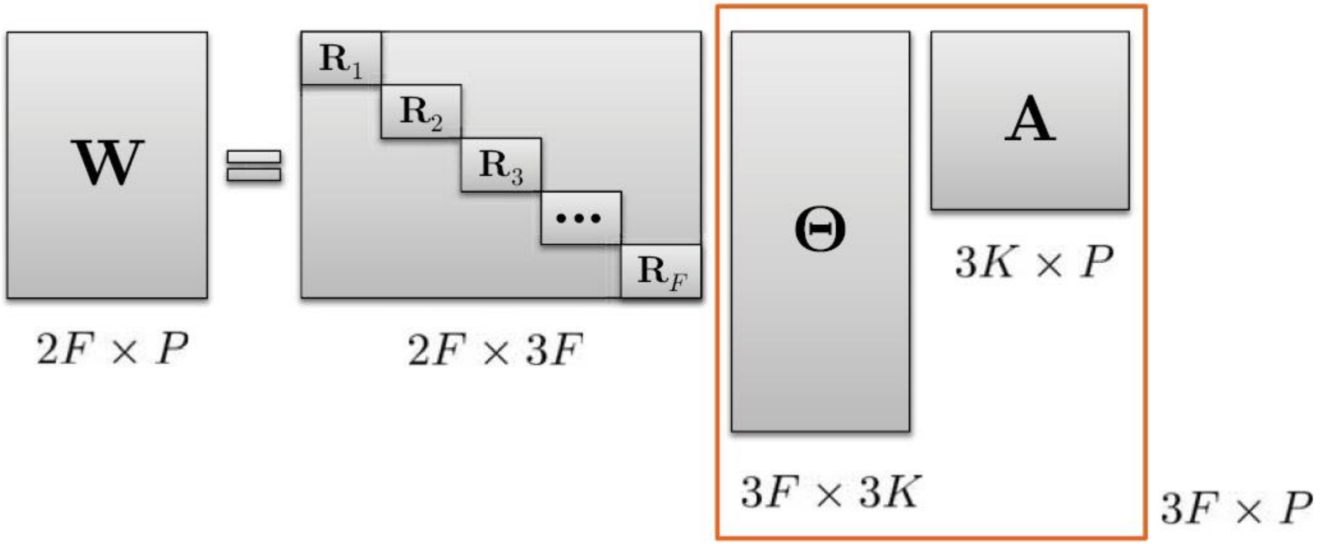
TRAJECTORY REPRESENTATION

of Dynamic Structure *Under Orthographic Projection*



TRAJECTORY REPRESENTATION

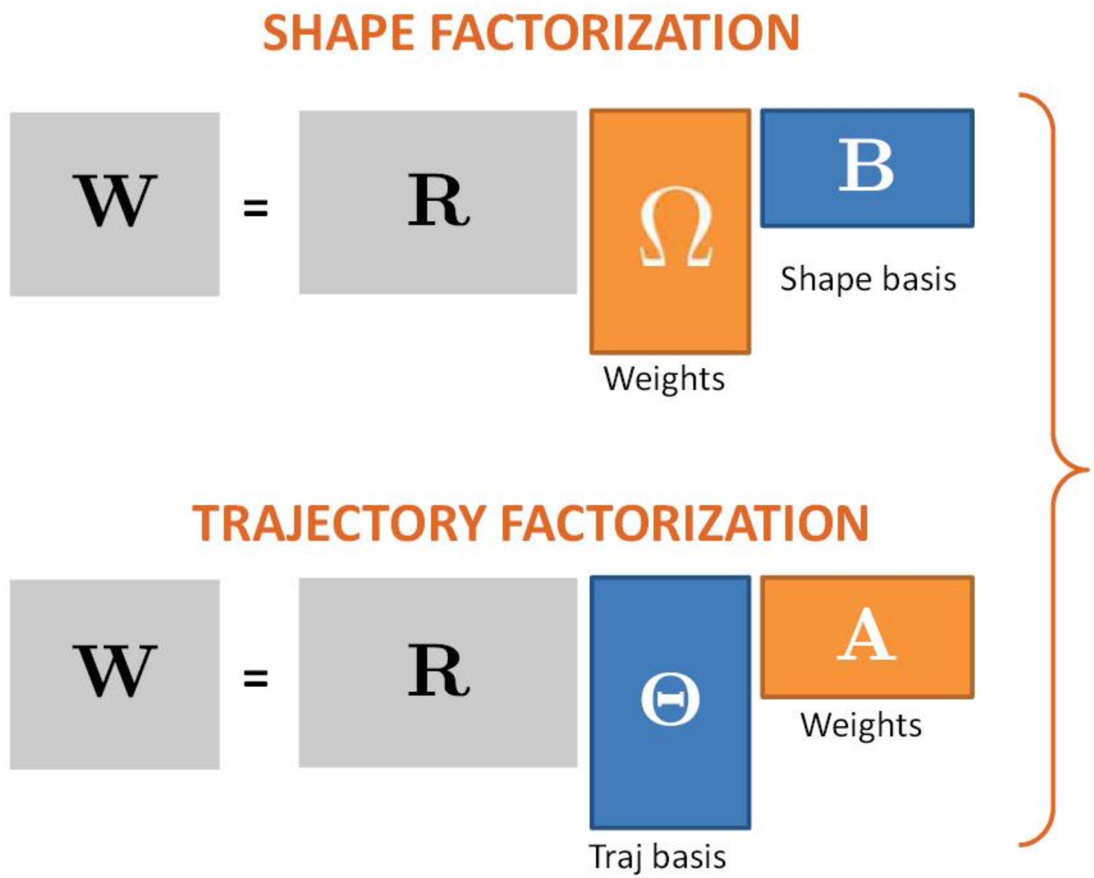
of Dynamic Structure *Under Orthographic Projection*



Structure $\mathbf{S}$, in trajectory subspace represented by K trajectory basis

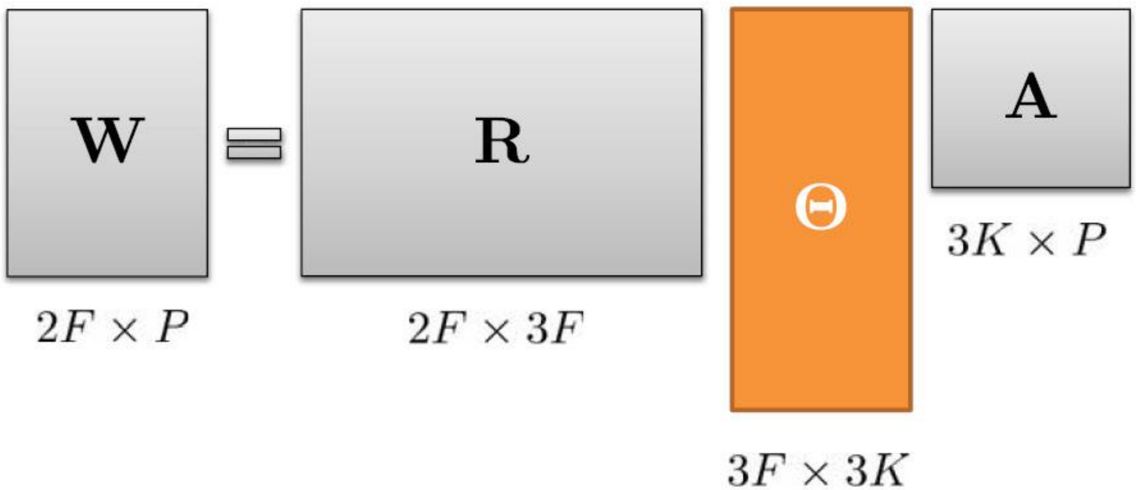
Duality

Weights and Bases



Shape weights are trajectory basis and trajectory weights are shape basis

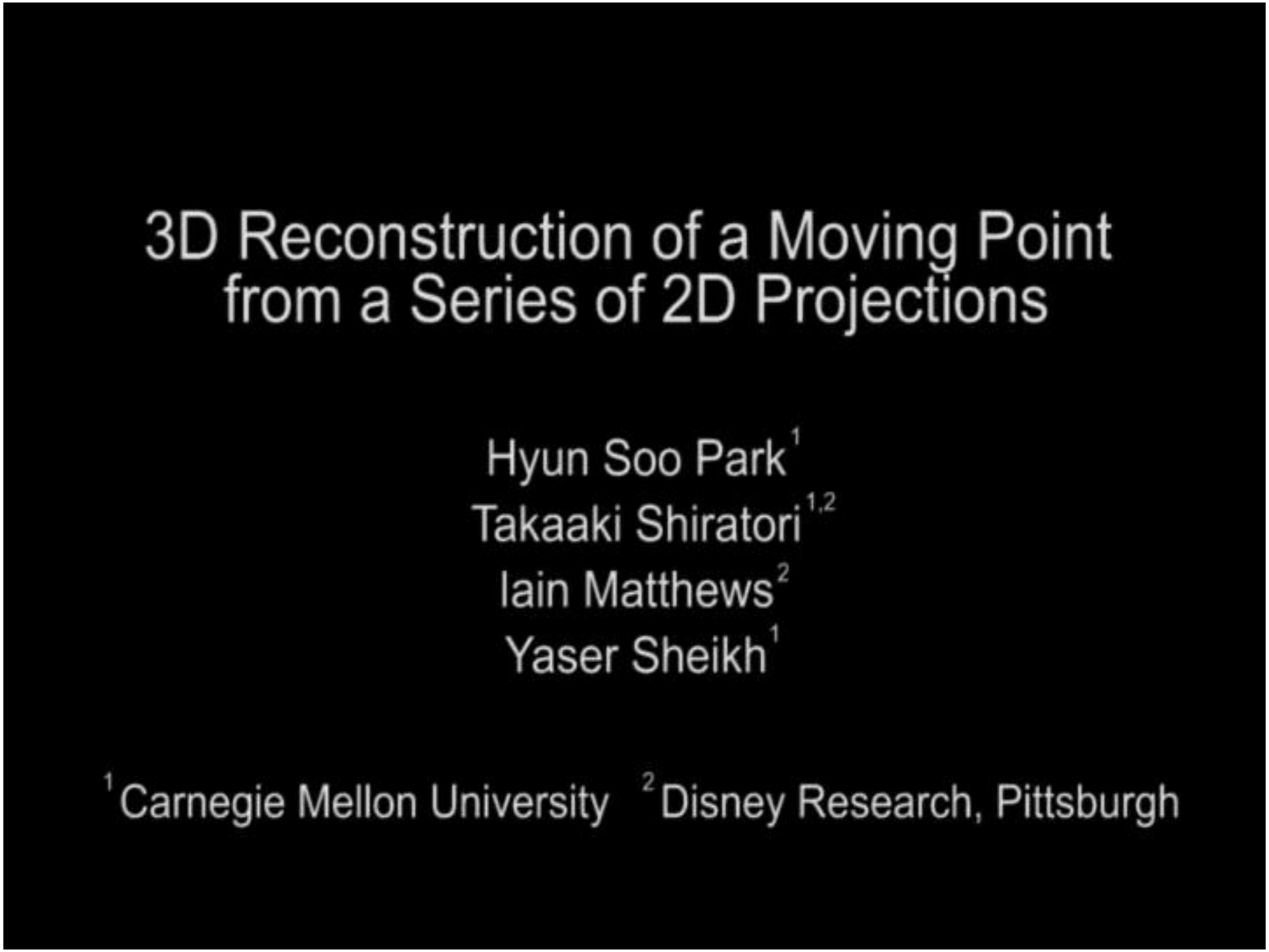
Estimating Structure via Trajectory Model



Object Independent Basis

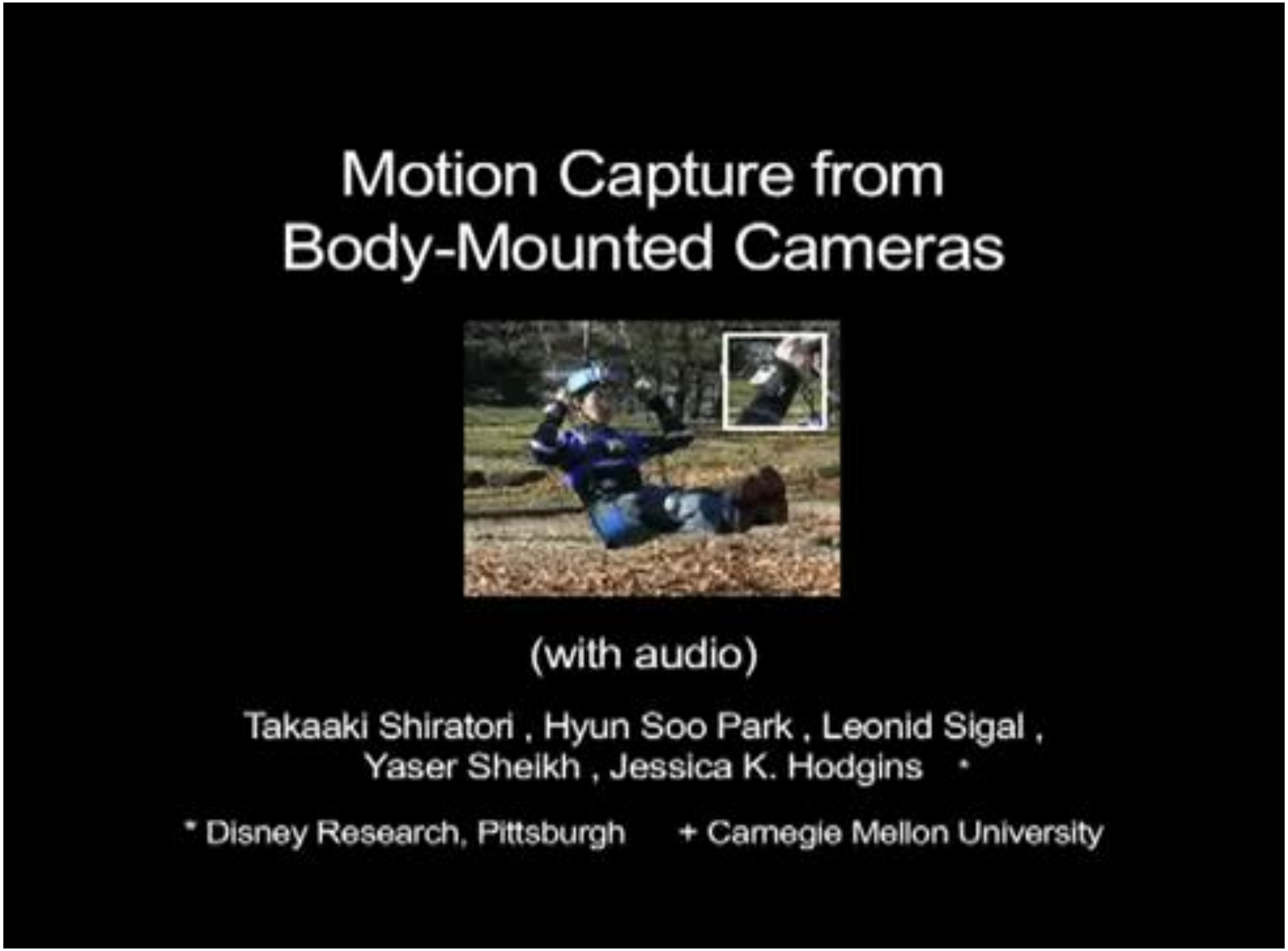
1. Deformation constrained by physical actuation
2. Trajectories vary smoothly and not randomly
3. Can be compactly represented by predefined basis
   e.g. Discrete Cosine Transform

Results



H. Park et al. 3D Reconstruction of a Moving Point from a Series of 2D Projections. ECCV'10.

Structure from motion



UNIVERSITÄT BONN