Master in
Computer Vision
Barcelona

Module: 3D Vision
Lecture 8: 3D reconstruction from multiple views.
Voxel-based methods.
Stratified methods.
Projective reconstruction.
Lecturer: Antonio Agudo

Outline

- 3D reconstruction from multiple views
- Voxel-based methods
  - Shape from silhouette
    - Voxel coloring
    - Space carving
- Structure from motion.
- Stratified reconstruction.
- Projective reconstruction: Factorization method

3D reconstruction from multiple views

- Calibrated case: Multi-view stereo, Shape from X
- Non-calibrated case: Structure from motion (in computer vision)
  SLAM (in robotics)

3D reconstruction from multiple views

- Calibrated case: Multi-view stereo, Shape from X
- Non-calibrated case: Structure from motion

Input data:
Images from different views, video sequence, images from databases, ...

3D reconstruction from multiple views

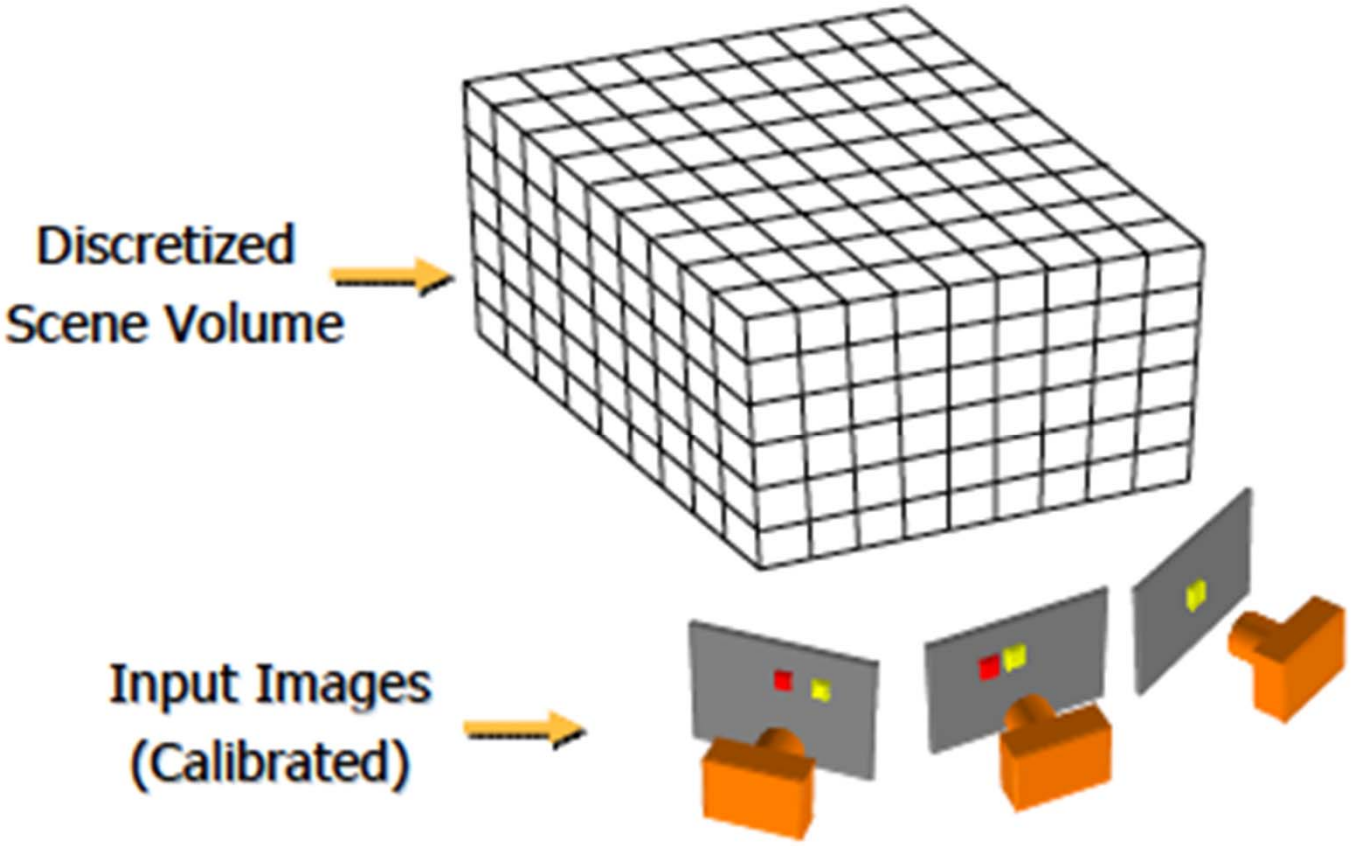
- Calibrated case: Multi-view stereo, Shape from X
- Non-calibrated case: Structure from motion

Input data:
Images from different views, video sequence, images from databases, ...

Representation of 3D shape:
Depth maps, voxels, point clouds, meshes (after reconstructing) ...

Voxel-based methods

Problem statement



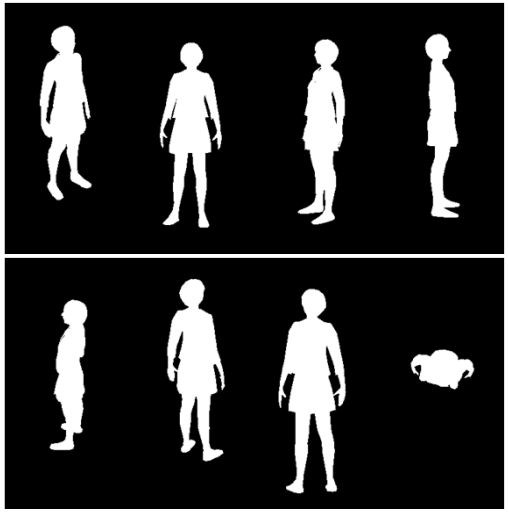
GOAL: Determine the **occupancy**, and eventually **color**, of voxels in **V**.

Image source: [S. Seitz]

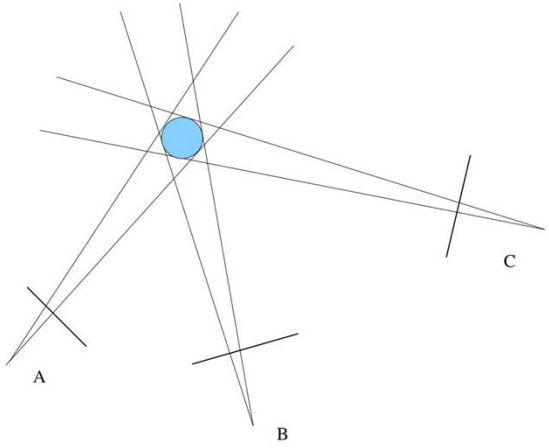
Shape from silhouette

Traditional approach: **Visual hull**

3D shape as the intersection of the back-projected silhouettes in the 3D space.



Silhouettes



Visual cones

3D Shape

Shape from silhouette

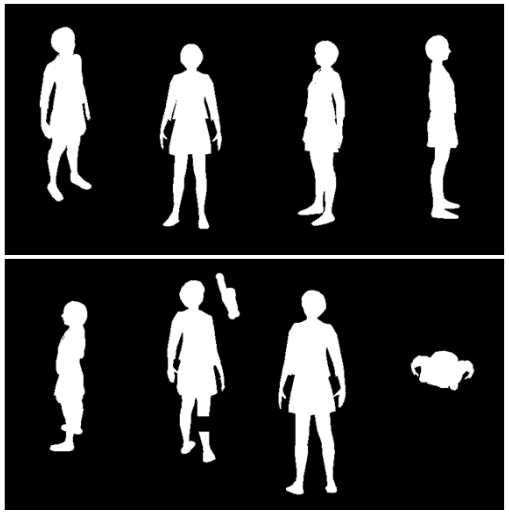
VISUAL HULL ALGORITHM

1. Iterate through all voxels in the volume
2. Project voxel to each image
3. If voxel projects inside silhouette in ALL views
   Mark the voxel as occupied

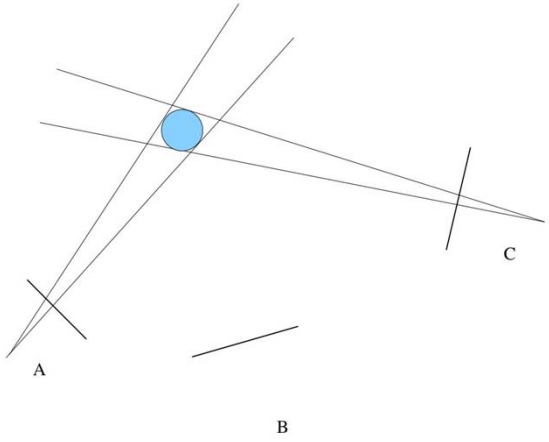
Shape from silhouette

Traditional approach: **Visual hull**

Fails in case of incomplete (inconsistent) silhouettes.
Robust methods exist.



Silhouettes



Visual cones

Shape

Shape from silhouette

OBSERVATION:
Visual hull methods **do not recover concavities**. Methods based on the photo-consistency of color images do recover them.

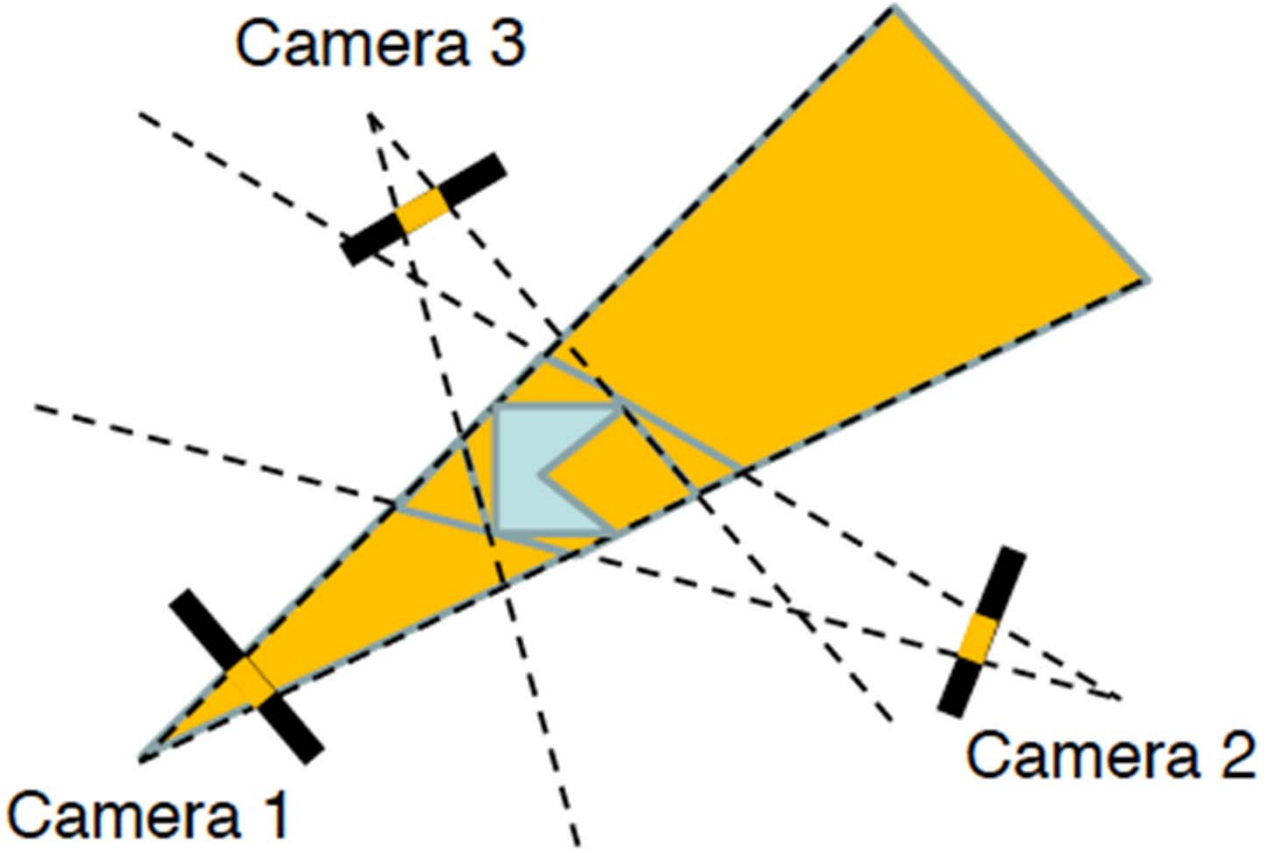


Image source: [A. Ladikos]

Voxel coloring

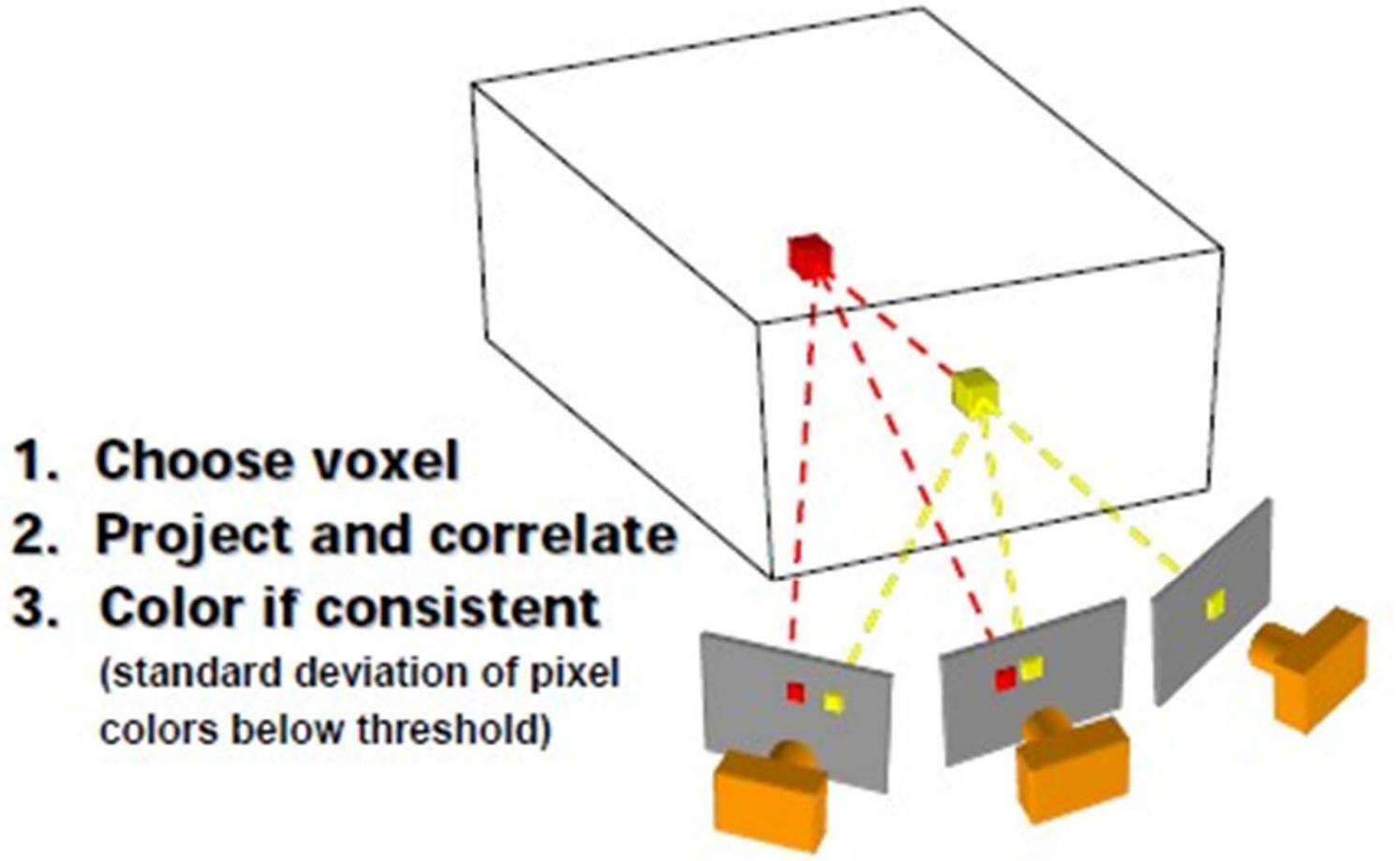


Image source: [S. Seitz]

Voxel coloring

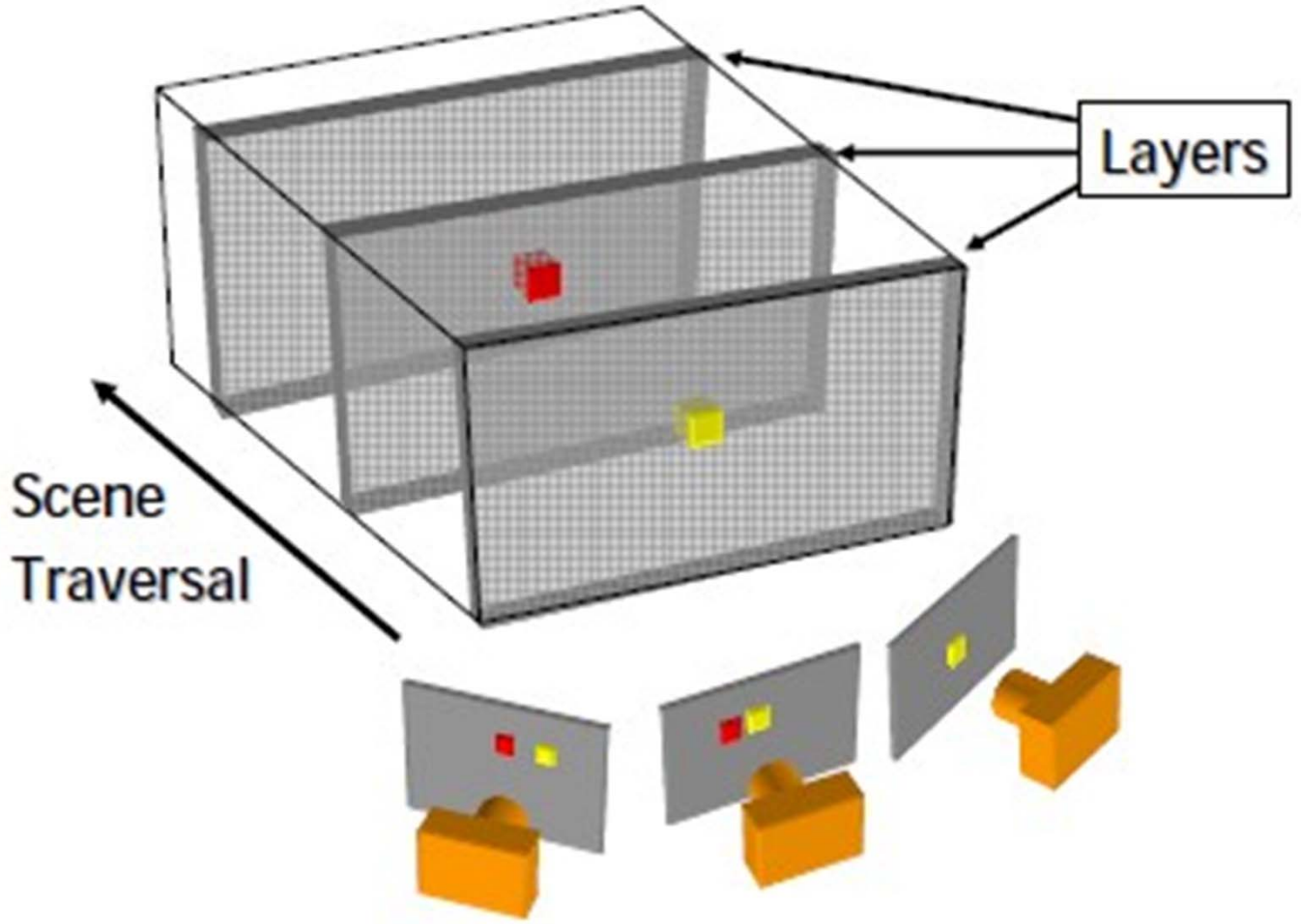


Image source: [S. Seitz]

Voxel coloring

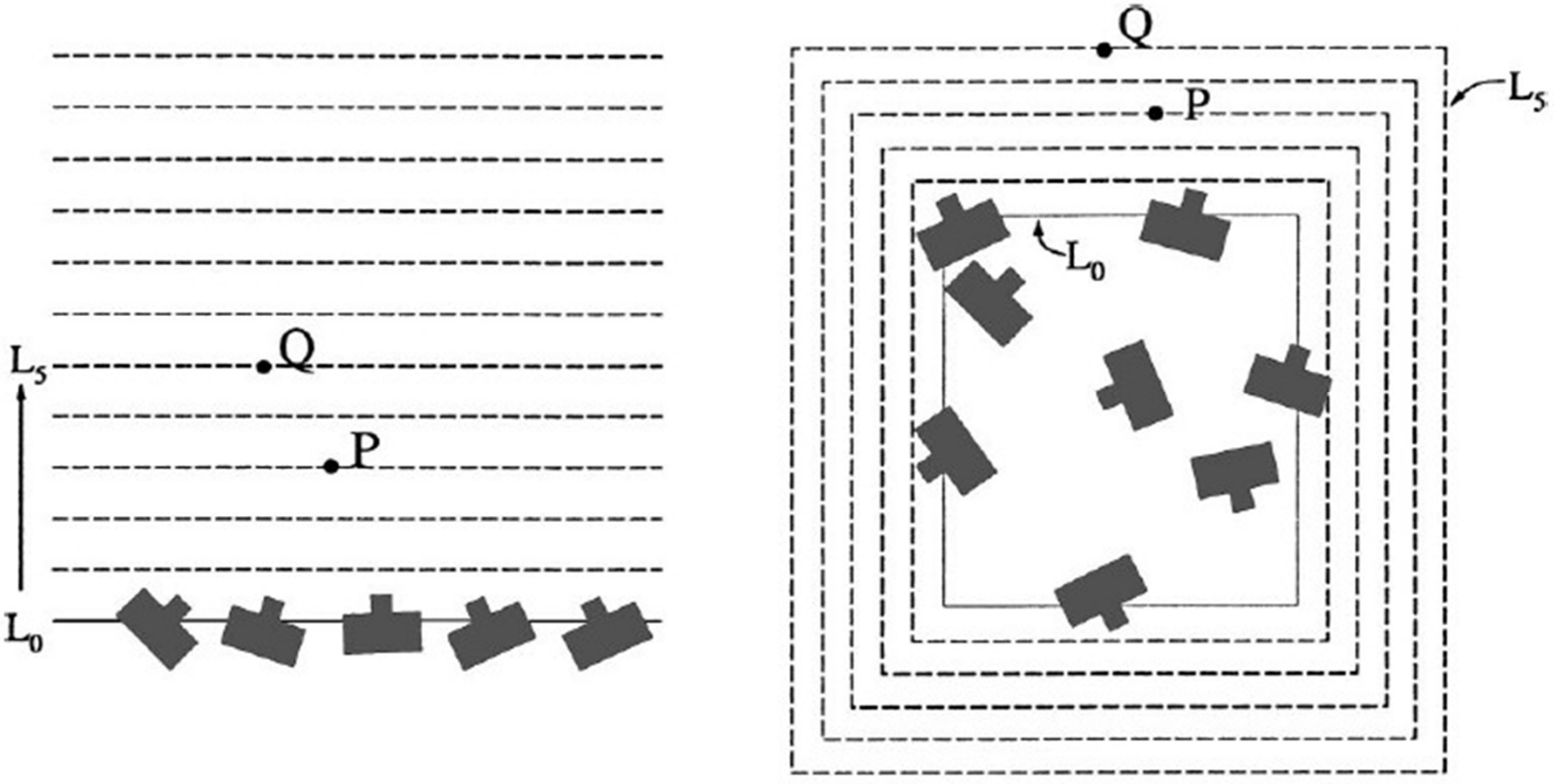
ALGORITHM

1. Iterate through the layers
2. Iterate through voxels in the layer
3. Project voxel to each image
4. Evaluate voxel photo-consistency
5. If photo-consistent
   - Color the voxel
   - Mark the image pixels (to determine voxel occlusion)

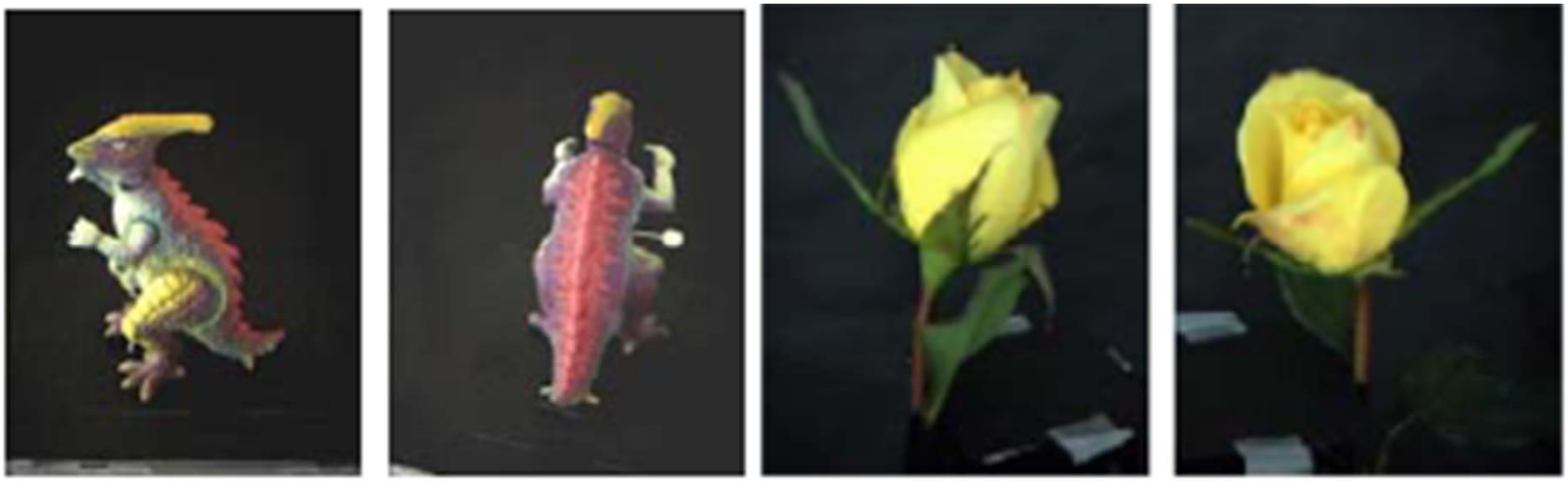
Voxel coloring

Constraint on the camera setting: if no scene point is contained within the convex hull of the camera centers the camera setting is compatible.

Examples of layered scene traversal



Voxel coloring



Space carving

The 3D shape recovery from a set of images is generally **ill-posed**: there may be multiple shapes that are consistent with the same set of images.

Image source: [S. Lazebnik]

Space carving

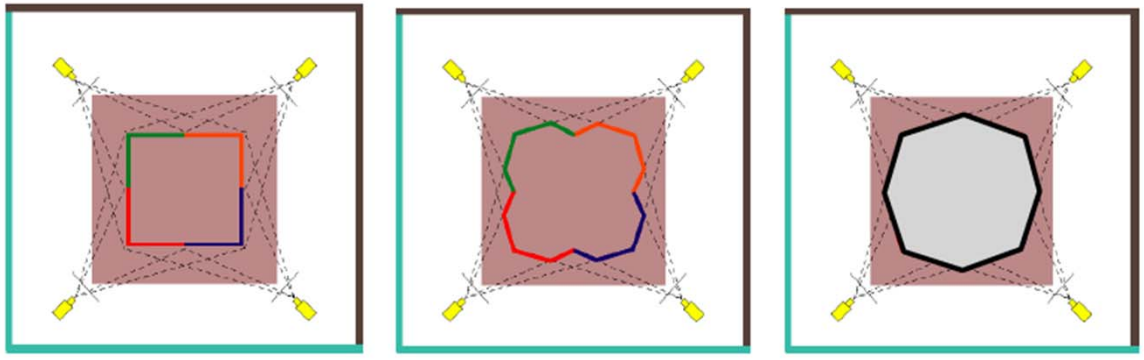
The 3D shape recovery from a set of images is generally **ill-posed**: there may be multiple shapes that are consistent with the same set of images. → Some sort of constraint has to be imposed

Image source: [S. Lazebnik]

Space carving

The 3D shape recovery from a set of images is generally **ill-posed**: there may be multiple shapes that are consistent with the same set of images. → Some sort of constraint has to be imposed
In particular:

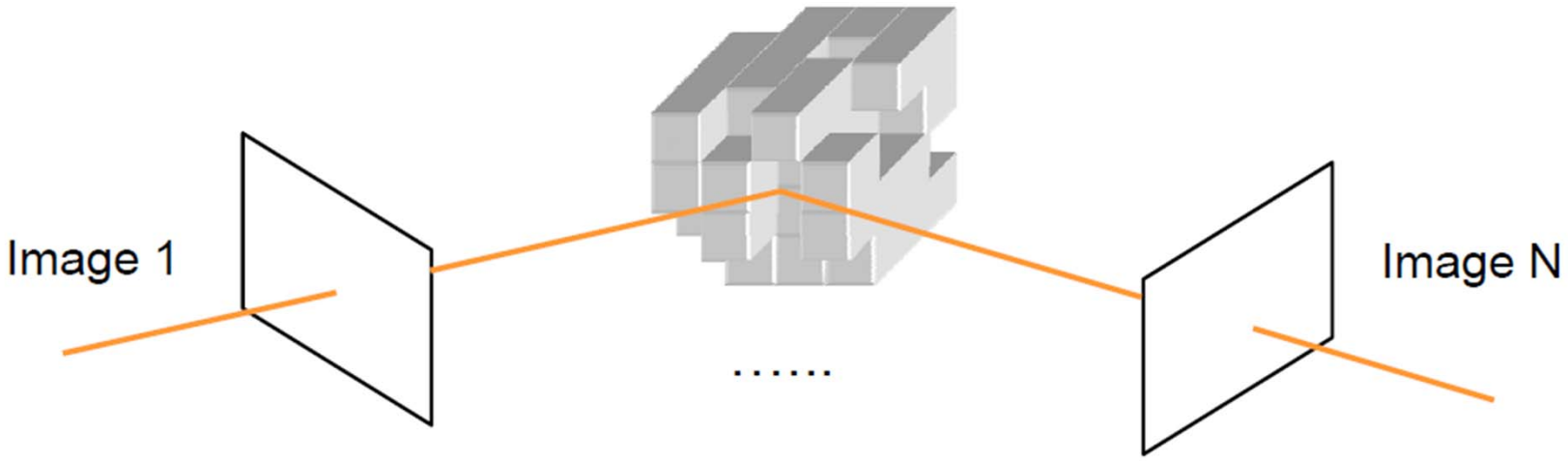
- The visual hull is defined as the maximal volume consistent with all the silhouettes.
- The voxel coloring uses the ordinal visibility constraint.
- Other multi-view stereo methods look for the shape with smoothest surface
- The space carving looks for the **photo hull**: the union of all photo-consistent shapes



True Scene Photo Hull Visual Hull

Image source: [S. Lazebnik]

Space carving

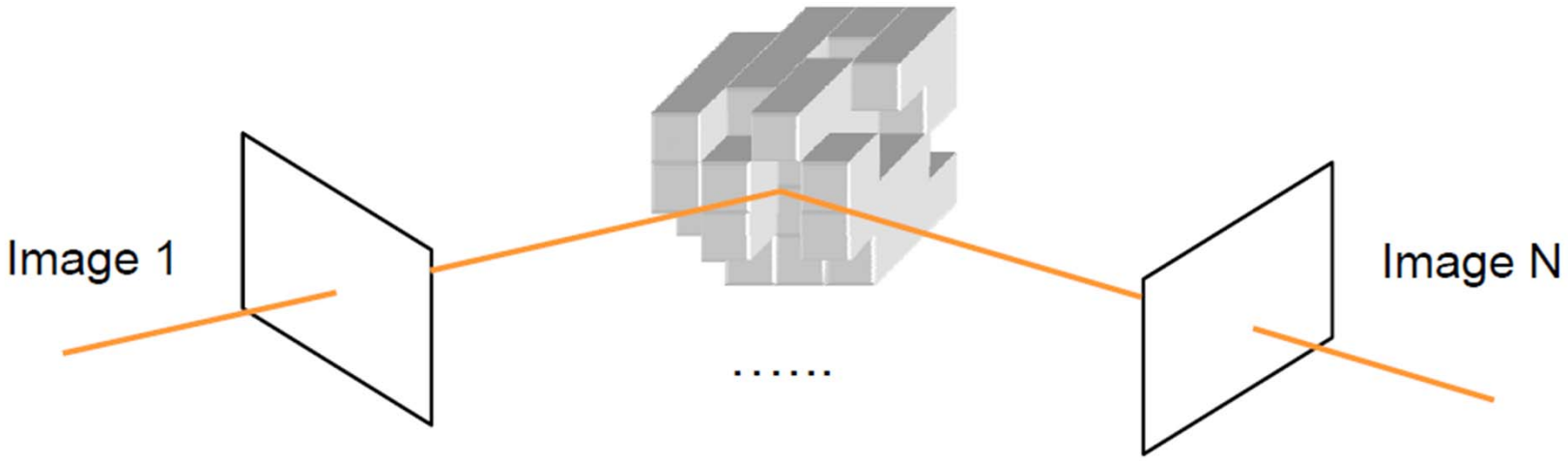


Space Carving Algorithm

- Initialize to a volume V containing the true scene
- Choose a voxel on the current surface
- Project to visible input images
- Carve if not photo-consistent
- Repeat until convergence

Image source: [S. Lazebnik]

Space carving



Space Carving Algorithm

- Initialize to a volume V containing the true scene
- Choose a voxel on the current surface
- Project to visible input images
- Carve if not photo-consistent
- Repeat until convergence

Needs to keep track of the scene visibility

Image source: [S. Lazebnik]

Space carving

Multi-view visibility ordering

Multi-view visibility orders do not exist in the general case BUT it is possible to define visibility orders that apply to a subset of the input cameras.

Image source: [Kutulakos and Seitz 2000]

Space carving

Multi-view visibility ordering

Multi-view visibility orders do not exist in the general case BUT it is possible to define visibility orders that apply to a subset of the input cameras.
→ No constraints on the camera setting

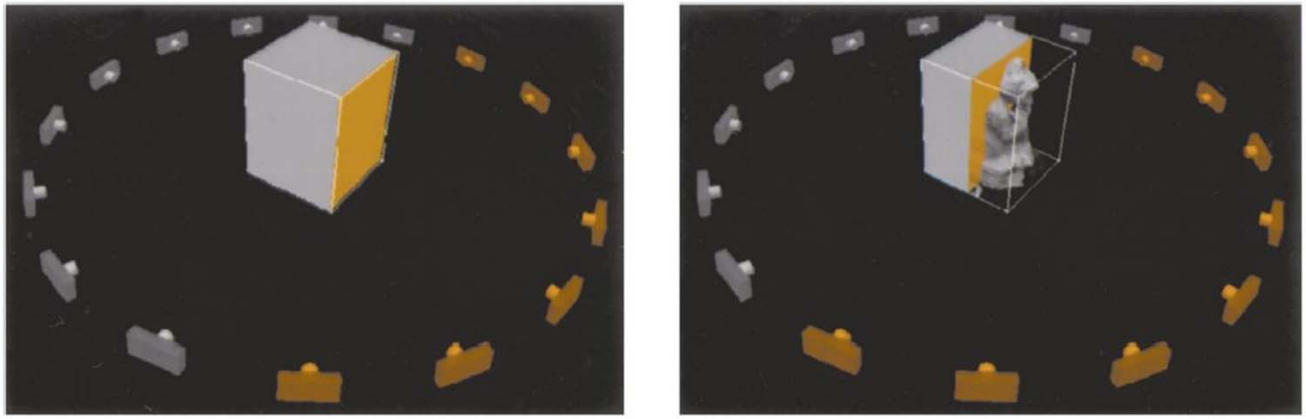
Image source: [Kutulakos and Seitz 2000]

Space carving

Multi-view visibility ordering

Multi-view visibility orders do not exist in the general case BUT it is possible to define visibility orders that apply to a subset of the input cameras.

→ No constraints on the camera setting



Multi-sweep space carving algorithm

Performs 6 sweeps through the volume, corresponding to the 6 principle directions (increasing and decreasing X, Y, and Z)

Image source: [Kutulakos and Seitz 2000]

Multi-view stereo methods

Volumetric methods based on an energy minimization.

They use a photo-consistency measure and sometimes combine the silhouettes information.

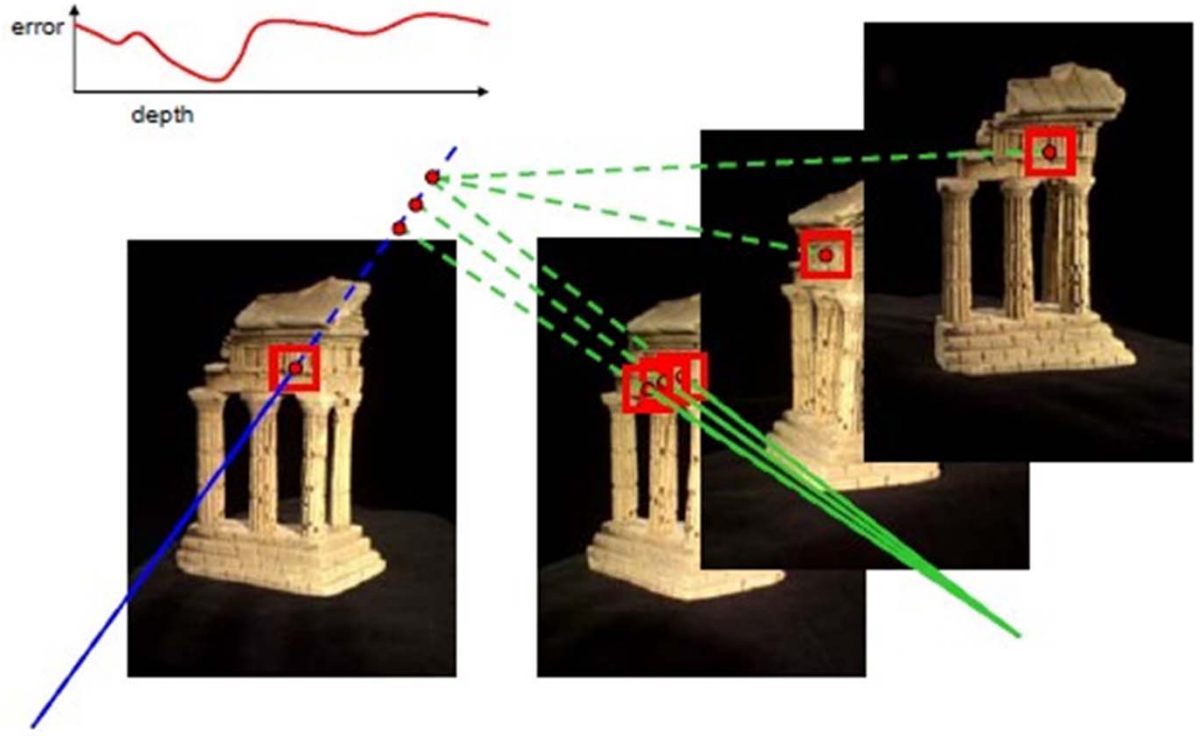
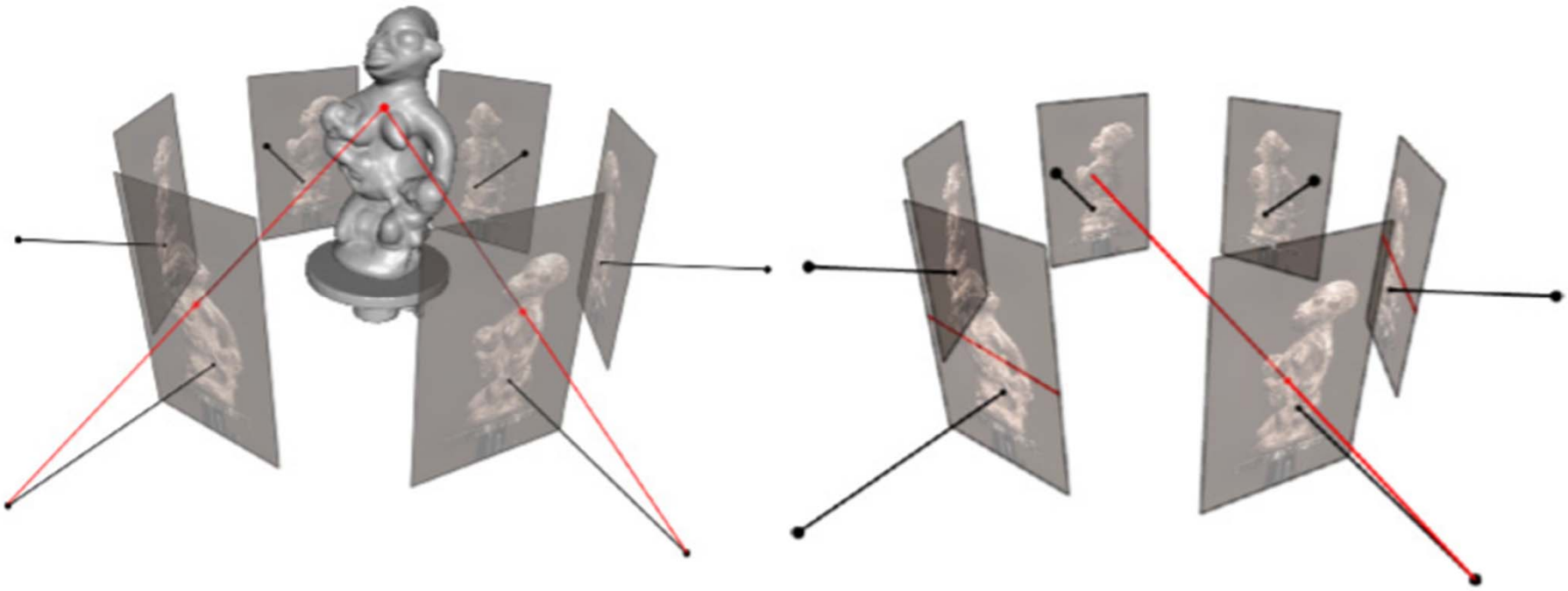


Image source: [S. Seitz]

Multi-view stereo methods



The 3D geometry of the structure defines a correspondence between pixeles in different images.

As the camera is known, the data association is equivalent to solve a 1D search problem (epipolar geometry).

Image source: [S. Seitz]

Multi-view stereo methods

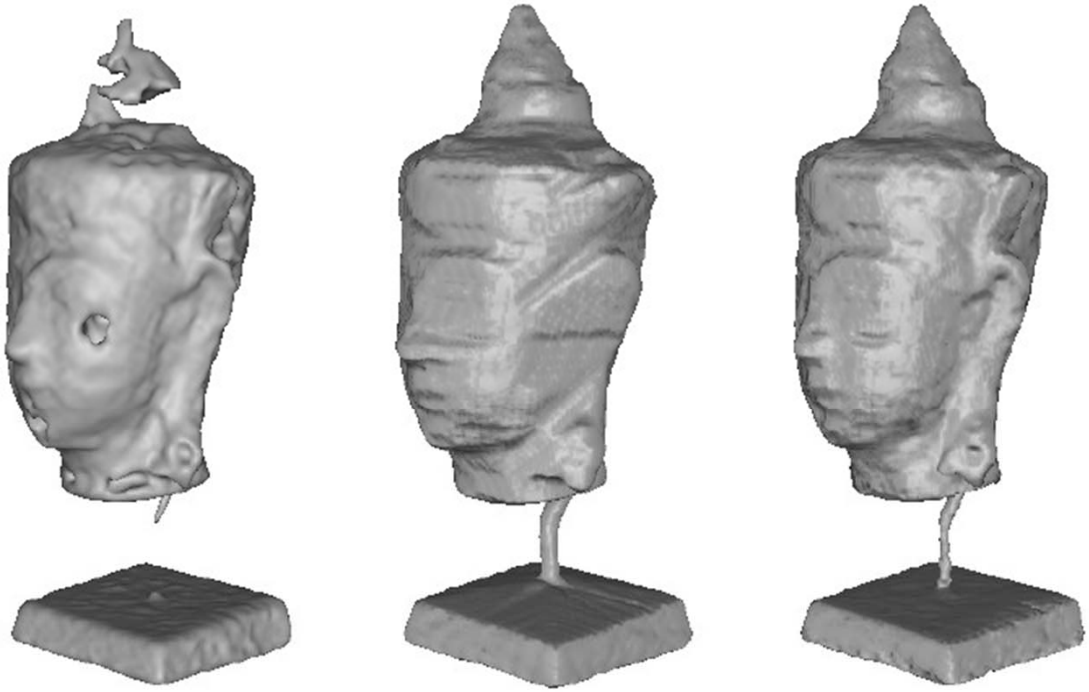


Image source: [Cremers and Kolev 2011]

Outline

- 3D reconstruction from multiple views
- Voxel-based methods
  - Shape from silhouette
    - Voxel coloring
    - Space carving
- Structure from motion.
- Stratified reconstruction.
- Projective reconstruction: Factorization method

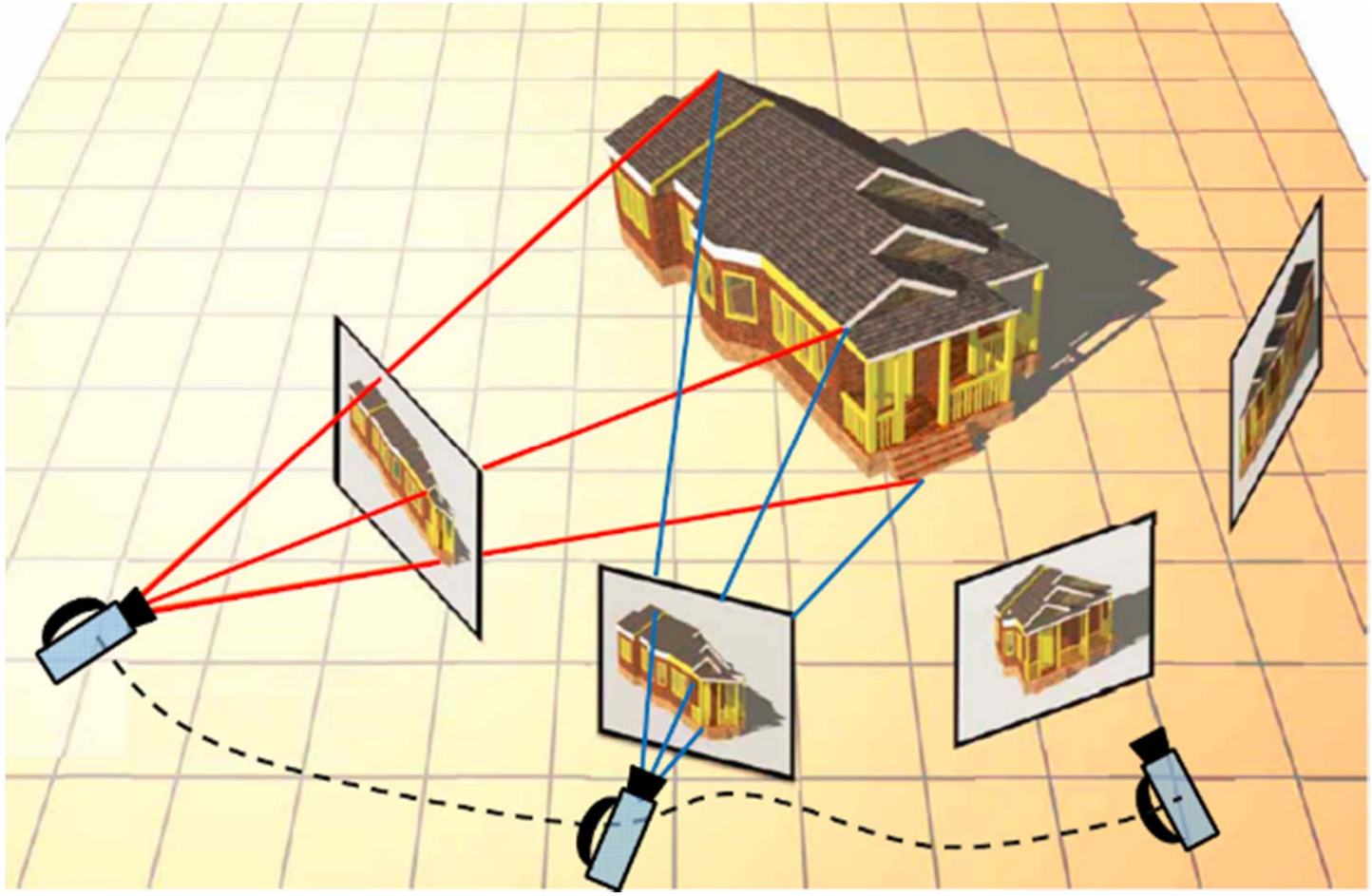
(Rigid) Structure from motion

Rigid Structure from Motion: 3D shape and camera motion

- The rigidity prior is enough to make the problem well posed
- Theory is solid and well-established
- Methods cannot be applied to non-rigid motions

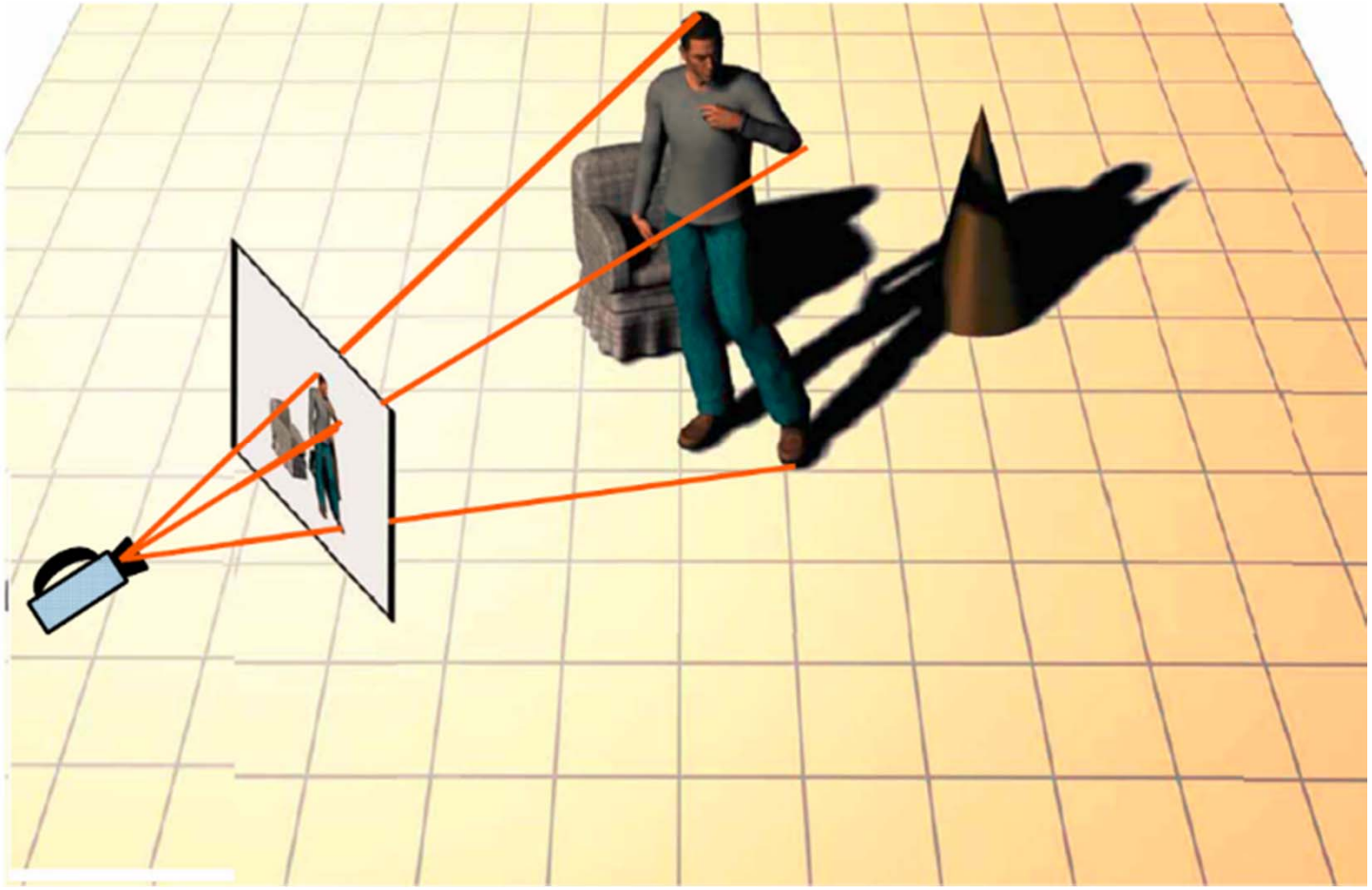
(Rigid) Structure from motion

- The rigidity prior is enough to make the problem well posed
- Theory is solid and well-established
- Methods cannot be applied to non-rigid motions



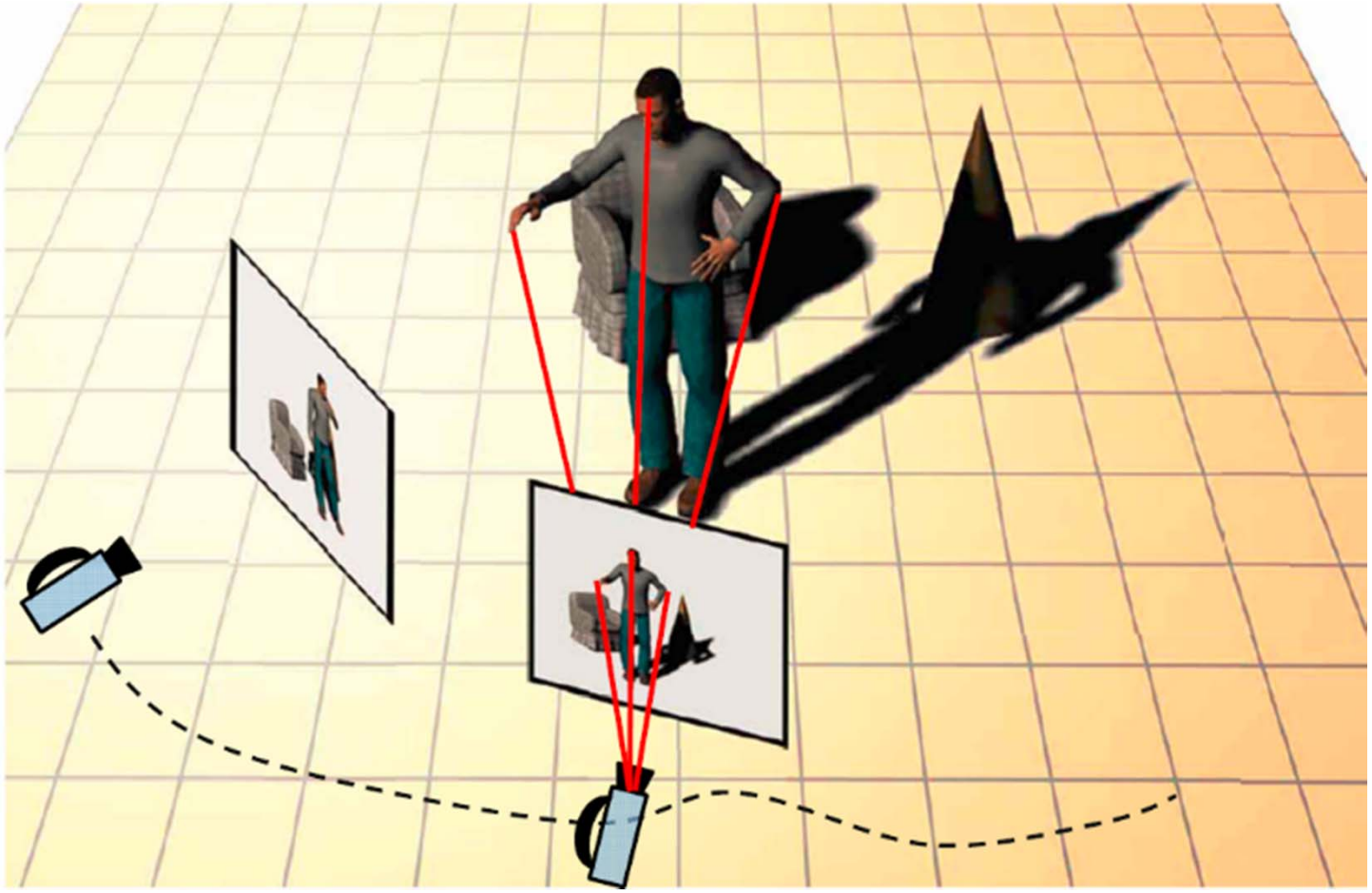
(Non-Rigid) Structure from motion

- Recovering non-rigid shape and motion from video is ill-posed
- Many different 3D shapes can have similar observations
- Additional priors are necessary



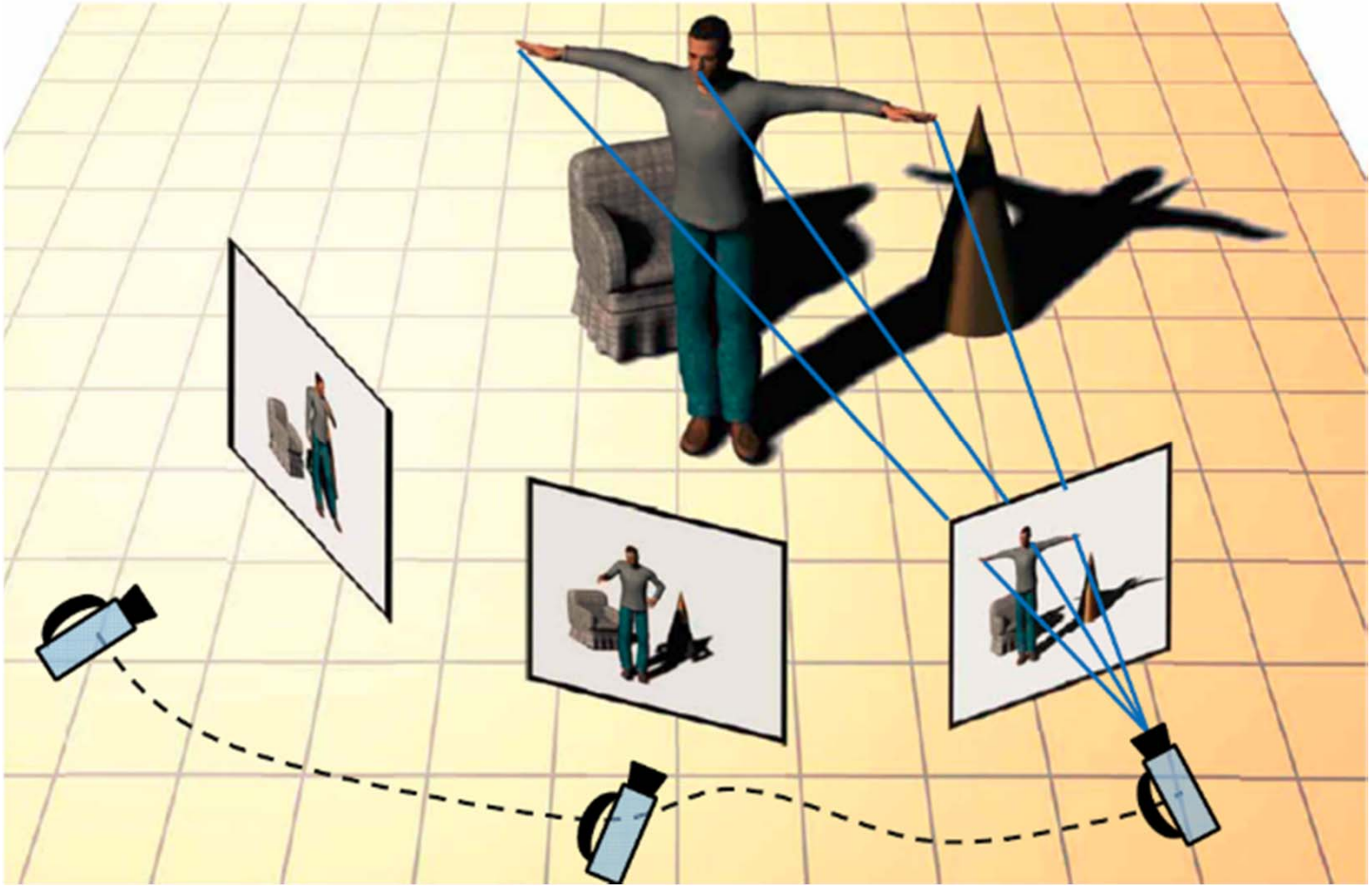
(Non-Rigid) Structure from motion

- Recovering non-rigid shape and motion from video is ill-posed
- Many different 3D shapes can have similar observations
- Additional priors are necessary



(Non-Rigid) Structure from motion

- Recovering non-rigid shape and motion from video is ill-posed
- Many different 3D shapes can have similar observations
- Additional priors are necessary



Structure from motion

Problem statement

Given a set of uncalibrated images and a set of image correspondences, compute the 3D points and the position, orientation, and *calibration of the cameras* (*P* matrices).

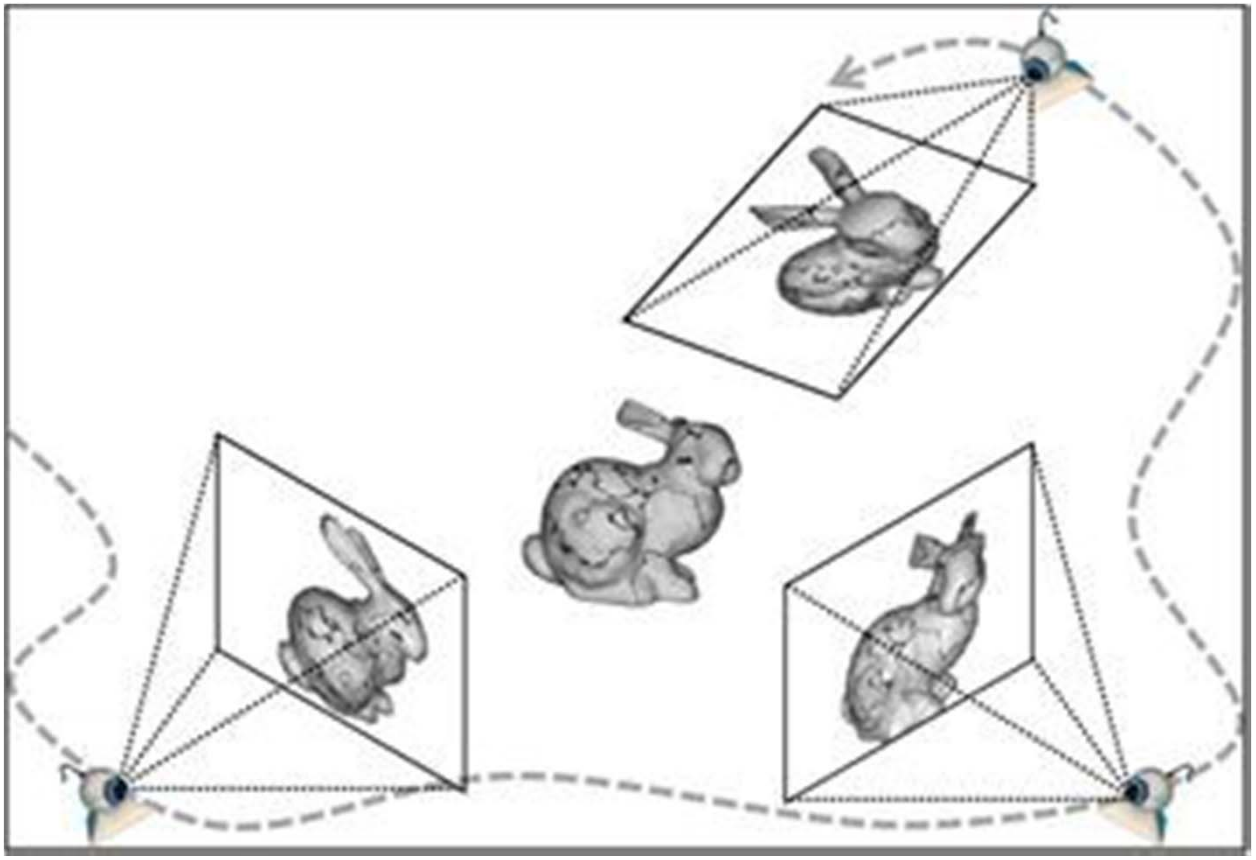


Image source: I. Mitsugami

Structure from motion

Image source: [Snavely et al. 2007]

Structure from motion

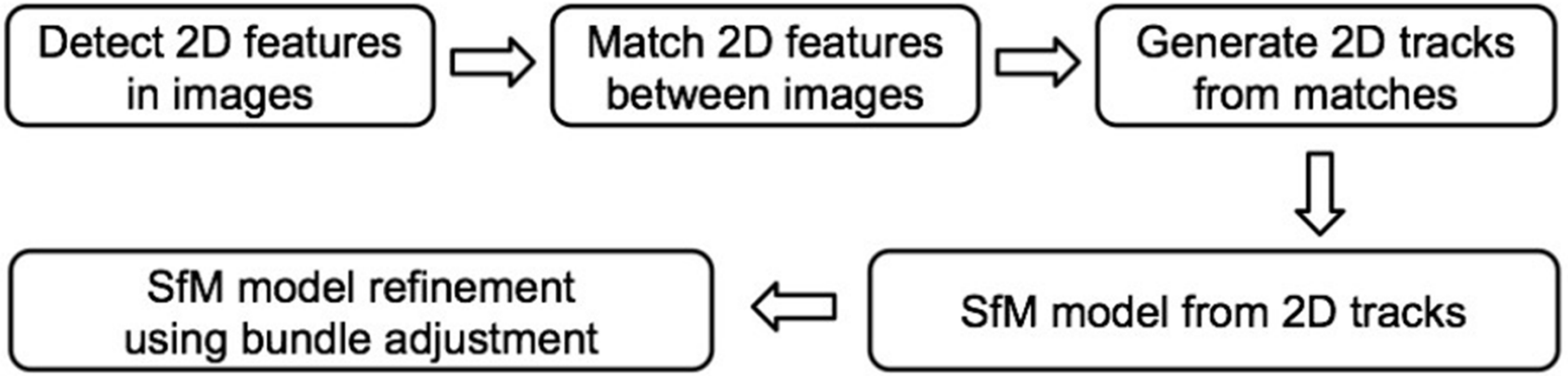
Video source: [Crandall et al. 2013]

Structure from motion + Semantic Segmentation

Video source: [Tateno et al. 2017]

Structure from motion

Main stages of a generic SfM pipeline



(Fine solution)
- Non-linear optimization

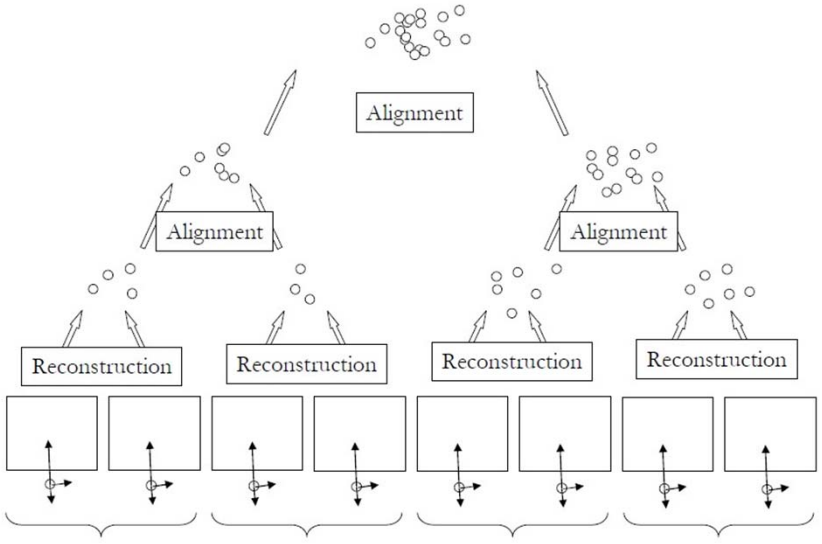
(Coarse solution)
- Closed-form solution by rigid factorization

Image source: [Furukawa and Hernández 2013]

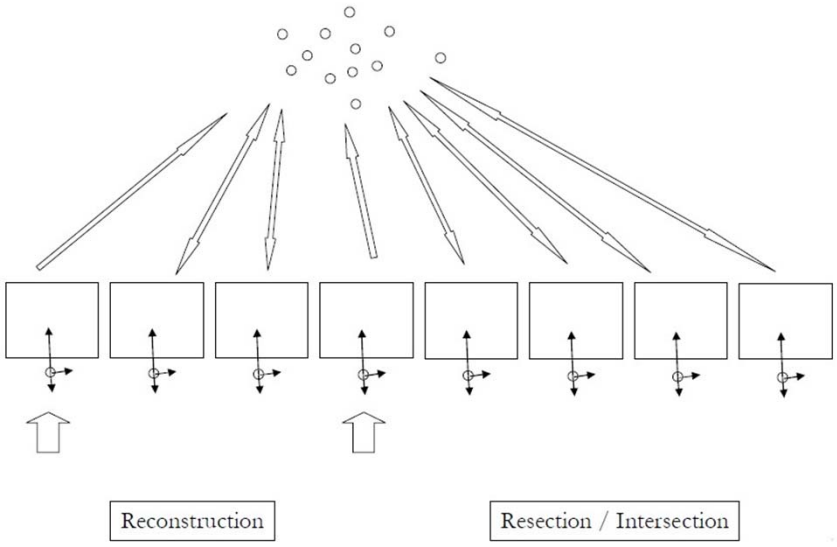
Structure from motion

Different approaches:

- Batch
  - Direct: all views at once
  - Hierarchical
- Sequential



Hierarchical



Sequential

Images source: A. Bartoli

Structure from motion: Batch vs. Sequential

NO FREE LUNCH

BATCH CASE	SEQUENTIAL CASE
✗ Estimation: After the video capture	✓ Estimation: As the data arrives
✗ Applicability: Off-line	✓ Applicability: On-line and real-time
✓ Accuracy: Depending on priors	✗ Accuracy: Less than batch

Outline

- 3D reconstruction from multiple views
- Voxel-based methods
  - Shape from silhouette
    - Voxel coloring
    - Space carving
- Structure from motion.
- Stratified reconstruction.
- Projective reconstruction: Factorization method

Projective ambiguity

Given a set of uncalibrated images and a set of image correspondences, compute the 3D points and the position, orientation, and calibration of the cameras (P matrices).

Image calibration contains an inherent **projective ambiguity**

$$\mathbf{x} = P\mathbf{X}, \text{ but also } \mathbf{x} = PH^{-1}H\mathbf{X} = \widehat{P}\widehat{\mathbf{X}}$$

Projective ambiguity

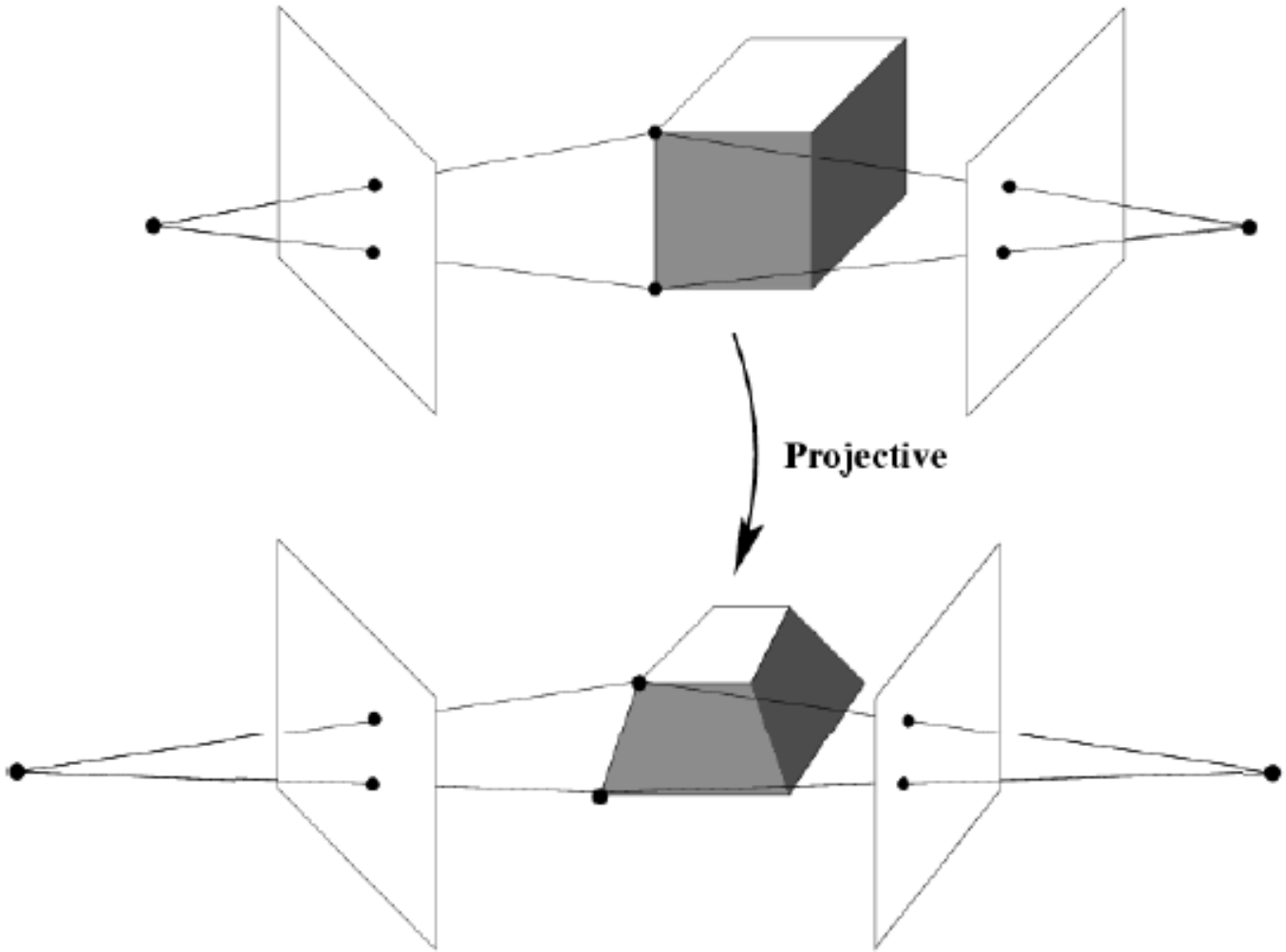


Image source: [Hartley and Zisserman 2004]

Projective reconstruction

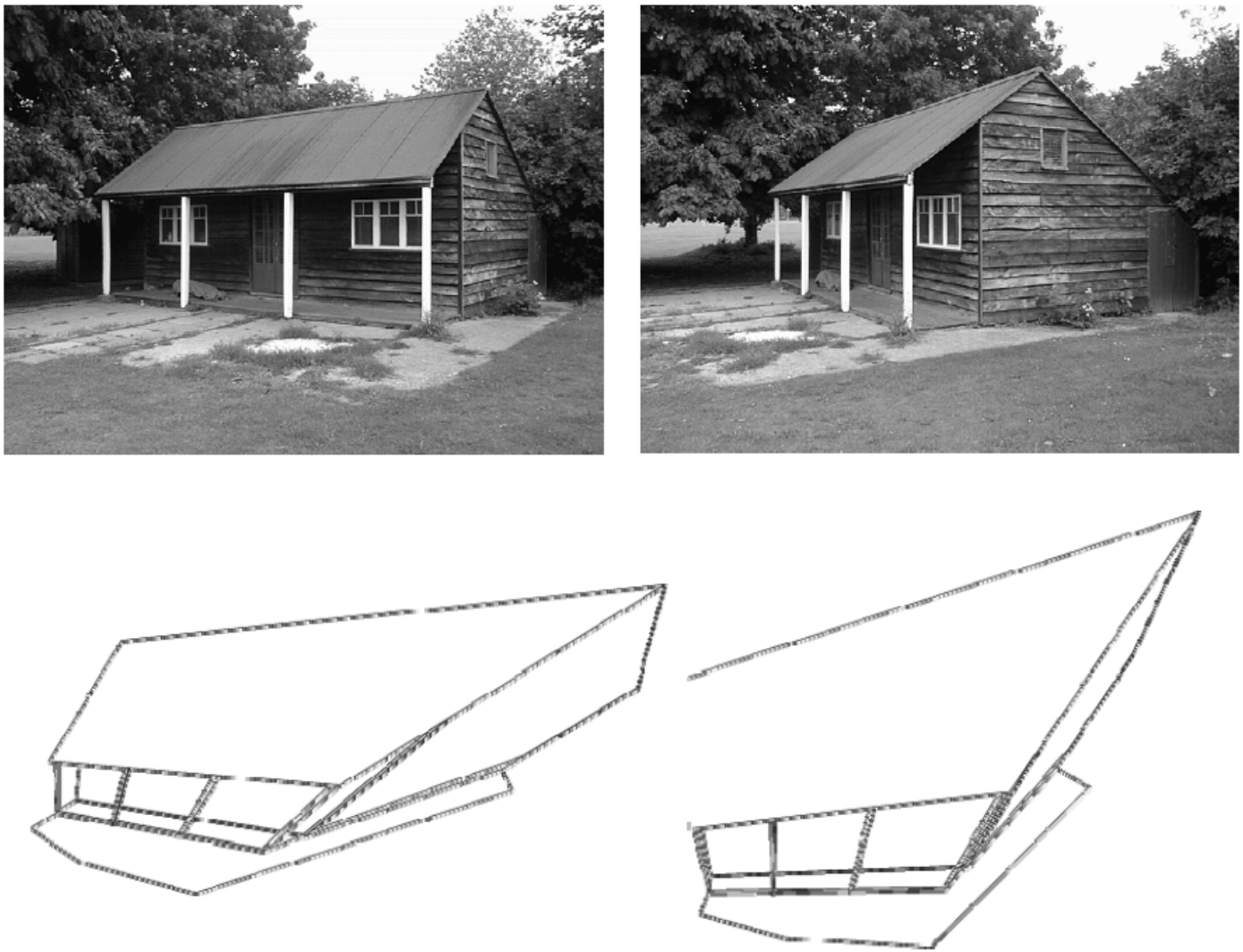


Image source: [Hartley and Zisserman 2004]

Stratified reconstruction

A solution to the projective ambiguity problem is the **stratified reconstruction**.

The main steps are:

1. Estimate a projective reconstruction
2. Upgrade the previous recons. to an affine reconstruction (OPTIONAL)
3. Upgrade the previous recons. to a metric reconstruction

Affine ambiguity

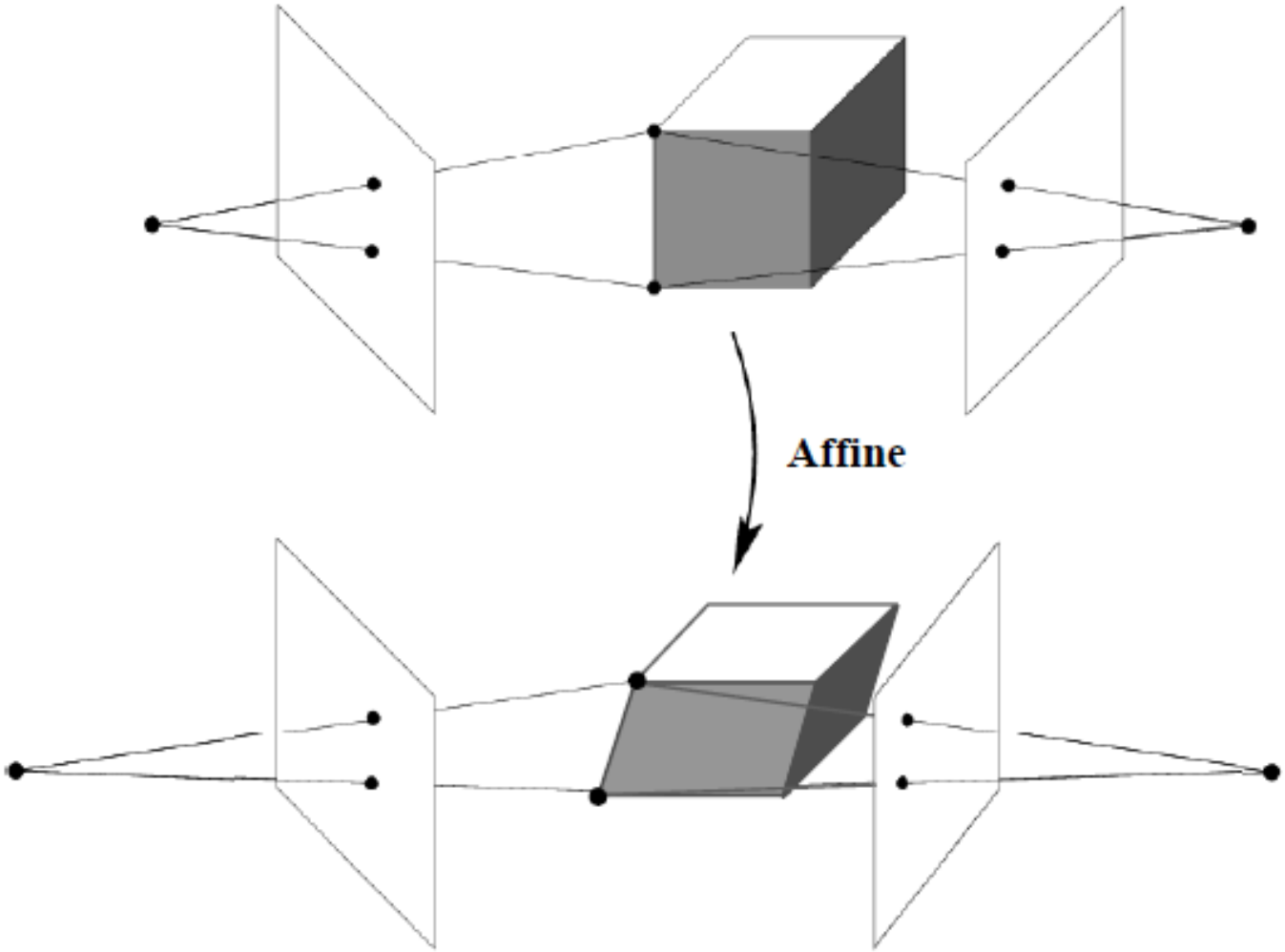


Image source: [Hartley and Zisserman 2004]

Affine reconstruction

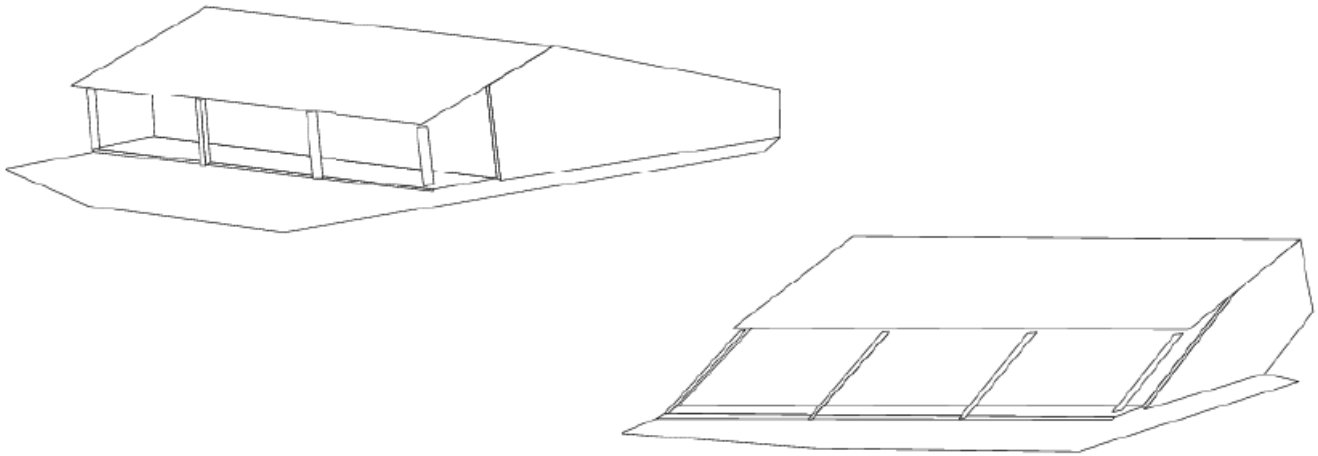


Image source: [Hartley and Zisserman 2004]

Similarity (or metric) ambiguity

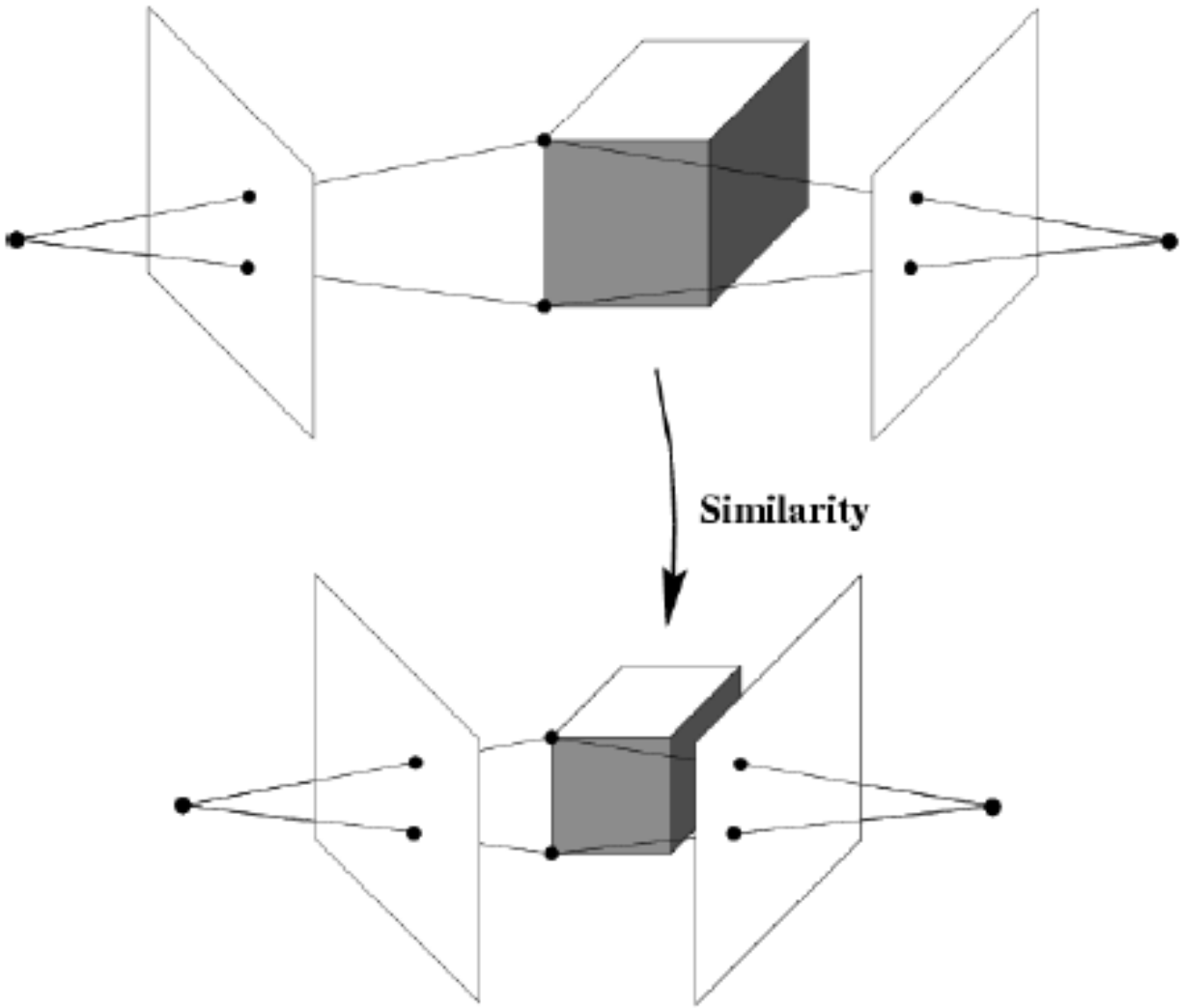


Image source: [Hartley and Zisserman 2004]

Metric reconstruction

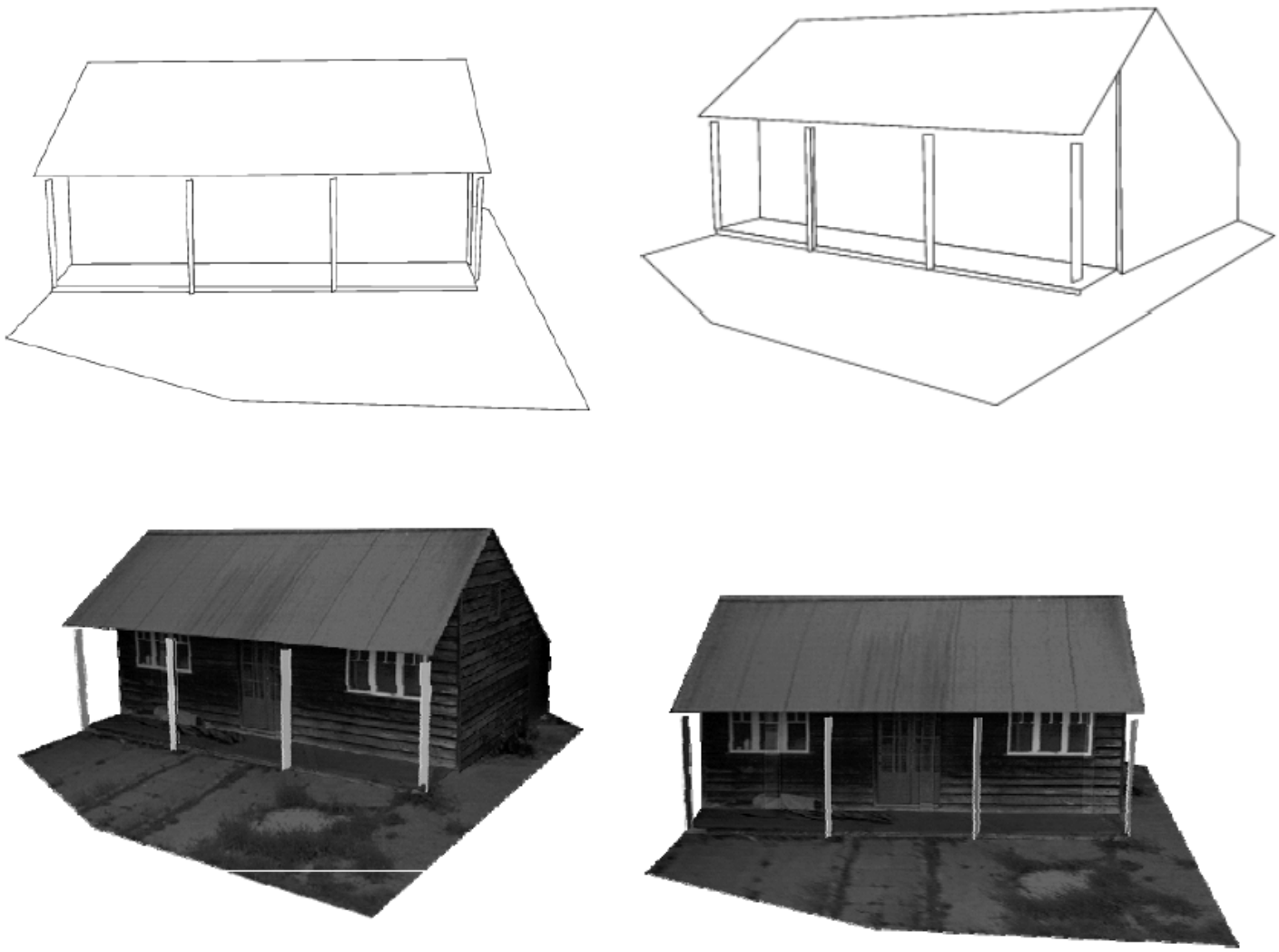


Image source: [Hartley and Zisserman 2004]

Outline

- 3D reconstruction from multiple views
- Voxel-based methods
  - Shape from silhouette
  - Voxel coloring
  - Space carving
- Stratified reconstruction
- Projective reconstruction: Factorization method

Projective reconstruction

Particular case of 2 views

If we estimate F we can extract two possible camera matrices:

$$P = [I \mid 0]$$

$$P' = [SF \mid e']$$

where S is any skew-symmetric matrix (such that P' has rank 3).
A good choice is: $S = [e']_x$.
The epipole may be computed from ${e'}^T F = 0$

Factorization method

Projective reconstruction method for 2 or more views
[P. Sturm and B. Triggs 1996]

Projective equations:

$$\mathbf{x}_j^i \equiv P^i \mathbf{X}_j \qquad \longrightarrow \qquad \lambda_j^i \mathbf{x}_j^i = P^i \mathbf{X}_j$$

where $j = 1, \ldots n$ denote the points, and
$i = 1, \ldots m$ denote the images (views)

Factorization method

Projective reconstruction method for 2 or more views
[P. Sturm and B. Triggs 1996]

Projective equations:

$$\mathbf{x}_j^i \equiv P^i \mathbf{X}_j \qquad \longrightarrow \qquad \lambda_j^i \mathbf{x}_j^i = P^i \mathbf{X}_j$$

where $j = 1, \ldots n$ denote the points, and
$i = 1, \ldots m$ denote the images (views)

Collect all projective eq's into a matrix equation:

$$\begin{bmatrix} \lambda_1^1 \mathbf{x}_1^1 & \lambda_2^1 \mathbf{x}_2^1 & \ldots & \lambda_n^1 \mathbf{x}_n^1 \\ \lambda_1^2 \mathbf{x}_1^2 & \lambda_2^2 \mathbf{x}_2^2 & \ldots & \lambda_n^2 \mathbf{x}_n^2 \\ \ldots & \ldots & \ldots & \ldots \\ \lambda_1^m \mathbf{x}_1^m & \lambda_2^m \mathbf{x}_2^m & \ldots & \lambda_n^m \mathbf{x}_n^m \end{bmatrix} = \begin{bmatrix} P^1 \\ P^2 \\ \ldots \\ P^m \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \ldots & \mathbf{X}_n \end{bmatrix}$$

λ_j^i are unknown scalar factors, called **projective depths**

Factorization method

Projective reconstruction method for 2 or more views
[P. Sturm and B. Triggs 1996]

Projective equations:

$$\mathbf{x}_j^i \equiv P^i \mathbf{X}_j \qquad \longrightarrow \qquad \lambda_j^i \mathbf{x}_j^i = P^i \mathbf{X}_j$$

where $j = 1, \ldots n$ denote the points, and
$i = 1, \ldots m$ denote the images (views)

Collect all projective eq's into a matrix equation:

$$\begin{bmatrix} \lambda_1^1 \mathbf{x}_1^1 & \lambda_2^1 \mathbf{x}_2^1 & \ldots & \lambda_n^1 \mathbf{x}_n^1 \\ \lambda_1^2 \mathbf{x}_1^2 & \lambda_2^2 \mathbf{x}_2^2 & \ldots & \lambda_n^2 \mathbf{x}_n^2 \\ \ldots & \ldots & \ldots & \ldots \\ \lambda_1^m \mathbf{x}_1^m & \lambda_2^m \mathbf{x}_2^m & \ldots & \lambda_n^m \mathbf{x}_n^m \end{bmatrix} = \begin{bmatrix} P^1 \\ P^2 \\ \ldots \\ P^m \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \ldots & \mathbf{X}_n \end{bmatrix}$$

λ_j^i are unknown scalar factors, called **projective depths**

Requires a set of points visible in all views!

Factorization method

We have

$$\underbrace{M}_{3m \times n} = \underbrace{\begin{bmatrix} P^1 \\ P^2 \\ \dots \\ P^m \end{bmatrix}}_{3m \times 4} \underbrace{\begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \dots & \mathbf{X}_n \end{bmatrix}}_{4 \times n}$$

$$M = \mathcal{P}_M \mathcal{X}_M$$

M is called the **measurement matrix**.

M has at most rank 4 $\rightarrow$ this suggests a factorization algorithm based on the SVD

Factorization method

$$M = \underbrace{U}_{3m \times n} \underbrace{D}_{n \times n} \underbrace{V^T}_{n \times n}$$

D should ideally have only four non-zero elements (M rank 4).

Factorization method

$$M = \underbrace{U}_{3m \times n} \underbrace{D}_{n \times n} \underbrace{V^T}_{n \times n}$$

D should ideally have only four non-zero elements (M rank 4).

We define D_4 as the $n \times 4$ submatrix of D.

Factorization method

$$M = \underbrace{U}_{3m \times n} \underbrace{D}_{n \times n} \underbrace{V^T}_{n \times n}$$

D should ideally have only four non-zero elements (M rank 4).

We define D_4 as the $n \times 4$ submatrix of D.

We write:

$$M = \underbrace{UD_4}_{3m \times 4} \underbrace{V_4^T}_{4 \times n}$$

where

$$UD_4 = \begin{bmatrix} P^1 \\ P^2 \\ \dots \\ P^m \end{bmatrix} \qquad V_4^T = \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \dots & \mathbf{X}_n \end{bmatrix}$$

Note: the factorization is not unique

Factorization method

Normalization of the data with a similarity transformation so that transformed points have zero mean and average distance from the origin of $\sqrt{2}$.

Apply the following similarity transformation to each image i:

$$H_s^i = \begin{pmatrix} s^i & 0 & -s^i c_x^i \\ 0 & s^i & -s^i c_y^i \\ 0 & 0 & 1 \end{pmatrix}$$

where centroid $c^i = (c_x^i, c_y^i)$, and $s^i = \frac{\sqrt{2}}{\text{mean dist to } c^i}$.

Limitation: the 3D points must be visible in all the views.

Factorization method

ALGORITHM

1. Determine a subset of scene points and cameras so that the measurement matrix is completely determined.
2. Normalize the set of points in each image (similarity transf. H_s).
3. Initialize all λ_j^i ($= 1$ or better initialization).
4. Alternate rescaling the rows of the depth matrix Λ (formed by λ_j^i) to have unit norm and the columns of Λ to have unit norm until Λ stops changing significantly (usually two loops).
5. Build the measurement matrix M.
6. Determine the SVD of $M = UDV^T$.
7. Let $\mathcal{P}_M = UD_4$ and $\mathcal{X}_M = V_4^T$.
8. If $\sum_i \sum_j d(x_j^i, P^i \mathbf{X}_j)^2$ converges then stop.
   Otherwise let $\lambda_j^i = (P^i \mathbf{X}_j)_3$ and go to Step 4.
9. Unnormalize the camera matrices $(H_s^i)^{-1} P^i$.
10. (Triangulate and resection the non-nucleus scene points and cameras).

References

[Hartley and Zisserman 2004] R.I. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, 2004.

[Hernández and Furukawa 2013] C. Hernández and Y. Furukawa, Multi-view stereo: A tutorial. Foundations and Trends® in Computer Graphics and Vision, 9 (1-2), 2013.

[K. Kutulakos and S. Seitz 2000] K. Kutulakos and S. Seitz, A theory of shape by space carving. Int. Journal of Computer Vision, 38(3) 2000.

[A. Laurentini 1994], The visual hull concept for silhouette-based image understanding, IEEE Trans. Pattern Analysis and Machine Learning, 16(2), 1994.

[S. Seitz and C. Dyer 1999] S. Seitz, and C. Dyer, Photorealistic Scene Reconstruction by Voxel Coloring. Int. Journal of Computer Vision, 35,(2) 1999.

[Snavely et al. 2007] N. Snavely, S. M. Seitz, R. Szeliski. Modeling the World from Internet Photo Collections. Int. Journal of Computer Vision, 2007.

[P. Sturm and B. Triggs 1996] P. Sturm and B. Triggs, A factorization based algorithm for multi-image projective structure and motion, European Conference on Computer Vision, 1996.

Master in
Computer Vision
Barcelona

Module: 3D Vision
Lecture 9: Auto-calibration.
Bundle adjustment.
Lecturer: Gloria Haro

Outline

- Auto-calibration
- Bundle adjustment

Auto-calibration

Auto-calibration is the process of **determining the internal camera parameters** directly from multiple uncalibrated images.

There are different methods for autocalibration. We will see the stratified methods which involve two steps:

- Affine reconstruction: it finds the plane at infinity Π_∞
- Metric reconstruction: it finds the image of the absolute conic $\omega = K^{-T}K^{-1}$ (and thus the internal parameters)

Auto-calibration

Problem statement

Assume we already have a projective reconstruction, i.e., we know

- the camera matrices: P_p^i, $i = 1, ..., m$
- the 3D points: $\mathbf{X}_{j,p}$, $j = 1, ..., n$

such that $\mathbf{x_j^i} \equiv P_p^i \mathbf{X}_{j,p}$ $\forall i, j$ (subindex p stands for *projective*)

Auto-calibration

Problem statement

Assume we already have a projective reconstruction, i.e., we know

- the camera matrices: P_p^i, $i = 1, ..., m$
- the 3D points: $\mathbf{X}_{j,p}$, $j = 1, ..., n$

such that $\mathbf{x_j^i} \equiv P_p^i \mathbf{X}_{j,p}\ \forall i, j$ (subindex p stands for *projective*)

There is a **projective ambiguity**:

$$\mathbf{x_j^i} \equiv P_p^i H^{-1} H \mathbf{X}_{j,p} = \hat{P}^i \hat{\mathbf{X}}_j$$

Auto-calibration

Problem statement

Assume we already have a projective reconstruction, i.e., we know

- the camera matrices: P_p^i, $i = 1, ..., m$
- the 3D points: $\mathbf{X}_{j,p}$, $j = 1, ..., n$

such that $\mathbf{x}_\mathbf{j}^\mathbf{i} \equiv P_p^i \mathbf{X}_{j,p}\ \forall i, j$ (subindex p stands for *projective*)

There is a **projective ambiguity**:

$$\mathbf{x}_\mathbf{j}^\mathbf{i} \equiv P_p^i H^{-1} H \mathbf{X}_{j,p} = \hat{P}^i \hat{\mathbf{X}}_j$$

Goal: Find a projective mapping $H_{e \leftarrow p}$ that maps the projective recons. to a metric one.

Auto-calibration

Problem statement

Assume we already have a projective reconstruction, i.e., we know

- the camera matrices: $P_p^i,\ i = 1, ..., m$
- the 3D points: $\mathbf{X}_{j,p},\ j = 1, ..., n$

such that $\mathbf{x_j^i} \equiv P_p^i \mathbf{X}_{j,p}\ \forall i, j$ (subindex p stands for *projective*)

There is a **projective ambiguity**:

$$\mathbf{x_j^i} \equiv P_p^i H^{-1} H \mathbf{X}_{j,p} = \hat{P}^i \hat{\mathbf{X}}_j$$

Goal: Find a projective mapping $H_{e \leftarrow p}$ that maps the projective recons. to a metric one.

After the mapping, in the Euclidean reference system we have:

- the camera matrices: $P_e^i = P_p^i H_{e \leftarrow p}^{-1},\ i = 1, ..., m$
- the 3D points: $\mathbf{X}_{j,e} = H_{e \leftarrow p} \mathbf{X}_{j,p},\ j = 1, ..., n$

Auto-calibration

By choosing the world reference system in the first camera we have:

$$P_e^1 = K[I|0]$$

We also may choose the projective reconstruction so that:

$$P_p^1 = [I|0]$$

Auto-calibration

By choosing the world reference system in the first camera we have:

$$P_e^1 = K[I|0]$$

We also may choose the projective reconstruction so that:

$$P_p^1 = [I|0]$$

Recall that a 4×4 projective matrix can be written as:

$$H_{e\leftarrow p}^{-1} = \begin{pmatrix} A & \tau \\ \mathbf{v}^T & v \end{pmatrix}$$

From $P_e^1 = P_p^1 H_{e\leftarrow p}^{-1}$, we have:

$$K[I|0] = [I|0] \begin{pmatrix} A & \tau \\ \mathbf{v}^T & v \end{pmatrix}$$

then:

$$A = K \text{ and } \tau = \mathbf{0}$$

Auto-calibration

We had:

$$A = K \text{ and } \tau = \mathbf{0}$$

If we fix $v = 1$, we may write:

$$H_{e \leftarrow p}^{-1} = \begin{pmatrix} K & \mathbf{0} \\ \mathbf{v}^T & 1 \end{pmatrix}$$

where K is the calibration matrix of the first camera.

Auto-calibration

We had:

$$A = K \text{ and } \tau = \mathbf{0}$$

If we fix $v = 1$, we may write:

$$H_{e \leftarrow p}^{-1} = \begin{pmatrix} K & \mathbf{0} \\ \mathbf{v}^T & 1 \end{pmatrix}$$

where K is the calibration matrix of the first camera.

Thus, we look for a matrix $H_{e \leftarrow p}$ with this structure so as to obtain an Euclidean (metric) reconstruction from a projective one.

Auto-calibration

We had:

$$A = K \text{ and } \tau = \mathbf{0}$$

If we fix $v = 1$, we may write:

$$H_{e \leftarrow p}^{-1} = \begin{pmatrix} K & \mathbf{0} \\ \mathbf{v}^T & 1 \end{pmatrix}$$

where K is the calibration matrix of the first camera.

Thus, we look for a matrix $H_{e \leftarrow p}$ with this structure so as to obtain an Euclidean (metric) reconstruction from a projective one.

Notice that

$$H_{e \leftarrow p} = \begin{pmatrix} K^{-1} & \mathbf{0} \\ -\mathbf{v}^T K^{-1} & 1 \end{pmatrix}$$

Auto-calibration

$$H_{e \leftarrow p} = \begin{pmatrix} K^{-1} & \mathbf{0} \\ -\mathbf{v}^T K^{-1} & 1 \end{pmatrix}$$

What is v?

Auto-calibration

$$H_{e\leftarrow p} = \begin{pmatrix} K^{-1} & \mathbf{0} \\ -\mathbf{v}^T K^{-1} & 1 \end{pmatrix}$$

What is v?

Observe that the coordinates of the plane at infinity in an Euclidean frame of $\mathbb{P}^3$ are:

$$\Pi_{\infty,e} = (0,0,0,1)^T$$

Remember how does a homography acts on a plane: $\Pi' = H^{-T}\Pi$. Then,

$$\Pi_{\infty,e} = (0,0,0,1)^T = H^{-T}_{e\leftarrow p}\Pi_{\infty,p}$$

Auto-calibration

$$H_{e\leftarrow p} = \begin{pmatrix} K^{-1} & \mathbf{0} \\ -\mathbf{v}^T K^{-1} & 1 \end{pmatrix}$$

What is v?

Observe that the coordinates of the plane at infinity in an Euclidean frame of $\mathbb{P}^3$ are:

$$\Pi_{\infty,e} = (0,0,0,1)^T$$

Remember how does a homography acts on a plane: $\Pi' = H^{-T}\Pi$. Then,

$$\Pi_{\infty,e} = (0,0,0,1)^T = H^{-T}_{e\leftarrow p}\Pi_{\infty,p}$$

and

$$\Pi_{\infty,p} = H^{T}_{e\leftarrow p}\Pi_{\infty,e} = \begin{pmatrix} K^{-T} & -K^{-T}\mathbf{v} \\ -\mathbf{0}^T & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -K^{-T}\mathbf{v} \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{p} \\ 1 \end{pmatrix}$$

Hence $\mathbf{p} = -K^{-T}\mathbf{v} \longrightarrow \mathbf{v} = -K^T\mathbf{p}$

Auto-calibration

Then, we have

$$H_{e\leftarrow p}^{-1} = \begin{pmatrix} K & \mathbf{0} \\ -\mathbf{p}^T K & 1 \end{pmatrix}$$

Auto-calibration

Then, we have

$$H_{e\leftarrow p}^{-1} = \begin{pmatrix} K & \mathbf{0} \\ -\mathbf{p}^T K & 1 \end{pmatrix} = \underbrace{\begin{pmatrix} I & \mathbf{0} \\ -\mathbf{p}^T & 1 \end{pmatrix}}_{H_{a\leftarrow p}^{-1}} \underbrace{\begin{pmatrix} K & \mathbf{0} \\ -\mathbf{0}^T & 1 \end{pmatrix}}_{H_{e\leftarrow a}^{-1}}$$

where $H_{a\leftarrow p}^{-1}$ is a projective transformation, and
$H_{e\leftarrow a}^{-1}$ is an affine transformation.

Auto-calibration

Then, we have

$$H_{e\leftarrow p}^{-1} = \begin{pmatrix} K & \mathbf{0} \\ -\mathbf{p}^T K & 1 \end{pmatrix} = \underbrace{\begin{pmatrix} I & \mathbf{0} \\ -\mathbf{p}^T & 1 \end{pmatrix}}_{H_{a\leftarrow p}^{-1}} \underbrace{\begin{pmatrix} K & \mathbf{0} \\ -\mathbf{0}^T & 1 \end{pmatrix}}_{H_{e\leftarrow a}^{-1}}$$

where $H_{a\leftarrow p}^{-1}$ is a projective transformation, and
$H_{e\leftarrow a}^{-1}$ is an affine transformation.

Notice that, $H_{e\leftarrow p} = H_{e\leftarrow a} H_{a\leftarrow p}$

Auto-calibration

Then, we have

$$H_{e\leftarrow p}^{-1} = \begin{pmatrix} K & \mathbf{0} \\ -\mathbf{p}^T K & 1 \end{pmatrix} = \underbrace{\begin{pmatrix} I & \mathbf{0} \\ -\mathbf{p}^T & 1 \end{pmatrix}}_{H_{a\leftarrow p}^{-1}} \underbrace{\begin{pmatrix} K & \mathbf{0} \\ -\mathbf{0}^T & 1 \end{pmatrix}}_{H_{e\leftarrow a}^{-1}}$$

where $H_{a\leftarrow p}^{-1}$ is a projective transformation, and
$H_{e\leftarrow a}^{-1}$ is an affine transformation.

Notice that, $H_{e\leftarrow p} = H_{e\leftarrow a} H_{a\leftarrow p}$

- $H_{a\leftarrow p}$ permits to upgrade the projective recons. into an affine one
  Affine reconstruction
- $H_{e\leftarrow a}$ permits to upgrade the affine recons. into a metric one
  Metric reconstruction

Auto-calibration – Affine reconstruction

We need to find the homography:

$$H_{a \leftarrow p} = \begin{pmatrix} I & \mathbf{0} \\ \mathbf{p}^T & 1 \end{pmatrix}$$

The essence of the affine reconstruction is then to locate the plane at infinity in the projective reconstruction frame.

Auto-calibration – Affine reconstruction

We need to find the homography:

$$H_{a \leftarrow p} = \begin{pmatrix} I & \mathbf{0} \\ \mathbf{p}^T & 1 \end{pmatrix}$$

The essence of the affine reconstruction is then to locate the plane at infinity in the projective reconstruction frame.

By applying $H_{a \leftarrow p}$ we will map $\mathbf{p}$ to $\Pi_\infty = (0, 0, 0, 1)^T$ and we will recover te paralellism.

Auto-calibration – Affine reconstruction

How do we identify p?
As every plane, the plane at infinity is determined by **three points on it**.

Procedure:
since $\Pi^T \mathbf{X}_i = 0$, $i = 1, 2, 3$, we have:

$$\underbrace{\begin{pmatrix} \mathbf{X}_1^T \\ \mathbf{X}_2^T \\ \mathbf{X}_3^T \end{pmatrix}}_{A\,(3\times 4 \text{ matrix})} \Pi = \mathbf{0}$$

If the three points are in general position (not on the same line), they provide linearly indep. eq. and the matrix they form is rank 3.

$\longrightarrow$ Then Π is obtained uniquely (up to scale) as the 1-dimensional right null space of A.

Auto-calibration – Affine reconstruction

How do we find the three points on the plane at infinity?

Auto-calibration – Affine reconstruction

How do we find the three points on the plane at infinity?
They are 3D vanishing points, coming from three pairs of parallel lines with different directions.

Auto-calibration – Affine reconstruction

How do we find the three points on the plane at infinity?
They are 3D vanishing points, coming from three pairs of parallel lines with different directions.

How do they may be computed from the images?

Auto-calibration – Affine reconstruction

How do we find the three points on the plane at infinity?
They are 3D vanishing points, coming from three pairs of parallel lines with different directions.

How do they may be computed from the images?

a) Pick three vanishing points in two images and find their corresponding 3D points by triangulation.

b) Identify a vanishing point $\mathbf{v}_i$ in one image and one of the lines ℓ'_i (image of one of the parallel lines of the scene leading to $\mathbf{v}'_i$) in the second image. Then, find $\mathbf{V}_i \in \mathbb{P}^3$ by solving:

$$\left.\begin{array}{r} \mathbf{v}_i \times P\mathbf{V}_i = \mathbf{0} \\ \ell_i'^T P'\mathbf{V}_i = 0 \end{array}\right\} \longrightarrow A\mathbf{V}_i = \mathbf{0}$$

$\mathbf{V}_i$ is the right null vector of A.

Auto-calibration

Auto-calibration – Metric reconstruction

The key to metric reconstruction is to find the image of the absolute conic in one of the images $\omega = K^{-T}K^{-1}$

Auto-calibration – Metric reconstruction

The key to metric reconstruction is to find the image of the absolute conic in one of the images $\omega = K^{-T}K^{-1}$

Suppose that the image of the absolute conic is known in some image to be ω, and one has an affine recons. in which the corresponding camera matrix is given by $P = [M|m]$. Then, the affine recons. may be transformed to a metric recons. by applying a 3D transformation of the form:

$$H_{e \leftarrow a} = \begin{pmatrix} A^{-1} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{pmatrix}$$

where A is obtained by Cholesky factorization: $AA^T = (M^T \omega M)^{-1}$

Auto-calibration – Metric reconstruction

The approach relies on identiying ω. There are various ways of doing this.

There are different kinds of constraints on ω:

- Constraints coming from scene orthogonality.
- Constraints coming from known internal parameters.
- Constraints arising from the same cameras (same matrix K) in all images.

Typically, a combination of these constraints is used.

Auto-calibration – Metric reconstruction

Constraints coming from scene orthogonality

If $\mathbf{v}_1$ and $\mathbf{v}_2$ is a pair of vanishing points arising from orthogonal scene lines, then we have a linear constraint on ω:

$$\mathbf{v}_1^T \omega \mathbf{v}_2 = 0$$

Auto-calibration – Metric reconstruction

Constraints coming from known internal parameters

Since

$$\omega = K^{-T}K^{-1} = \begin{pmatrix} \omega_{11} & \omega_{12} & \omega_{13} \\ \omega_{12} & \omega_{22} & \omega_{23} \\ \omega_{13} & \omega_{23} & \omega_{33} \end{pmatrix}$$

knowledge about some restrictions on the internal parameters contained in K may be used to constraint or determine the elements of ω.

Auto-calibration – Metric reconstruction

Constraints coming from known internal parameters

Since

$$\omega = K^{-T}K^{-1} = \begin{pmatrix} \omega_{11} & \omega_{12} & \omega_{13} \\ \omega_{12} & \omega_{22} & \omega_{23} \\ \omega_{13} & \omega_{23} & \omega_{33} \end{pmatrix}$$

knowledge about some restrictions on the internal parameters contained in K may be used to constraint or determine the elements of ω.

In the case where the camera is known to have zero skew, then $\omega_{12} = 0$.

Auto-calibration – Metric reconstruction

Constraints coming from known internal parameters

Since

$$\omega = K^{-T}K^{-1} = \begin{pmatrix} \omega_{11} & \omega_{12} & \omega_{13} \\ \omega_{12} & \omega_{22} & \omega_{23} \\ \omega_{13} & \omega_{23} & \omega_{33} \end{pmatrix}$$

knowledge about some restrictions on the internal parameters contained in K may be used to constraint or determine the elements of ω.

In the case where the camera is known to have zero skew, then $\omega_{12} = 0$.

If the pixels are square, that is, zero skew and $\alpha_x = \alpha_y$, then: $\omega_{11} = \omega_{22}$.

Auto-calibration – Metric reconstruction

Combination of the previous constraints

Five constraints on ω:

$$\mathbf{u}^T \omega \mathbf{v} = 0$$

$$\mathbf{u}^T \omega \mathbf{z} = 0$$

$$\mathbf{v}^T \omega \mathbf{z} = 0$$

$$\omega_{11} = \omega_{22}$$

$$\omega_{12} = 0$$

In matrix form: $A\omega_V = \mathbf{0}$,
where $\omega_V = (\omega_{11}, \omega_{12}, \omega_{13}, \omega_{22}, \omega_{23}, \omega_{33})^T$

$$A = \begin{pmatrix} u_1 v_1 & u_1 v_2 + u_2 v_1 & u_1 v_3 + u_3 v_1 & u_2 v_2 & u_2 v_3 + u_3 v_2 & u_3 v_3 \\ u_1 z_1 & u_1 z_2 + u_2 z_1 & u_1 z_3 + u_3 z_1 & u_2 z_2 & u_2 z_3 + u_3 z_2 & u_3 z_3 \\ v_1 z_1 & v_1 z_2 + v_2 z_1 & v_1 z_3 + v_3 z_1 & v_2 z_2 & v_2 z_3 + v_3 z_2 & v_3 z_3 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 \end{pmatrix}$$

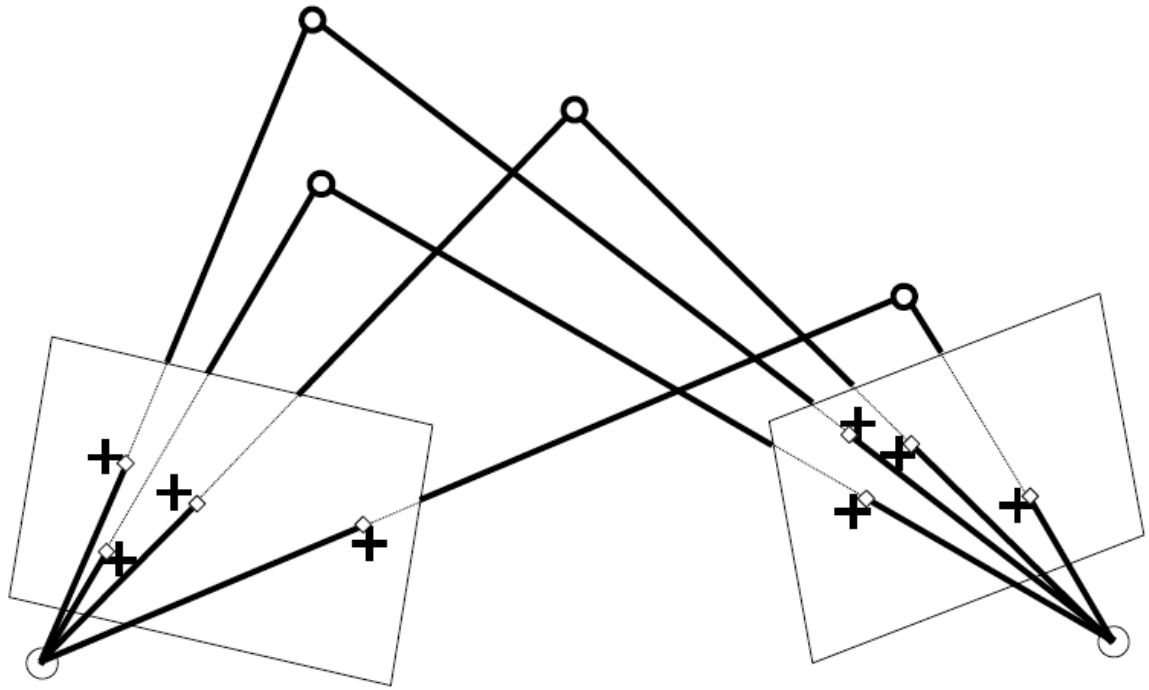
The solution ω_V is the null vector of A.

Outline

- Auto-calibration
- Bundle adjustment

Bundle adjustment

If the image measurements are noisy, then the equations $x_j^i = P^i \mathbf{X}_j$ will not be satisified exactly.



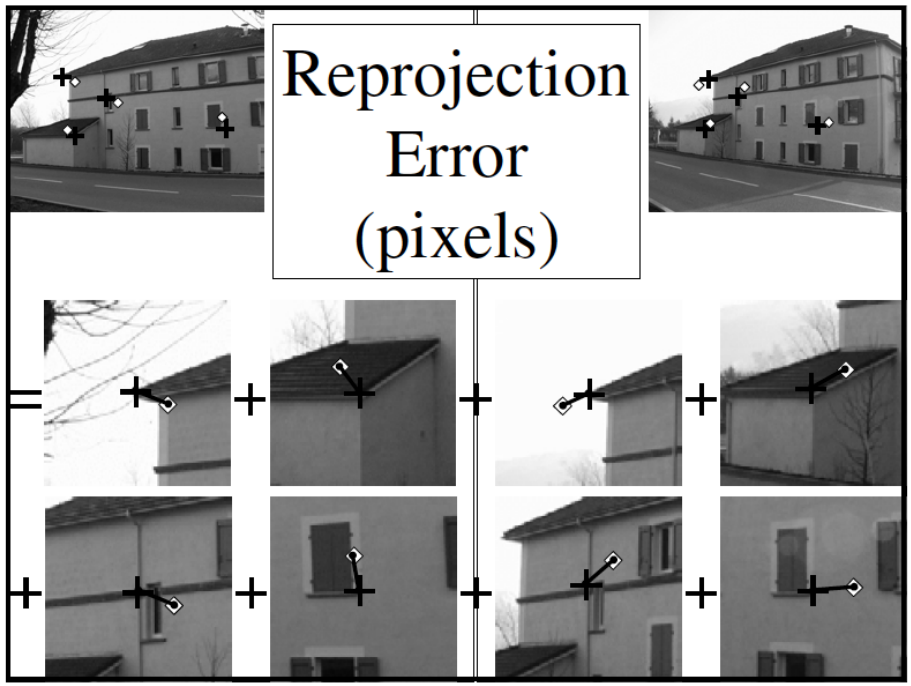
We then seek for matrices $\hat{P}^i$ and 3D points $\hat{\mathbf{X}}_j$ such that $\hat{\mathbf{x}}_j^i = \hat{P}^i \hat{\mathbf{X}}_j$ and $\hat{\mathbf{x}}_j^i$ as much close as possible to $\mathbf{x}_j^i$.

Image source: A. Bartoli

Bundle adjustment

Strategy: Minimize the reprojection error

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



In the case where the noise has Gaussian distribution, by minimizing this function we are looking for the Maximum Likelihood estimator.

Image source: A. Bartoli

Bundle adjustment

This estimation is known as bundle adjustment since it involves adjusting the bundle of rays between each camera and the set of 3D points.
It is generally used as a final step for any reconstruction algorithm.

PROS:

- It is tolerant to missing data
- It is robust to noise (provides the ML estimate)
- It may be extended to include priors and constraints on camera parameters or point positions.

CONS:

- Non-convex problem, it requires a good initialization
- It can become a large minimization problem because of the number of parameters involved.

Bundle adjustment

Usually the Levenberg-Marquardt method is used to minimize the problem.

There are many variants on how to proceed to reduce the complexity of the probem:

- Do not include all the views or the points, and fill these in later by resectioning or triangulation respectively.
- Partition the data into sets, bundle adjust each set separately and then merge.
- Alternate minimization of P's and $\mathbf{X}$'s.
- Sparse methods.