Reconnaissance d'objets et vision artificielle

http://www.di.ens.fr/willow/teaching/recvis09

Lecture 3

A refresher on camera geometry
Image alignment and 3D alignment

Check it out!

Cours de "Computational photography"
de Frédo Durand
Le jeudi de 9h30 a 12h30 Salle Info 2

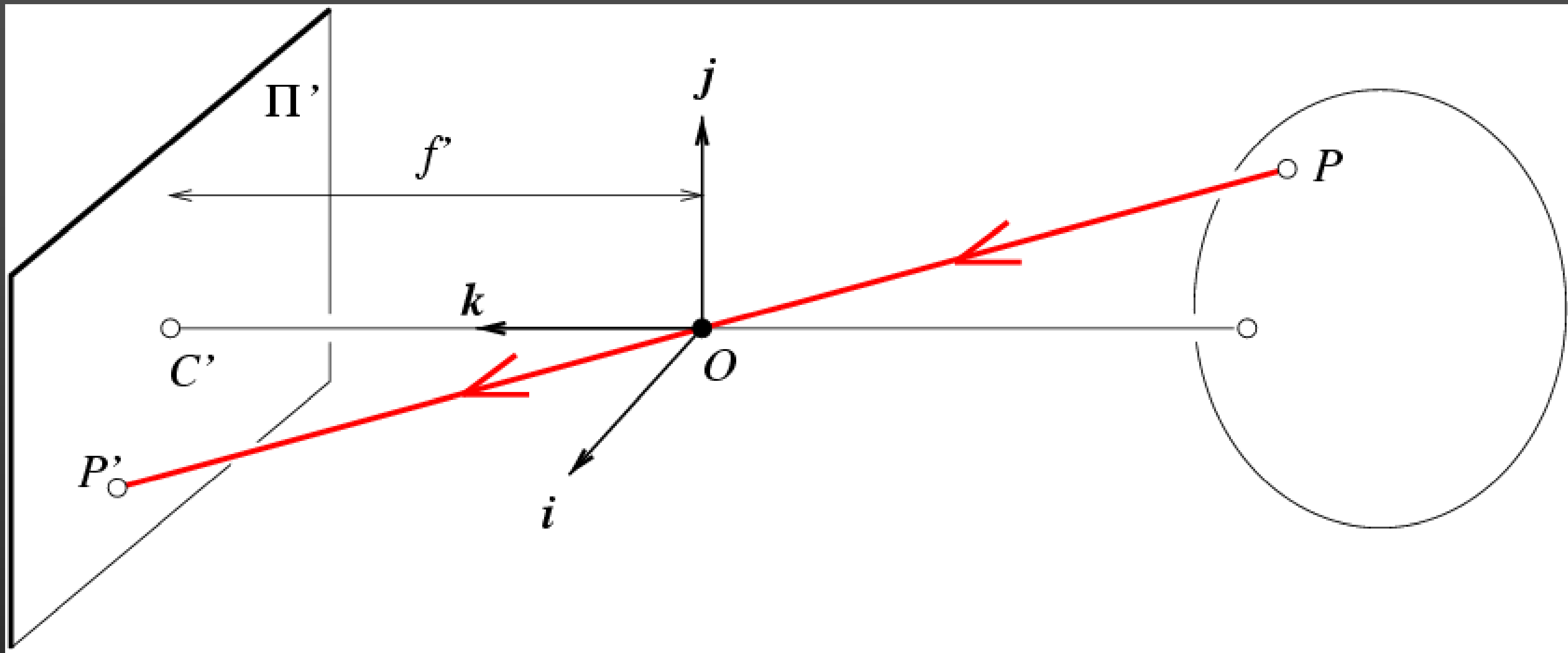
http://people.csail.mit.edu/fredo/Classes/Comp_Photo_ENS/

N'oubliez pas!

Premier exercice de programmation du le 27 octobre

http://www.di.ens.fr/willow/teaching/recvis09/assignment1/

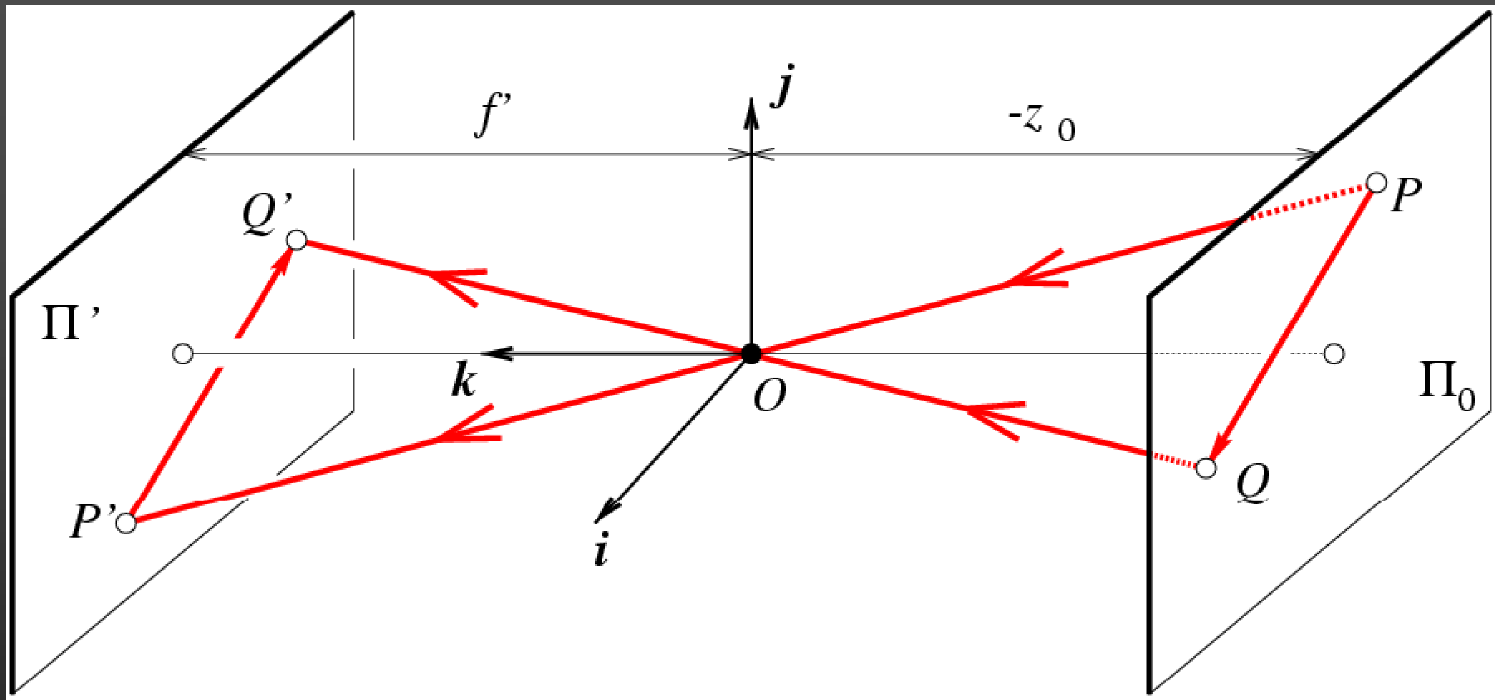
Pinhole perspective equation



$$\begin{cases} x' = f'\dfrac{x}{z} \\[2ex] y' = f'\dfrac{y}{z} \end{cases}$$

NOTE: z is always negative..

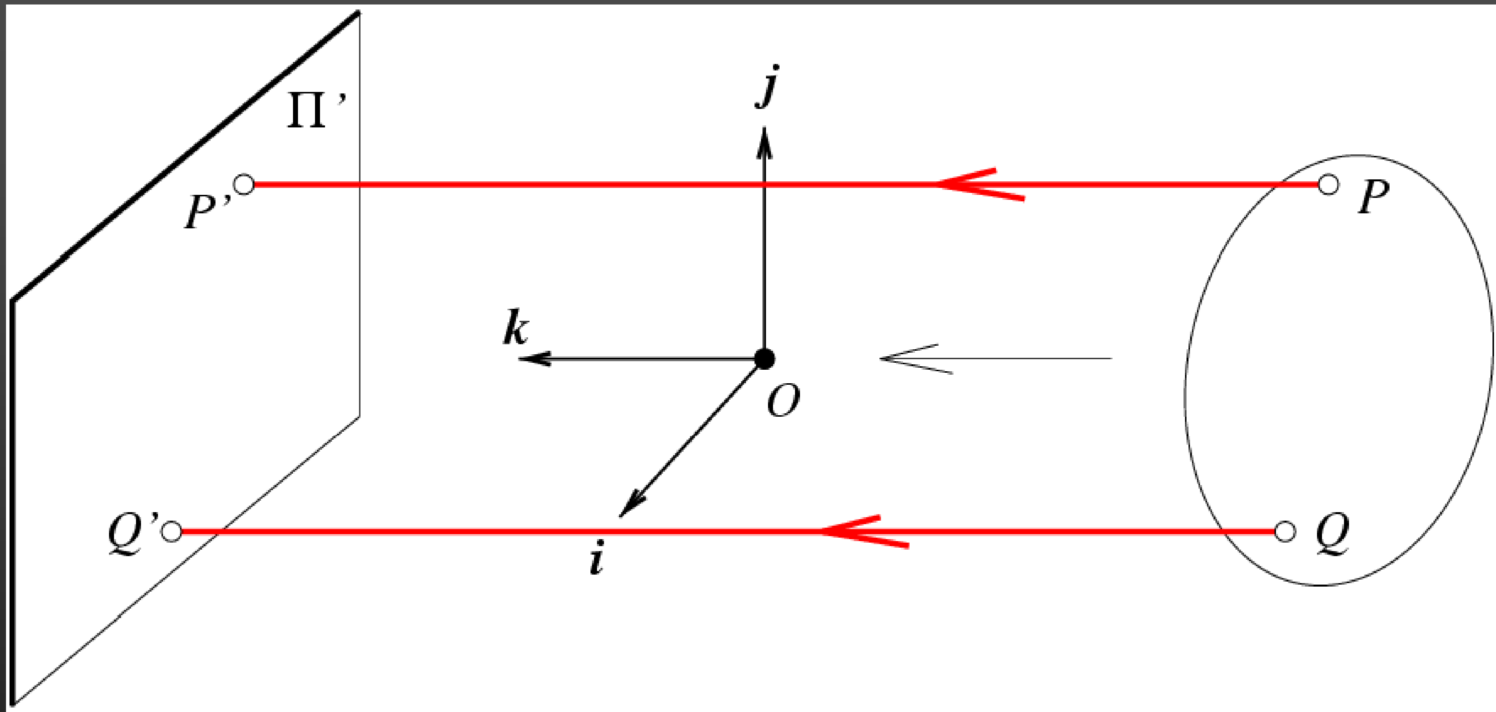
Affine models: Weak perspective projection



$$\boxed{\begin{cases} x' = -mx \\ y' = -my \end{cases}} \quad \text{where} \quad m = -\frac{f'}{z_0}$$ is the magnification.

When the scene relief is small compared its distance from the Camera, *m* can be taken constant: weak perspective projection.

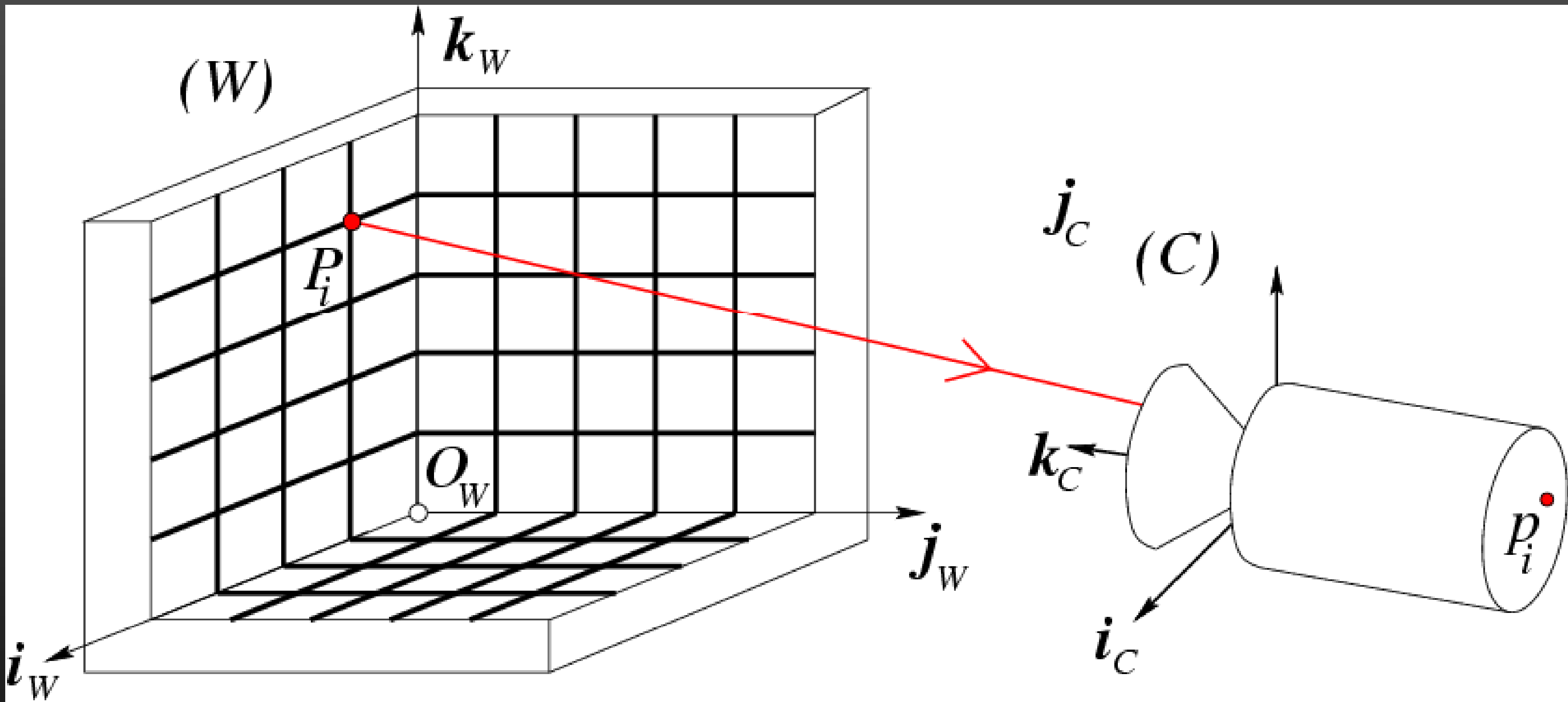
Affine models: Orthographic projection



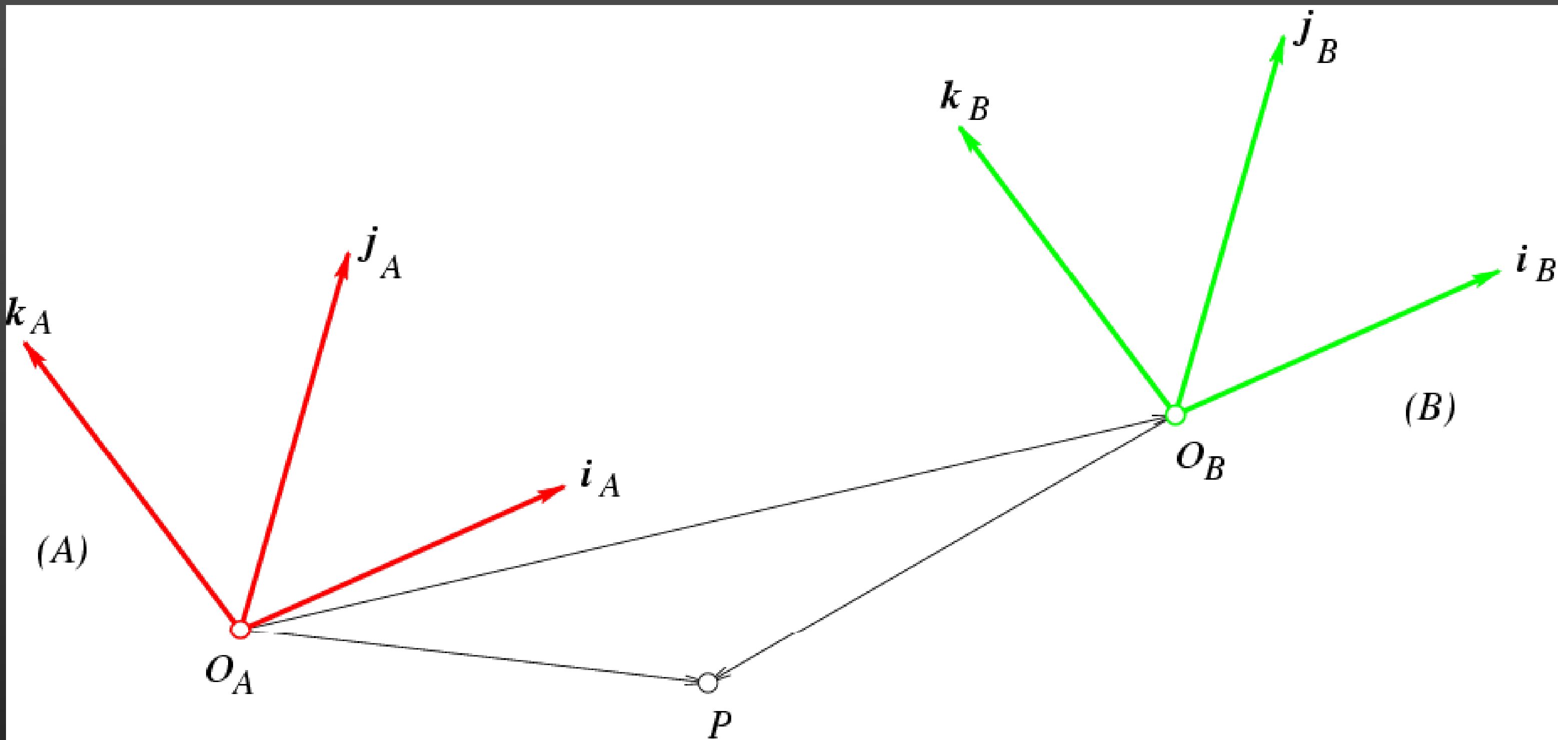
$$\begin{cases} x' = x \\ y' = y \end{cases}$$

When the camera is at a (roughly constant) distance from the scene, take m=1.

Analytical camera geometry

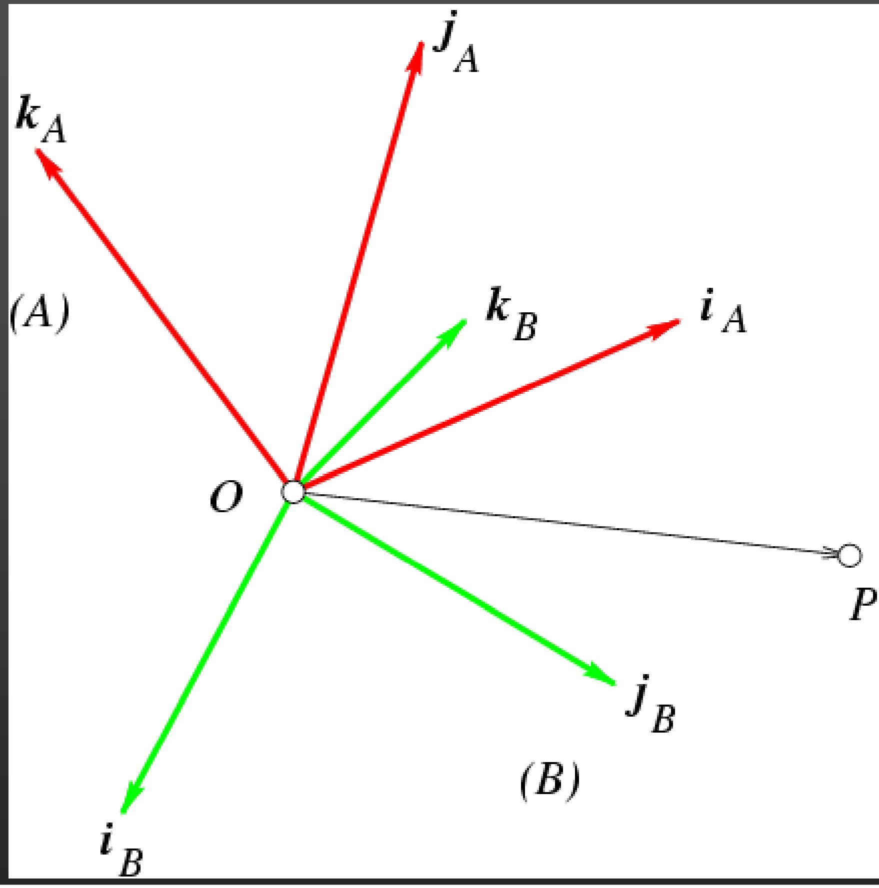


Coordinate Changes: Pure Translations



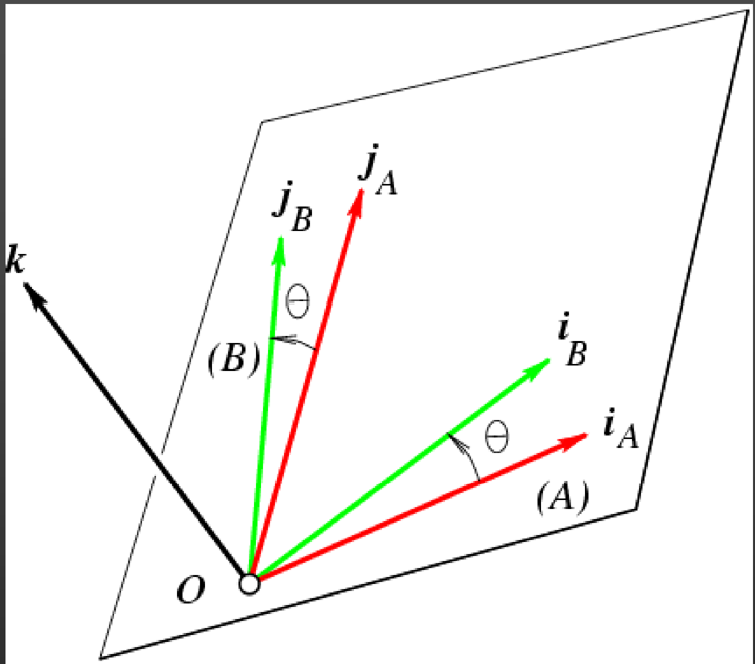
$$\overrightarrow{O_BP} = \overrightarrow{O_BO_A} + \overrightarrow{O_AP} \Leftrightarrow {}^{B}P = {}^{A}P + {}^{B}O_A$$

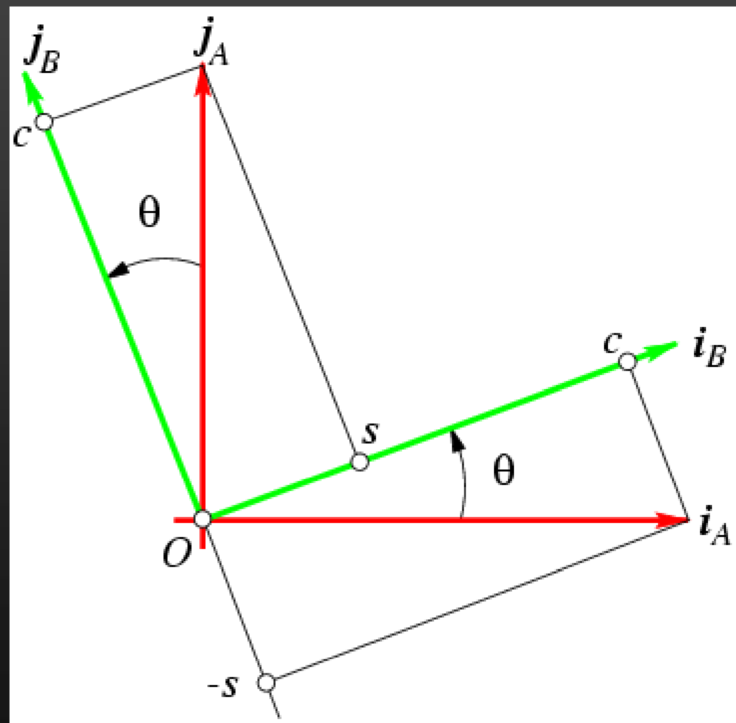
Coordinate Changes: Pure Rotations



$$
{}^{B}_{A}R = \begin{bmatrix} \mathbf{i}_A . \mathbf{i}_B & \mathbf{j}_A . \mathbf{i}_B & \mathbf{k}_A . \mathbf{i}_B \\ \mathbf{i}_A . \mathbf{j}_B & \mathbf{j}_A . \mathbf{j}_B & \mathbf{k}_A . \mathbf{j}_B \\ \mathbf{i}_A . \mathbf{k}_B & \mathbf{j}_A . \mathbf{k}_B & \mathbf{k}_A . \mathbf{k}_B \end{bmatrix} = \begin{bmatrix} {}^{A}\mathbf{i}_B^T \\ {}^{A}\mathbf{j}_B^T \\ {}^{A}\mathbf{k}_B^T \end{bmatrix} = \begin{bmatrix} {}^{B}\mathbf{i}_A & {}^{B}\mathbf{j}_A & {}^{B}\mathbf{k}_A \end{bmatrix}
$$

Coordinate Changes: Rotations about the z Axis





$$
{}^B_A R = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}
$$

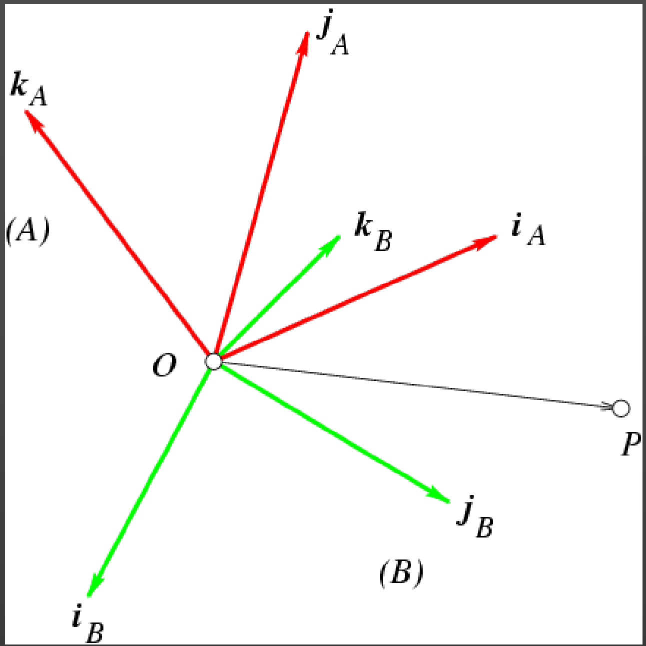
A rotation matrix is characterized by the following properties:

- Its inverse is equal to its transpose, and
- its determinant is equal to 1.

Or equivalently:

- Its rows (or columns) form a right-handed orthonormal coordinate system.

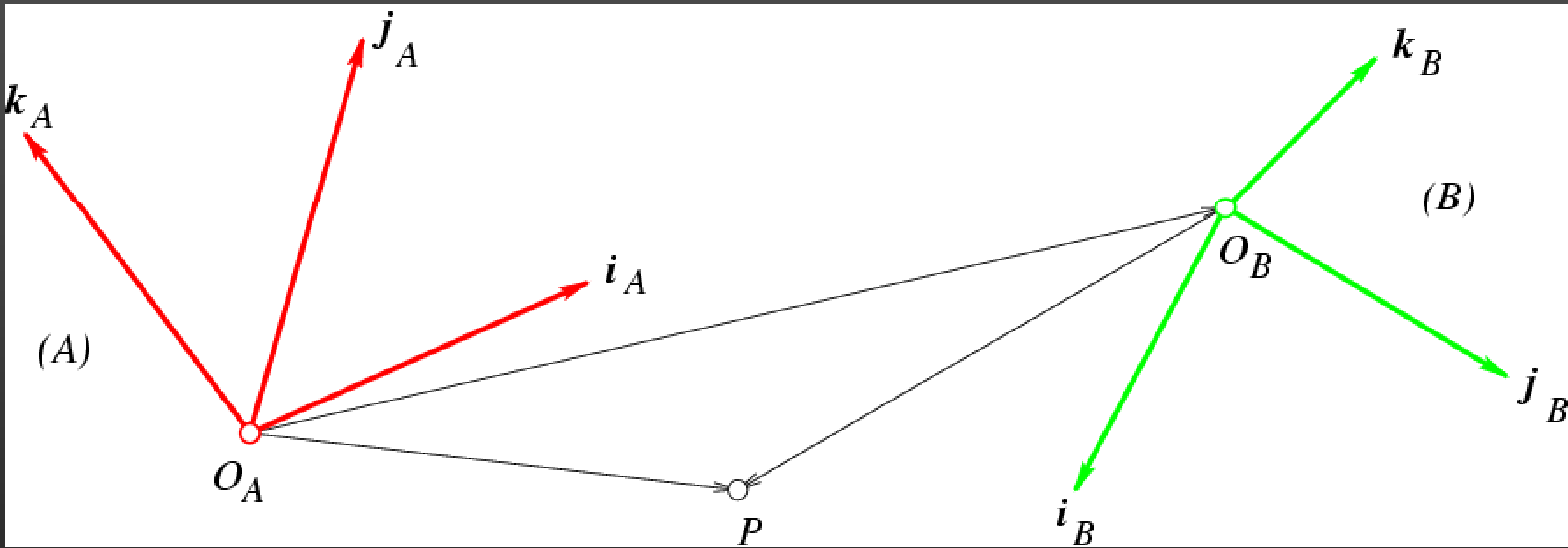
Coordinate changes: pure rotations



$$\overrightarrow{OP} = \begin{bmatrix} \mathbf{i}_A & \mathbf{j}_A & \mathbf{k}_A \end{bmatrix} \begin{bmatrix} {}^A x \\ {}^A y \\ {}^A z \end{bmatrix} = \begin{bmatrix} \mathbf{i}_B & \mathbf{j}_B & \mathbf{k}_B \end{bmatrix} \begin{bmatrix} {}^B x \\ {}^B y \\ {}^B z \end{bmatrix}$$

$$\Rightarrow \quad {}^B P = {}^B_A R \, {}^A P$$

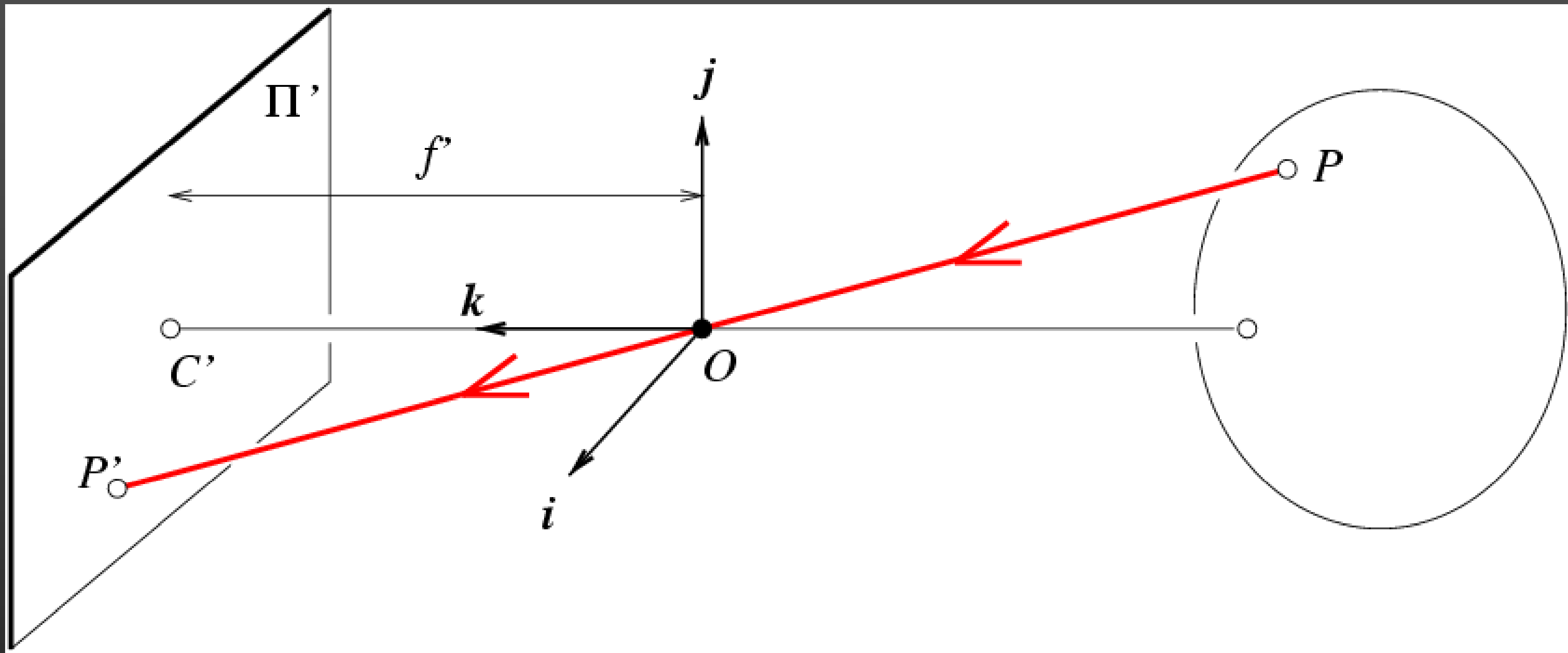
Coordinate Changes: Rigid Transformations



$$\begin{bmatrix} {}^BP \\ 1 \end{bmatrix} = \begin{bmatrix} {}^B_AR & {}^BO_A \\ \mathbf{0}^T & 1 \end{bmatrix}\begin{bmatrix} {}^AP \\ 1 \end{bmatrix} = \begin{bmatrix} {}^B_AR\,{}^AP + {}^BO_A \\ 1 \end{bmatrix} = {}^B_AT\begin{bmatrix} {}^AP \\ 1 \end{bmatrix}$$

$${}^BP = {}^B_AR\,{}^AP + {}^BO_A$$

Pinhole perspective equation



$$\begin{cases} x' = f'\dfrac{x}{z} \\ y' = f'\dfrac{y}{z} \end{cases}$$

NOTE: z is always negative..

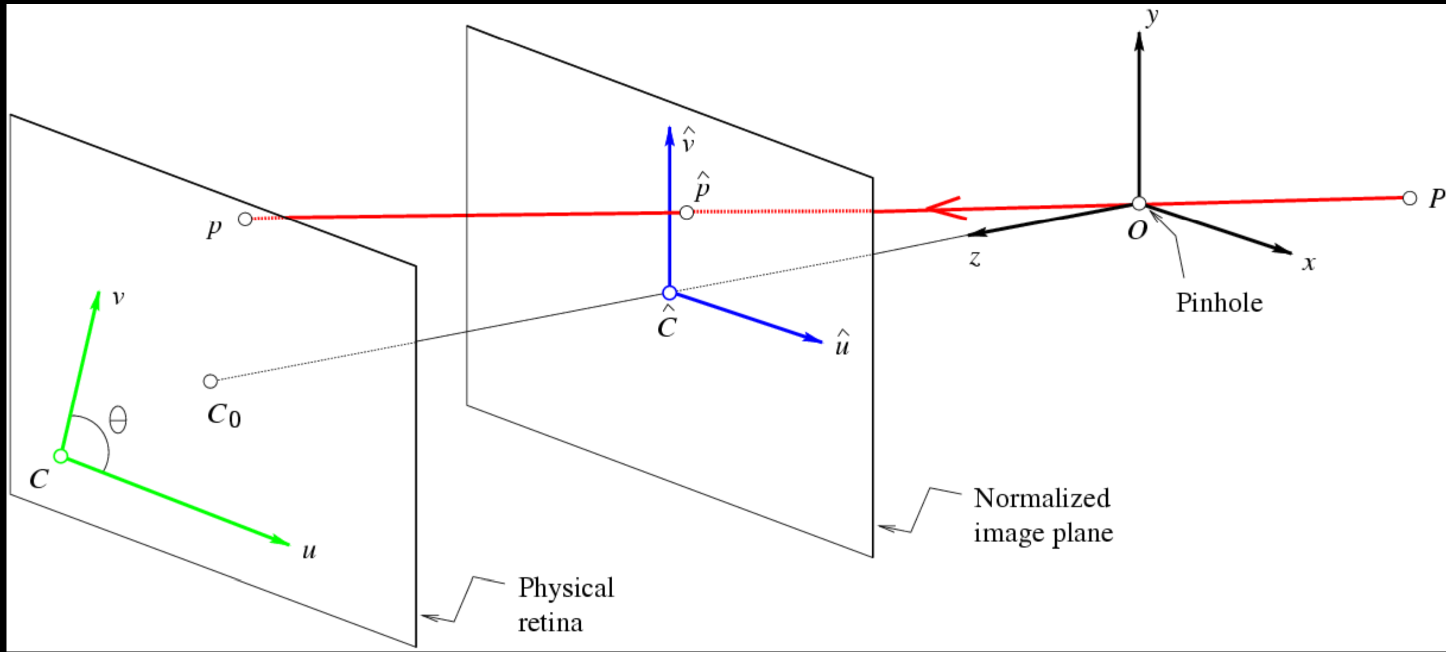
The intrinsic parameters of a camera

Units:

k, l : pixel/m

f : m

α, β : pixel



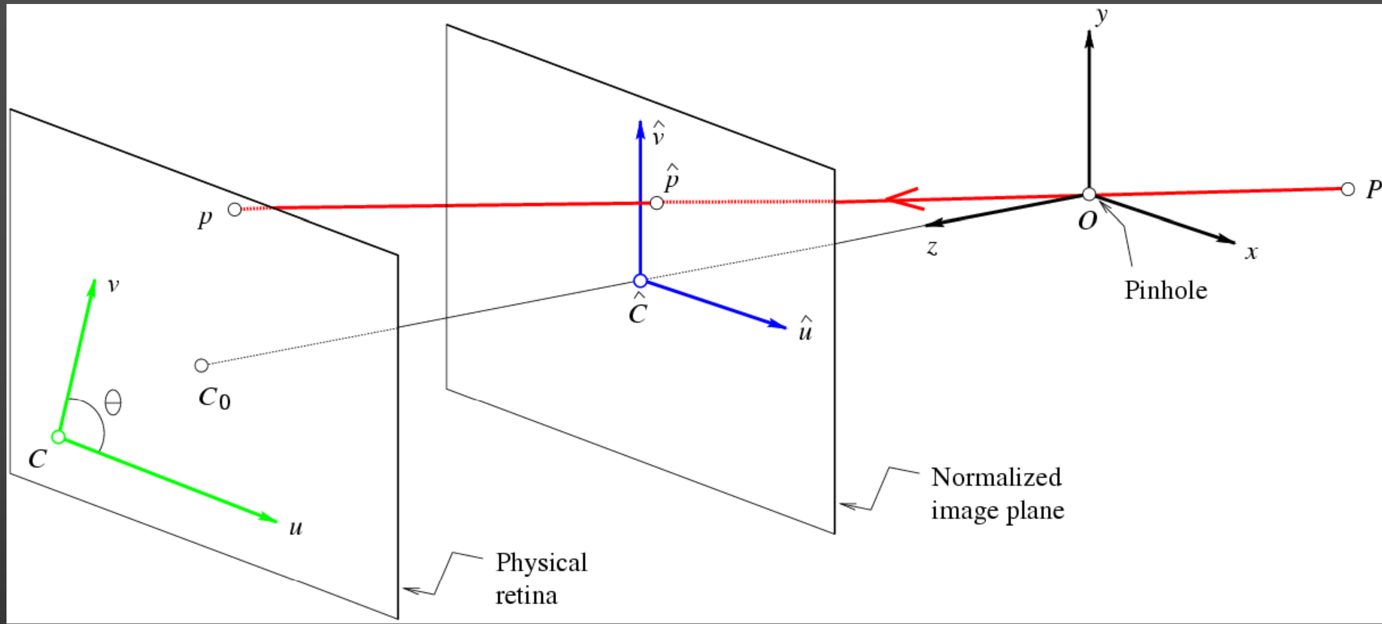
$$\begin{cases} \hat{u} = \dfrac{x}{z} \\ \hat{v} = \dfrac{y}{z} \end{cases} \iff \hat{\boldsymbol{p}} = \frac{1}{z} \begin{pmatrix} \mathrm{Id} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{P} \\ 1 \end{pmatrix}$$

Normalized image coordinates

Physical image coordinates

$$\begin{cases} u = kf\dfrac{x}{z} \\ v = lf\dfrac{y}{z} \end{cases}$$

The intrinsic parameters of a camera



Calibration matrix

$$\boldsymbol{p} = \mathcal{K}\hat{\boldsymbol{p}}, \quad \text{where} \quad \boldsymbol{p} = \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \quad \text{and} \quad \mathcal{K} \stackrel{\text{def}}{=} \begin{pmatrix} \alpha & -\alpha\cot\theta & u_0 \\ 0 & \dfrac{\beta}{\sin\theta} & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

The perspective projection equation

$$\boldsymbol{p} = \frac{1}{z}\mathcal{M}\boldsymbol{P}, \quad \text{where} \quad \mathcal{M} \stackrel{\text{def}}{=} (\,\mathcal{K} \quad \mathbf{0}\,)$$

The extrinsic parameters of a camera

- When the camera frame (C) is different from the world frame (W),

$$\begin{pmatrix} {}^C P \\ 1 \end{pmatrix} = \begin{pmatrix} {}^C_W\mathcal{R} & {}^C O_W \\ \mathbf{0}^T & 1 \end{pmatrix} \begin{pmatrix} {}^W P \\ 1 \end{pmatrix}.$$

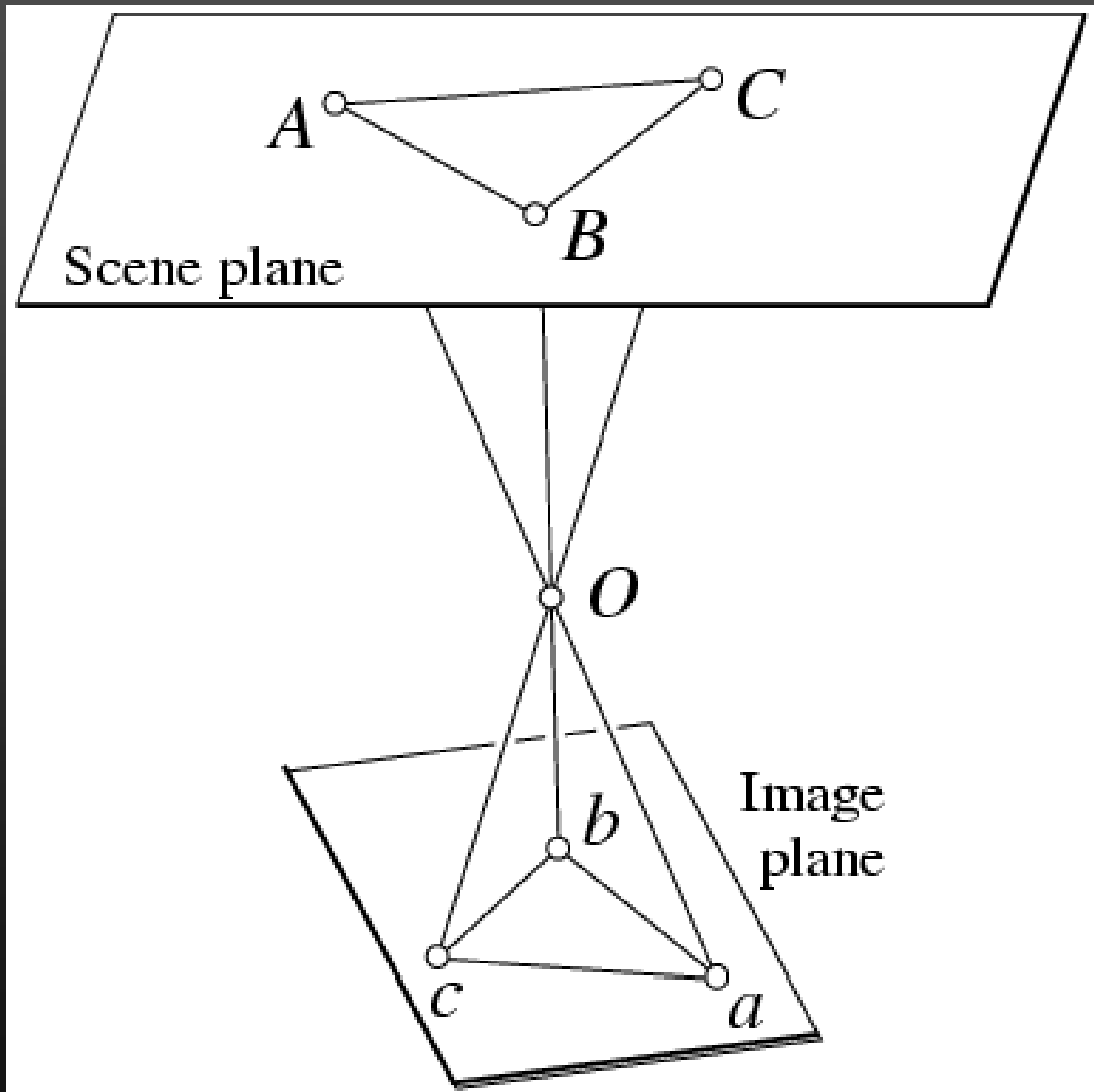
- Thus,

$$\boxed{\boldsymbol{p} = \frac{1}{z}\mathcal{M}\boldsymbol{P},} \quad \text{where} \quad \begin{cases} \mathcal{M} = \mathcal{K}\,(\,\mathcal{R} \quad \boldsymbol{t}\,)\,, \\ \mathcal{R} = {}^C_W\mathcal{R}, \\ \boldsymbol{t} = {}^C O_W, \\ \boldsymbol{P} = \begin{pmatrix} {}^W P \\ 1 \end{pmatrix}. \end{cases}$$

- Note: z is *not* independent of $\mathcal{M}$ and $\boldsymbol{P}$:

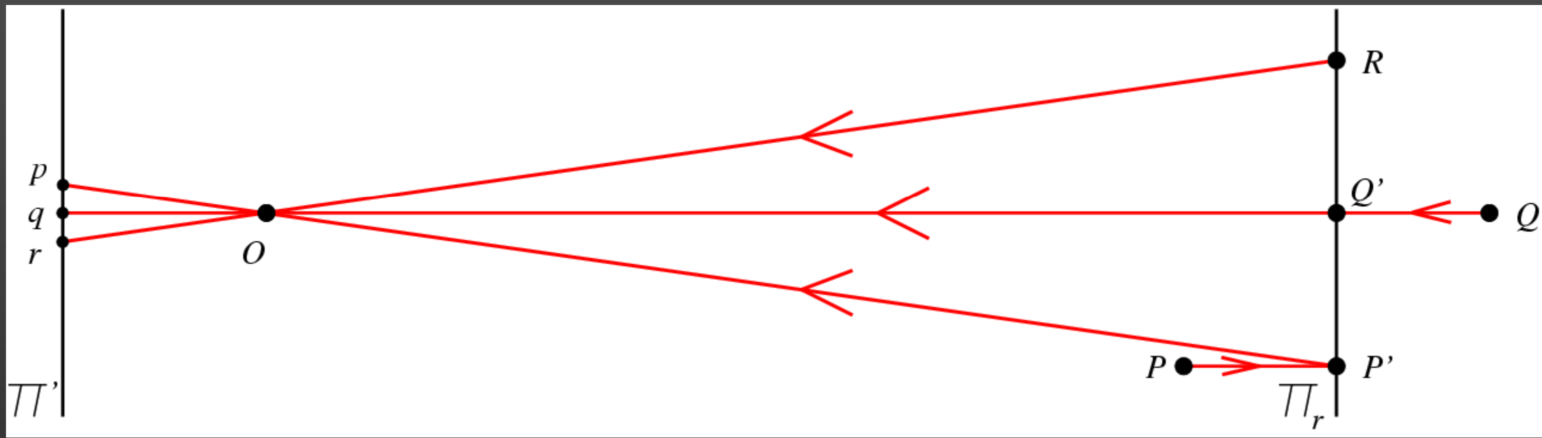
$$\mathcal{M} = \begin{pmatrix} \boldsymbol{m}_1^T \\ \boldsymbol{m}_2^T \\ \boldsymbol{m}_3^T \end{pmatrix} \Longrightarrow z = \boldsymbol{m}_3 \cdot \boldsymbol{P}, \quad \text{or} \quad \begin{cases} u = \dfrac{\boldsymbol{m}_1 \cdot \boldsymbol{P}}{\boldsymbol{m}_3 \cdot \boldsymbol{P}}, \\ v = \dfrac{\boldsymbol{m}_2 \cdot \boldsymbol{P}}{\boldsymbol{m}_3 \cdot \boldsymbol{P}}. \end{cases}$$

Perspective projections induce projective transformations between planes

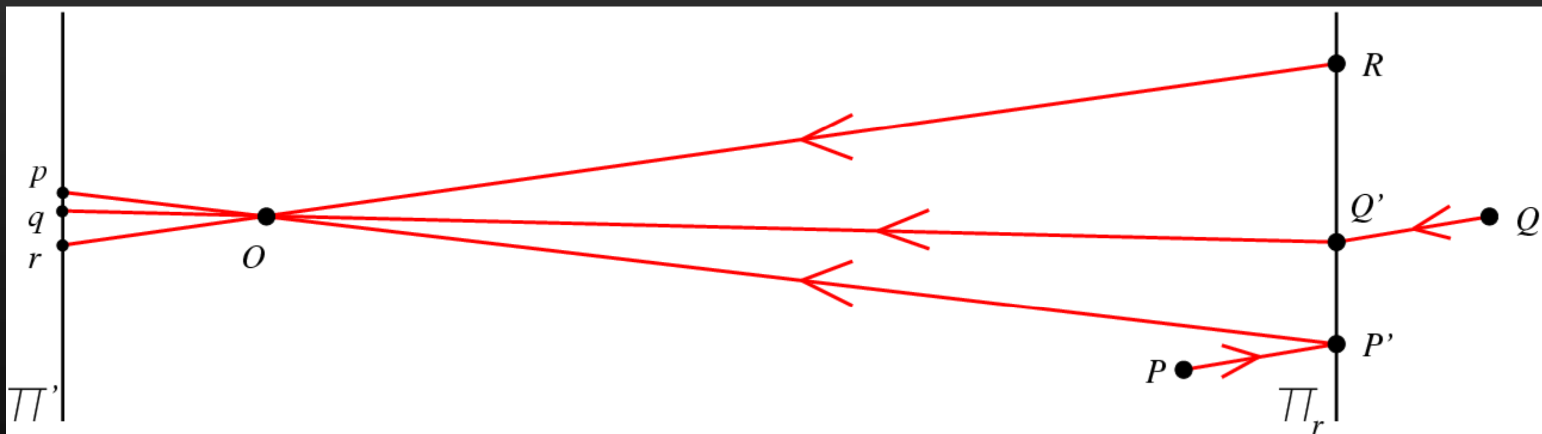


Affine cameras

Weak-perspective projection

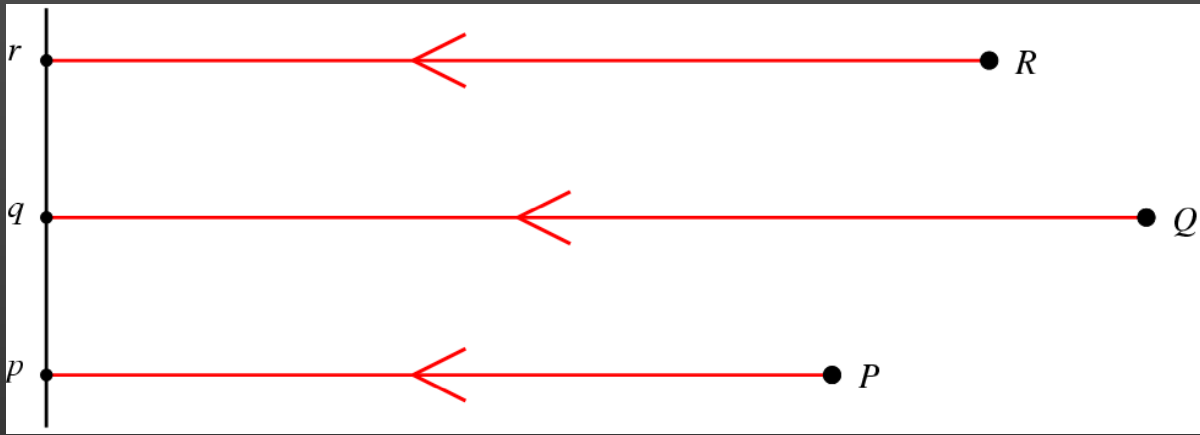


Paraperspective projection

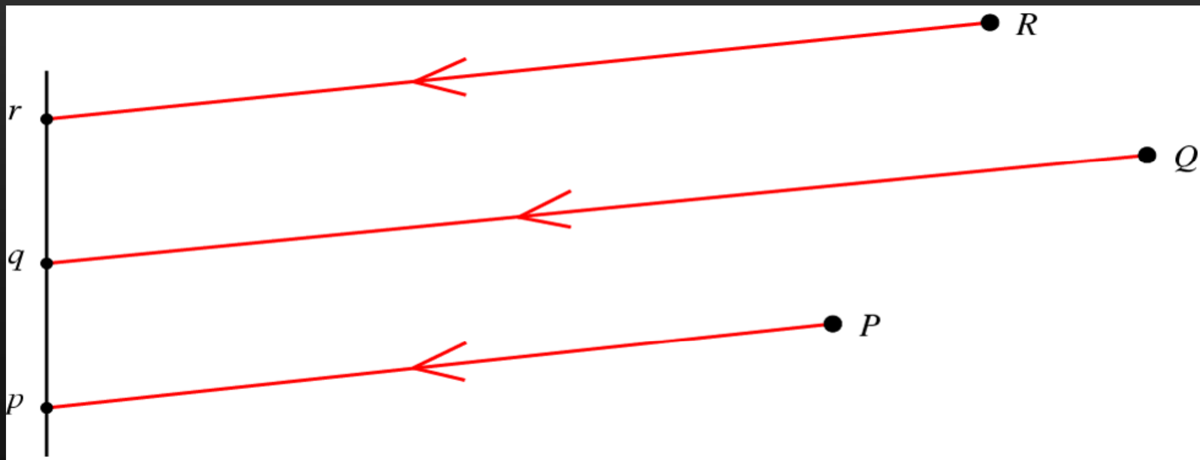


More affine cameras

Orthographic projection



Parallel projection



Weak-perspective projection model

$$\boldsymbol{p} = \frac{1}{z_r}\mathcal{M}\boldsymbol{P}$$

(p and P are in homogeneous coordinates)

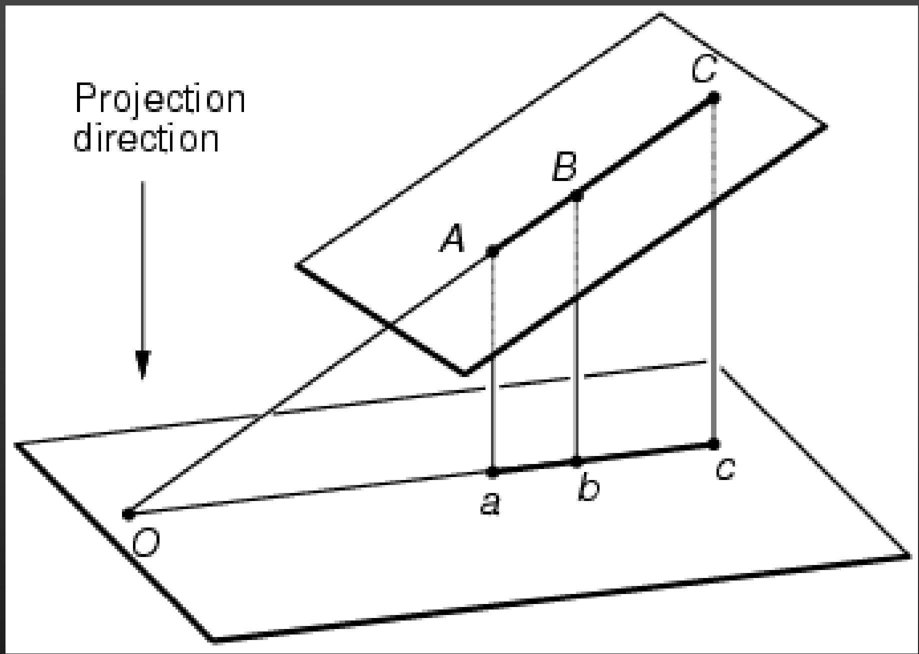
$$p = M\,P$$

(P is in homogeneous coordinates)

$$p = A\,P + b$$

(neither p nor P is in hom. coordinates)

Affine projections induce affine transformations from planes onto their images.



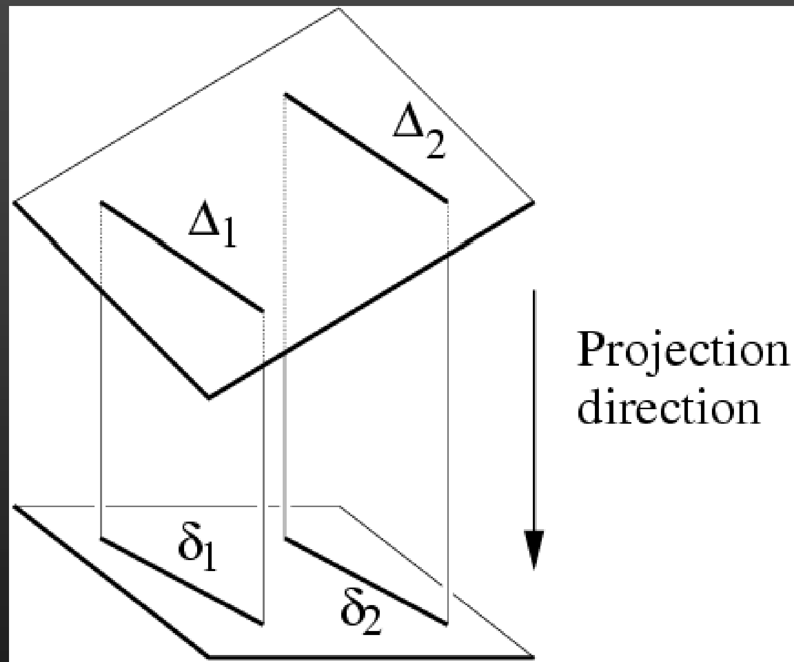


Image alignment task

- It helps to be able to compare *descriptors* of local patches surrounding interest points (cf last lecture).
- This is not strictly necessary. We will concentrate here on the geometry of the problem.

Dealing with outliers

The set of putative matches still contains a very high percentage of outliers

How do we fit a geometric transformation to a small subset of all possible matches?

Possible strategies:

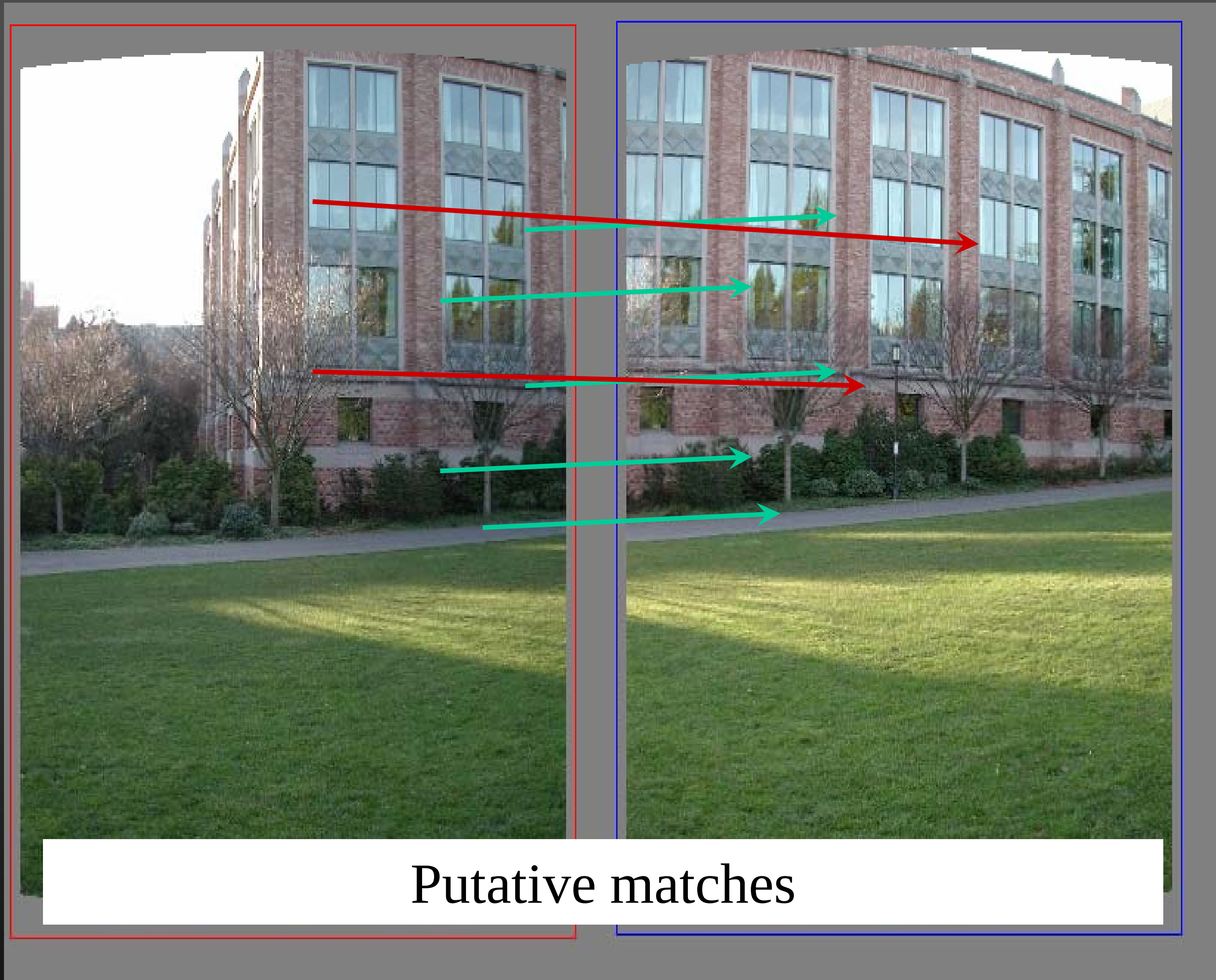
- RANSAC
- Incremental alignment
- Hough transform
- Hashing

Strategy 1: RANSAC

RANSAC loop (Fischler & Bolles, 1981):

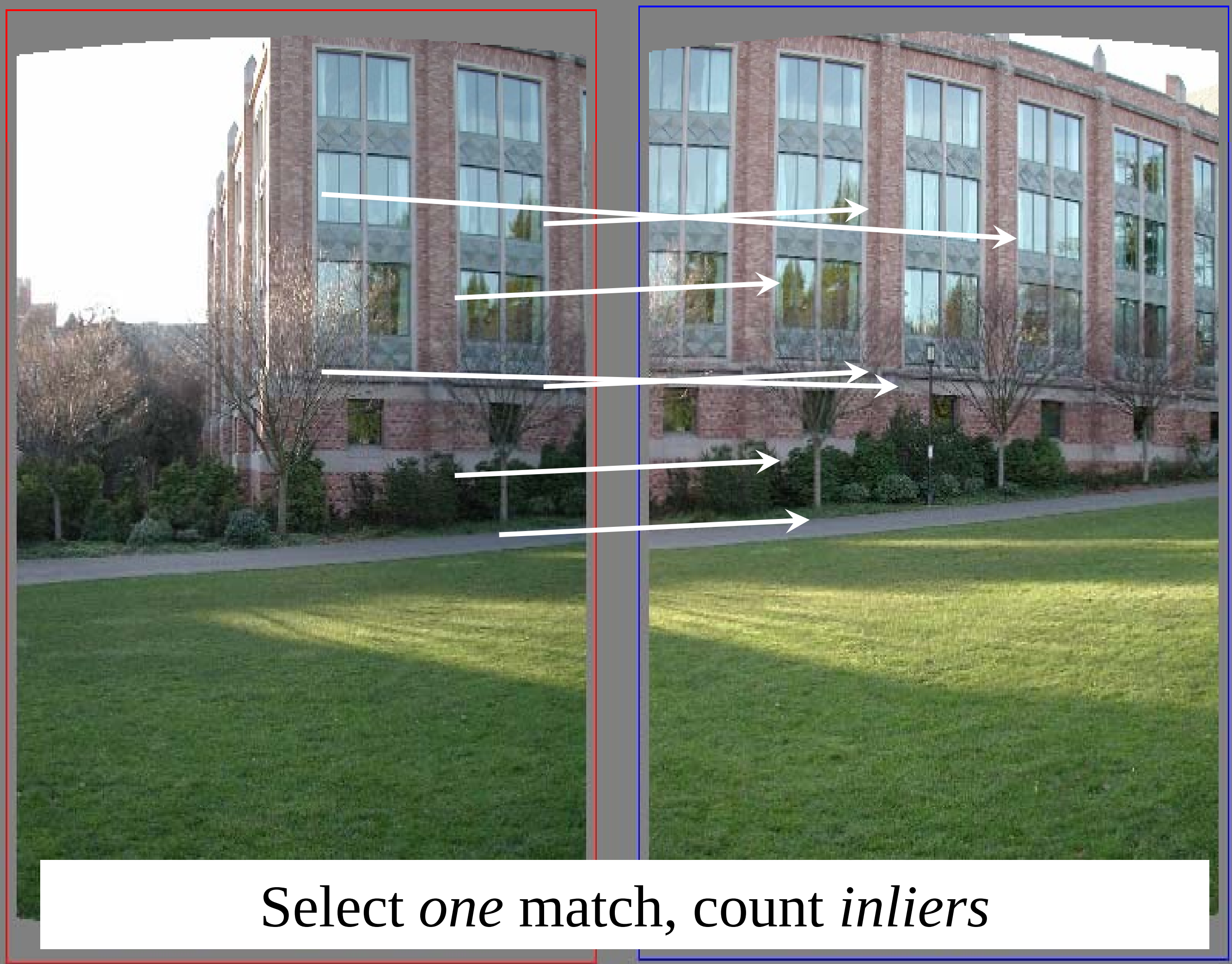
- Randomly select a *seed group* of matches
- Compute transformation from seed group
- Find *inliers* to this transformation
- If the number of inliers is sufficiently large, re-compute least-squares estimate of transformation on all of the inliers
- Keep the transformation with the largest number of inliers

RANSAC example: Translation



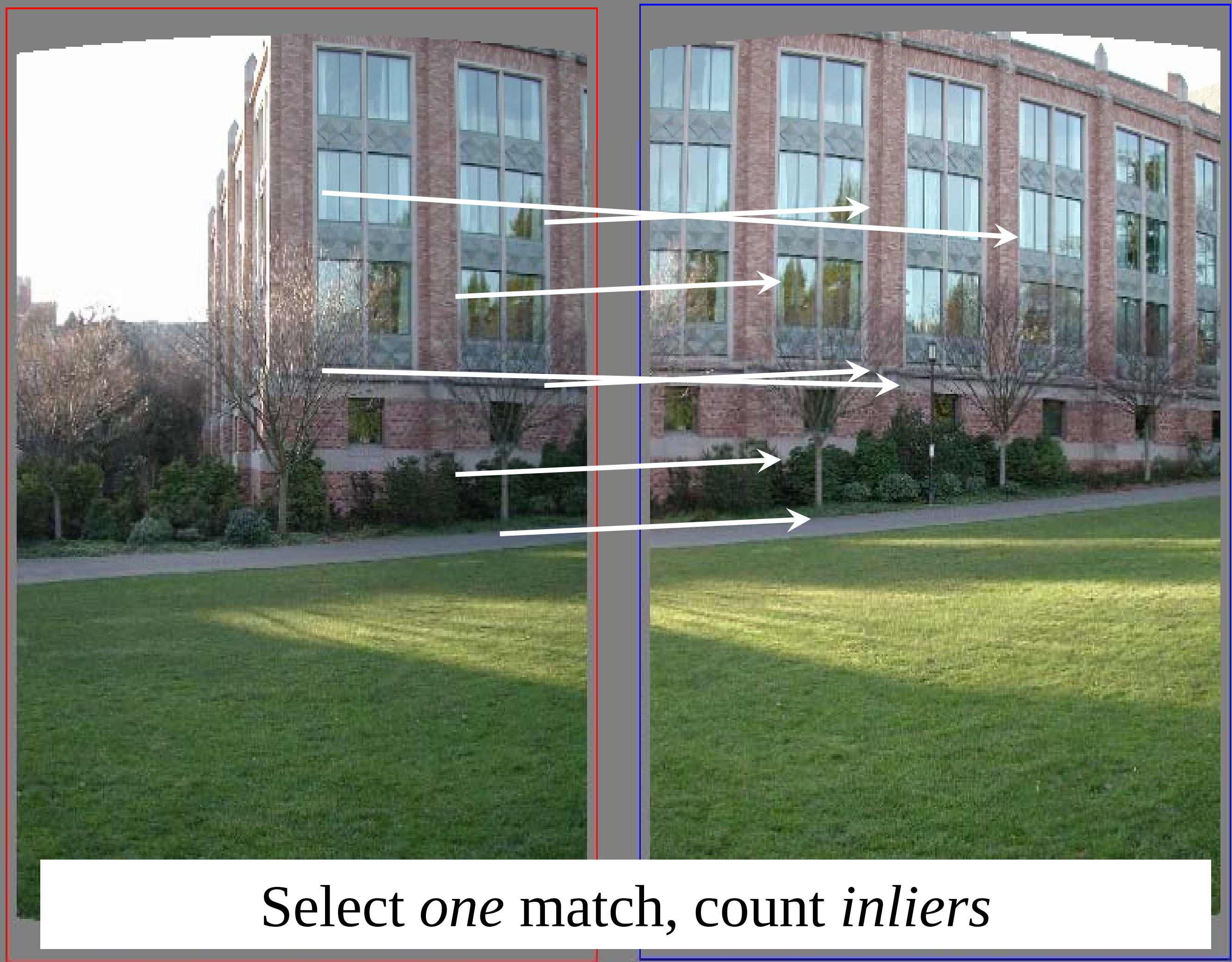
Putative matches

RANSAC example: Translation



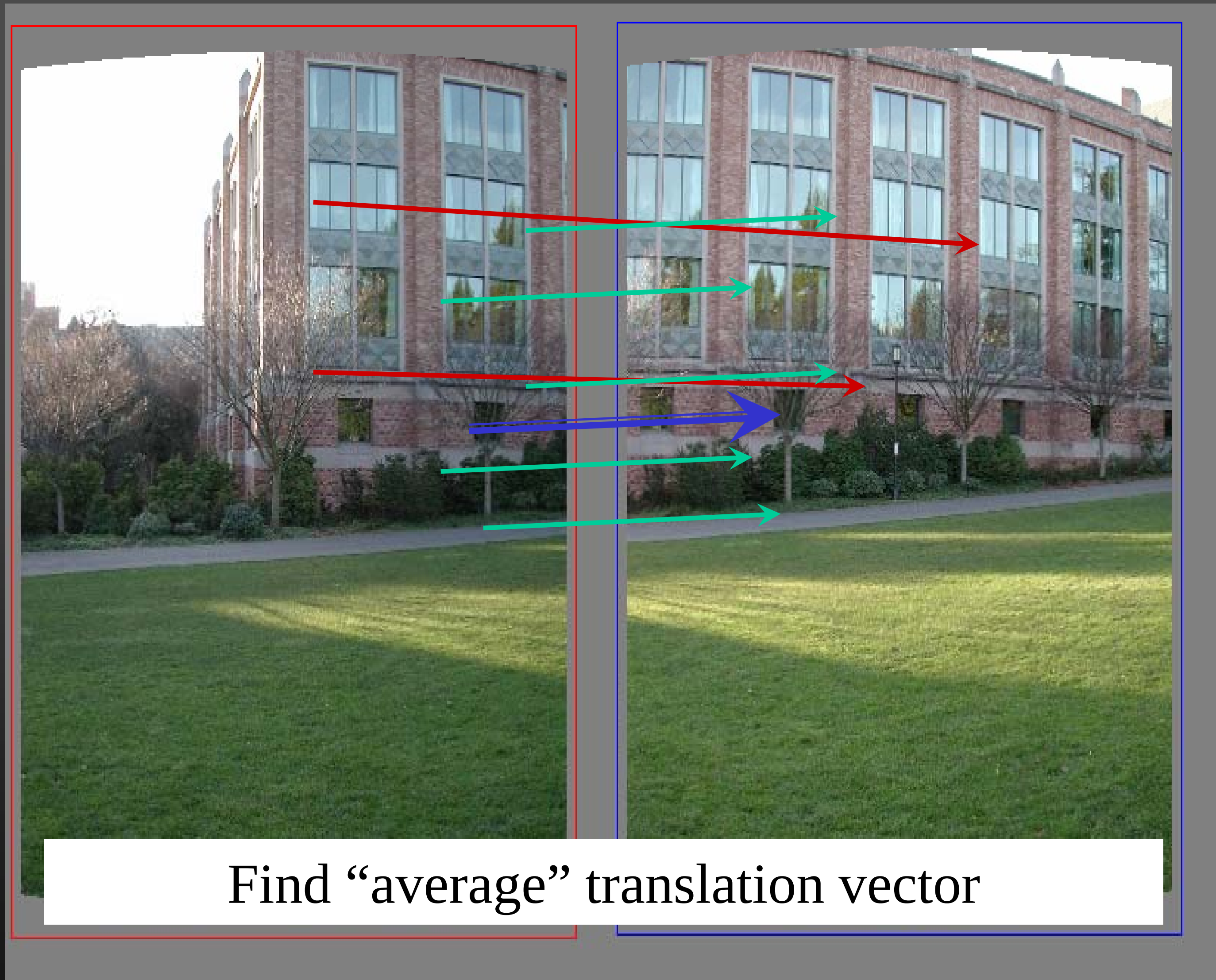
Select *one* match, count *inliers*

RANSAC example: Translation



Select *one* match, count *inliers*

RANSAC example: Translation



Find “average” translation vector

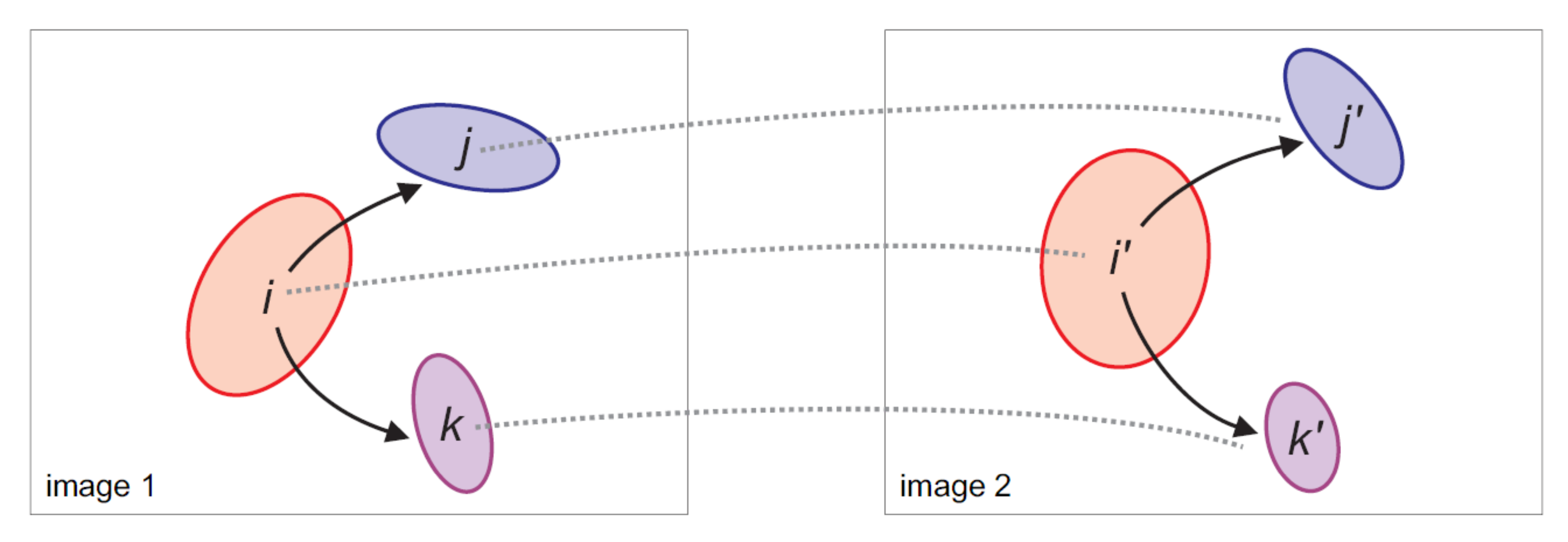
Strategy 2: Incremental alignment

Take advantage of strong locality constraints: only pick close-by matches to start with, and gradually add more matches in the same neighborhood

Approach introduced in [Ayache & Faugeras, 1982; Hebert & Faugeras, 1983; Gaston & Lozano-Perez, 1984]

Illustrated here with the method from S. Lazebnik, C. Schmid and J. Ponce, “Semi-local affine parts for object recognition”, BMVC 2004

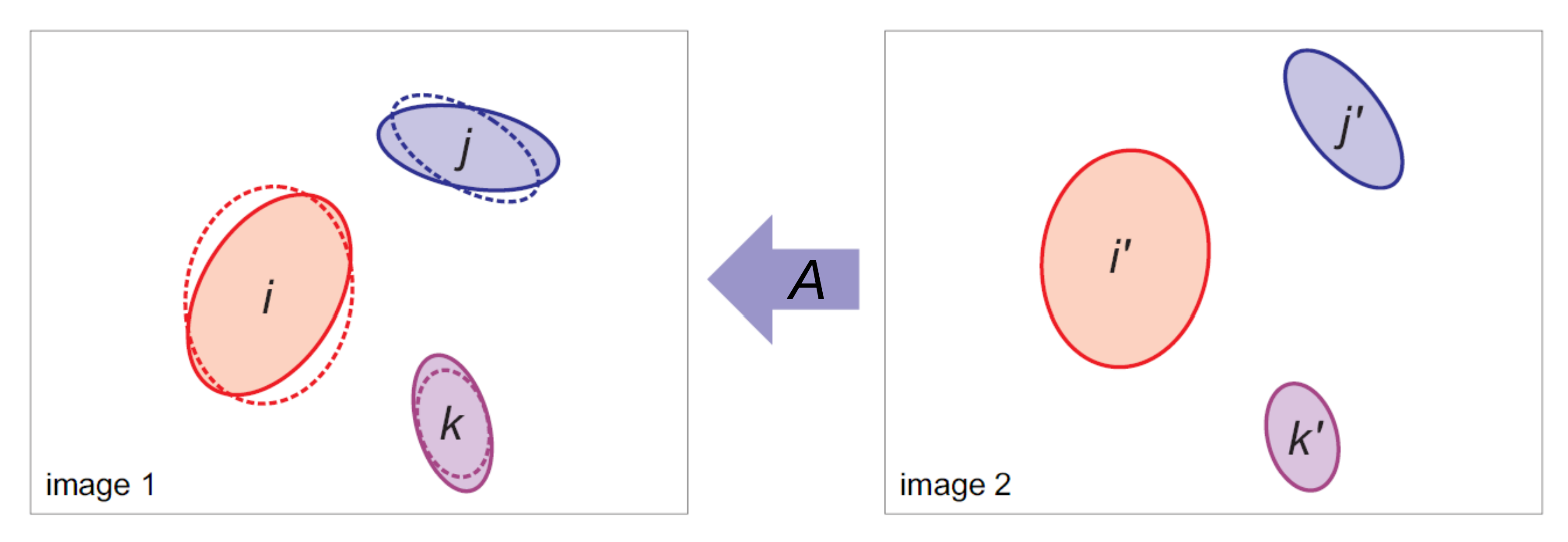
Incremental alignment: Details



Generating seed groups:

- Identify triples of neighboring features (*i*, *j*, *k*) in first image
- Find all triples (*i'*, *j'*, *k'*) in the second image such that *i'* (resp. *j'*, *k'*) is a putative match of *i* (resp. *j*, *k*), and *j'*, *k'* are neighbors of *i'*

Incremental alignment: Details

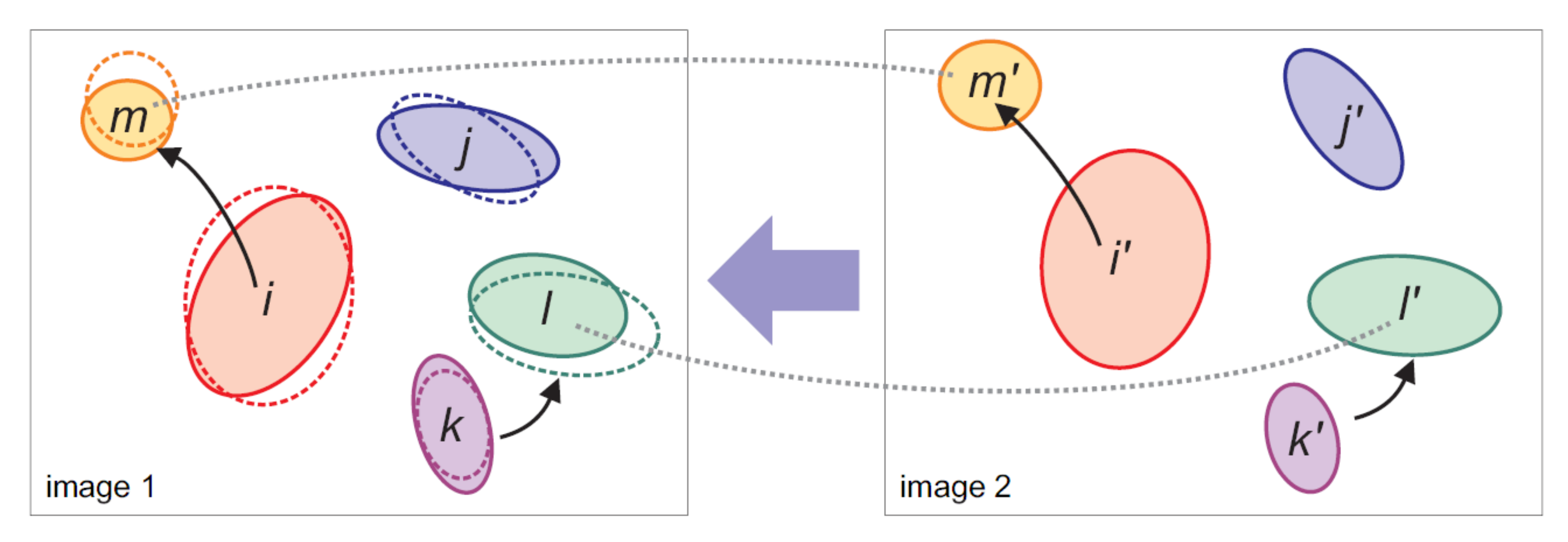


Beginning with each seed triple, repeat:

- Estimate the aligning transformation between corresponding features in current group of matches
- Grow the group by adding other consistent matches in the neighborhood

Until the transformation is no longer consistent
or no more matches can be found

Incremental alignment: Details

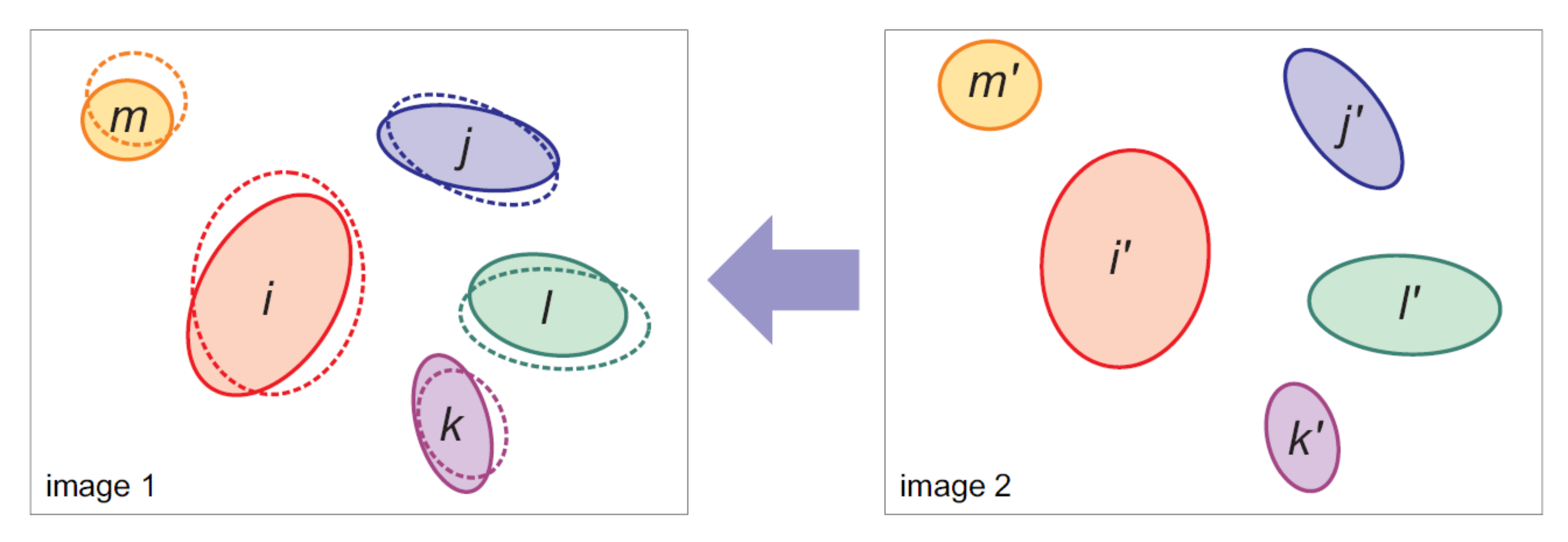


Beginning with each seed triple, repeat:

- Estimate the aligning transformation between corresponding features in current group of matches
- Grow the group by adding other consistent matches in the neighborhood

Until the transformation is no longer consistent
or no more matches can be found

Incremental alignment: Details

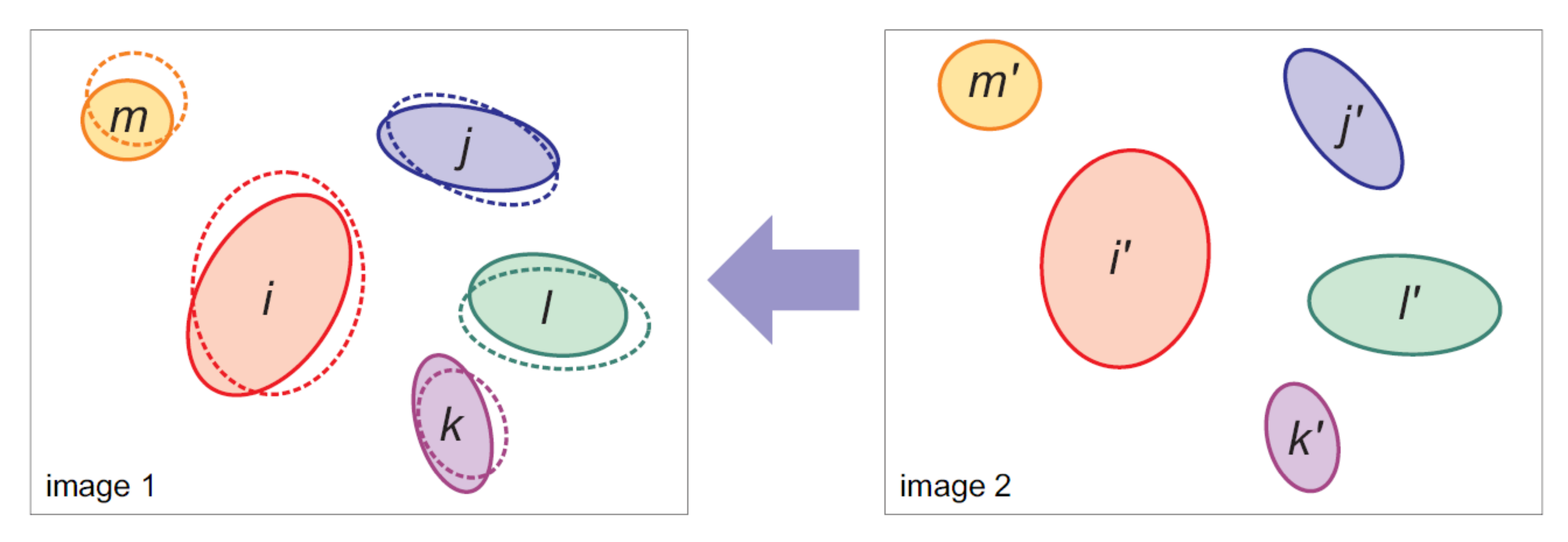


Beginning with each seed triple, repeat:

- Estimate the aligning transformation between corresponding features in current group of matches
- Grow the group by adding other consistent matches in the neighborhood

Until the transformation is no longer consistent or no more matches can be found

Incremental alignment: Details



Beginning with each seed triple, repeat:

- Estimate the aligning transformation between corresponding features in current group of matches
- Grow the group by adding other consistent matches in the neighborhood

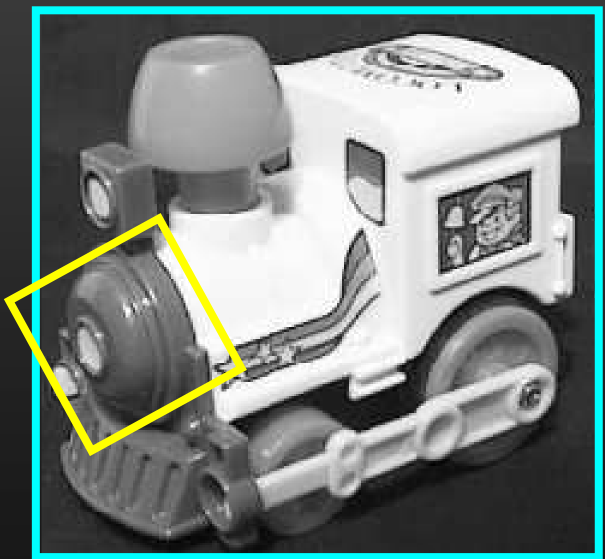
Until the transformation is no longer consistent
or no more matches can be found

Strategy 3: Hough transform

Suppose our features are scale- and rotation-covariant

- Then a single feature match provides an alignment hypothesis (translation, scale, orientation)

model



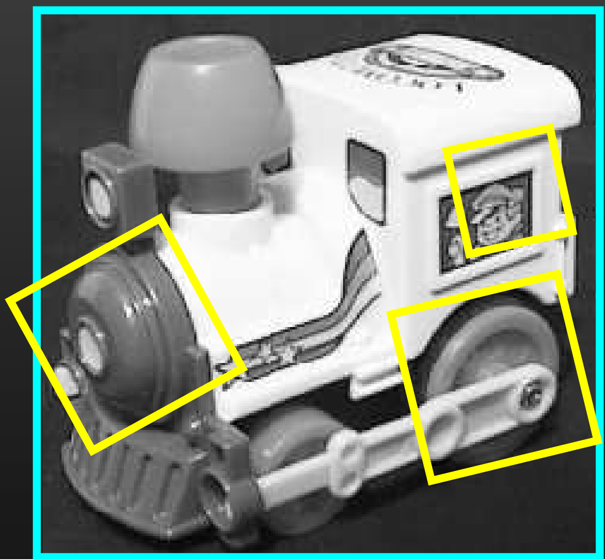
David G. Lowe. **“Distinctive image features from scale-invariant keypoints”,** *IJCV* 60 (2), pp. 91-110, 2004.

Strategy 3: Hough transform

Suppose our features are scale- and rotation-covariant

- Then a single feature match provides an alignment hypothesis (translation, scale, orientation)
- Of course, a hypothesis obtained from a single match is unreliable
- Solution: let each match vote for its hypothesis in a Hough space with very coarse bins

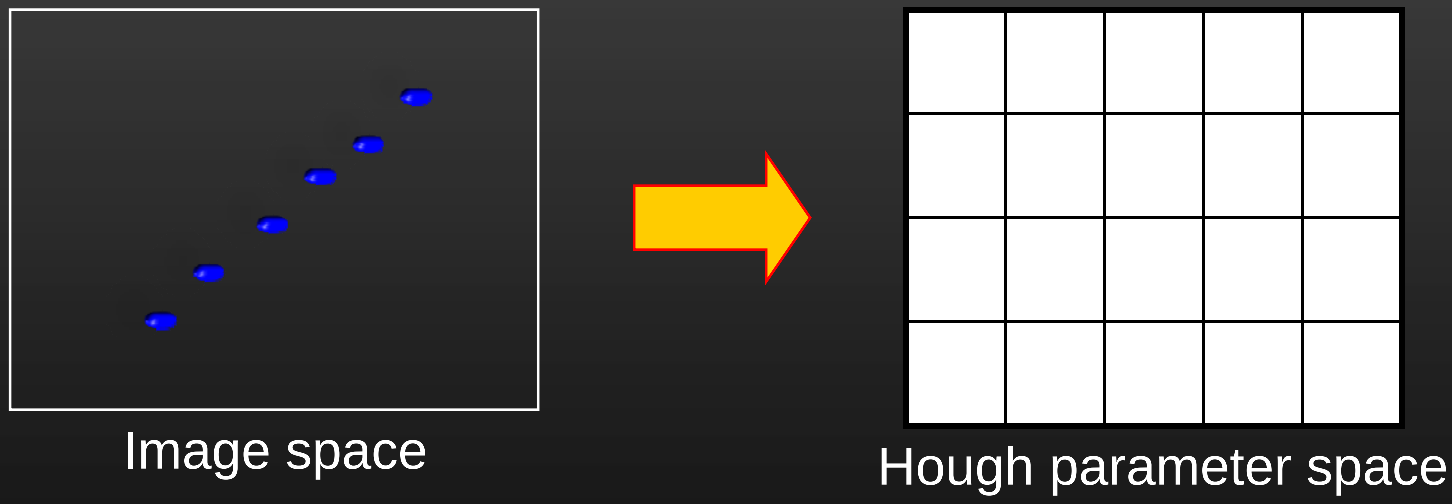
model



David G. Lowe. **“Distinctive image features from scale-invariant keypoints”,** *IJCV* 60 (2), pp. 91-110, 2004.

Hough transform

- An early type of voting scheme
- General outline:
  - Discretize parameter space into bins
  - For each feature point in the image, put a vote in every bin in the parameter space that could have generated this point
  - Find bins that have the most votes

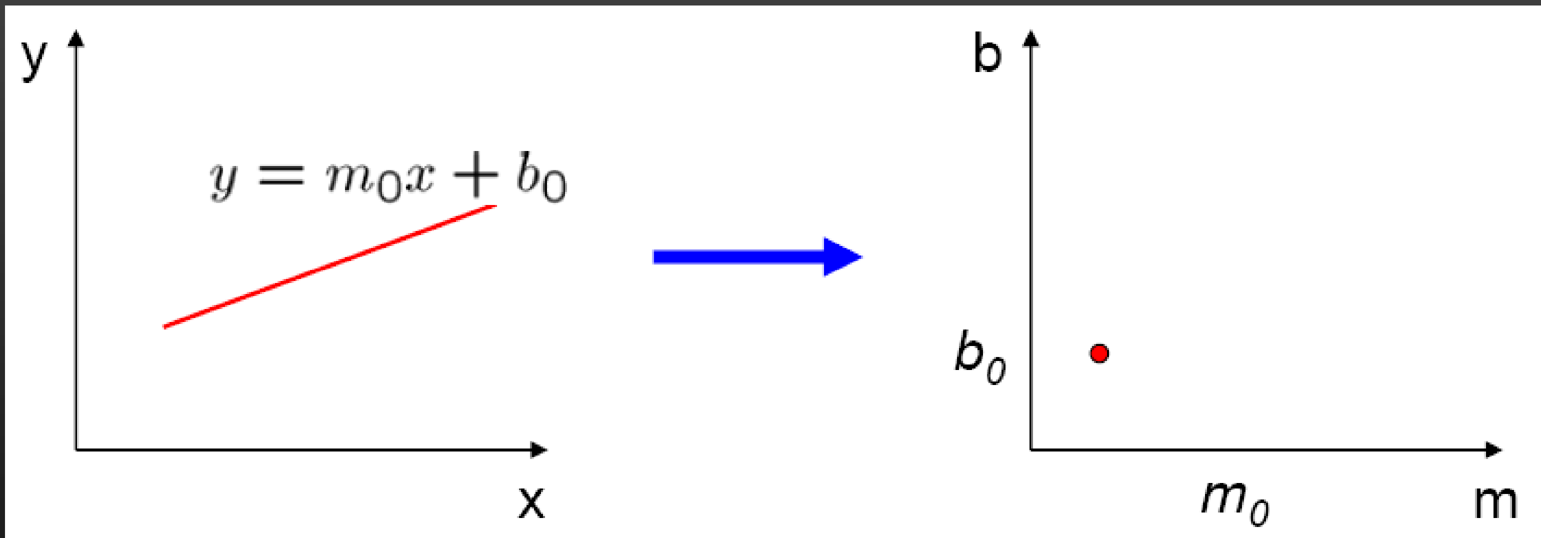


P.V.C. Hough, *Machine Analysis of Bubble Chamber Pictures,* Proc. Int. Conf. High Energy Accelerators and Instrumentation, 1959

Parameter space representation

- A line in the image corresponds to a point in Hough space

Image space   Hough parameter space



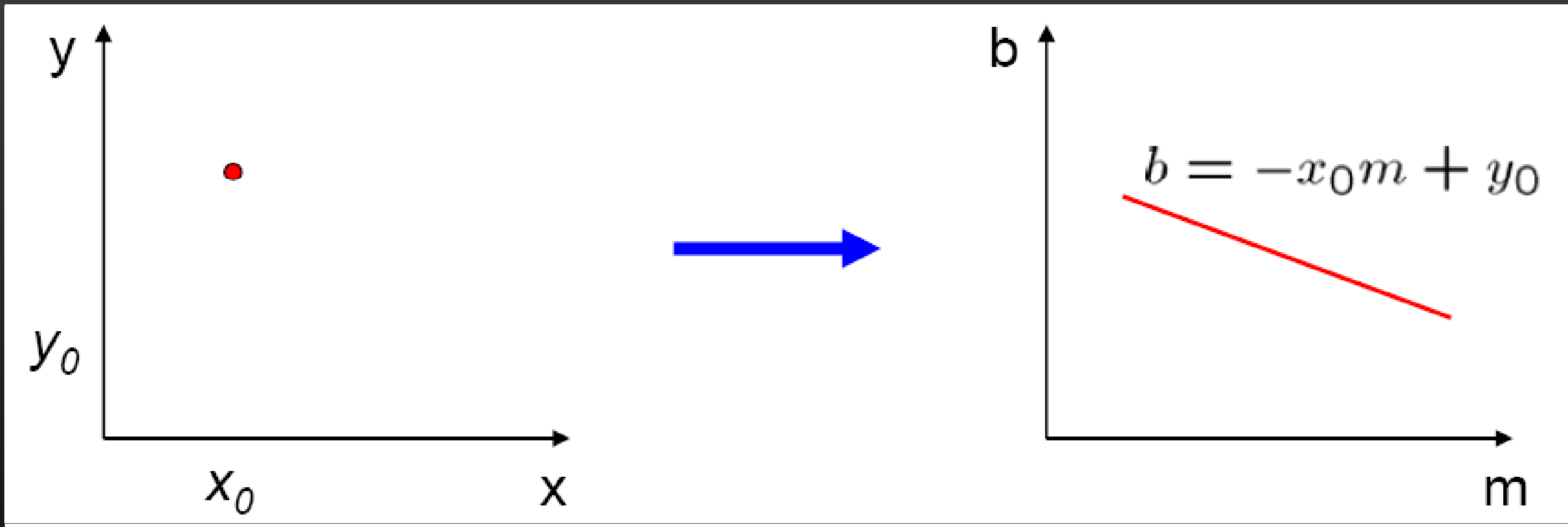
Source: K. Grauman

Parameter space representation

- What does a point (x_0, y_0) in the image space map to in the Hough space?
  - Answer: the solutions of $b = -x_0 m + y_0$
  - This is a line in Hough space

Image space

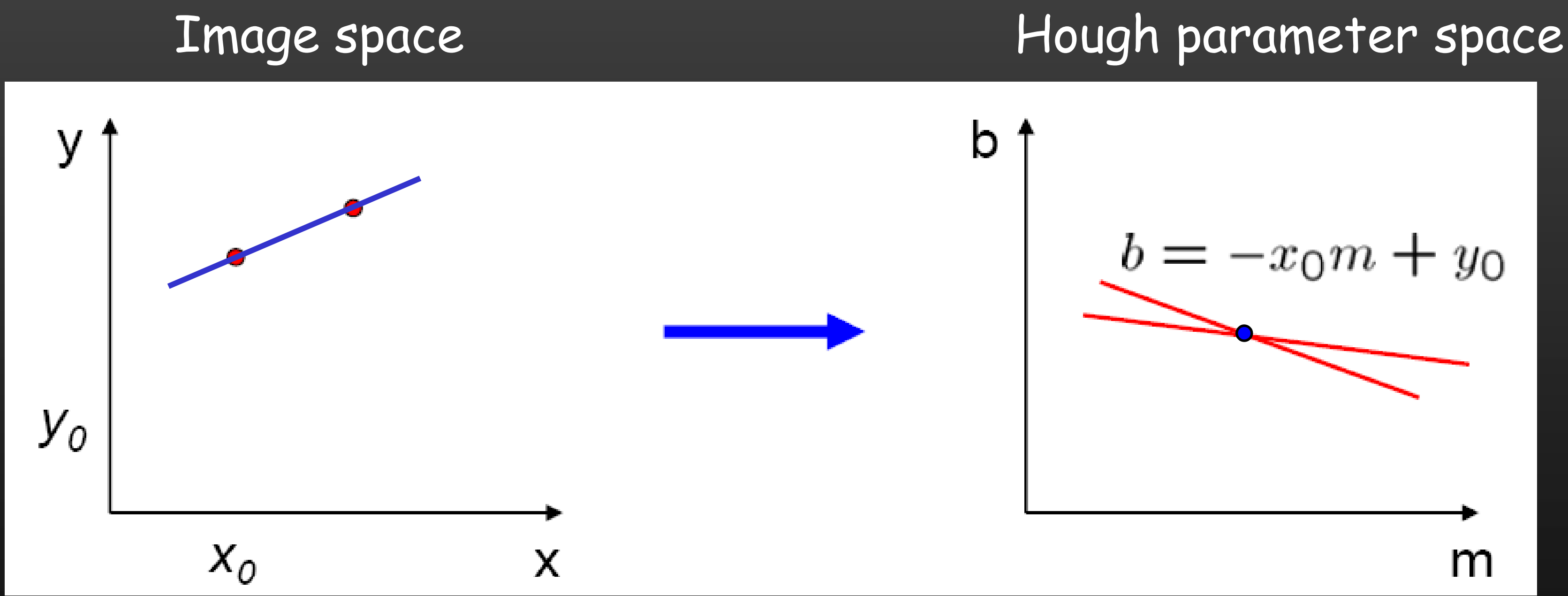
Hough parameter space



Source: K. Grauman

Parameter space representation

- Where is the line that contains both (x_0, y_0) and (x_1,y_1)?
  - It is the intersection of the lines $b = -x_0m + y_0$ and $b = -x_1m + y_1$



Source: K. Grauman

Hough transform details (D. Lowe's system)

Training phase: For each model feature, record 2D location, scale, and orientation of model (relative to normalized feature frame)

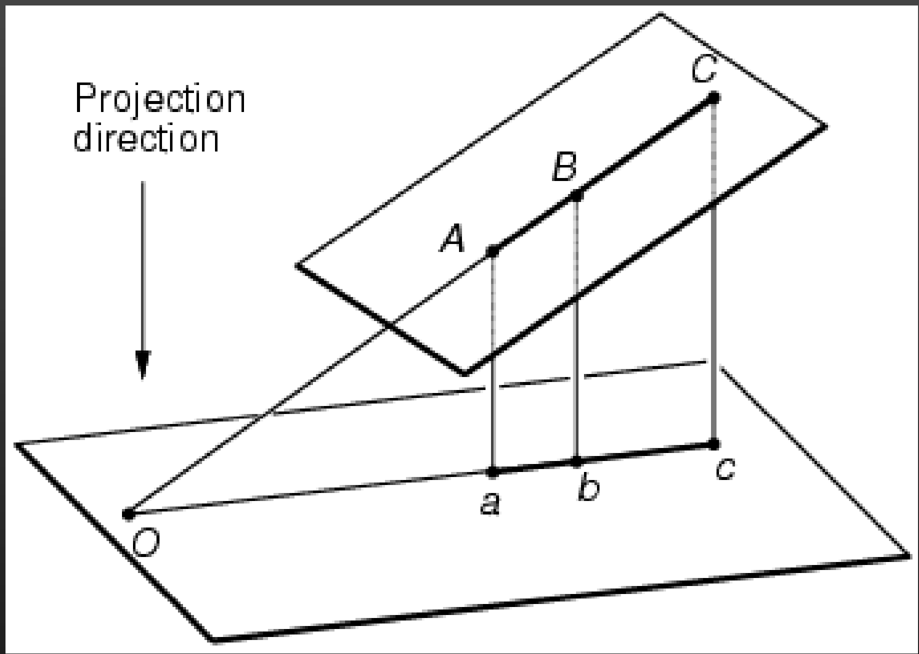
Test phase: Let each match between a test and a model feature vote in a 4D Hough space

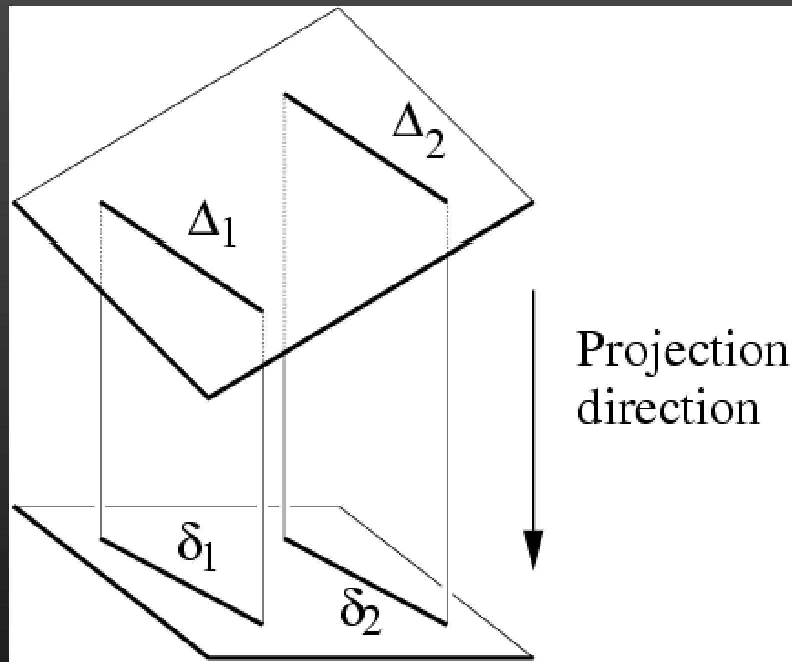
- Use broad bin sizes of 30 degrees for orientation, a factor of 2 for scale, and 0.25 times image size for location
- Vote for two closest bins in each dimension

Find all bins with at least three votes and perform geometric verification

- Estimate least squares affine transformation
- Use stricter thresholds on transformation residual
- Search for additional features that agree with the alignment

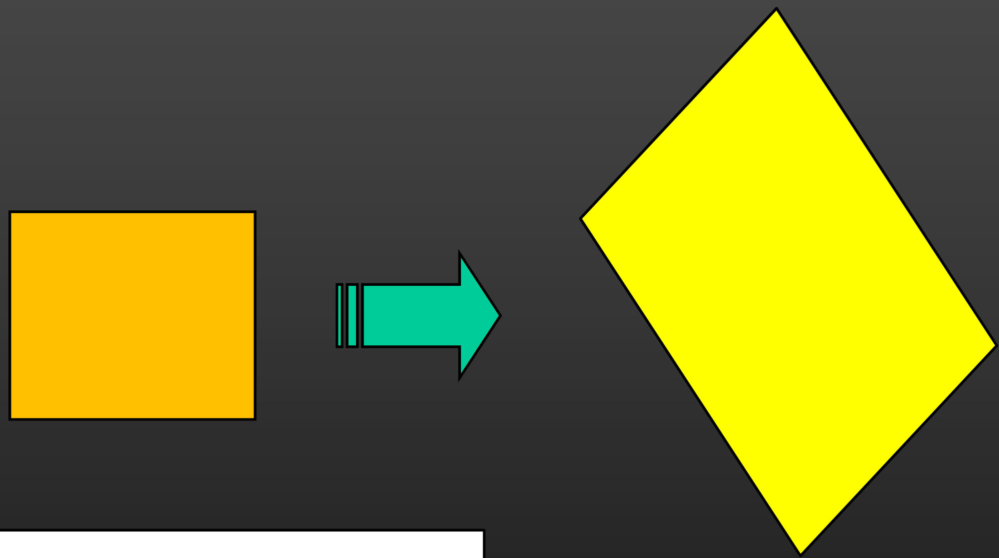
Affine projections induce affine transformations from planes onto their images.





Affine transformations

An affine transformation maps a parallelogram onto another parallelogram



$$\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

Fitting an affine transformation

Equation for affine transformation:

$$\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

9 entries, 6 degrees of freedom

2 equations in 6 unknowns

U **a** = **u**’

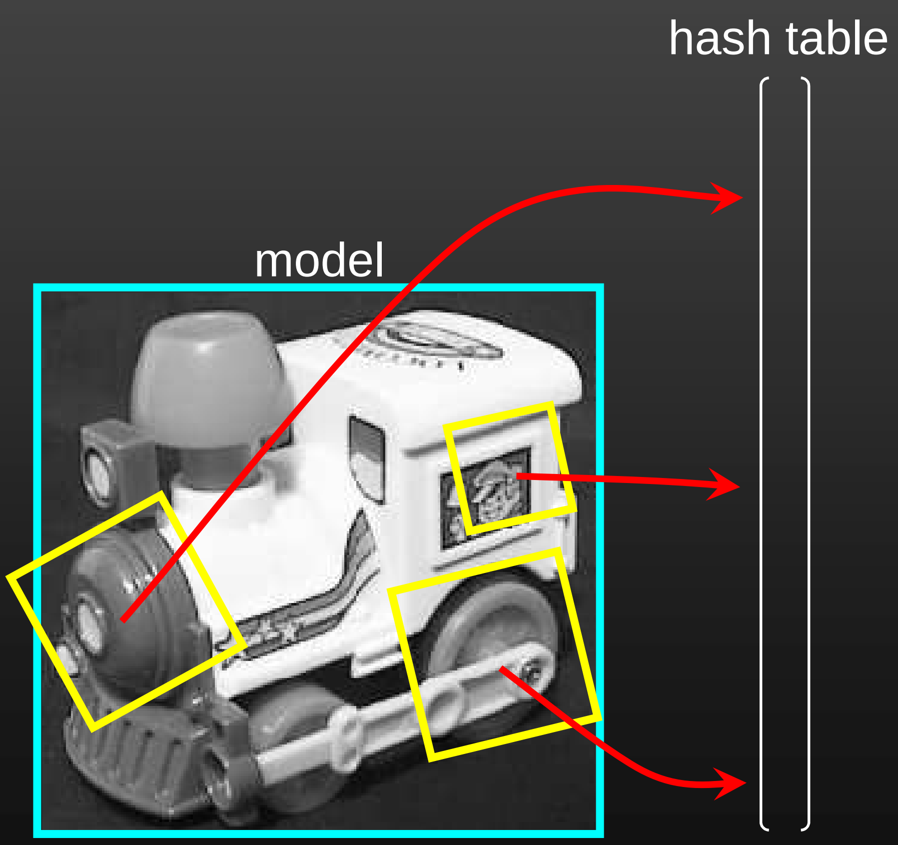
In general uniquely determined by 3 correspondences

Linear least squares for more correspondences

$$\begin{bmatrix} u & v & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & u & v & 1 \end{bmatrix} \begin{bmatrix} a_{11} \\ a_{12} \\ b_1 \\ a_{21} \\ a_{22} \\ b_2 \end{bmatrix} = \begin{bmatrix} u' \\ v' \end{bmatrix}$$

Strategy 4: Hashing

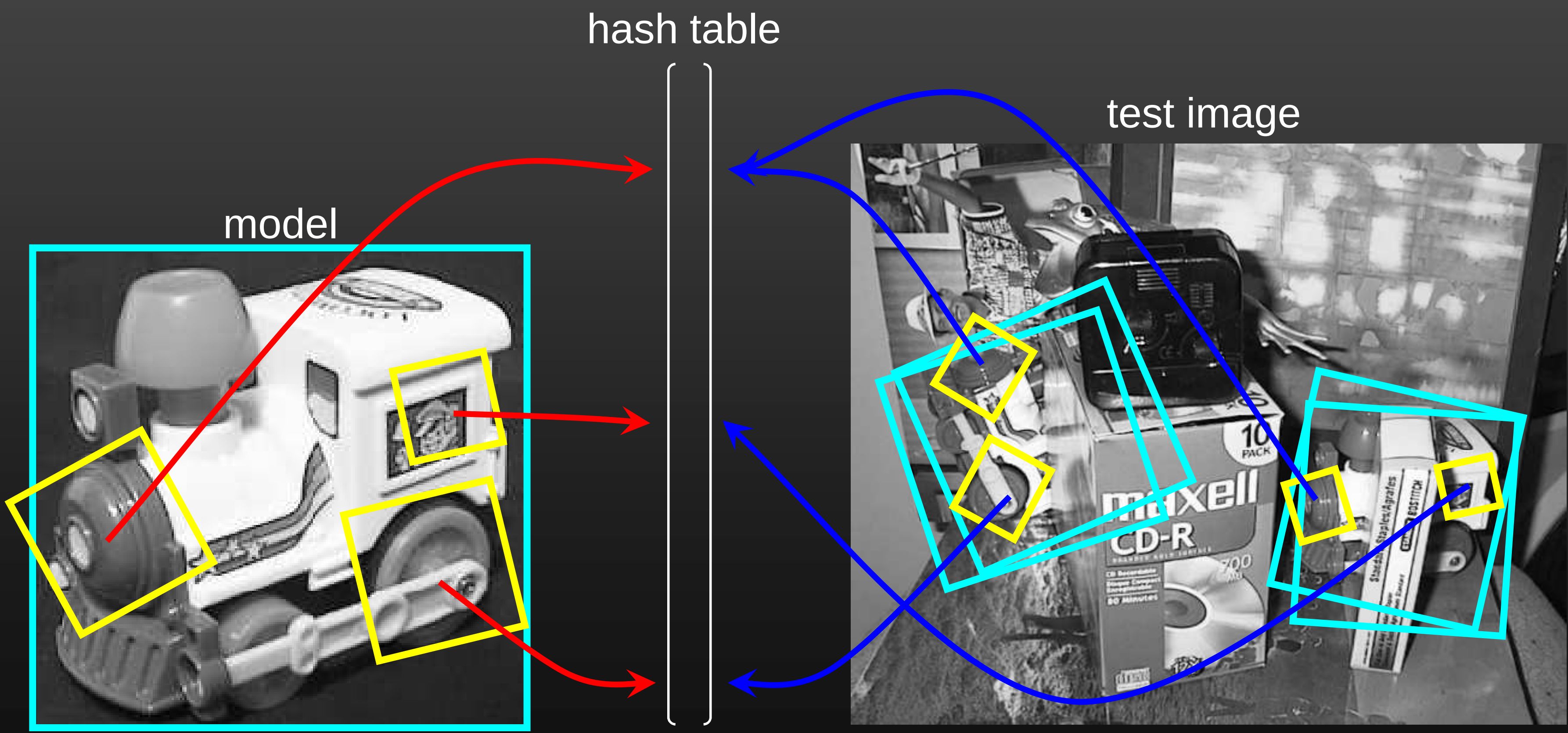
Make each invariant image feature into a low-dimensional “key” that indexes into a table of hypotheses



Strategy 4: Hashing

Make each invariant image feature into a low-dimensional “key” that indexes into a table of hypotheses

Given a new test image, compute the hash keys for all features found in that image, access the table, and look for consistent hypotheses

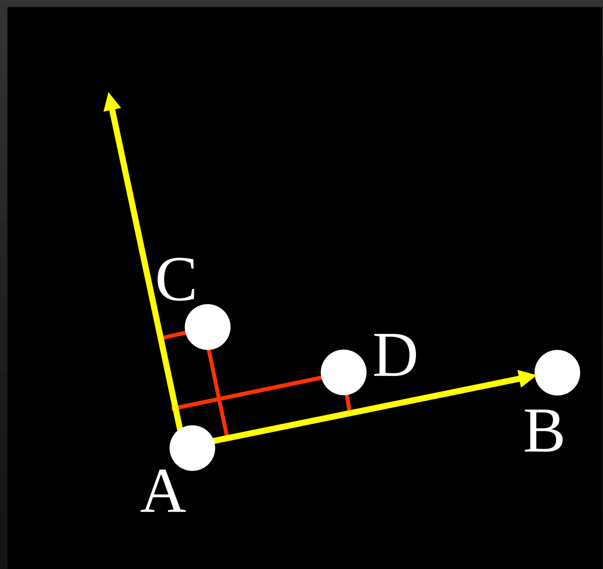


Strategy 4: Hashing

Make each invariant image feature into a low-dimensional "key" that indexes into a table of hypotheses

Given a new test image, compute the hash keys for all features found in that image, access the table, and look for consistent hypotheses

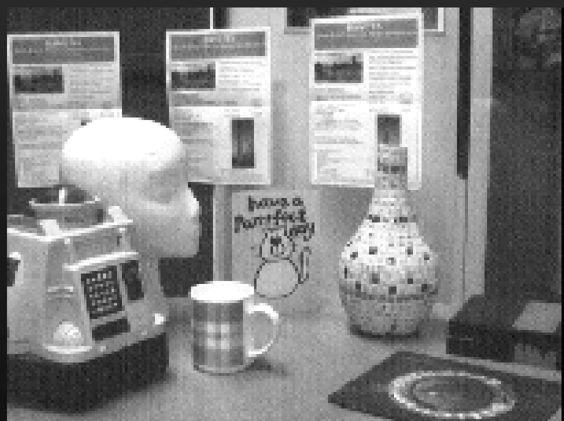
This can even work when we don't have any feature descriptors: we can take n-tuples of neighboring features and compute invariant hash codes from their geometric configurations



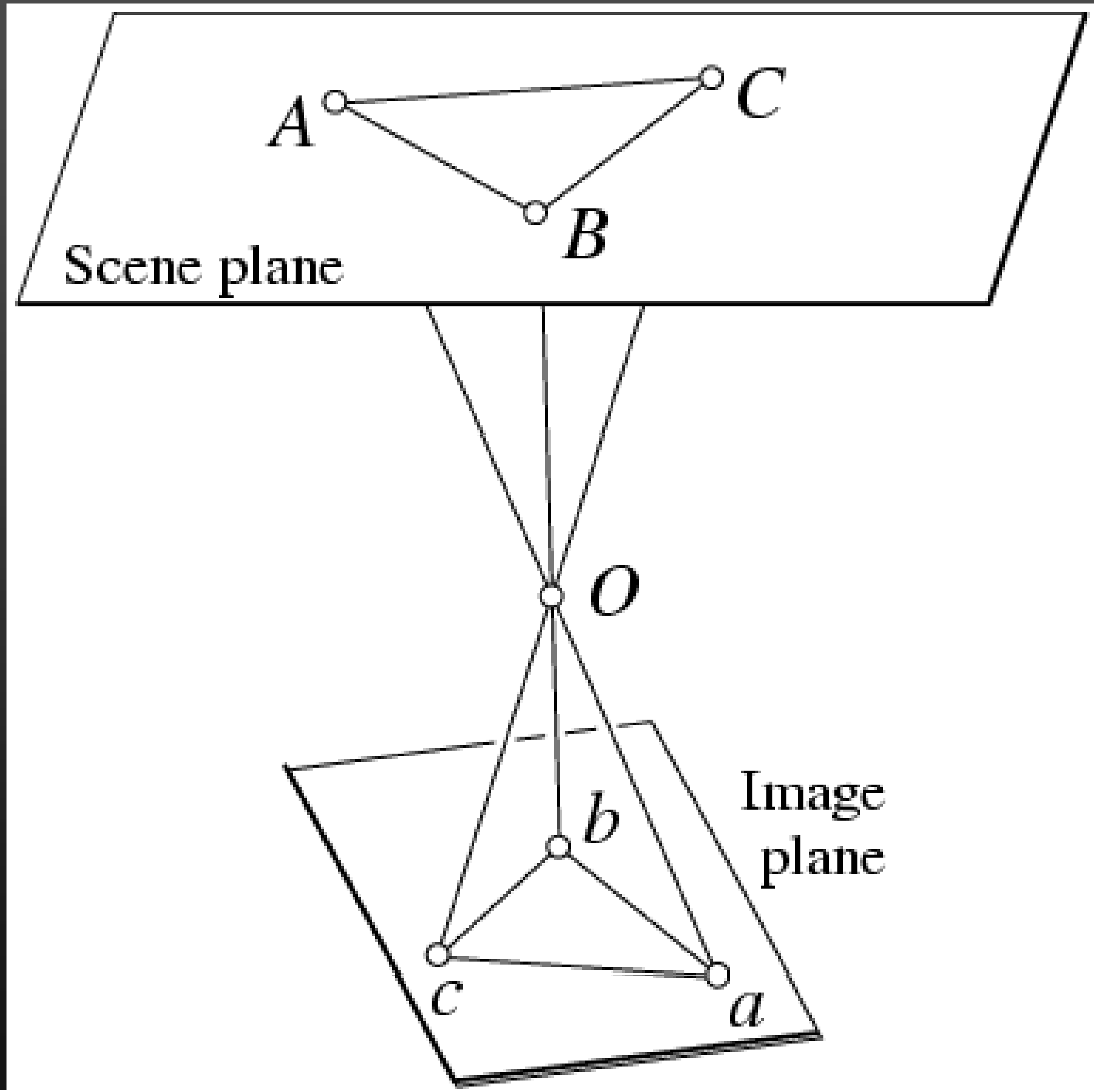
Beyond affine transformations

What is the transformation between two views of a planar surface?


What is the transformation between images from two cameras that share the same center?

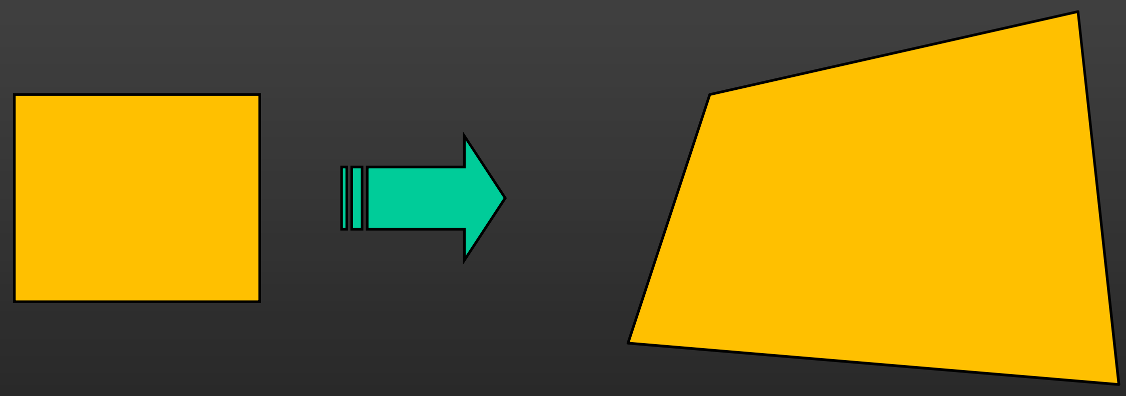



Perspective projections induce projective transformations between planes



Beyond affine transformations

Homography: plane projective transformation (transformation taking a quad to another arbitrary quad)



Fitting a homography

Recall: homogenenous coordinates

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Converting *to* homogenenous image coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

Converting *from* homogenenous image coordinates

Fitting a homography

Recall: homogenenous coordinates

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Converting *to* homogenenous image coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

Converting *from* homogenenous image coordinates

Equation for homography:

$$\lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Fitting a homography

Equation for homography:

$$\lambda \begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$\lambda \mathbf{x}_i' = \mathbf{H}\,\mathbf{x}_i = \begin{bmatrix} \mathbf{h}_1^T \\ \mathbf{h}_2^T \\ \mathbf{h}_3^T \end{bmatrix} \mathbf{x}_i$$

9 entries, 8 degrees of freedom (scale is arbitrary)

$$\mathbf{x}_i' \times \mathbf{H}\,\mathbf{x}_i = 0$$

$$\mathbf{x}_i' \times \mathbf{H}\,\mathbf{x}_i = \begin{bmatrix} y_i'\,\mathbf{h}_3^T\mathbf{x}_i - \mathbf{h}_2^T\mathbf{x}_i \\ \mathbf{h}_1^T\mathbf{x}_i - x_i'\,\mathbf{h}_3^T\mathbf{x}_i \\ x_i'\,\mathbf{h}_2^T\mathbf{x}_i - y_i'\,\mathbf{h}_1^T\mathbf{x}_i \end{bmatrix}$$

$$\begin{bmatrix} 0^T & -\mathbf{x}_i^T & y_i'\,\mathbf{x}_i^T \\ \mathbf{x}_i^T & 0^T & -x_i'\,\mathbf{x}_i^T \\ -y_i'\,\mathbf{x}_i^T & x_i'\,\mathbf{x}_i^T & 0^T \end{bmatrix} \begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{pmatrix} = 0$$

3 equations, only 2 linearly independent

Direct linear transform

$$\begin{bmatrix} 0^T & \mathbf{x}_1^T & -y_1' \mathbf{x}_1^T \\ \mathbf{x}_1^T & 0^T & -x_1' \mathbf{x}_1^T \\ \cdots & \cdots & \cdots \\ 0^T & \mathbf{x}_n^T & -y_n' \mathbf{x}_n^T \\ \mathbf{x}_n^T & 0^T & -x_n' \mathbf{x}_n^T \end{bmatrix} \begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{pmatrix} = 0$$

$$\mathbf{A}\,\mathbf{h} = 0$$

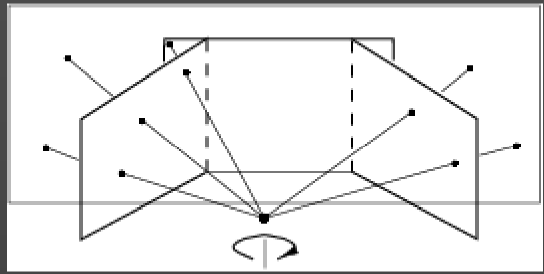
H has 8 degrees of freedom (9 parameters, but scale is arbitrary)

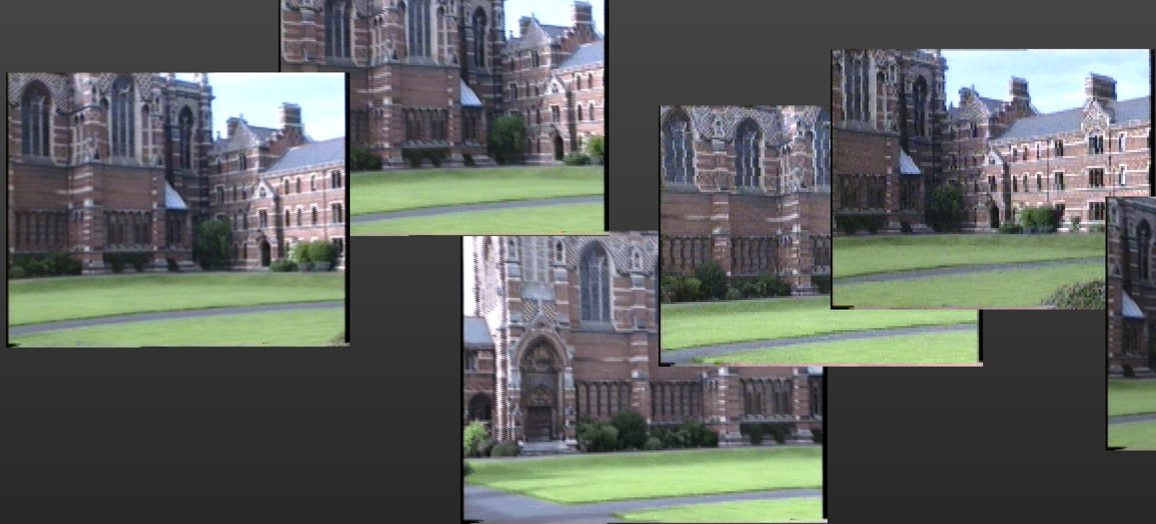
One match gives us two linearly independent equations

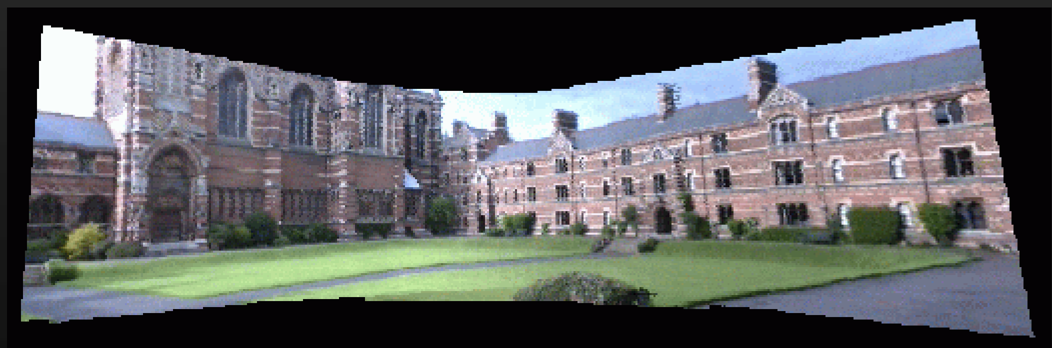
Four matches needed for a minimal solution (null space of 8x9 matrix)

More than four: homogeneous least squares

Application: Panorama stitching



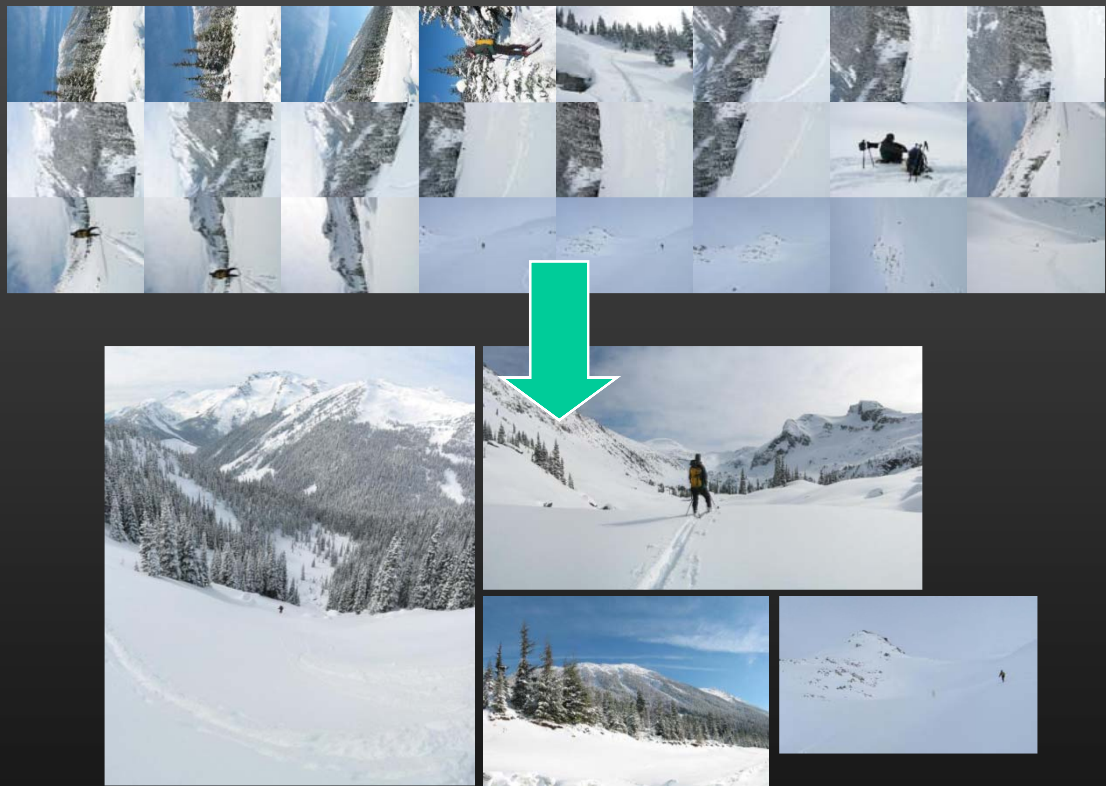




Images courtesy of A. Zisserman.

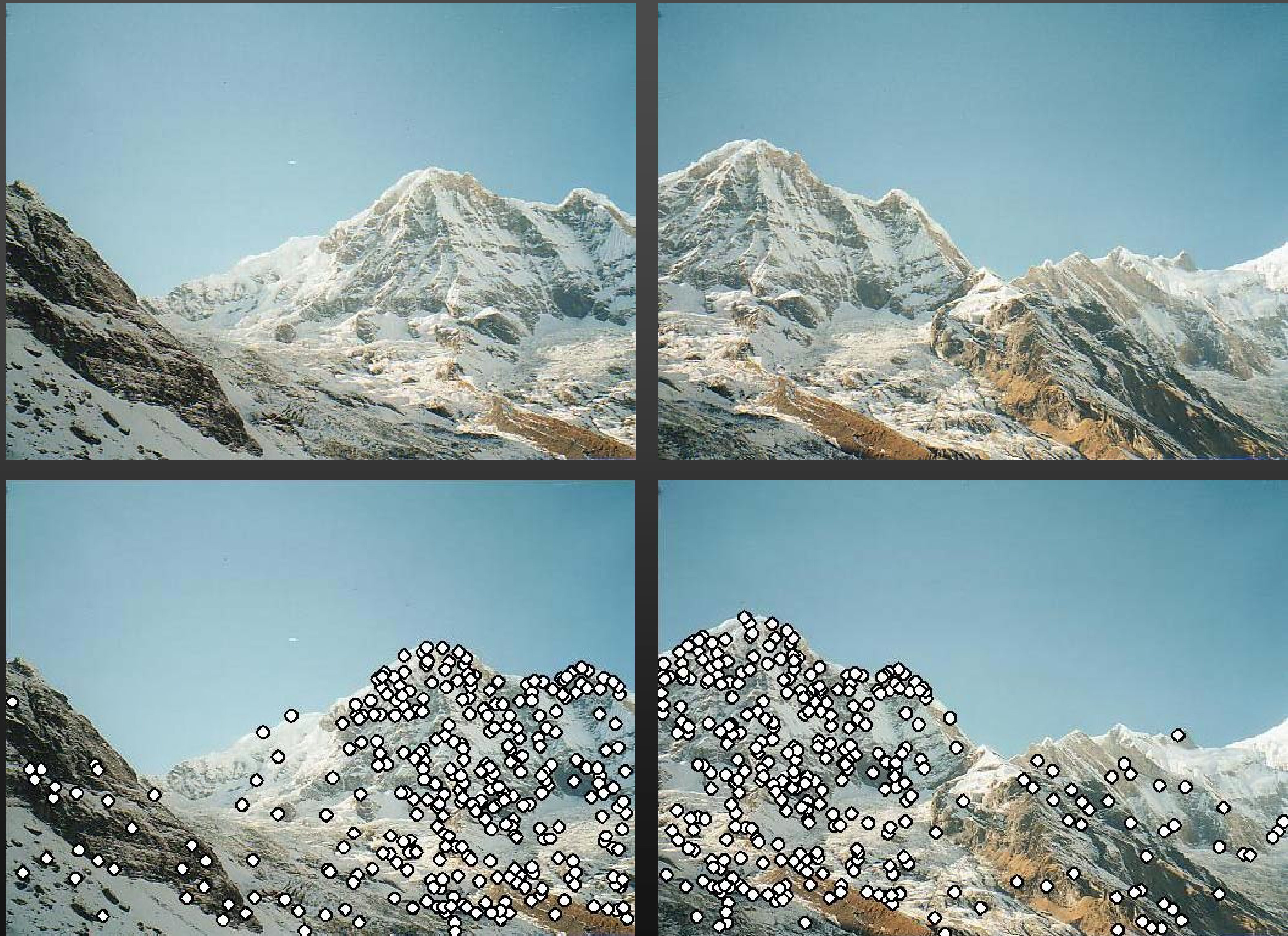
Recognizing panoramas

Given contents of a camera memory card, automatically figure out which pictures go together and stitch them together into panoramas

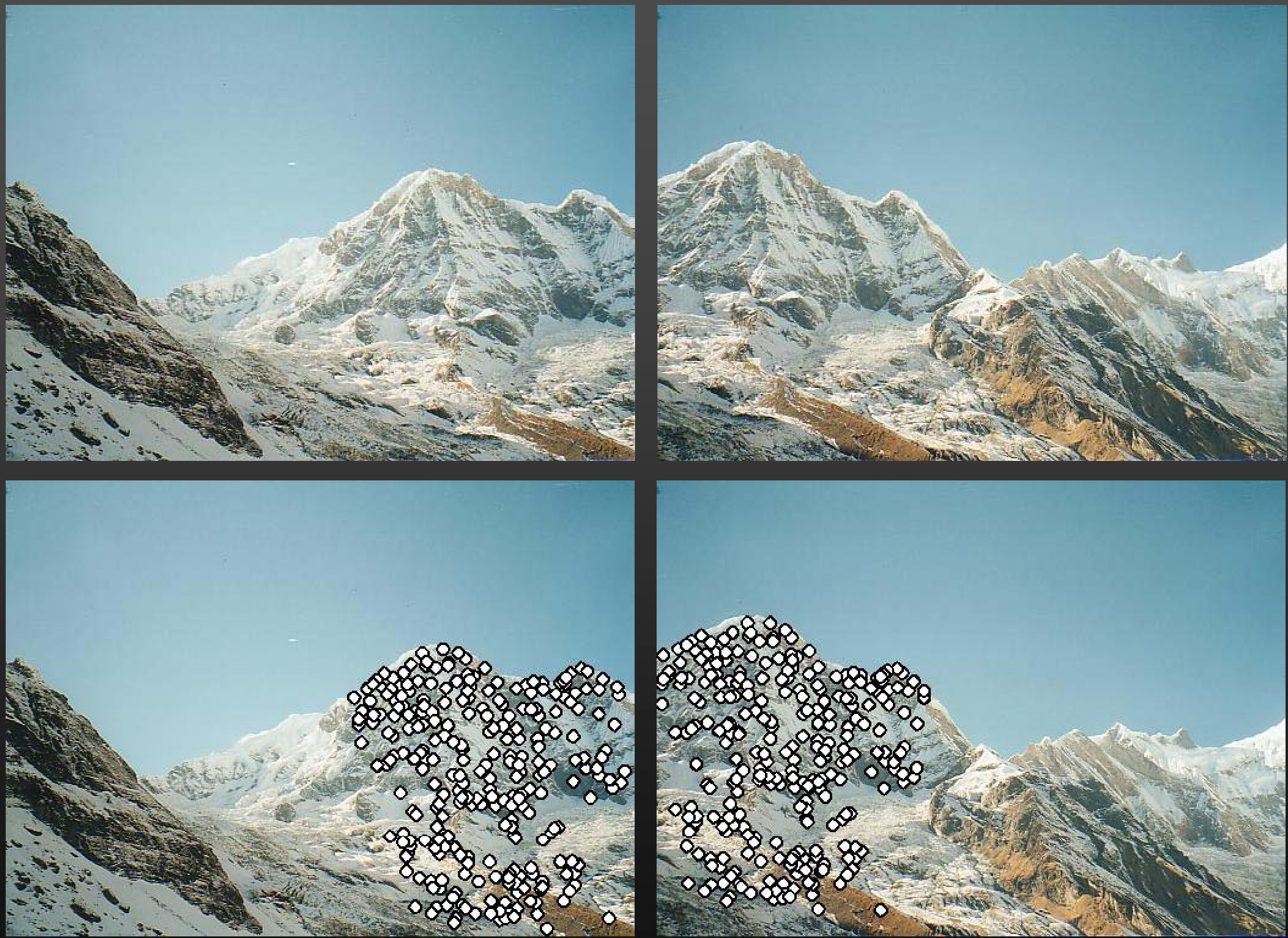


M. Brown and D. Lowe, “Recognizing panoramas”, ICCV 2003.

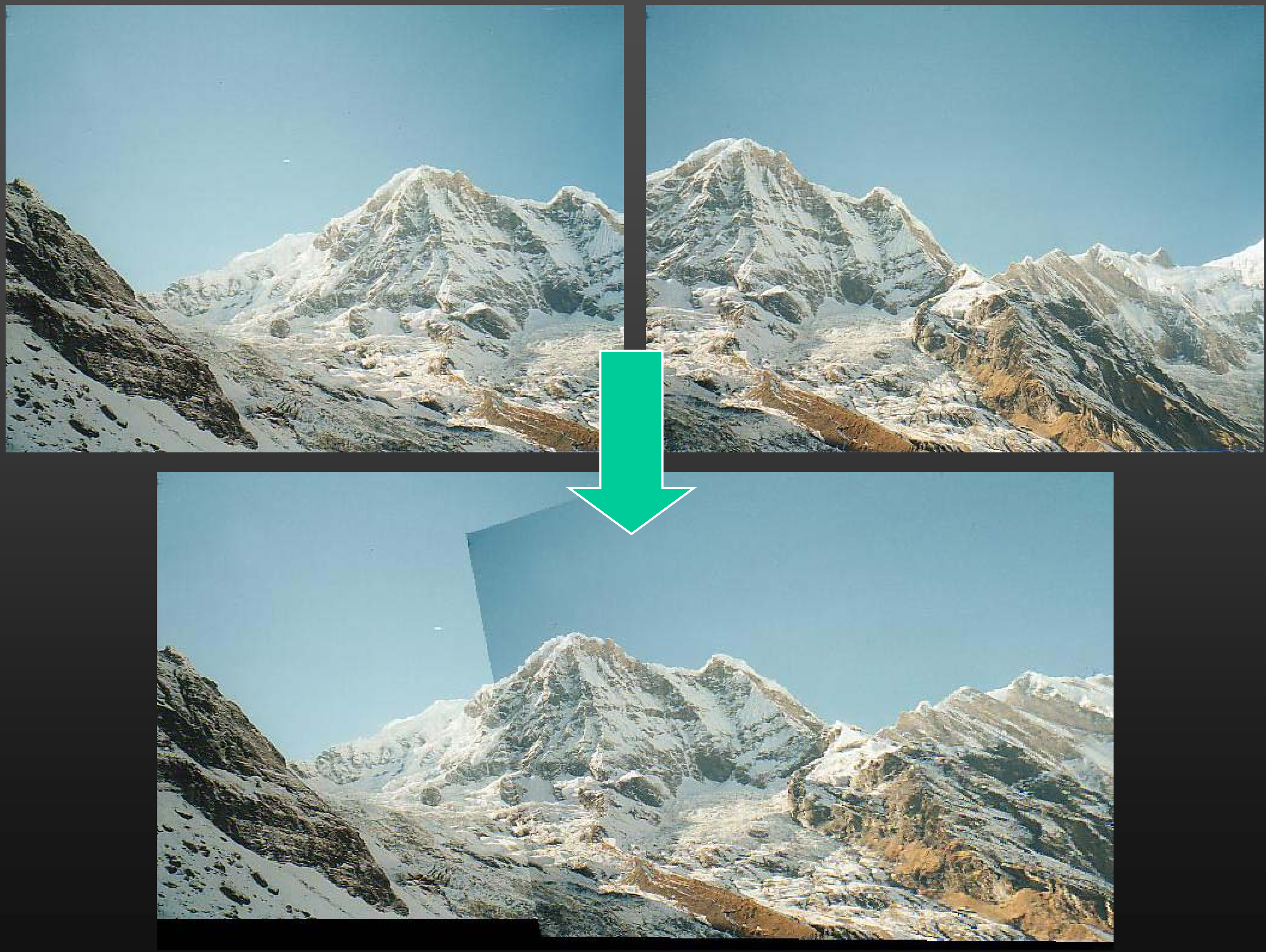
1. Estimate homography (RANSAC)



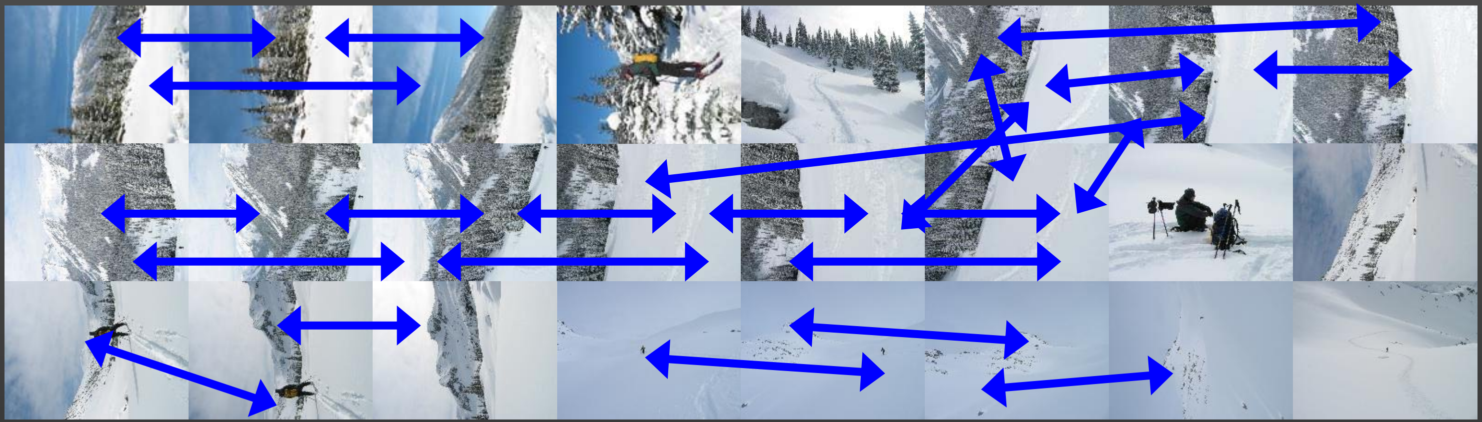
1. Estimate homography (RANSAC)



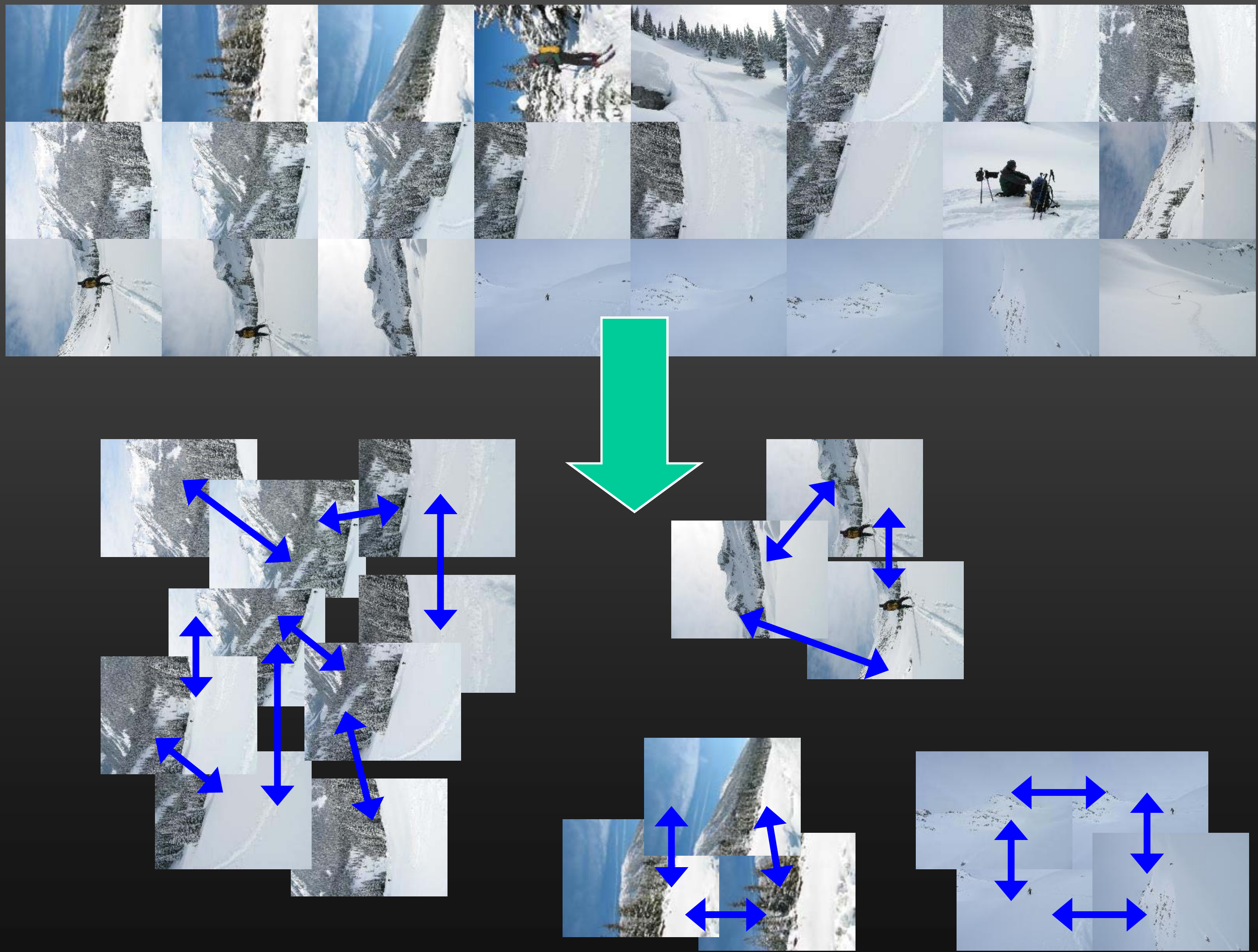
1. Estimate homography (RANSAC)



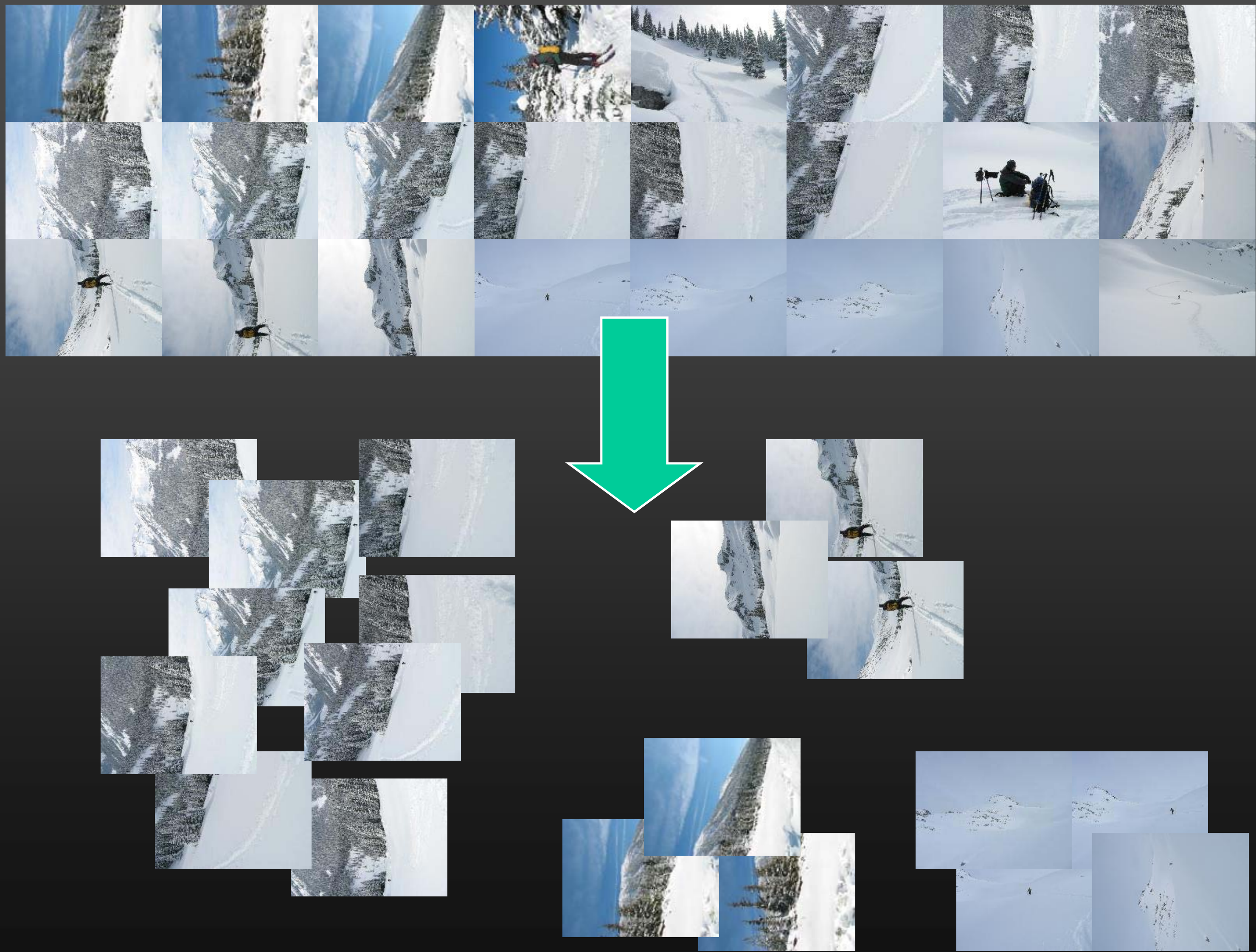
2. Find connected sets of images



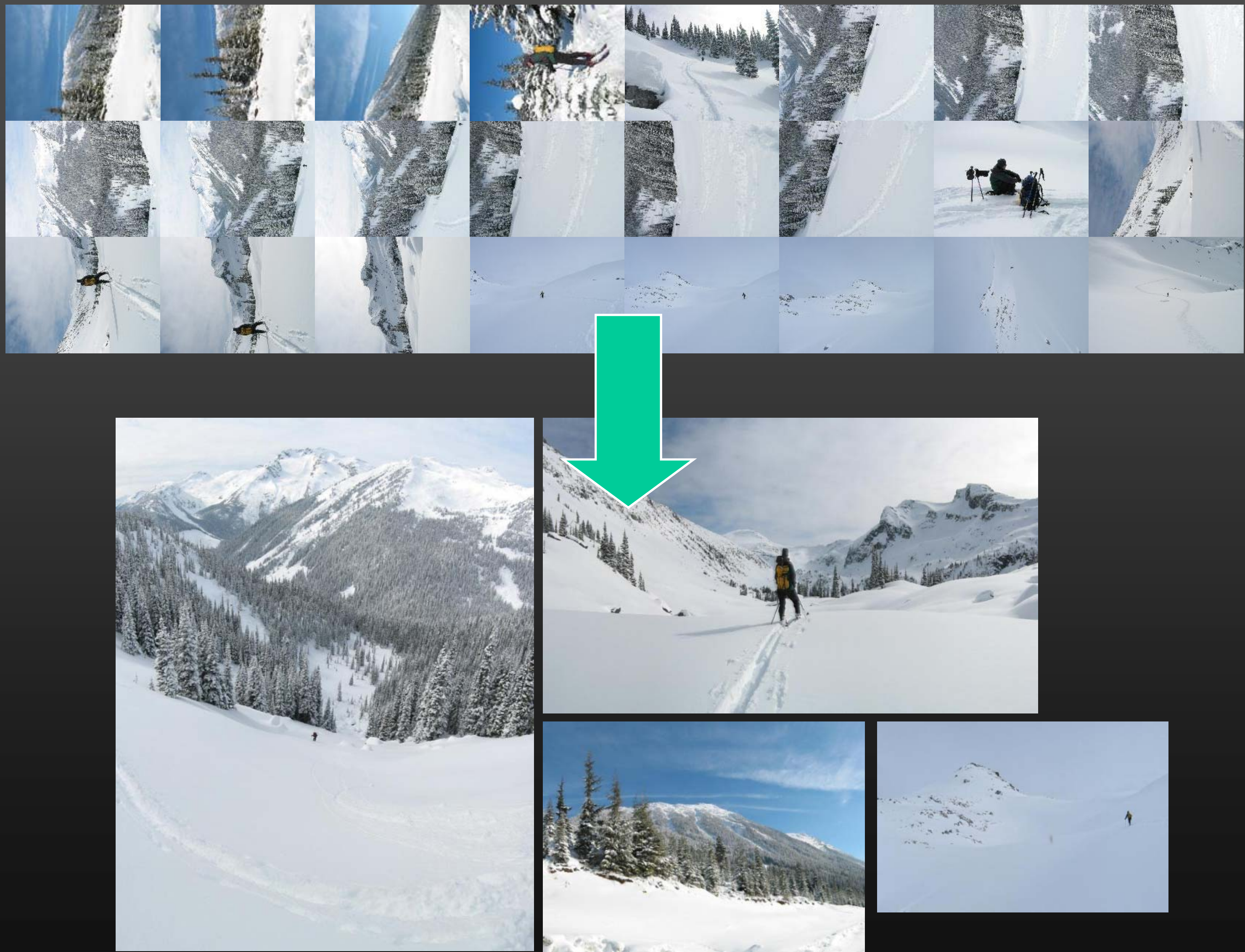
2. Find connected sets of images



2. Find connected sets of images



3. Stitch and blend the panoramas



Results

Issues in alignment-based applications

Choosing the geometric alignment model

- Tradeoff between "correctness" and robustness (also, efficiency)

Choosing the descriptor

- "Rich" imagery (natural images): high-dimensional patch-based descriptors (e.g., SIFT)
- "Impoverished" imagery (e.g., star fields): need to create invariant geometric descriptors from k-tuples of point-based features

Strategy for finding putative matches

- Small number of images, one-time computation (e.g., panorama stitching): brute force search
- Large database of model images, frequent queries: indexing or hashing
- Heuristics for feature-space pruning of putative matches

Issues in alignment-based applications

Choosing the geometric alignment model

Choosing the descriptor

Strategy for finding putative matches

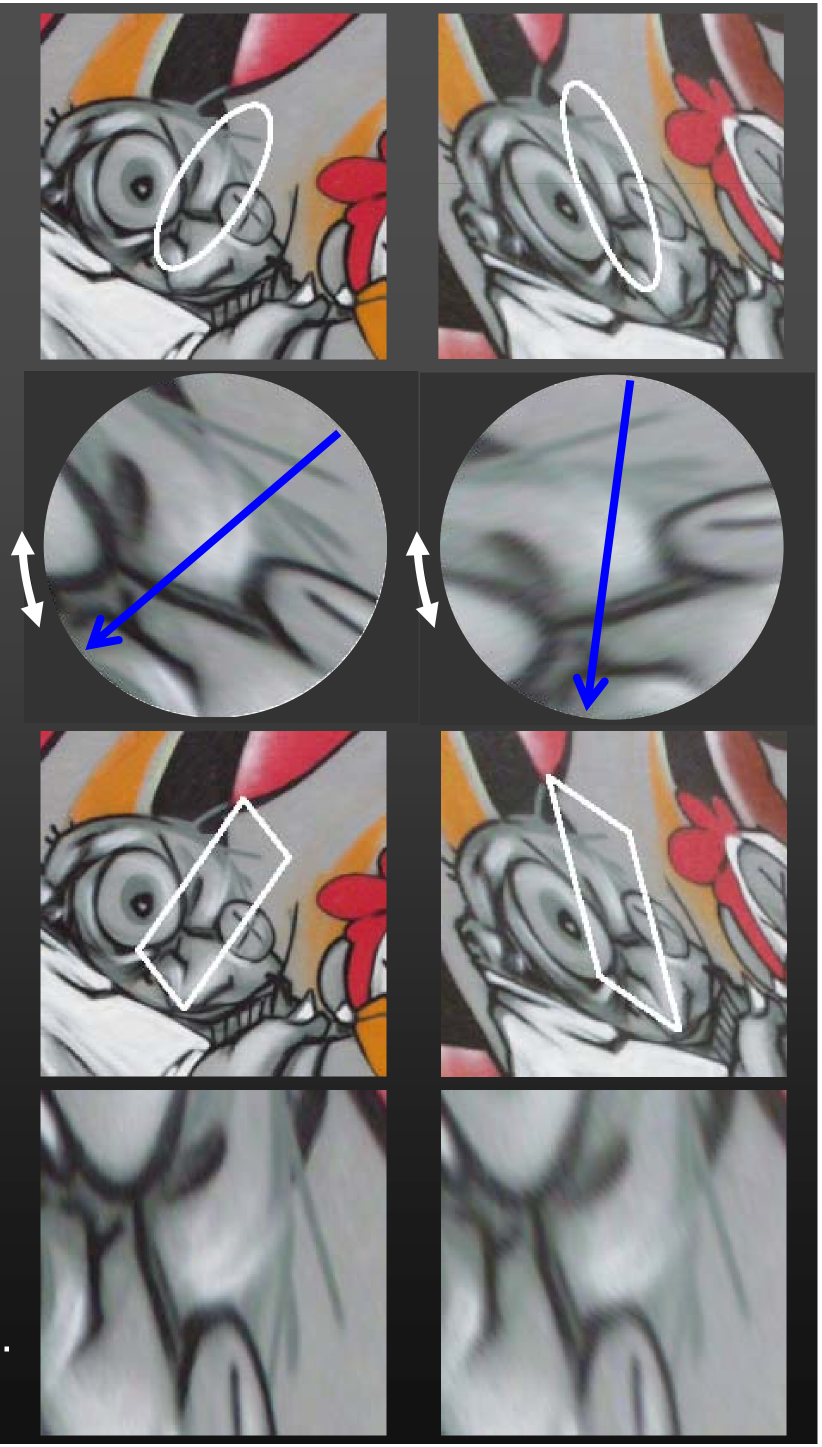
Hypothesis generation strategy

- Relatively large inlier ratio: RANSAC
- Small inlier ratio: locality constraints, Hough transform

Hypothesis verification strategy

- Size of consensus set, residual tolerance depend on inlier ratio and expected accuracy of the model
- Possible refinement of geometric model
- Dense verification

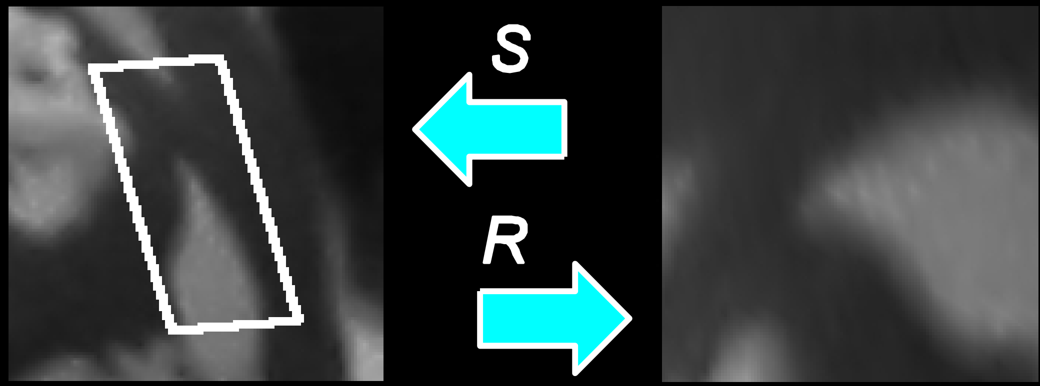
Affine Patches for 3D Alignment



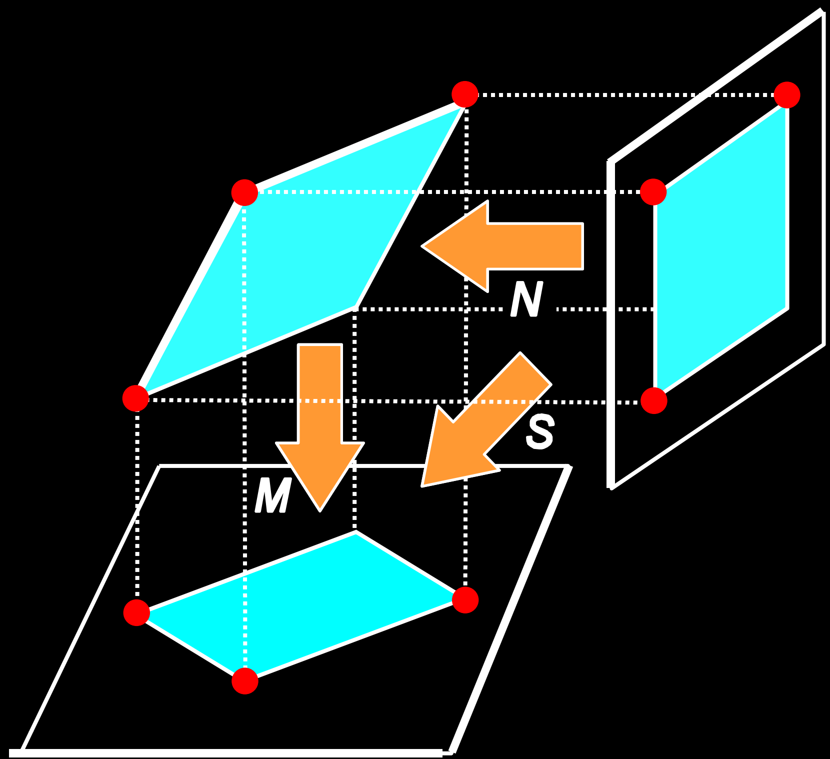
Repeatibility, covariance, invariance

Tell & Carlsson (2000); Kadir & Brady (2001); Matas et al. (2001); Tuytelaars & Van Gool (2002)

Modeling and recognizing 3D rigid solids



Johnson & Hebert (1998); Lowe (1999)



Idea: $\underline{S} = \underline{M} \times \underline{N}$

- The (smooth) surface of a solid is never globally planar,
- but it is always locally planar

$\underline{S} \rightarrow \underline{M}\ ,\ \underline{N}$

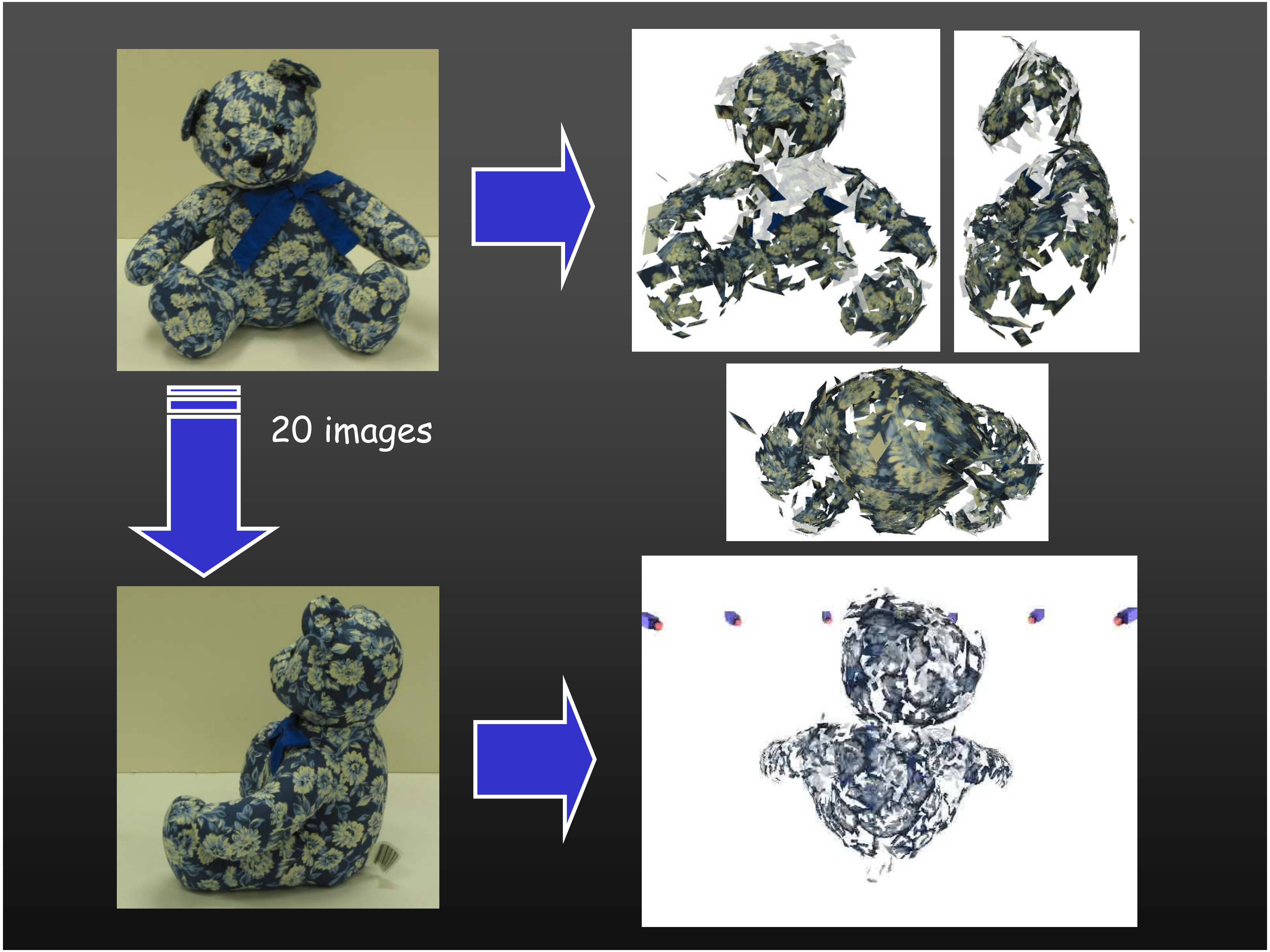
$E \leftarrow |\underline{S} - \underline{M}\,\underline{N}|$

Tomasi & Kanade (1992)

Rothganger et al. (CVPR'03)

Duda & Hart (1972); Weiss (1987); Burns et al. (1992); Mundy et al. (1992, 1994); Rothwell et al. (1992)

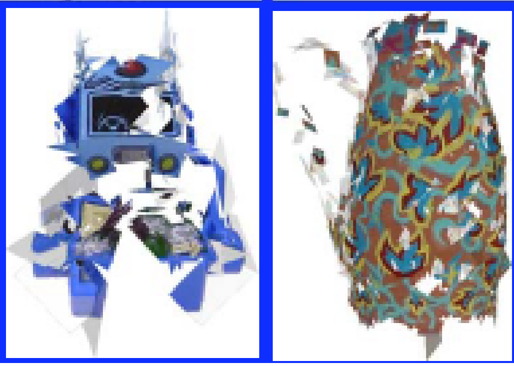
Ayache & Faugeras (1982); Hebert & Faugeras (1983); Gaston et al. (1984); Huttenlocher & Ullman (1987)


20 images

Dataset: 149 training images of 8 objects

Number of images per object

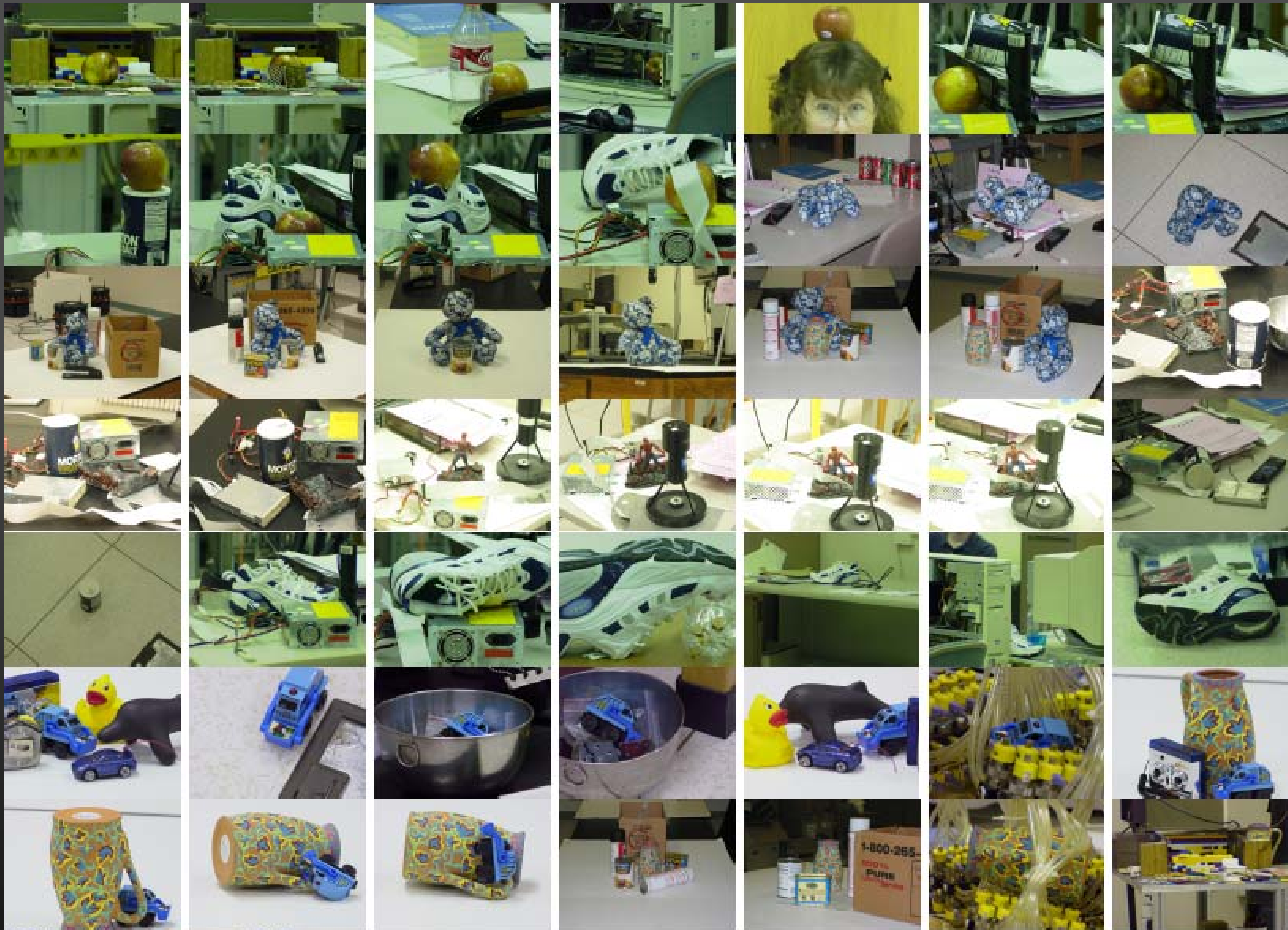
Apple	Bear	Rubble	Salt	Shoe	Spidey	Truck	Vase
29	20	16	16	16	16	16	20

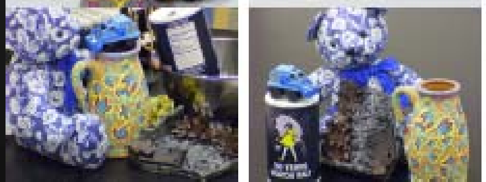


8 learned 3D object models (Rothganger et al.'04)

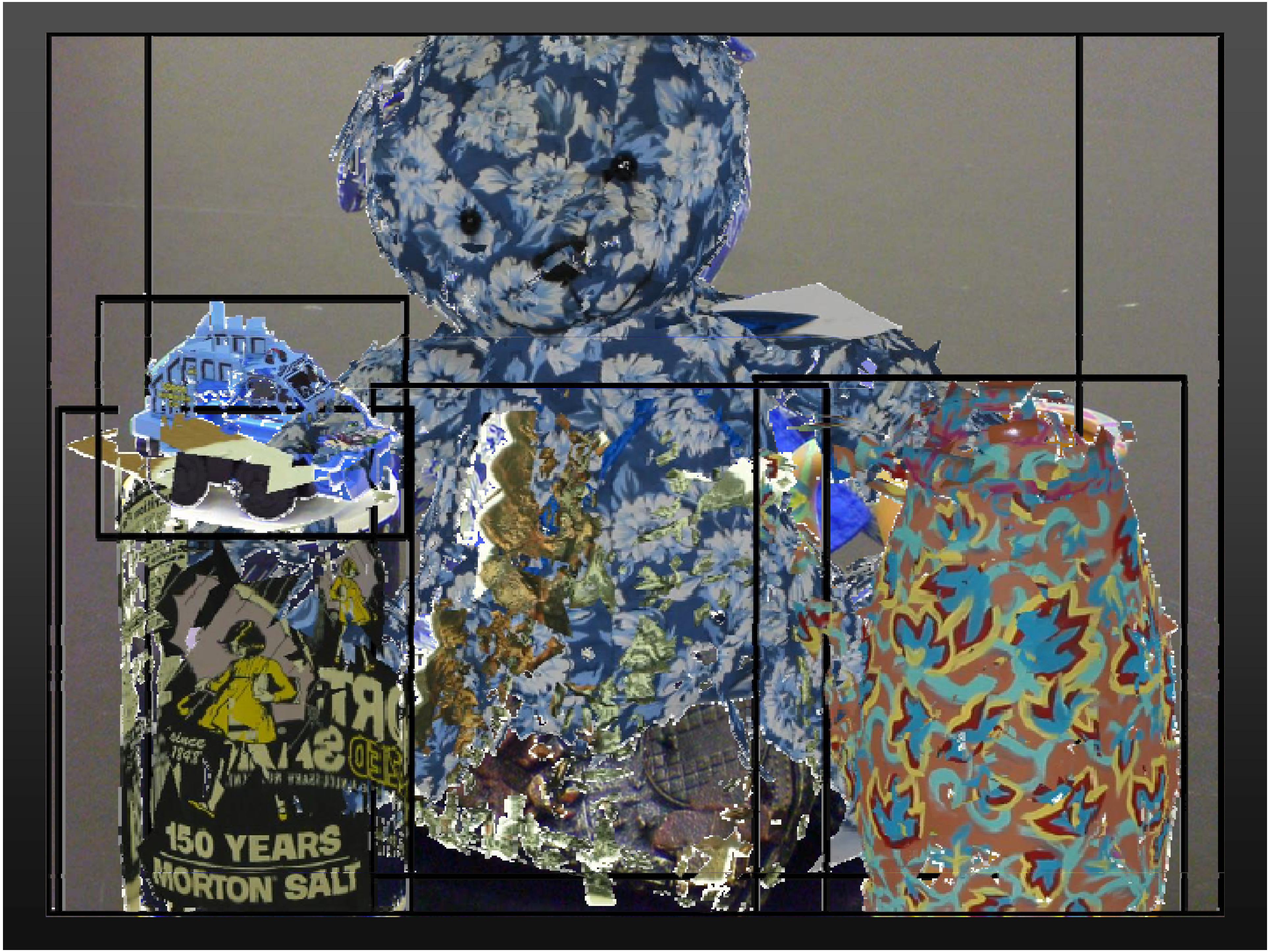
Number of patches per model

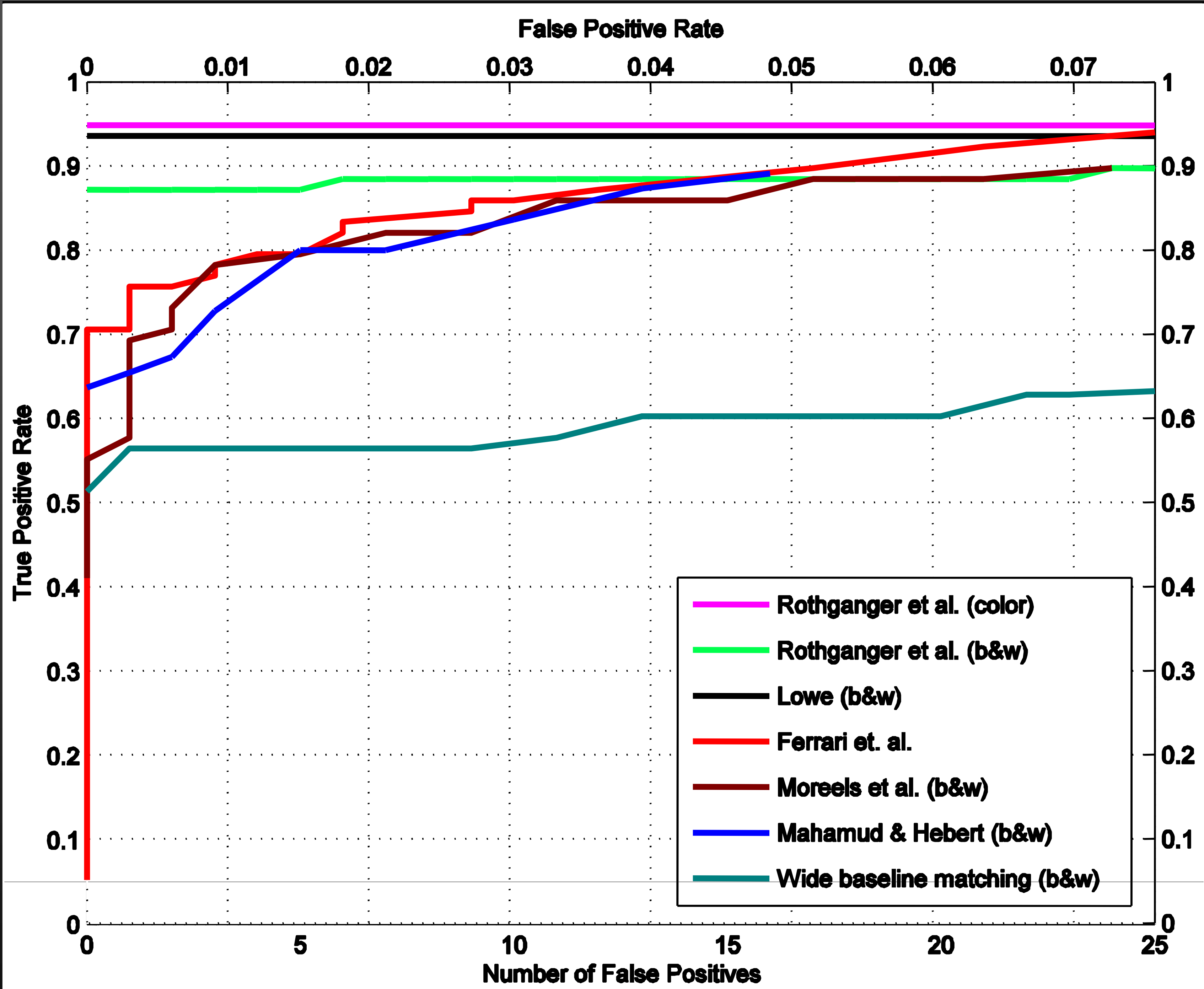
Apple	Bear	Rubble	Salt	Shoe	Spidey	Truck	Vase
759	4014	727	866	488	526	518	1085





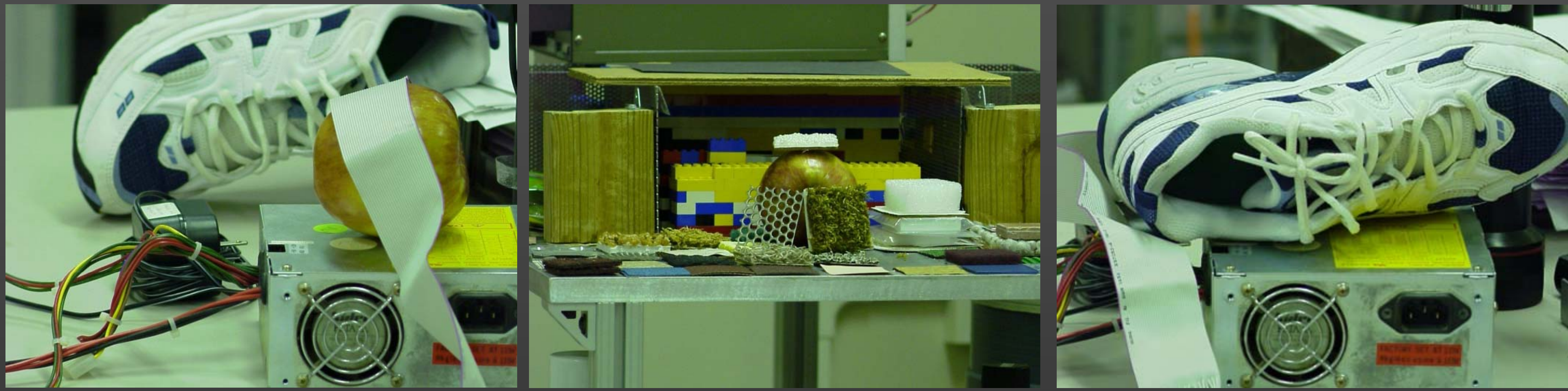
Dataset: 51 test images with 1 to 5 of the 8 objects present in each image.


150 YEARS
MORTON SALT



False Positive Rate
0
0.01
0.02
0.03
0.04
0.05
0.06
0.07
True Positive Rate
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Rothganger et al. (color)
Rothganger et al. (b&w)
Lowe (b&w)
Ferrari et. al.
Moreels et al. (b&w)
Mahamud & Hebert (b&w)
Wide baseline matching (b&w)
0
5
10
15
20
25
Number of False Positives

The four failures



Some successes