Lecture 9A

Other topics .

To be covered:

- Alternative methods to compute fundamental matrix
- Self-calibration
- 3 views and the trifocal tensor
- Multiple-view reconstruction
- RANSAC with fundamental matrix
- Bundle adjustment
- Calibrated reconstruction
  - 5-point algorithm
- Stereo
- Optical flow
- SLAM

Videos

- Photosynth
- Being There
- 3D Image Stabilization
- Motion magnification

Algebraic Minimization Algorithm

The algebraic minimization algorithm

Enforcing the singularity constraint

- SVD method minimizes $||\mathtt{F}' - \mathtt{F}||$.
- simple and rapid.
- Not optimal
- Treats all entries of $\mathtt{F}$ equally.
- However, some entries of $\mathtt{F}$ are more tightly constrained by the data.

The Algebraic Method

- Minimize $||\mathtt{A}\mathbf{f}'||$ subject to $||\mathbf{f}'|| = 1$ **AND** $\det \mathtt{F}' = 0$.
- $\det \mathtt{F}' = 0$ is a cubic constraint.
- Requires an iterative solution.
- However, simple iterative method works.

Solution assuming known epipole

- We may write $\mathtt{F} = \mathtt{M}[\mathbf{e}]_\times$, where $\mathbf{e}$ is epipole.
- $\mathtt{F}$ of this form is singular.
- Assume $\mathbf{e}$ is known, find $\mathtt{M}$.
- Write $\mathtt{F} = \mathtt{M}[\mathbf{e}]_\times$ as $\mathbf{f} = \mathbf{E}\mathbf{m}$

$$\begin{pmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{pmatrix} = \begin{bmatrix} 0 & -e_3 & e_2 & & & & & & \\ e_3 & 0 & -e_1 & & & & & & \\ -e_2 & e_1 & 0 & & & & & & \\ & & & 0 & -e_3 & e_2 & & & \\ & & & e_3 & 0 & -e_1 & & & \\ & & & -e_2 & e_1 & 0 & & & \\ & & & & & & 0 & -e_3 & e_2 \\ & & & & & & e_3 & 0 & -e_1 \\ & & & & & & -e_2 & e_1 & 0 \end{bmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{31} \\ m_{32} \\ m_{33} \end{pmatrix}$$

Solution assuming known epipole

- We may write $\mathtt{F} = \mathtt{M}[\mathbf{e}]_\times$, where $\mathbf{e}$ is epipole.
- $\mathtt{F}$ of this form is singular.
- Assume $\mathbf{e}$ is known, find $\mathtt{M}$.
- Write $\mathtt{F} = \mathtt{M}[\mathbf{e}]_\times$ as $\mathbf{f} = \mathbf{E}\mathbf{m}$

$$\begin{pmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{pmatrix} = \begin{bmatrix} 0 & -e_3 & e_2 & & & & & & \\ e_3 & 0 & -e_1 & & & & & & \\ -e_2 & e_1 & 0 & & & & & & \\ & & & 0 & -e_3 & e_2 & & & \\ & & & e_3 & 0 & -e_1 & & & \\ & & & -e_2 & e_1 & 0 & & & \\ & & & & & & 0 & -e_3 & e_2 \\ & & & & & & e_3 & 0 & -e_1 \\ & & & & & & -e_2 & e_1 & 0 \end{bmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{31} \\ m_{32} \\ m_{33} \end{pmatrix}$$

Solution assuming known epipole

- We may write $\mathtt{F} = \mathtt{M}[\mathbf{e}]_\times$, where $\mathbf{e}$ is epipole.
- $\mathtt{F}$ of this form is singular.
- Assume $\mathbf{e}$ is known, find $\mathtt{M}$.
- Write $\mathtt{F} = \mathtt{M}[\mathbf{e}]_\times$ as $\mathbf{f} = \mathbf{E}\mathbf{m}$

$$\begin{pmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{pmatrix} = \begin{bmatrix} 0 & -e_3 & e_2 & & & & & & \\ e_3 & 0 & -e_1 & & & & & & \\ -e_2 & e_1 & 0 & & & & & & \\ & & & 0 & -e_3 & e_2 & & & \\ & & & e_3 & 0 & -e_1 & & & \\ & & & -e_2 & e_1 & 0 & & & \\ & & & & & & 0 & -e_3 & e_2 \\ & & & & & & e_3 & 0 & -e_1 \\ & & & & & & -e_2 & e_1 & 0 \end{bmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{31} \\ m_{32} \\ m_{33} \end{pmatrix}$$

Solution assuming known epipole - continued

- Write $\mathtt{Af} = \mathtt{AEm}$.
- Minimize $||\mathtt{AEm}||$ subject to $||\mathtt{Em}|| = 1$.
- This is a linear least-squares estimation problem.
- Non-iterative algorithm involving SVD is possible
- Reference : Hartley, “Minimizing Algebraic Error”, Royal Society Proceedings, 1998.

Iterative Algebraic Estimation

Find the fundamental matrix $\mathtt{F}$ that minimizes the algebraic error $||\mathbf{Af}||$ subject to $||\mathbf{f}|| = 1$ and $\det \mathtt{F} = 0$.

- **Concept :** Vary epipole $\mathbf{e}$ to minimize the algebraic error $||\mathbf{Af}'|| = ||\mathtt{AE}\mathbf{m}||$.
- **Remark :** Each choice of epipole $\mathbf{e}$ defines a minimimum error vector $\mathtt{AE}\mathbf{m}$ as above.
- Use Levenberg-Marquardt method to minimize this error.
- Simple 3×9 minimization problem.
  - 3 inputs – the coordinates of the epipole
  - 9 outputs – the algebraic error vector $\mathbf{Af}' = \mathtt{AE}\mathbf{m}$.
- Each step requires estimation of $\mathbf{m}$ using SVD method.
- Tricks can be used to avoid SVD (see Hartley-Royal-Society).

Minimization of Geometric Error

The Gold Standard (ML) Method

Minimizing the Gold-Standard error function.

- Initial 3D reconstruction :

$$\begin{aligned} \mathtt{P} &= [\mathtt{I} \mid \mathbf{0}] \\ \mathtt{P}' &= [\mathtt{M} \mid \mathbf{t}] \\ \mathbf{X}_i &= (\mathtt{X}_i, \mathtt{Y}_i, 1, \mathtt{T}_i)^\top \end{aligned}$$

- Compute $\hat{\mathbf{x}}_i = \mathtt{P}\mathbf{X}_i = (\mathtt{X}_i, \mathtt{Y}_i, 1)^\top$ and $\hat{\mathbf{x}}'_i = \mathtt{P}'\mathbf{X}_i$.
- Iterate over $\mathtt{P}'$ and $\mathbf{X}_i = (\mathtt{X}_i, \mathtt{Y}_i, 1, \mathtt{T}_i)^\top$ to minimize cost function :

$$\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2$$

- Total of $3n + 12$ parameters.
  - 12 parameters for the camera matrix $\mathtt{P}'$
  - 3 parameters for each point $\mathbf{X}_i$.
  - Once $\mathtt{P}' = [\mathtt{M} \mid \mathbf{t}]$ is found, compute $\mathtt{F} = [\mathbf{t}]_\times \mathtt{M}$.

The Gold Standard (ML) Method

Minimizing the Gold-Standard error function.

- Initial 3D reconstruction :

$$\begin{aligned} \mathtt{P} &= [\mathtt{I} \mid \mathbf{0}] \\ \mathtt{P}' &= [\mathtt{M} \mid \mathbf{t}] \\ \mathbf{X}_i &= (\mathtt{X}_i, \mathtt{Y}_i, 1, \mathtt{T}_i)^\top \end{aligned}$$

- Compute $\hat{\mathbf{x}}_i = \mathtt{P}\mathbf{X}_i = (\mathtt{X}_i, \mathtt{Y}_i, 1)^\top$ and $\hat{\mathbf{x}}'_i = \mathtt{P}'\mathbf{X}_i$.
- Iterate over $\mathtt{P}'$ and $\mathbf{X}_i = (\mathtt{X}_i, \mathtt{Y}_i, 1, \mathtt{T}_i)^\top$ to minimize cost function :

$$\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2$$

- Total of $3n + 12$ parameters.
  - 12 parameters for the camera matrix $\mathtt{P}'$
  - 3 parameters for each point $\mathbf{X}_i$.
  - Once $\mathtt{P}' = [\mathtt{M} \mid \mathbf{t}]$ is found, compute $\mathtt{F} = [\mathbf{t}]_\times \mathtt{M}$.

Sparse Levenberg-Marquardt

Reference : Hartley- Azores

(i) Coordinates of $\mathbf{X}_i$ do not affect $\hat{\mathbf{x}}_j$ or $\hat{\mathbf{x}}'_j$ (for $i \neq j$).

(ii) Sparse LM takes advantage of sparseness

(iii) Linear time in n (number of points).

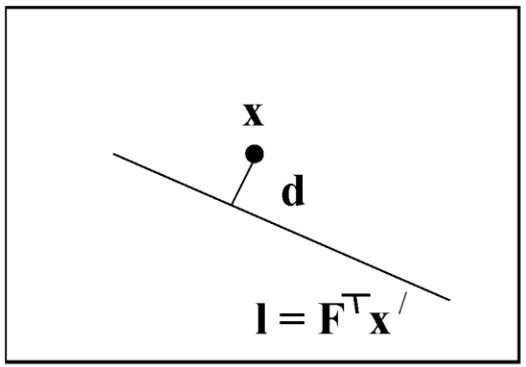
(iv) Reference : Hartley- Azores

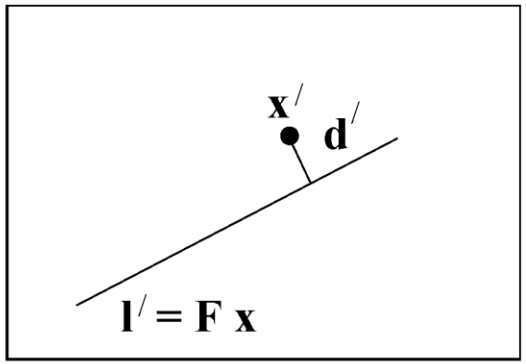
Parametrization of rank-2 matrices

Both epipoles as parameters. The resulting form of $\mathtt{F}$ is

$$\mathtt{F} = \begin{bmatrix} a & b & \alpha a + \beta b \\ c & d & \alpha c + \beta d \\ \alpha' a + \beta' c & \alpha' b + \beta' d & \alpha' \alpha a + \alpha' \beta b + \beta' \alpha c + \beta' \beta d \end{bmatrix}$$

Epipolar distance





- Point correspondence $\mathbf{x}' \leftrightarrow \mathbf{x}$:
- Point $\mathbf{x} \mapsto$ epipolar line $\mathtt{F}\mathbf{x}$.
- Epipolar distance is distance of point $\mathbf{x}'$ to epipolar line $\mathtt{F}\mathbf{x}$.
- Write $\mathtt{F}\mathbf{x} = (\lambda, \mu, \nu)^\top$ and $\mathbf{x}' = (x', y', 1)^\top$.
- Distance is

$$d(\mathbf{x}', \mathtt{F}\mathbf{x}) = \mathbf{x}'^\top \mathtt{F}\mathbf{x}(\lambda^2 + \mu^2)^{-1/2}$$

Epipolar distance - continued

- Epipolar distance may be written as

$$d(\mathbf{x}', \mathtt{F}\mathbf{x}) = \frac{\mathbf{x}'^{\top}\mathtt{F}\mathbf{x}}{((\mathtt{F}\mathbf{x})_1^2 + (\mathtt{F}\mathbf{x})_2^2)^{1/2}}$$

- Total cost function :

$$\sum_i d(\mathbf{x}'_i, \mathtt{F}\mathbf{x}_i)^2 = \sum_i \frac{(\mathbf{x}_i'^{\top}\mathtt{F}\mathbf{x}_i)^2}{(\mathtt{F}\mathbf{x}_i)_1^2 + (\mathtt{F}\mathbf{x}_i)_2^2}$$

- Total cost : sum over all $\mathbf{x}'_i \leftrightarrow \mathbf{x}_i$.
- Minimize this cost function over parametrization of $\mathtt{F}$.

Symmetric epipolar distance

- Epipolar distance function is not symmetric.
- Prefer sum of distances in both images.
- Symmetric cost function is

$$d(\mathbf{x}', \mathtt{F}\mathbf{x})^2 + d(\mathbf{x}, \mathtt{F}^\top \mathbf{x}')^2$$

- Sum over all points :

$$\text{Cost} = \sum_i (\mathbf{x}_i'^\top \mathtt{F}\mathbf{x}_i)^2 \left(\frac{1}{(\mathtt{F}\mathbf{x}_i)_1^2 + (\mathtt{F}\mathbf{x}_i)_2^2} + \frac{1}{(\mathtt{F}^\top \mathbf{x}_i')_1^2 + (\mathtt{F}^\top \mathbf{x}_i')_2^2} \right)$$

Problem

- Points near the epipole have a disproportionate influence.
- Small deviation in point makes big difference to epipolar line.

Luong / Zhang's other error function

$$\sum_i \frac{(\mathbf{x}_i'^{\top}\mathtt{F}\mathbf{x}_i)^2}{(\mathtt{F}\mathbf{x}_i)_1^2 + (\mathtt{F}\mathbf{x}_i)_2^2 + (\mathtt{F}^{\top}\mathbf{x}_i')_1^2 + (\mathtt{F}^{\top}F\mathbf{x}_i')_2^2}$$

Represents a first-order approximation to geometric error.

Sampson distance

More Algorithm Comparison

Experimental procedure

(i) Find matched points in image pair.

(ii) Select n matched points at random

(iii) Compute the fundamental matrix

(iv) Compute epipolar distance for all other points.

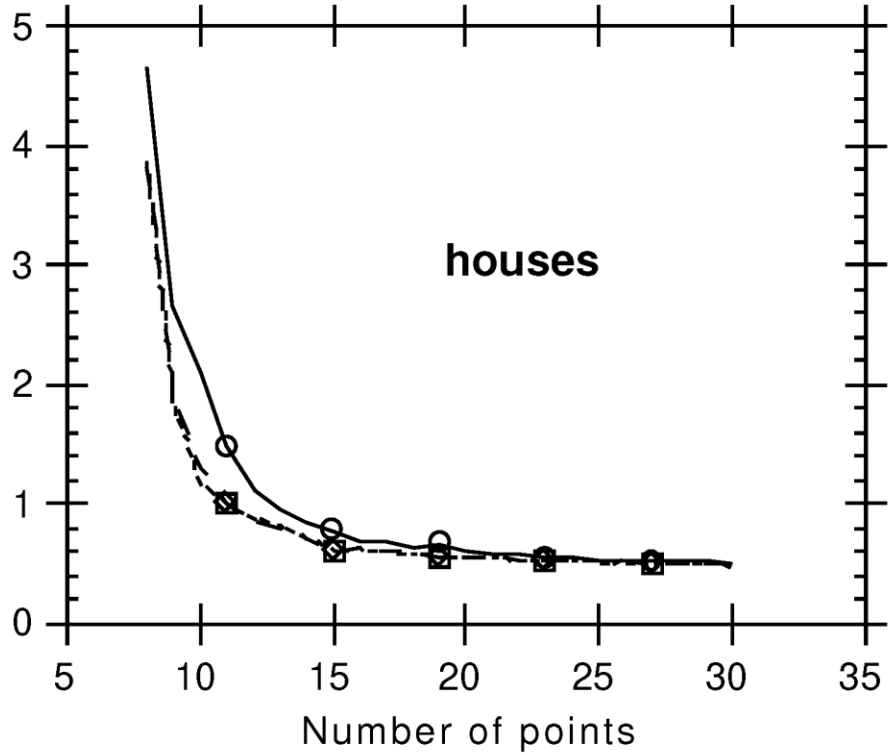
(v) Repeat 100 times for each n and collect statistics.

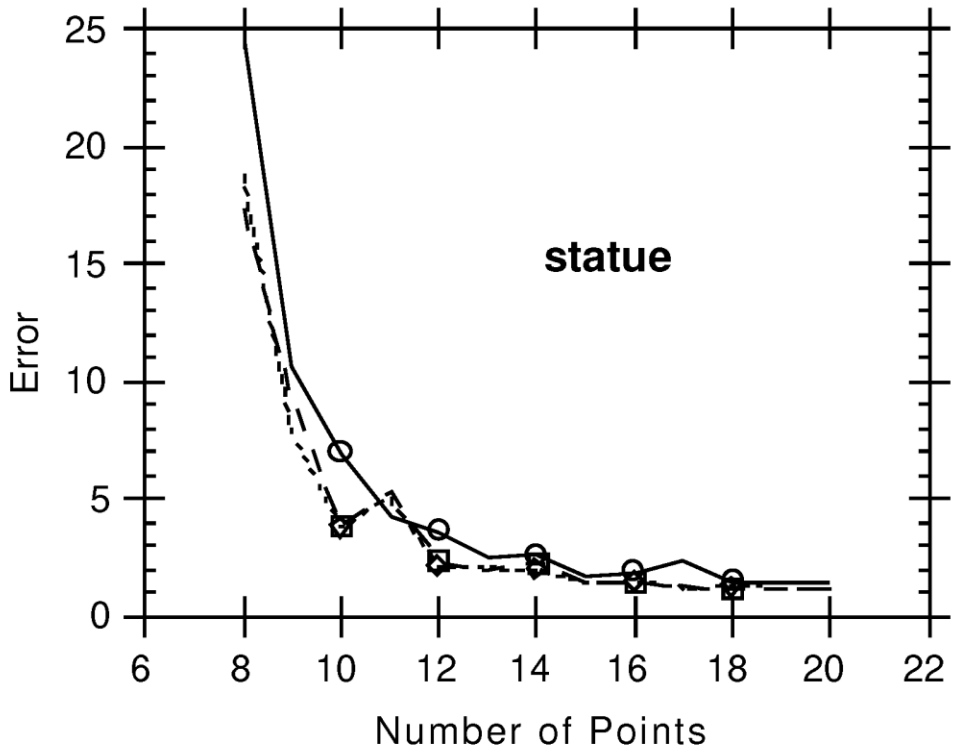
The error is defined as

$$\frac{1}{N}\sum_{i}^{N}(d(\mathbf{x}_i', \mathtt{F}\mathbf{x}_i) + d(\mathbf{x}_i, \mathtt{F}^\top\mathbf{x}_i'))$$

i.e Average symmetric epipolar distance.

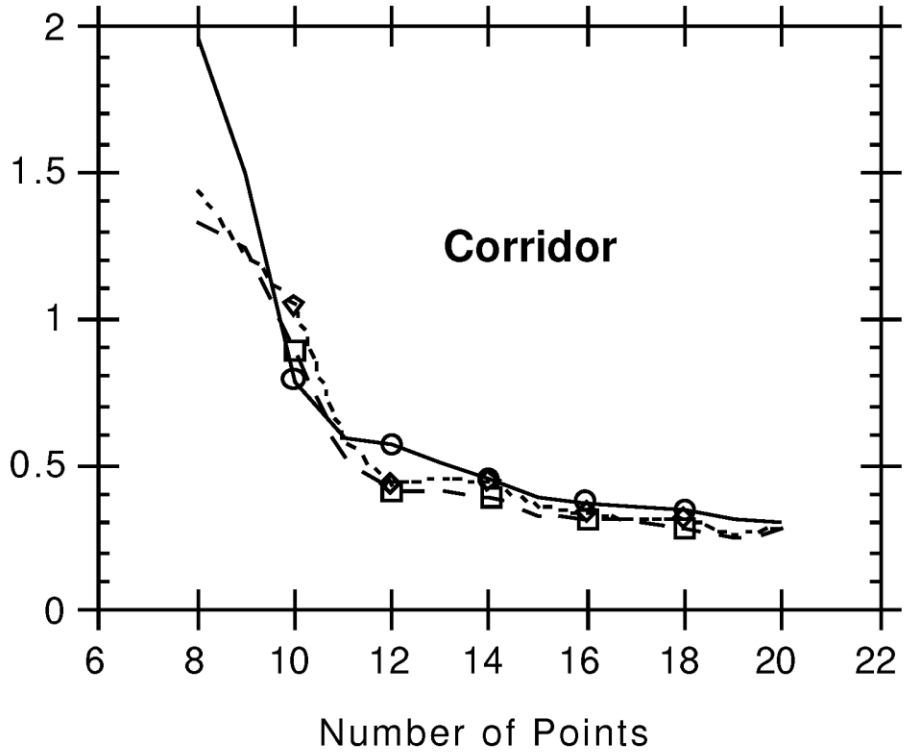
Results

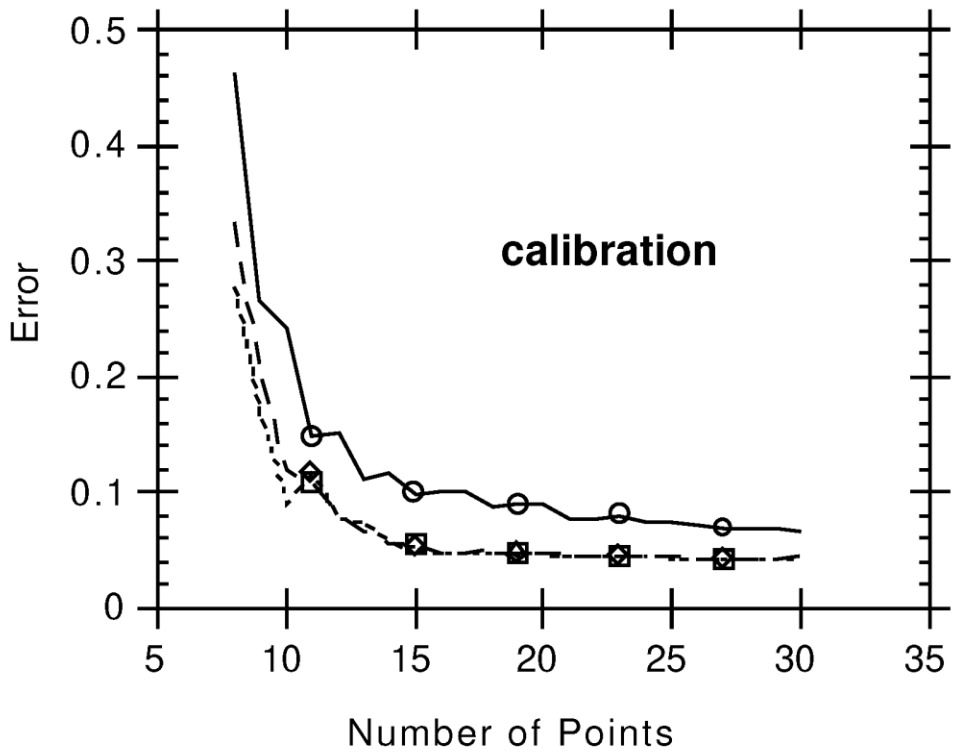




Normalized 8-point, geometric and algebraic error.

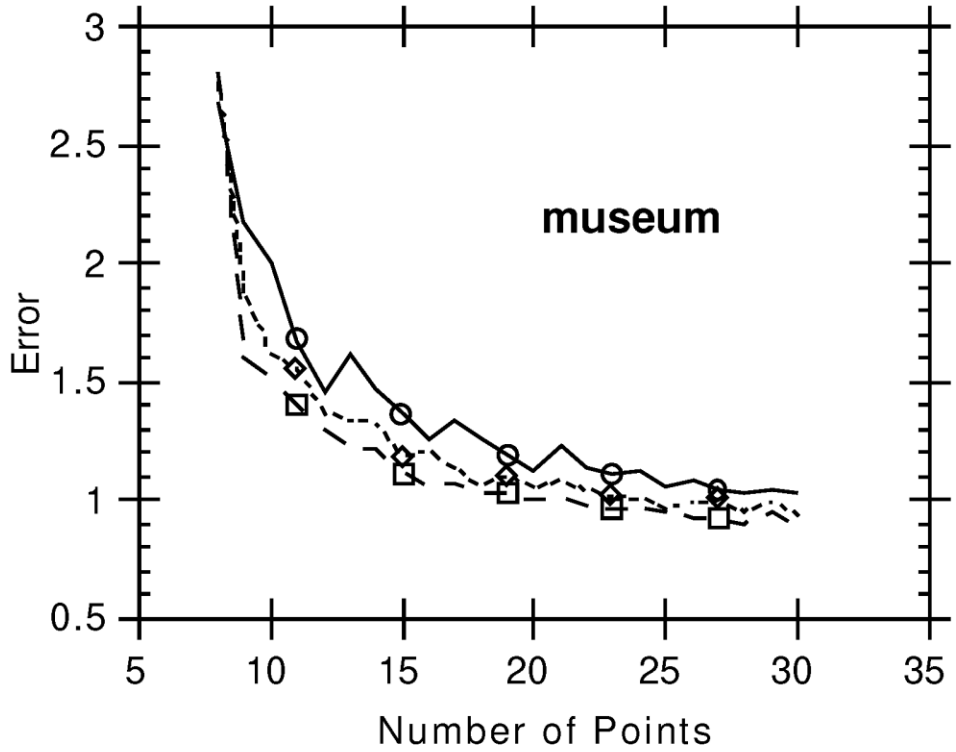
Results - continued





Normalized 8-point, geometric and algebraic error.

Results - continued



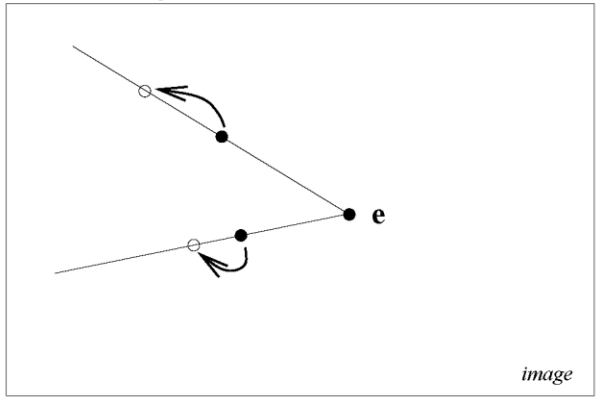
Normalized 8-point, geometric and algebraic error.

Special cases of F-computation

Using special motions can simplify the computation of the fundamental matrix.

Pure translation

- Can assume $\mathtt{P} = [\mathtt{I}|\mathbf{0}]$ and $\mathtt{P}' = [\mathtt{I}|\mathbf{t}]$.
- $\mathtt{F} = [\mathbf{t}]_\times$.
- $\mathtt{F}$ is skew-symmetric – has 2 dof.
- Being skew-symmetric, automatically has rank 2.



For a pure translation the epipole can be estimated from the image motion of two points.

Cameras with the same principal plane

- Principal plane of the camera is the third row of $\mathtt{P}$.
- Cameras have the same third row.
- Affine cameras - last row is $(0, 0, 0, 1)^\top$.

Simple correspondences exist :

$$(x', y', 0)\mathtt{F}(x, y, 0)^\top = 0$$

for any $(x', y', 0)^\top$ and $(x, y, 0)^\top$.

$\mathtt{F}$ has the following form :

$$\mathtt{F} = \begin{bmatrix} & & a \\ & & b \\ c & d & e \end{bmatrix}$$

Degeneracies

Correspondences are degenerate if they satisfy more than one $\mathtt{F}$.

$$\mathbf{x}_i{}^{\top}\mathtt{F}_1\mathbf{x}_i{}^{\top} = 0 \quad \text{and} \quad \mathbf{x}_i'\mathtt{F}_2\mathbf{x}_i = 0 \qquad\qquad (1 \leq i \leq n) \quad .$$

Points on a ruled quadric

(i) If all the points and the two camera centres lie on a ruled quadric, then there are three possible fundamental matrices.

(ii) points lie in a plane. The correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ lead to a 3-parameter family of possible fundamental matrices $\mathtt{F}$ (note, one of the parameters accounts for scaling the matrix so there is only a *two*-parameter family of homogeneous matrices).

(iii) Two cameras at the same point :

- The fundamental matrix does not exist.
- There is no such thing as an epipolar plane, and epipolar lines are not defined.
- Correspondences $\mathbf{x}'_i \leftrightarrow \mathbf{x}_i$ give at least a 2-parameter family of $\mathtt{F}$.

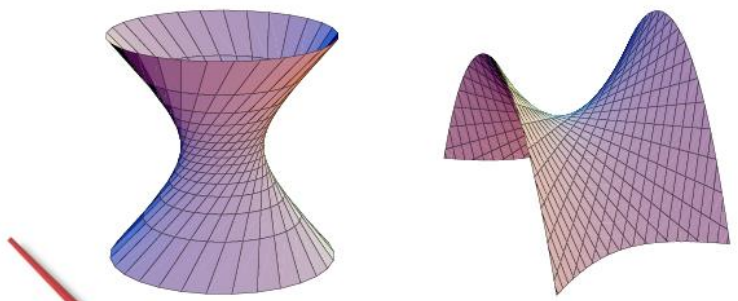


Fig. 3.3. **Ruled quadrics.** *Two examples of a hyperboloid of one sheet are given. These surfaces are given by equations* $\mathrm{X}^2 + \mathrm{Y}^2 = \mathrm{Z}^2 + 1$ *and* $\mathrm{XY} = \mathrm{Z}$ *respectively, and are projectively equivalent. Note that these two surfaces are made up of two sets of disjoint straight lines, and that each line from one set meets each line from the other set. The two quadrics shown here are projectively (though not affinely) equivalent.*

Automatic Estimation of Epipolar Geometry

Problem Statement

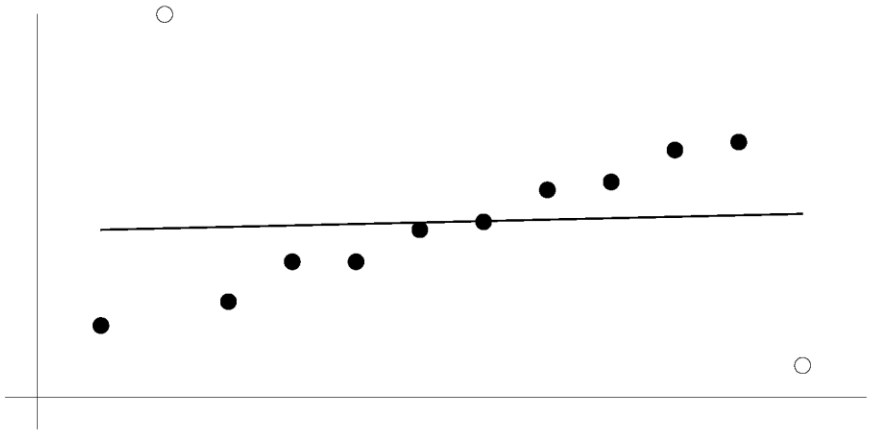
Given Image pair

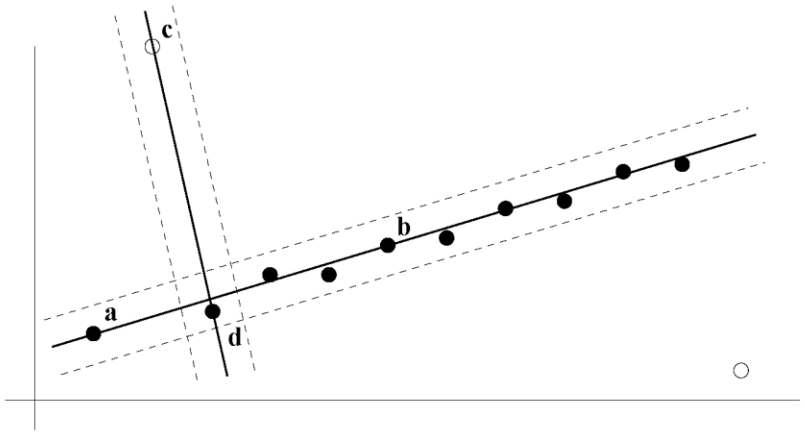
Find The fundamental matrix $\mathtt{F}$ and correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$.

- Compute image points
- Compute correspondences
- Compute epipolar geometry

Robust line estimation

Fit a line to 2D data containing outliers





There are two problems:

(i) a line fit to the data $\min_{\mathbf{l}} \sum_i d_{\perp i}^2$; and,

(ii) a classification of the data into inliers (valid points) and outliers.

RANdom SAmple Consensus (RANSAC)

[Fischler and Bolles, 1981]

Objective Robust fit of a model to a data set S which contains outliers.

Algorithm

(i) Randomly select a sample of s data points from S and instantiate the model from this subset.

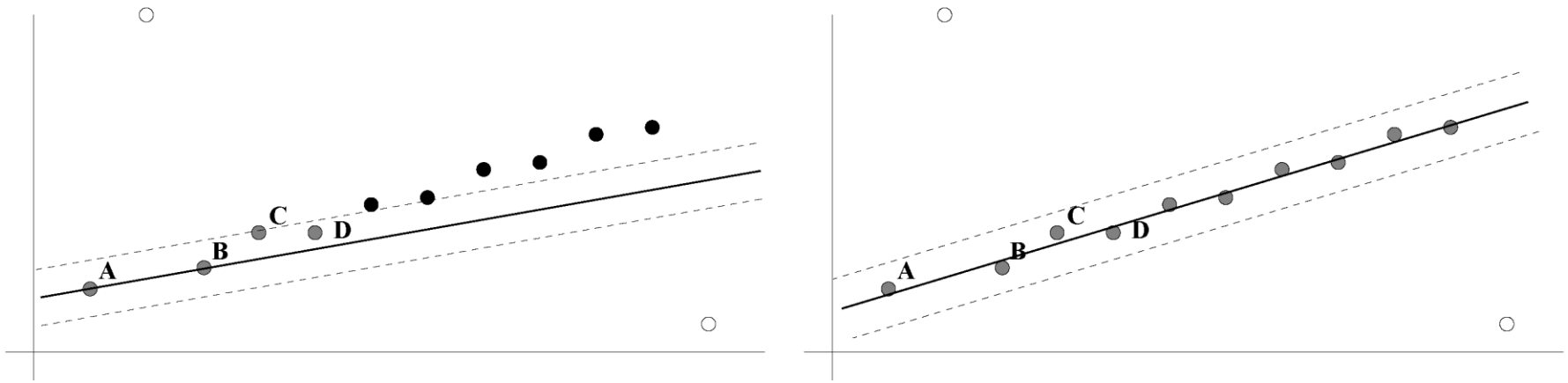
(ii) Determine the set of data points S_i which are within a distance threshold t of the model. The set S_i is the consensus set of the sample and defines the inliers of S.

(iii) If the size of S_i (the number of inliers) is greater than some threshold T, re-estimate the model using all the points in S_i and terminate.

(iv) If the size of S_i is less than T, select a new subset and repeat the above.

(v) After N trials the largest consensus set S_i is selected, and the model is re-estimated using all the points in the subset S_i.

Robust ML estimation



An improved fit by

- A better minimal set
- Robust MLE: instead of $\min_{\mathbf{l}} \sum_i d_{\perp i}^2$

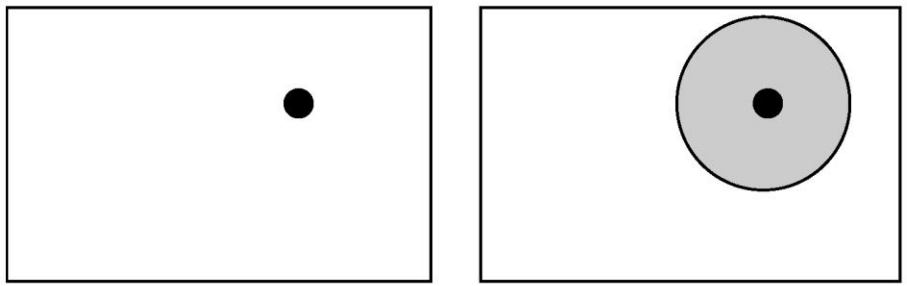
$$\min_{\mathbf{l}} \sum_i \gamma\left(d_{\perp i}\right) \qquad \text{with} \quad \gamma(e) = \begin{cases} e^2 & e^2 < t^2 \text{ inlier} \\ t^2 & e^2 \geq t^2 \text{ outlier} \end{cases}$$

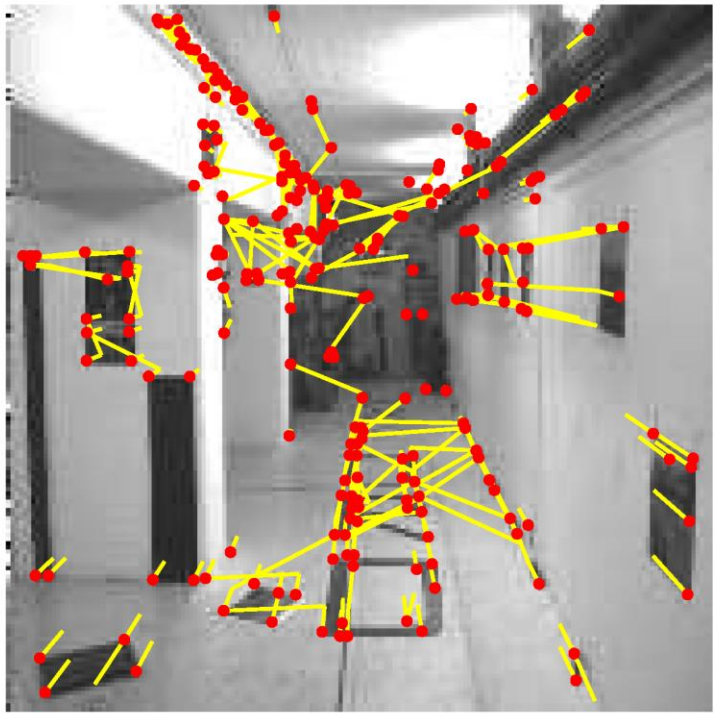
Feature extraction: “Corner detection”

Interest points [Harris]

- 100s of points per image

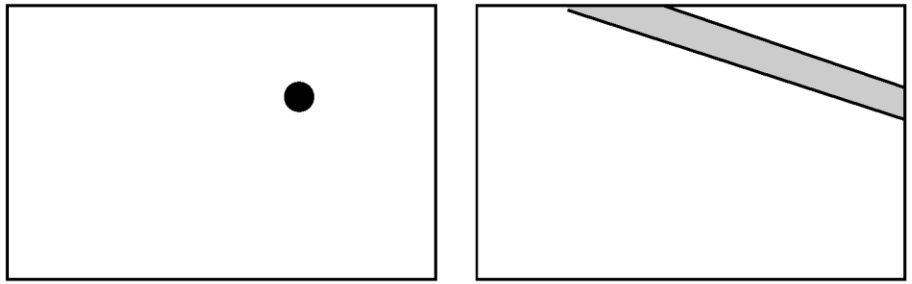
Correlation matching

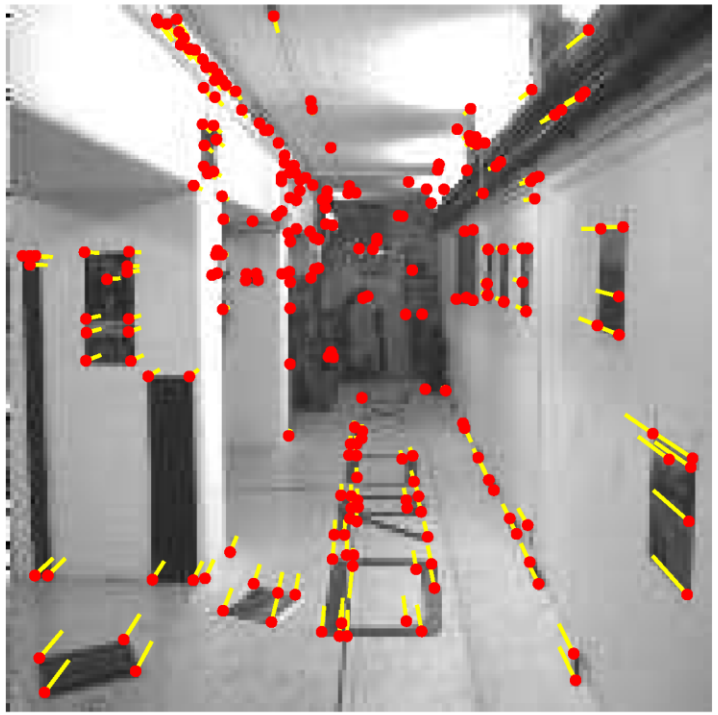




- Match each corner to most similar looking corner in the other image
- Many wrong matches (10-50%), but enough to compute the fundamental matrix.

Correspondences consistent with epipolar geometry





- Use RANSAC robust estimation algorithm
- Obtain correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ and $\mathtt{F}$
- Guided matching by epipolar line
- Typically: final number of matches is about 200-250, with distance error of $\sim$0.2 pixels.

Automatic Estimation of F and correspondences

Algorithm based on RANSAC [Torr]

(i) Interest points: Compute interest points in each image.

(ii) Putative correspondences: use cross-correlation and proximity.

(iii) RANSAC robust estimation:

Repeat

(a) Select random sample of 7 correspondences

(b) Compute F

(c) Measure support (number of inliers)

Choose the F with the largest number of inliers.

(iv) MLE: re-estimate F from inlier correspondences.

(v) Guided matching: generate additional matches.

Automatic Estimation of F and correspondences

Algorithm based on RANSAC [Torr]

(i) Interest points: Compute interest points in each image.

(ii) Putative correspondences: use cross-correlation and proximity.

(iii) RANSAC robust estimation:

Repeat

(a) Select random sample of 7 correspondences

(b) Compute F

(c) Measure support (number of inliers)

Choose the F with the largest number of inliers.

(iv) MLE: re-estimate F from inlier correspondences.

(v) Guided matching: generate additional matches.

How many samples?

For probability p of no outliers:

$$N = \log(1-p)/\log(1-(1-\epsilon)^s)$$

- N, number of samples
- s, size of sample set
- ϵ, proportion of outliers

e.g. for $p = 0.95$

Sample size	Proportion of outliers ϵ						
s	5%	10%	20%	25%	30%	40%	50%
2	2	2	3	4	5	7	11
3	2	3	5	6	8	13	23
4	2	3	6	8	11	22	47
5	3	4	8	12	17	38	95
6	3	4	10	16	24	63	191
7	3	5	13	21	35	106	382
8	3	6	17	29	51	177	766

Adaptive RANSAC

- $N = \infty$, sample_count$= 0$.
- While $N >$ sample_count Repeat
  - Choose a sample and count the number of inliers.
  - Set $\epsilon = 1 - (\text{number of inliers})/(\text{total number of points})$
  - Set N from ϵ with $p = 0.99$.
  - Increment the sample_count by one.
- Terminate.

e.g. for a sample size of 4

Number of inliers	1 - ϵ	Adaptive N
6	2%	20028244
10	3%	2595658
44	16%	6922
58	21%	2291
73	26%	911
151	56%	43

Part 2 : Three-view and Multiple-view Geometry

Computing a Metric Reconstruction

Two View Reconstruction Ambiguity

Given: image point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$,
compute a reconstruction:

$$\{\mathtt{P},\ \mathtt{P}',\ \mathbf{X}_i\} \quad \text{with} \quad \mathbf{x}_i = \mathtt{P}\mathbf{X}_i \quad \mathbf{x}'_i = \mathtt{P}'\mathbf{X}_i$$

Ambiguity

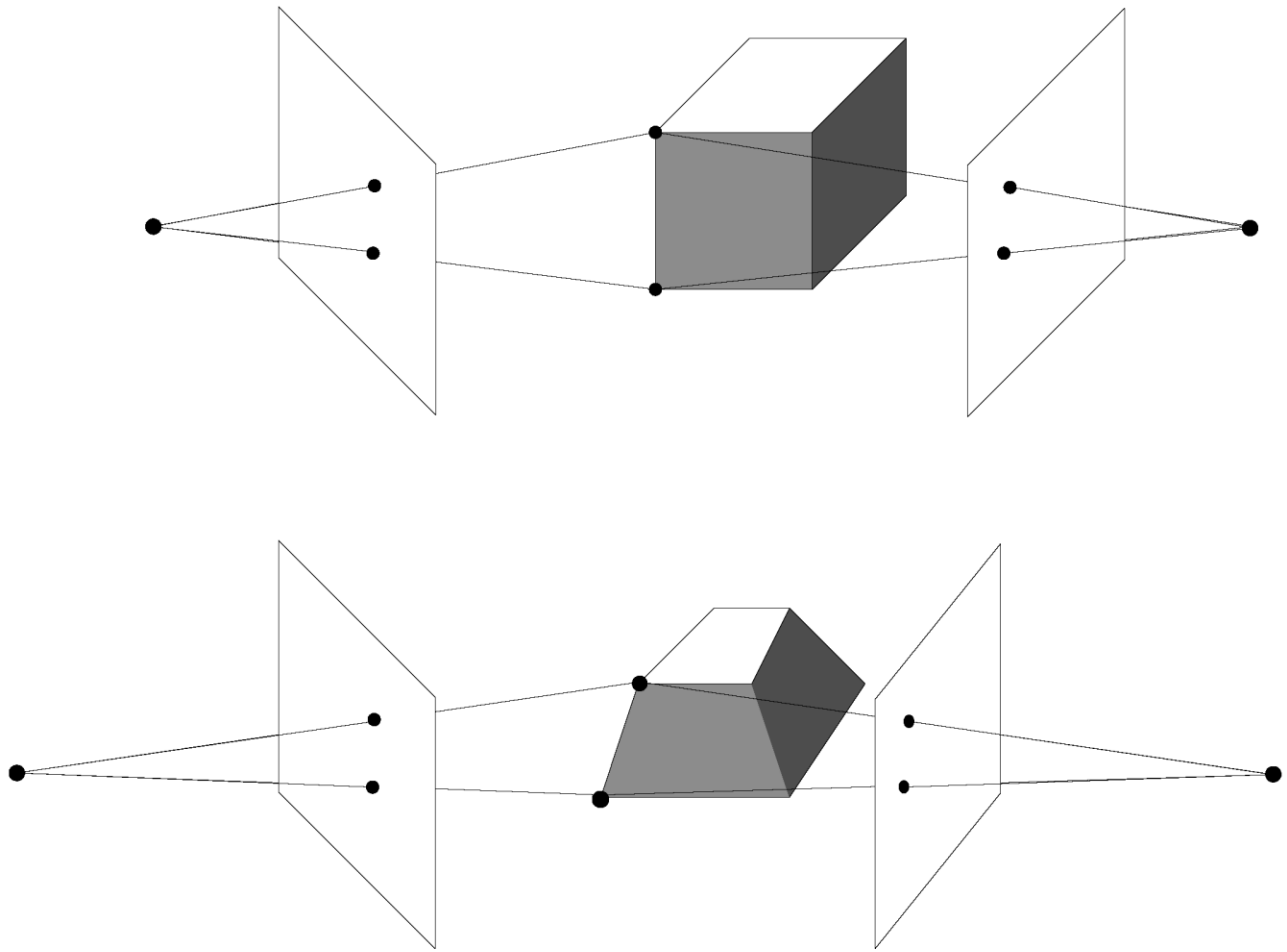
$$\mathbf{x}_i \ = \ \mathtt{P}\mathbf{X}_i = \mathtt{P}\ \mathtt{H}(\mathtt{H})^{-1}\ \mathbf{X}_i = \tilde{\mathtt{P}}\tilde{\mathbf{X}}_i$$

$$\mathbf{x}'_i \ = \ \mathtt{P}'\mathbf{X}_i = \mathtt{P}'\ \mathtt{H}(\mathtt{H})^{-1}\ \mathbf{X}_i = \tilde{\mathtt{P}}'\tilde{\mathbf{X}}_i$$

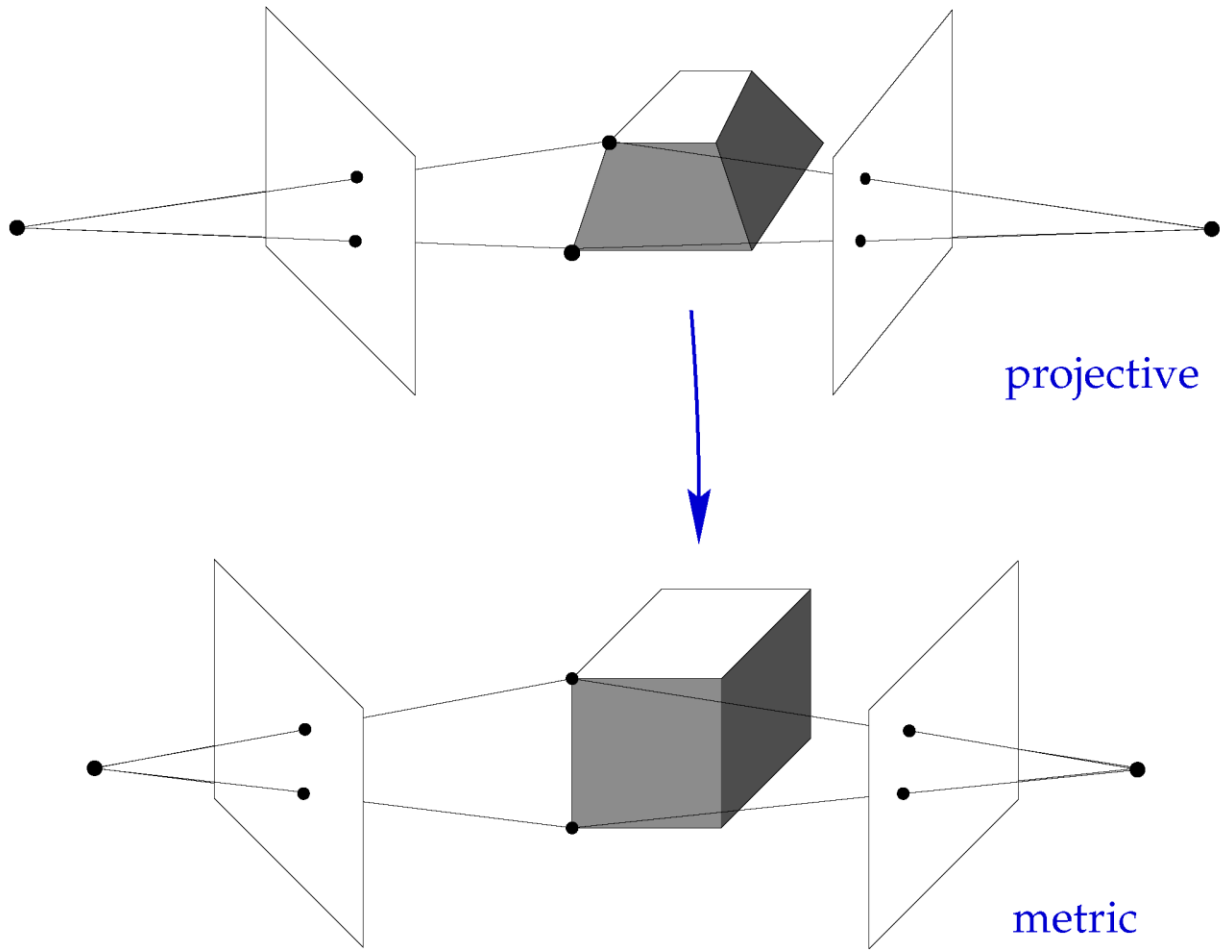
$\{\tilde{\mathtt{P}},\ \tilde{\mathtt{P}}',\ \tilde{\mathbf{X}}_i\}$ is an equivalent Projective Reconstruction.

Reconstruction from two views

Given only image points and their correspondence,
what can be determined?



Metric Reconstruction



Correct: angles, length ratios.

Algebraic Representation of Metric Reconstruction

Compute $\mathtt{H}$

$$\{\mathtt{P}^1,\ \mathtt{P}^2, ..., \mathtt{P}^m,\ \mathbf{X}_i\} \quad \xrightarrow[\mathtt{H}]{} \quad \{\mathtt{P}^1_{\mathrm{M}},\ \mathtt{P}^2_{\mathrm{M}}, ..., \mathtt{P}^m_{\mathrm{M}},\ \mathbf{X}^{\mathrm{M}}_i\}$$

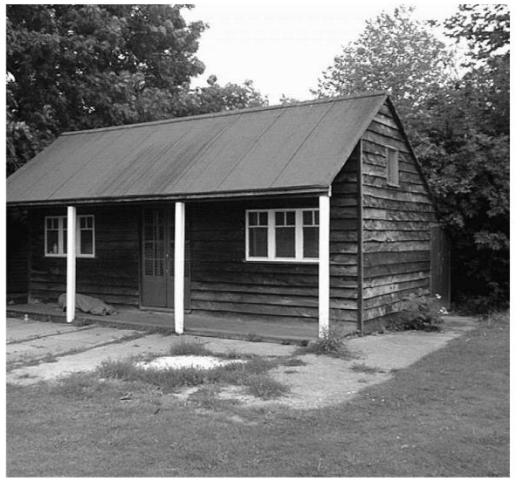
Projective Reconstruction Metric Reconstruction

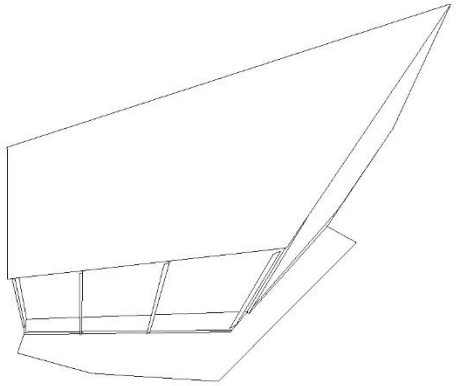
- Remaining ambiguity is rotation (3), translation (3) and scale (1).
- Only 8 parameters required to rectify entire sequence $(15 - 7 = 8)$.

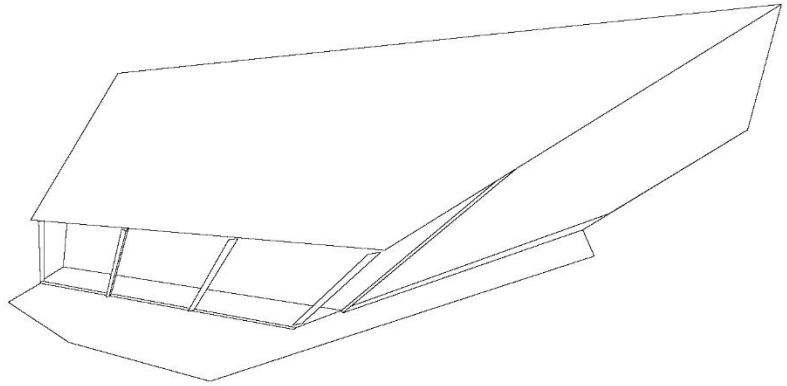
How?

- Calibration points: position of 5 scene points.
- Scene geometry: e.g. parallel lines/planes, orthogonal lines/planes, length ratios.
- Auto-calibration: e.g. camera aspect ratio constant for sequence.

Projective Reconstruction

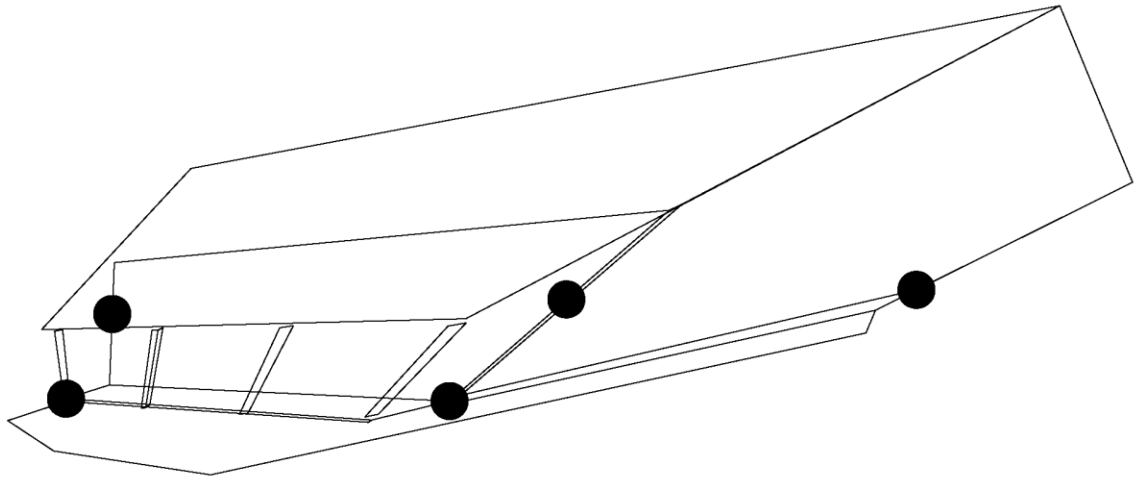


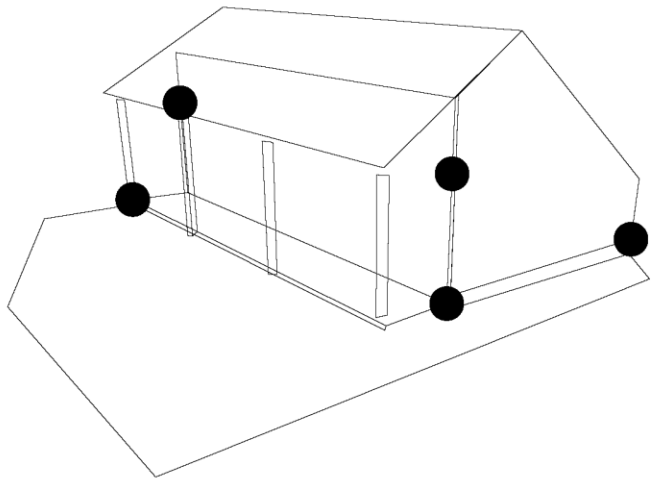




Direct Metric Reconstruction

Use 5 or more 3D points with known Euclidean coordinates to determine $\mathtt{H}$





Stratified Reconstruction

Given a projective reconstruction $\{\mathtt{P}^j, \mathbf{X}_i\}$, compute a metric reconstruction via an intermediate affine reconstruction.

(i) affine reconstruction: Determine the vector $\mathbf{p}$ which defines π_∞. An affine reconstruction is obtained as $\{\mathtt{P}^j\mathtt{H}_\mathrm{P}, \mathtt{H}_\mathrm{P}^{-1}\mathbf{X}_i\}$ with

$$\mathtt{H}_\mathrm{P} = \begin{bmatrix} \mathtt{I} & \mathbf{0} \\ -\mathbf{p}^\top & 1 \end{bmatrix}$$

(ii) Metric reconstruction: is obtained as $\{\mathtt{P}_\mathrm{A}^j\mathtt{H}_\mathrm{A}, \mathtt{H}_\mathrm{A}^{-1}\mathbf{X}_{\mathrm{A}_i}\}$ with

$$\mathtt{H}_\mathrm{A} = \begin{bmatrix} \mathtt{K} & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{bmatrix}$$

Stratified Reconstruction

- Start with a projective reconstruction.
- Find transformation to upgrade to affine reconstruction.
  - Equivalent to finding the plane at infinity.
- Find transformation to upgrade to metric (Euclidean) reconstruction.
  - Equivalent to finding the “absolute conic”
- Equivalent to camera calibration
  - If camera calibration is known then metric reconstruction is possible.
  - Metric reconstruction implies knowledge of angles – camera calibration.

Anatomy of a 3D projective transformation

- General 3D projective transformation represented by a 4×4 matrix.

$$\mathtt{H} = \begin{bmatrix} s\mathtt{RK} & \mathbf{t} \\ \mathbf{v}^\top & 1 \end{bmatrix} = \begin{bmatrix} s\mathtt{R} & \mathbf{t} \\ & 1 \end{bmatrix} \begin{bmatrix} \mathtt{K} & \\ & 1 \end{bmatrix} \begin{bmatrix} \mathtt{I} & \\ \mathbf{v}^\top & 1 \end{bmatrix}$$
$$= \text{metric} \times \text{affine} \times \text{projective}$$

Stratified reconstruction ...

(i) Apply the transformations one after the other :

- Projective transformation – reduce to affine ambiguity

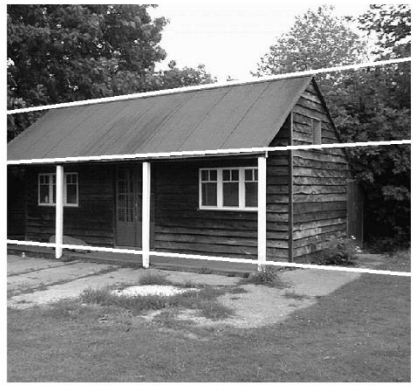
$$\begin{bmatrix} \mathtt{I} & \\ \mathbf{v}^\top & 1 \end{bmatrix}$$

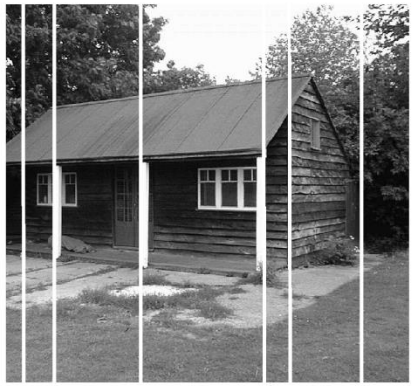
- Affine transformation – reduce to metric ambiguity

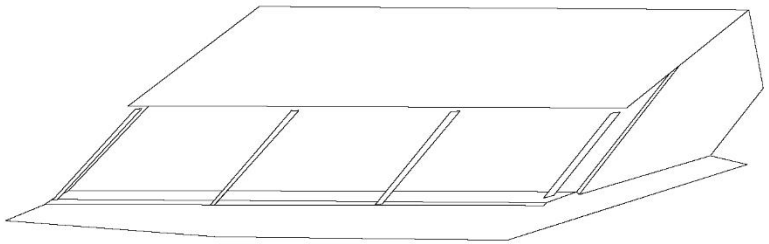
$$\begin{bmatrix} \mathtt{K} & \\ & 1 \end{bmatrix}$$

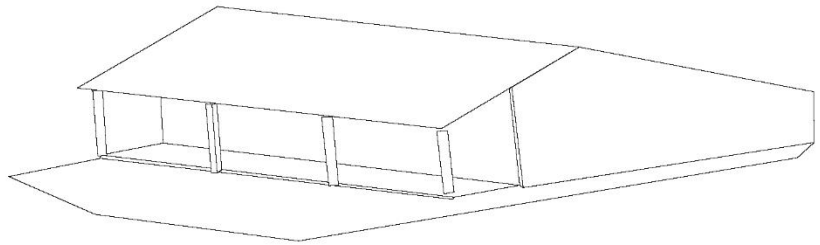
- Metric ambiguity of scene remains

Reduction to affine









Affine reduction using scene constraints - parallel lines

Reduction to affine

Other scene constraints are possible :

- Ratios of distances of points on line (e.g. equally spaced points).
- Ratios of distances on parallel lines.

Points lie in front of the viewing camera.

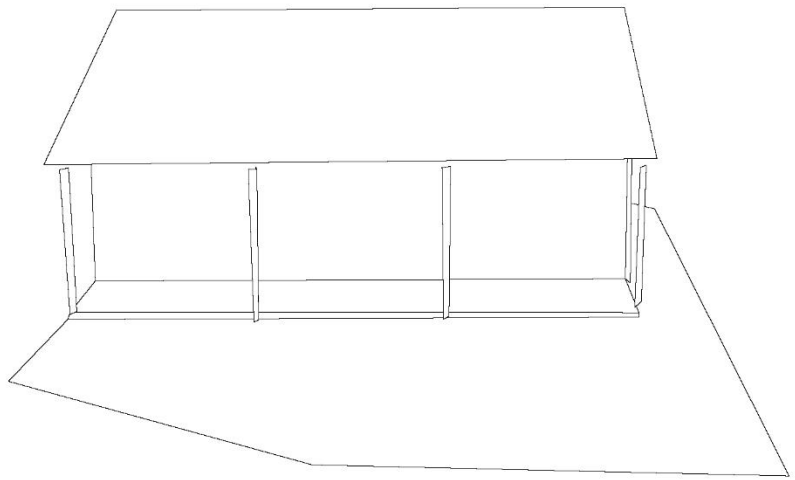
- Constrains the position of the plane at infinity.
- Linear-programming problem can be used to set bounds on the plane at infinity.
- Gives so-called “quasi-affine” reconstruction.
- Reference : Hartley-Azores.

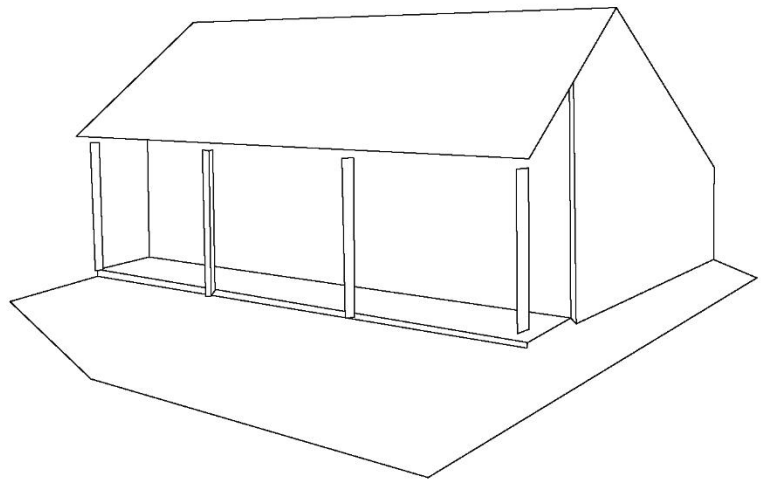
Reduction to affine . . .

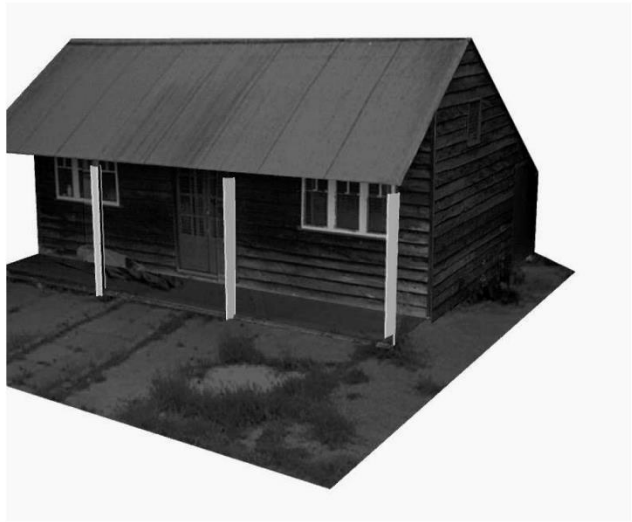
Common calibration of cameras.

- With 3 or more views, one can find (in principle) the position of the plane at infinity.
- Iteration over the entries of projective transform : $\begin{bmatrix} \mathtt{I} & \\ \mathbf{v}^\top & 1 \end{bmatrix}$.
- Not always reliable.
- Generally reduction to affine is difficult.

Metric Reconstruction



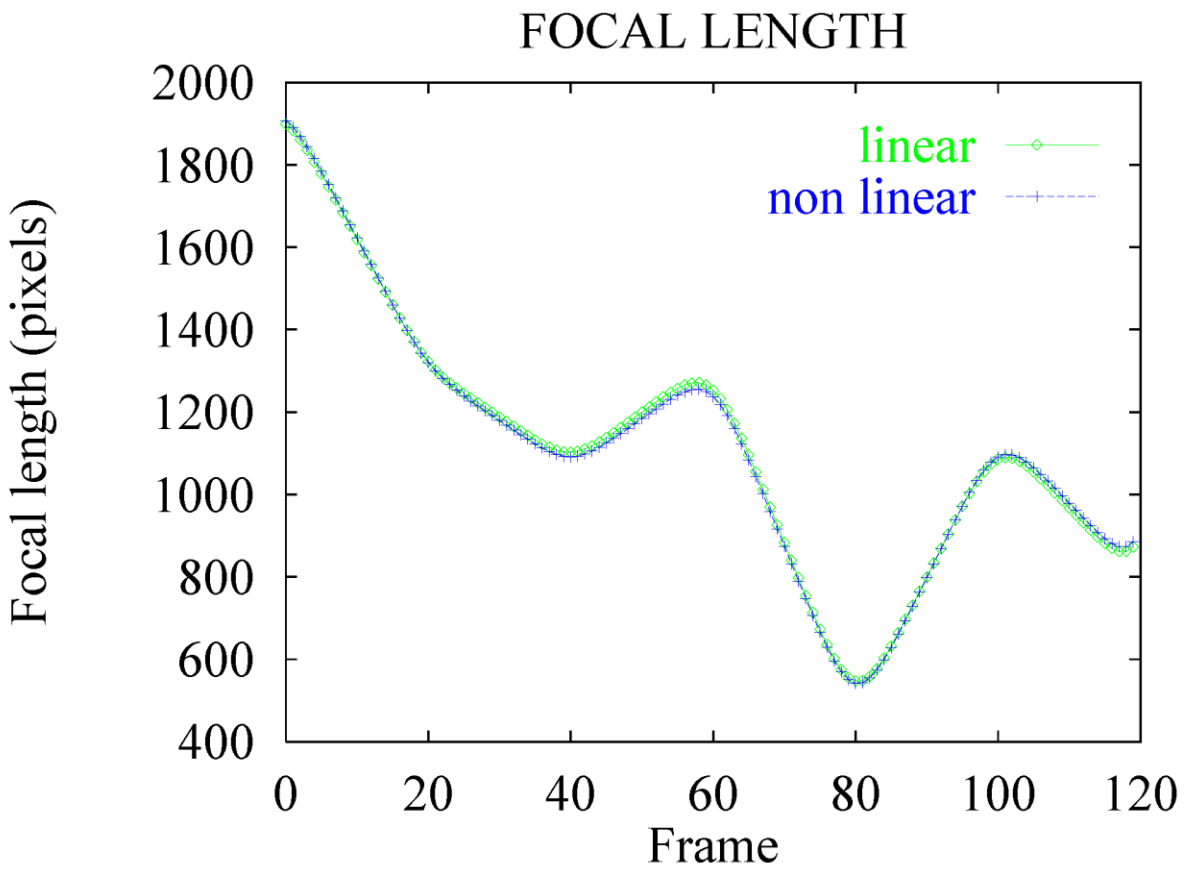




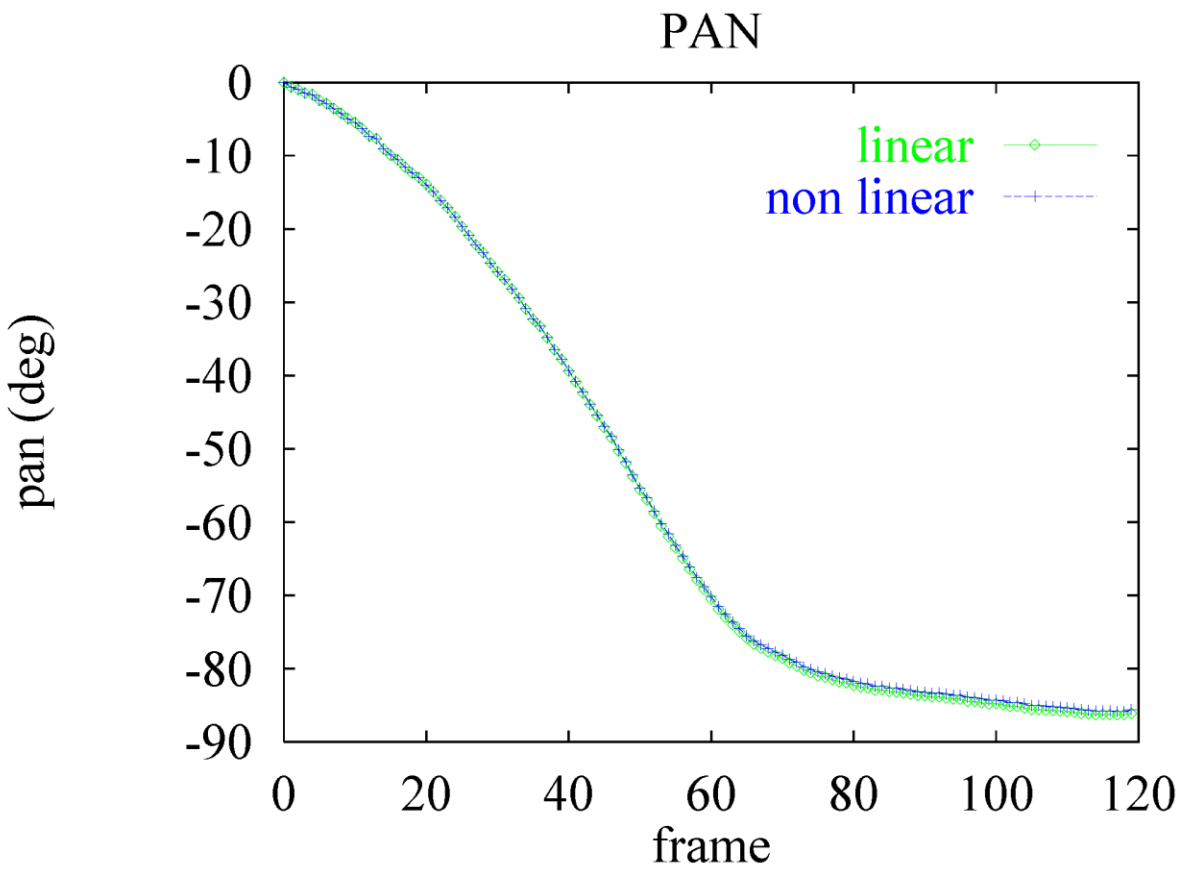
Nice Video

Self calibration

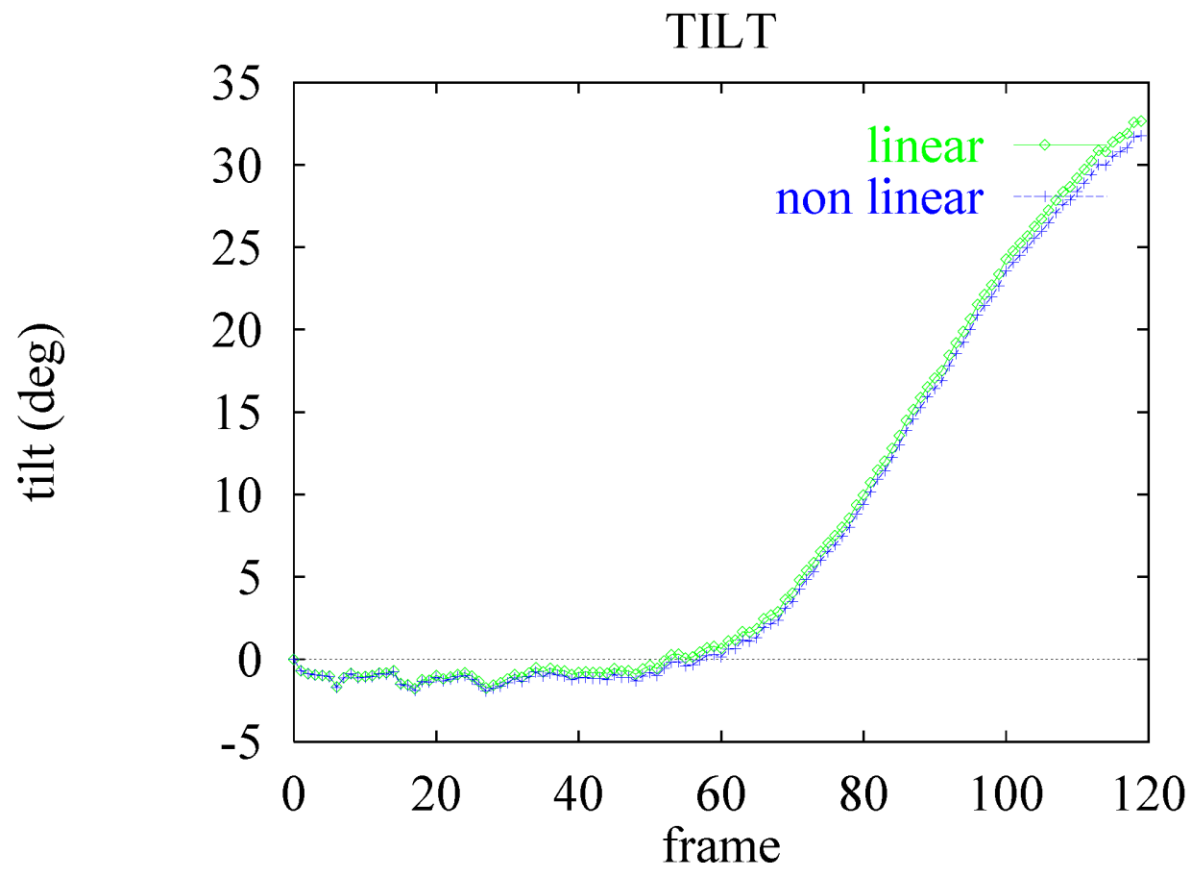
Nice Calibration Results – Focal Length



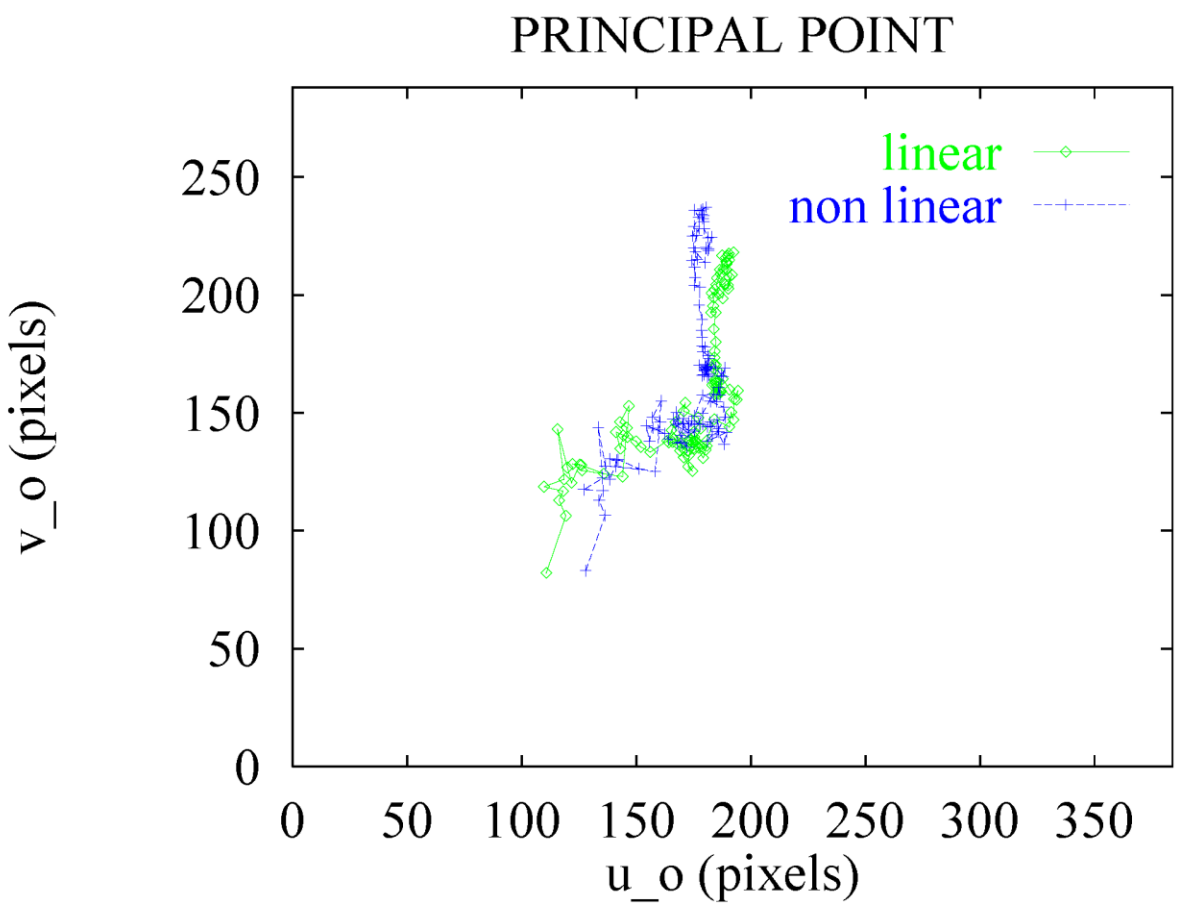
Nice Calibration Results – Pan angle



Nice Calibration Results – Tilt angle

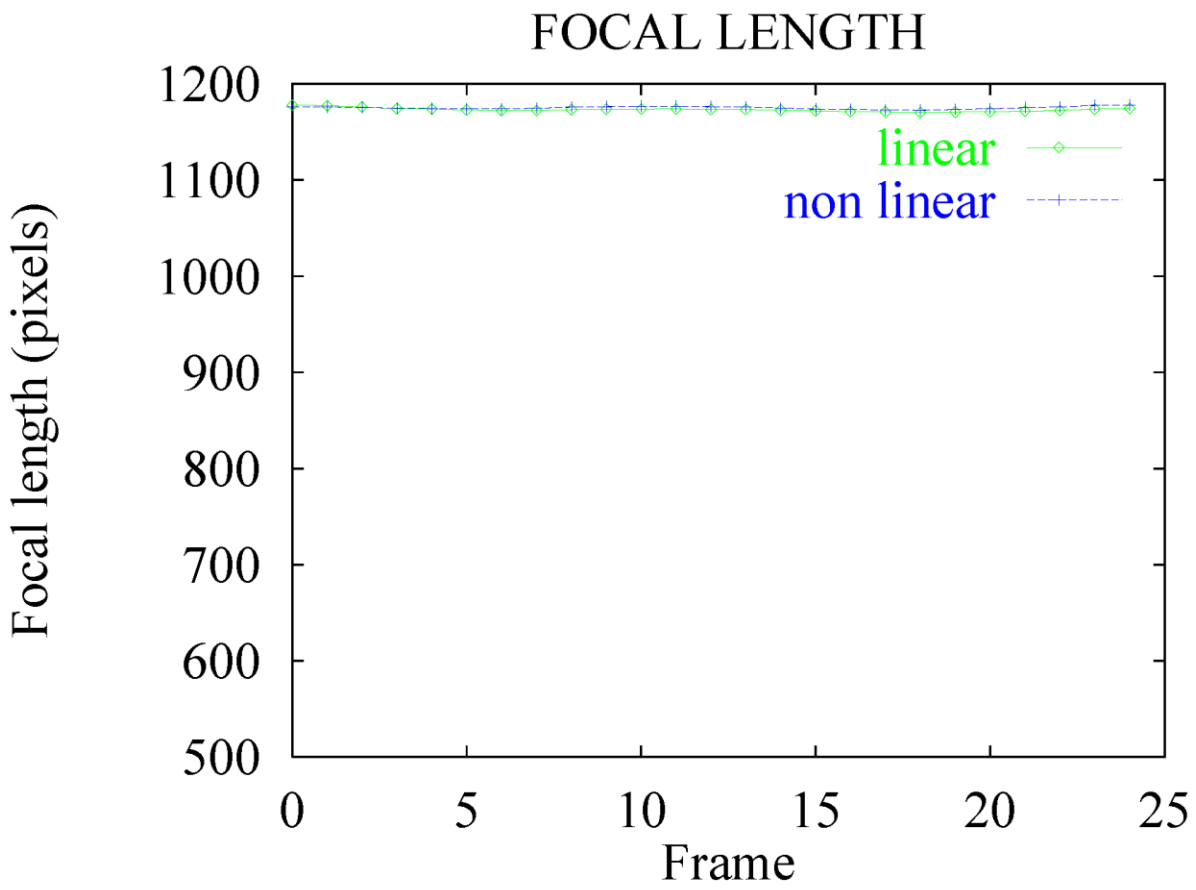


Nice Calibration Results – Principal point

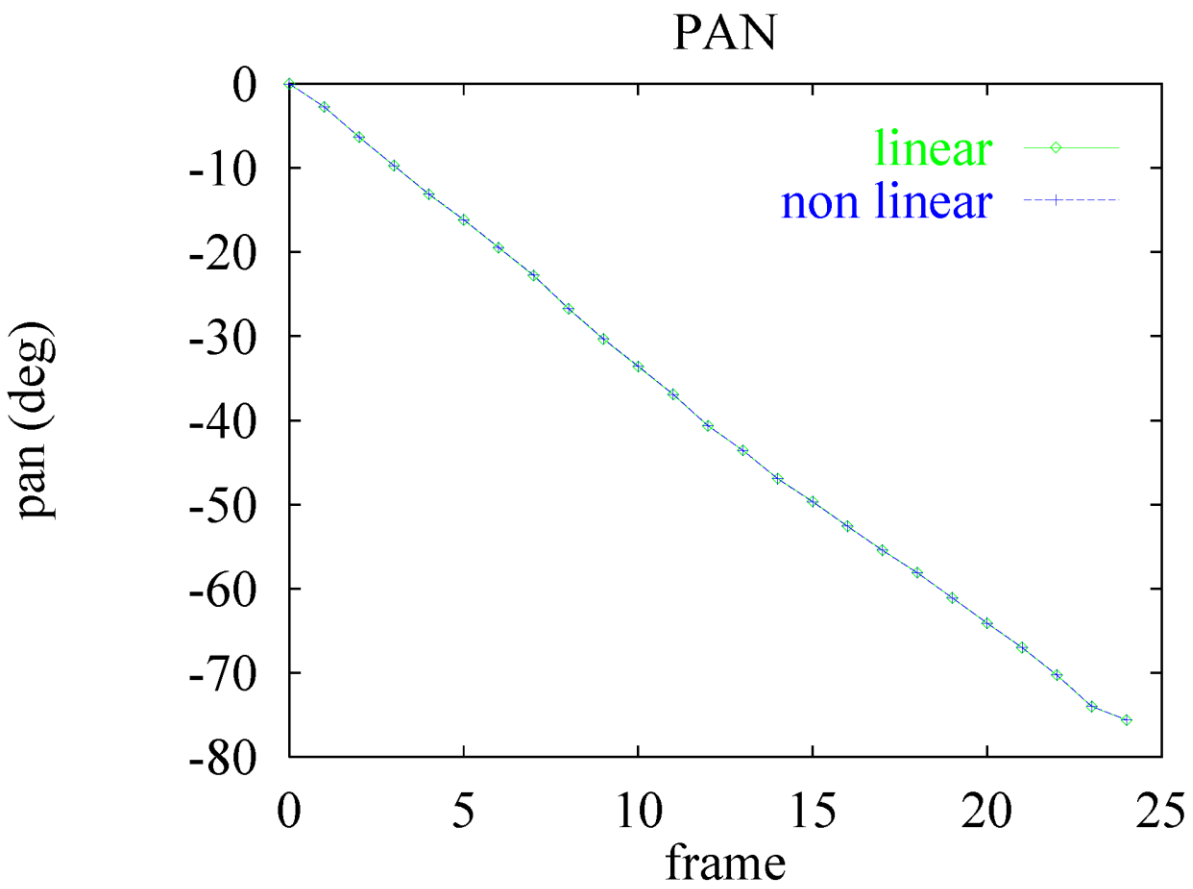


Keble College Video

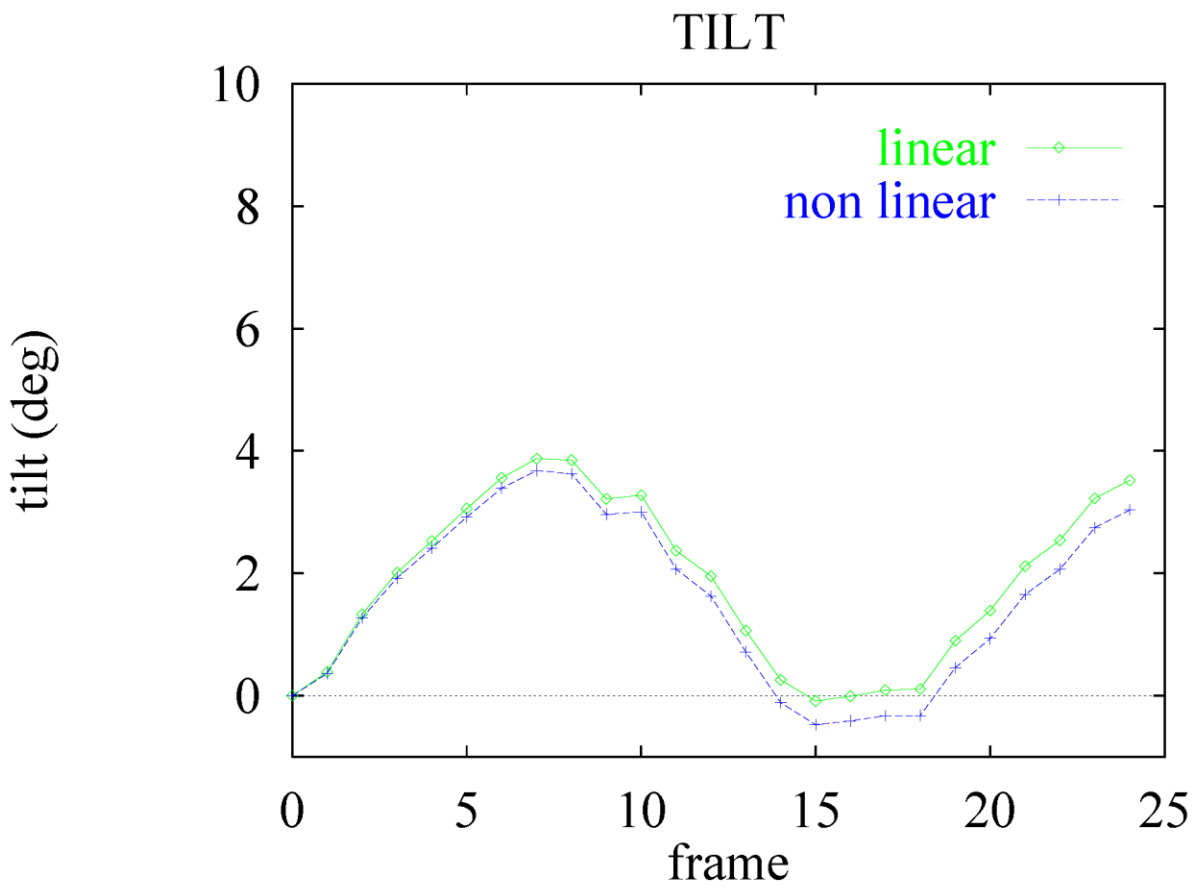
Keble Calibration Results – Focal Length



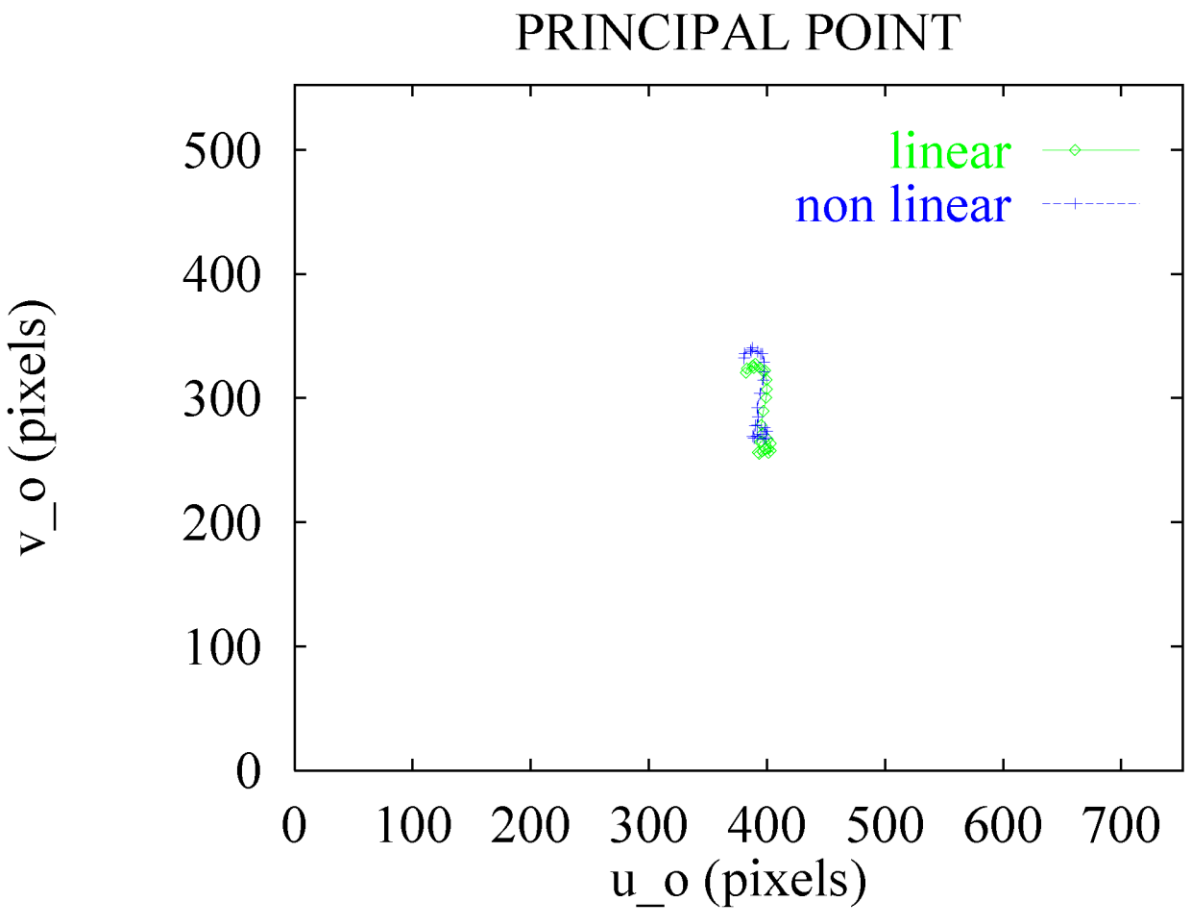
Keble Calibration Results – Pan angle



Keble Calibration Results – Tilt angle



Keble Calibration Results – Focal Length



The Trifocal Tensor

The Trifocal Tensor

(i) Defined for three views.

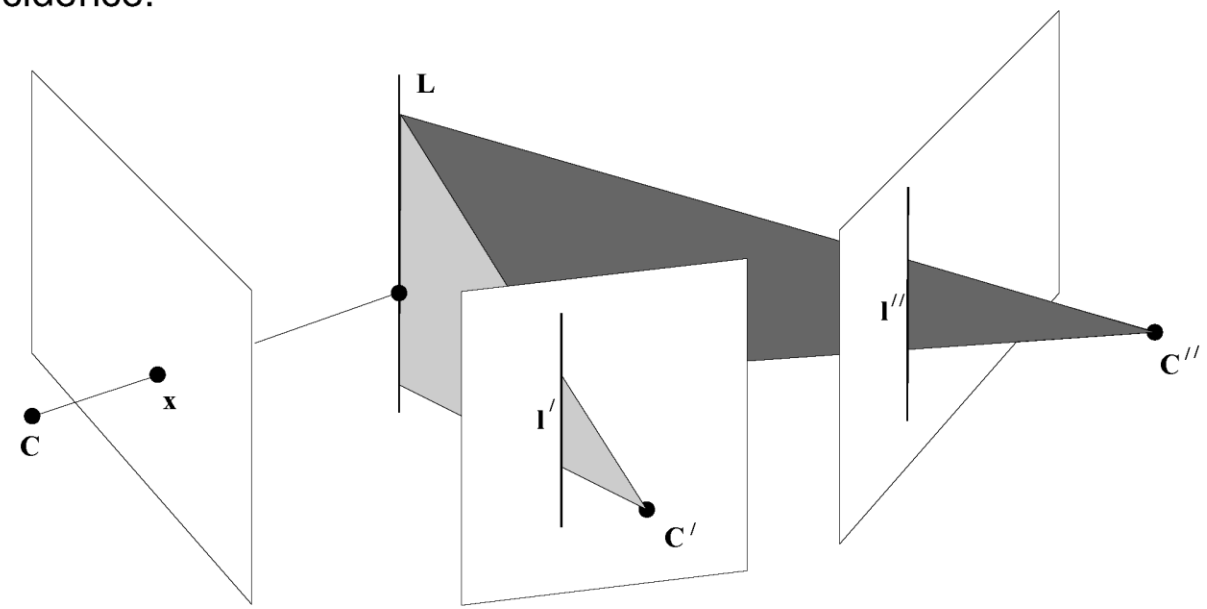
(ii) Plays a similar rôle to Fundmental matrix for two views.

(iii) Unlike fundamental matrix, trifocal tensor also relates lines in three views.

(iv) Mixed combinations of lines and points are also related.

Geometry of three views

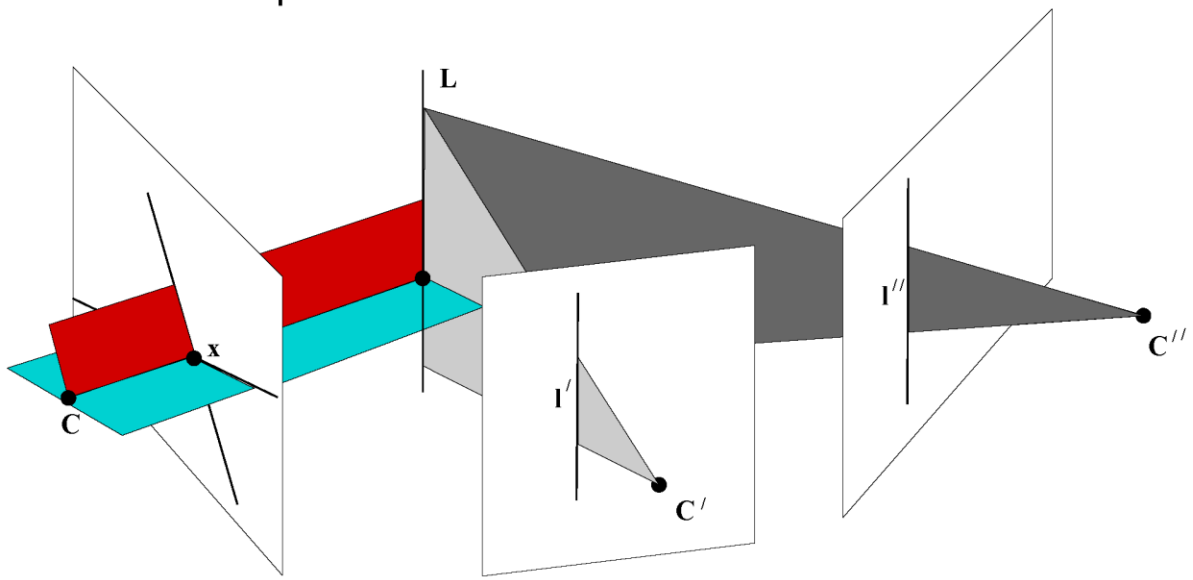
Point-line-line incidence.



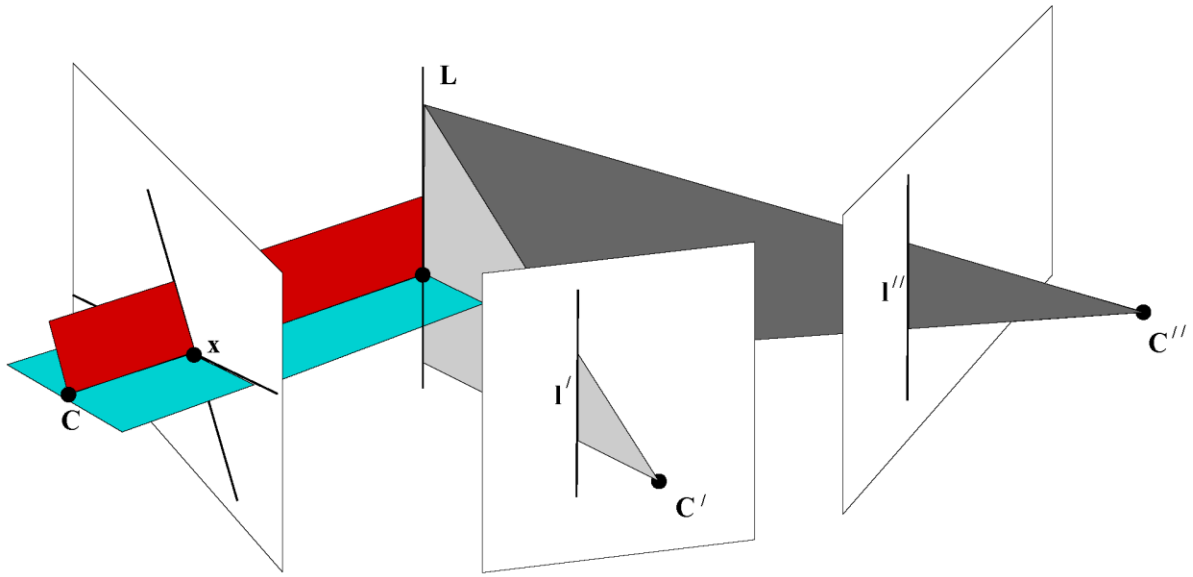
- Correspondence $\mathbf{x} \leftrightarrow \mathbf{l}' \leftrightarrow \mathbf{l}''$

Geometry of three views . . .

- Let $\mathbf{l}^{(1)}$ and $\mathbf{l}^{(2)}$ be two lines that meet in $\mathbf{x}$.
- General line $\mathbf{l}$ back-projects to a plane $\mathbf{l}^\top \mathtt{P}$.
- Four plane are $\mathbf{l}^{(1)\top}\mathtt{P}$, $\mathbf{l}^{(2)\top}\mathtt{P}'$, $\mathbf{l}'^\top\mathtt{P}'$ and $\mathbf{l}''^\top\mathtt{P}''$
- The four planes meet in a point.



Basic Trifocal constraint . . .



- Let $l_r^{(1)}$ and $l_s^{(2)}$ be two lines that meet in point x^i.
- The four lines $l_r^{(1)}$, $l_s^{(2)}$, l'_j and l''_k back-project to planes in space.
- The four planes meet in a point.

The trifocal relationship

Four planes meet in a point means determinant is zero.

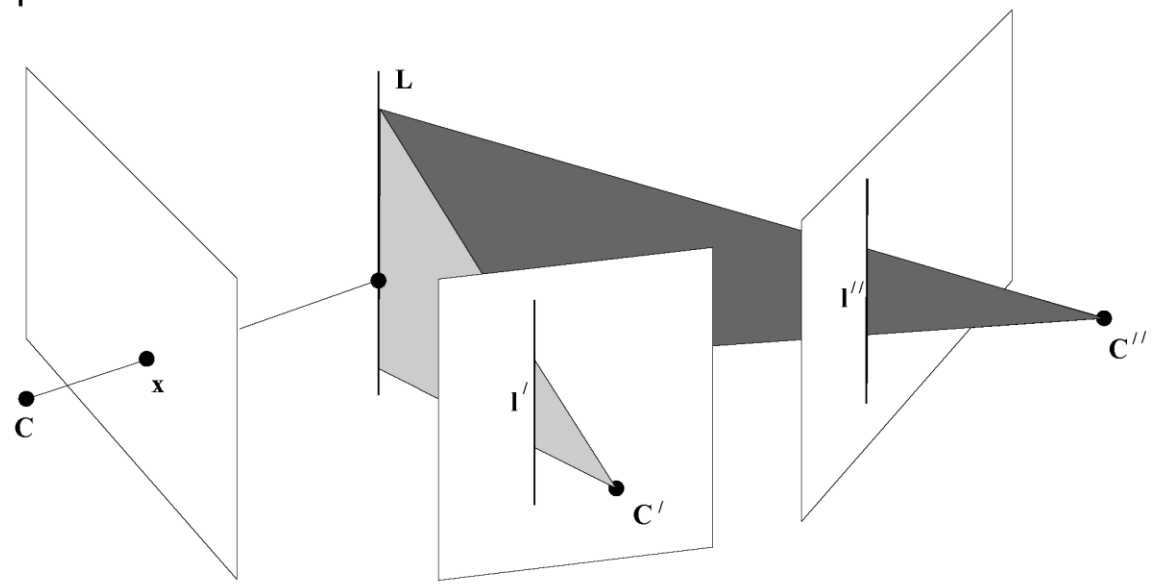
$$\det \begin{bmatrix} \mathbf{l}^{(1)\top}\mathtt{P} \\ \mathbf{l}^{(2)\top}\mathtt{P} \\ \mathbf{l}'^{\top}\mathtt{P}' \\ \mathbf{l}''^{\top}\mathtt{P}'' \end{bmatrix} = 0$$

- This is a linear relationship in the line coordinates.
- Also (less obviously) linear in the entries of the point $\mathbf{x} = \mathbf{l}^{(1)} \times \mathbf{l}^{(2)}$.

This is the trifocal tensor relationship.

Basic Trifocal constraint

Basic relation is a point-line-line incidence.



- Point x^i in image 1
- lines l'_j and l''_k in images 2 and 3.

Summary of incidence relations

(i) Point in first view, lines in second and third

$$l'_j \, l''_k \, T_i^{jk} x^i = 0$$

(ii) Point in first view, point in second and line in third

$$x^i x'^j l''_k \, \epsilon_{jpr} \mathcal{T}_i^{pk} = 0_r$$

(iii) Point in first view, line in second and point in third

$$x^i \, l'_j \, x''^k \, \epsilon_{kqs} \mathcal{T}_i^{jq} = 0_s$$

(iv) Point in three views

$$x^i x'^j x''^k \, \epsilon_{jpr} \epsilon_{kqs} \mathcal{T}_i^{pq} = 0_{rs}$$

Computation of the trifocal tensor

Summary of relations

Other point or line correspondences also yield constraints on T.

Correspondence	Relation	number of equations
three points	$x^i\, x'^j\, x''^k \epsilon_{jqu}\epsilon_{krv} T_i^{qr} = 0_{uv}$	4
two points, one line	$x^i\, x'^j\, l''_r \epsilon_{jqu} T_i^{qr} = 0_u$	2
one point, two lines	$x^i\, l'_q\, l''_r T_i^{qr} = 0$	1
three lines	$l_p\, l'_q\, l''_r \epsilon^{piw} T_i^{qr} = 0^w$	2

Trilinear Relations

Solving the equations

Given 26 equations we can solve for the 27 entries of T.

- Need 7 point correspondences
- or 13 line correspondences
- or some mixture.

Total set of equations has the form

$$\mathbf{Et} = \mathbf{0}$$

- With 26 equations find an exact solution.
- With more equations, least-squares solution.

Solving the equations ...

- Solution :
  - Take the SVD : $\mathtt{E} = \mathtt{UDV}^\top$.
  - Solution is the last column of $\mathtt{V}$ corresponding to smallest singular value).
  - Minimizes $||\mathtt{E}\mathbf{t}||$ subject to $||\mathbf{t}|| = 1$.
- Normalization of data is *essential*.

Automatic Estimation of a Projective Reconstruction for a Sequence

Outline

first frame of video

(i) Projective reconstruction: 2-views, 3-views, N-views

(ii) Obtaining correspondences over N-views

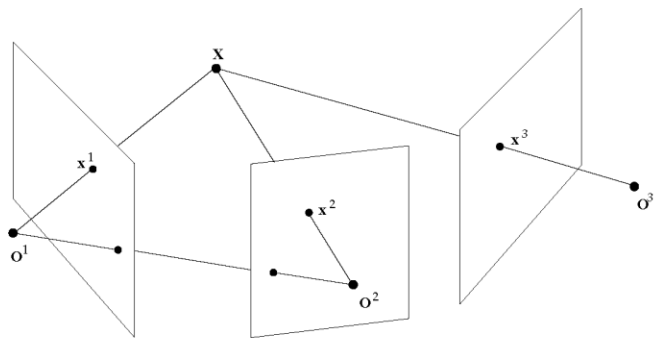
Reconstruction from three views

Given: image point correspondences $\mathbf{x}_i^1 \leftrightarrow \mathbf{x}_i^2 \leftrightarrow \mathbf{x}_i^3$,
compute a projective reconstruction:

$$\{\mathtt{P}^1,\ \mathtt{P}^2,\ \mathtt{P}^3,\ \mathbf{X}_i\} \quad \text{with} \quad \mathbf{x}_i^j = \mathtt{P}^j\,\mathbf{X}_i$$

What is new?

verify correspondences



- 3-view tensor: the trifocal tensor
- Compute from 6 image point correspondences.
- Automatic algorithm similar to F. [Torr & Zisserman]

Automatic Estimation of the trifocal tensor and correspondences

(i) Pairwise matches: Compute point matches between view pairs using robust `F` estimation.

(ii) Putative correspondences: over three views from two view matches.

(iii) RANSAC robust estimation:

Repeat

(a) Select random sample of 6 correspondences

(b) Compute T (1 or 3 solutions)

(c) Measure support (number of inliers)

Choose the T with the largest number of inliers.

(iv) MLE: re-estimate T from inlier correspondences.

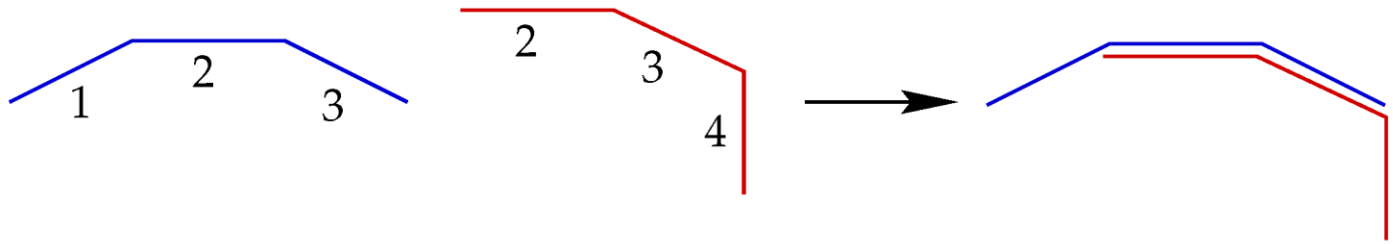
(v) Guided matching: generate additional matches.

Projective Reconstruction for a Sequence

(i) Compute all 2-view reconstructions for consecutive frames.

(ii) Compute all 3-view reconstructions for consecutive frames.

(iii) Extend to sequence by hierarchical merging:



(iv) Bundle-adjustment: minimize reprojection error

$$\min_{\mathrm{P}^j\ \mathbf{X}_i} \sum_{i \in \text{points}} \sum_{j \in \text{frames}} d\left(\mathbf{x}_i^j,\ \mathrm{P}^j \mathbf{X}_i\right)^2$$

(v) Automatic algorithm [Fitzgibbon & Zisserman]

Cameras and Scene Geometry for an Image Sequence

Given video

first frame of video

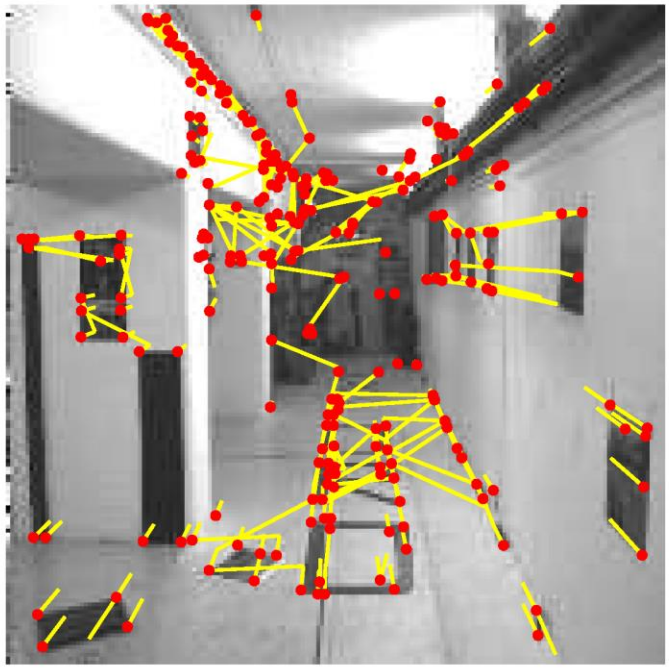
- Point correspondence (tracking).
- Projective Reconstruction.

Interest points computed for each frame

first frame of video

- About 500 points per frame

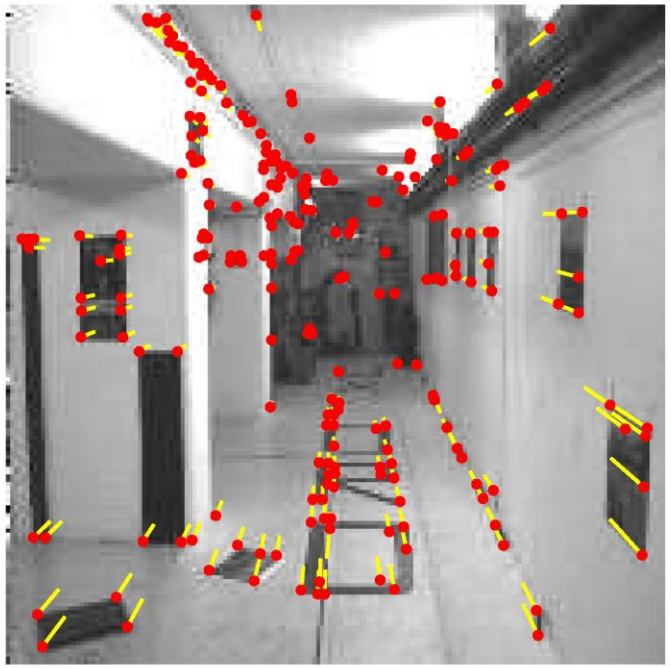
Point tracking: Correlation matching



first frame of video

- 10-50% wrong matches

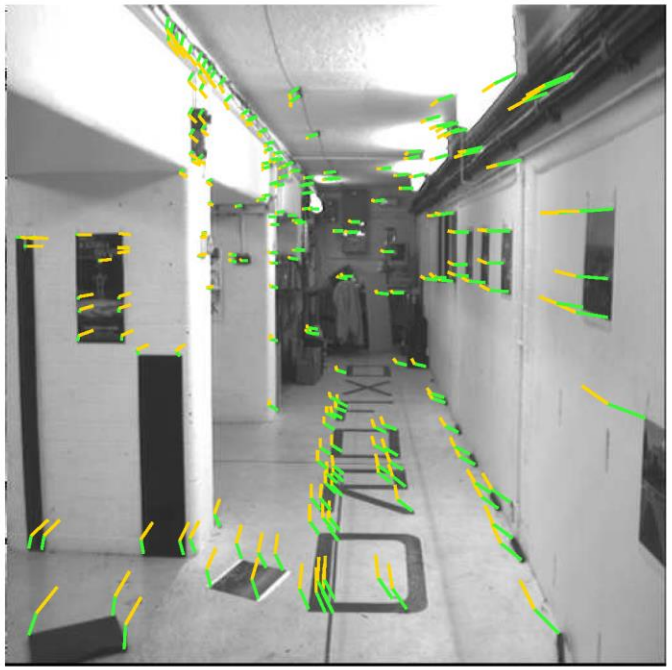
Point tracking: Epipolar-geometry guided matching



first frame of video

- Compute `F` so that matches consistent with epipolar geometry.
- Many fewer false matches, but still a loose constraint.

Point tracking: Trifocal tensor guided matching



first frame of video

- Compute trifocal tensor so that matches consistent with 3-views.
- Tighter constraint, so even fewer false matches.
- Three views is the last significant improvement.

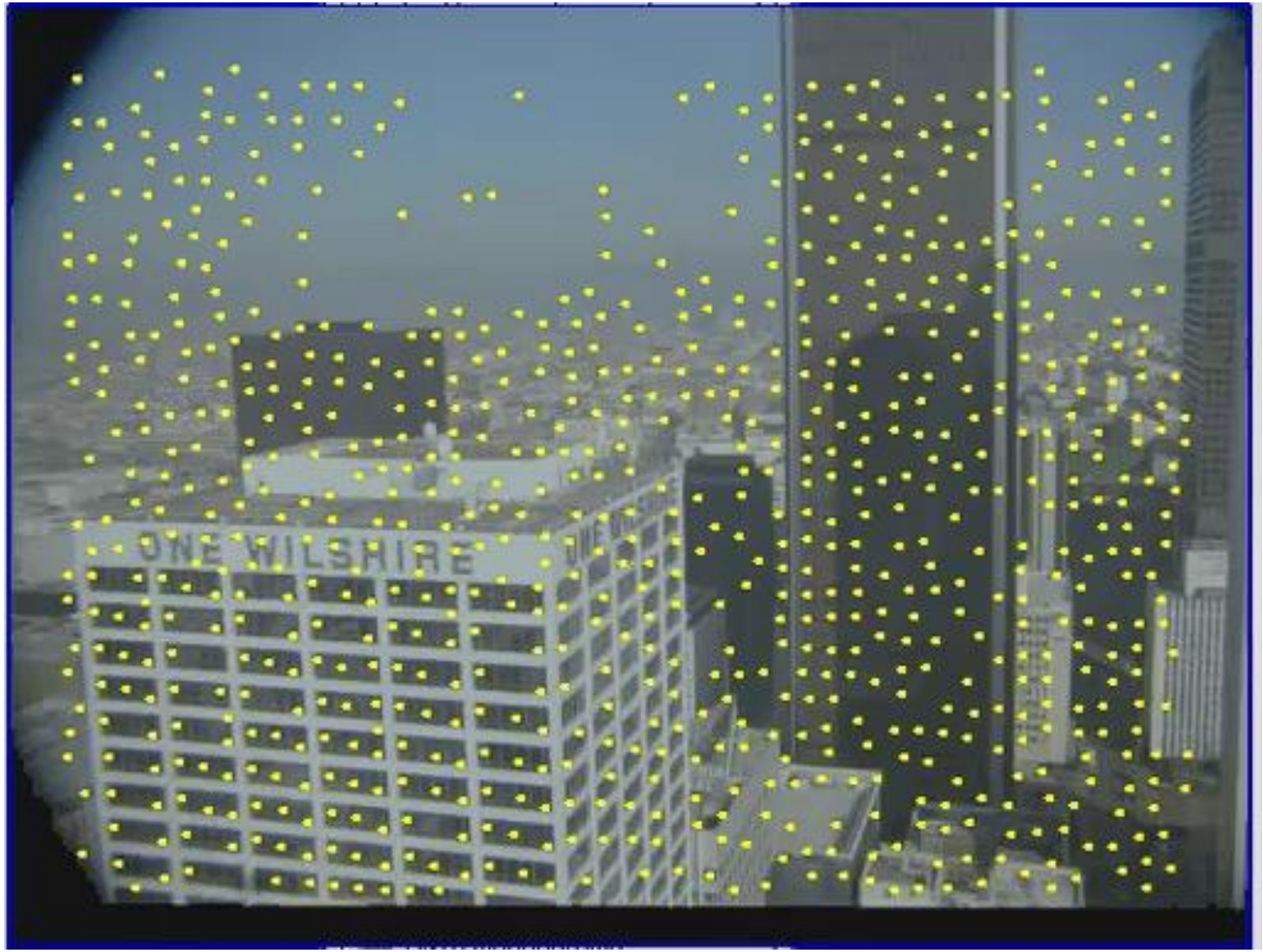
Reconstruction from Point Tracks

Compute 3D points and cameras from point tracks

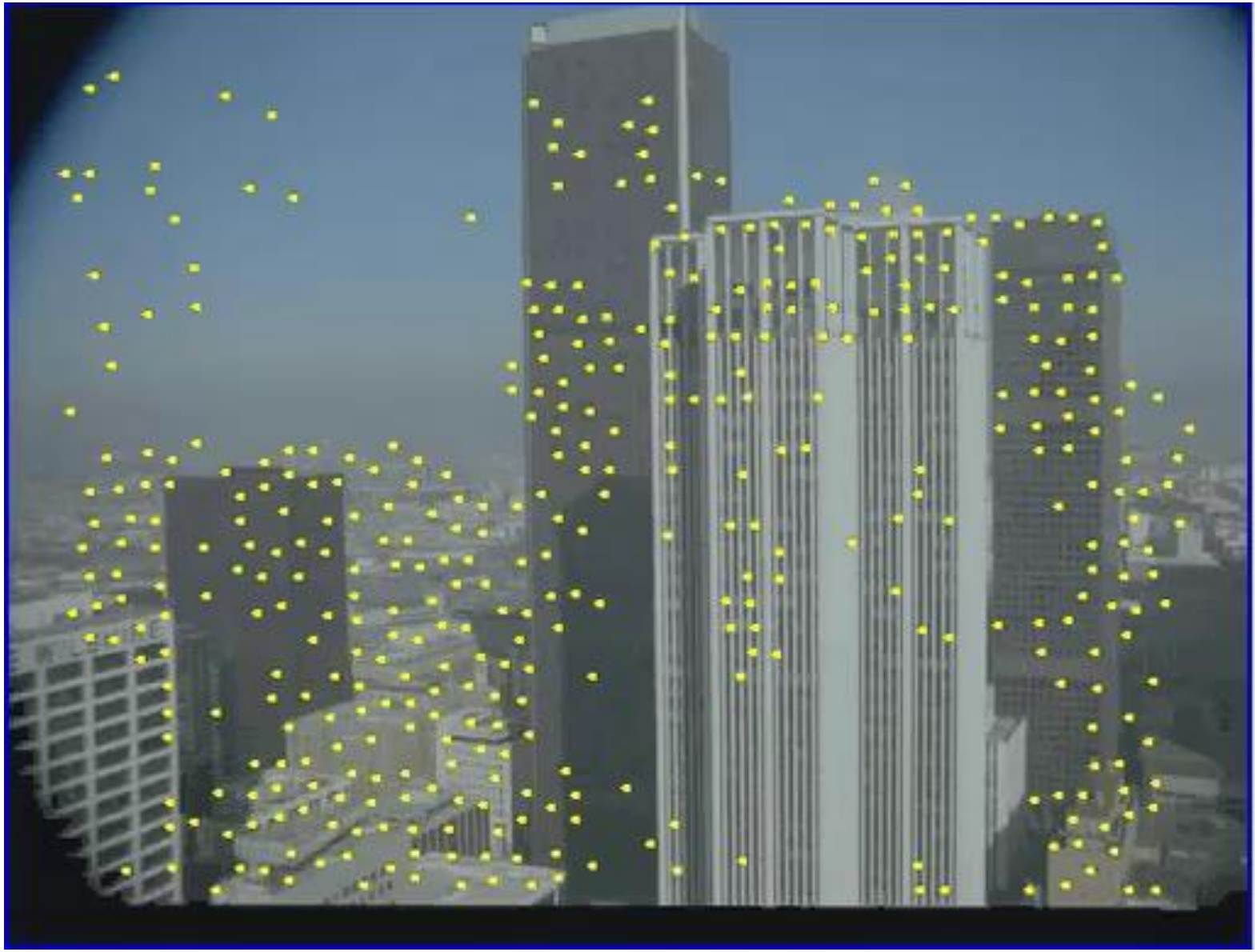
a frame of the video

- Hierarchical merging of sub-sequences.
- Bundle adjustment.

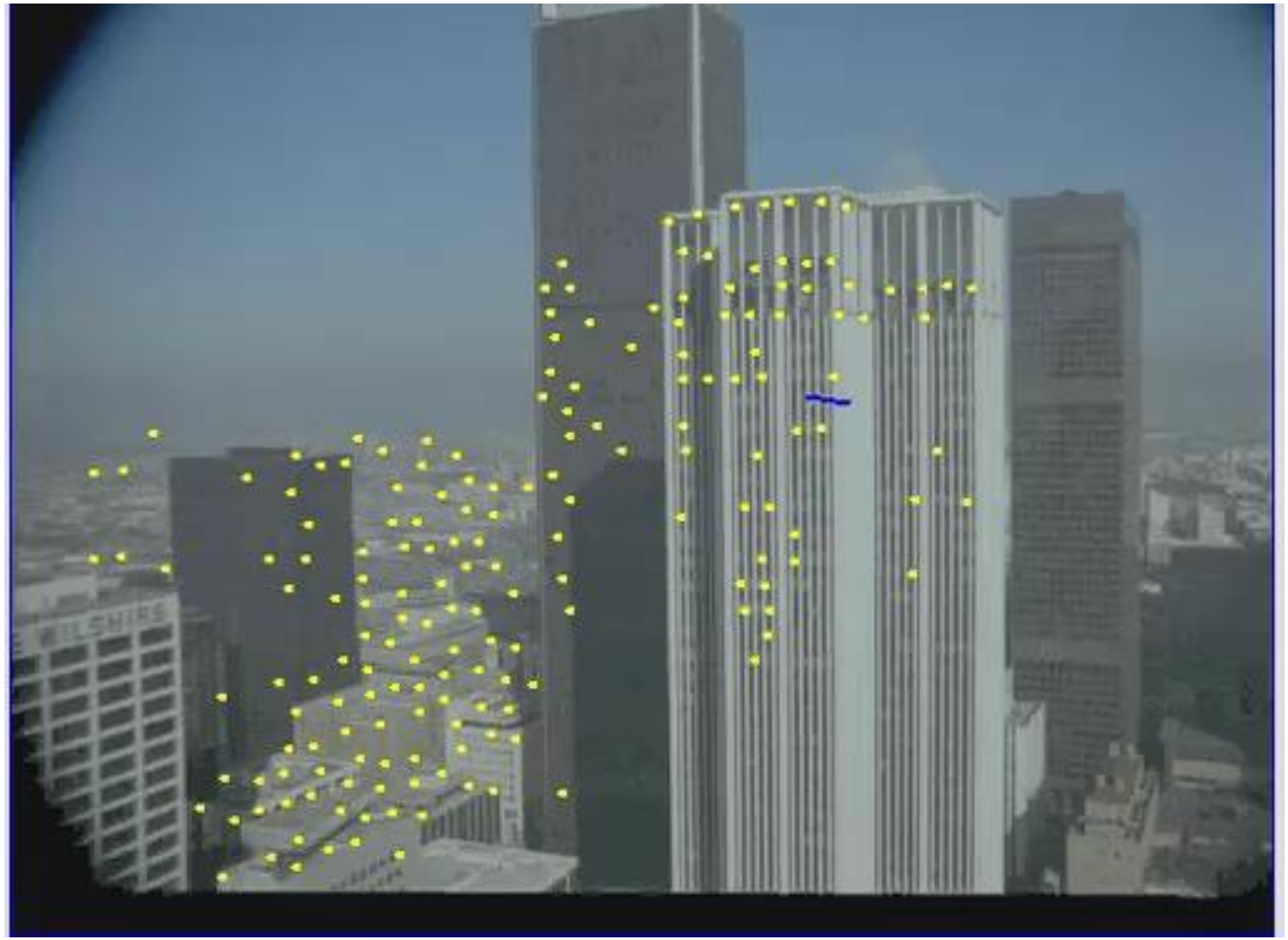
Raw matched points



Weeded using fundamental matrix



Weeded using trifocal tensor



An Efficient Solution to the Five-Point Relative Pose Problem

David Nistér
Sarnoff Corporation
CN5300, Princeton, NJ 08530
dnister@sarnoff.com

Five-Point Motion Estimation Made Easy

Hongdong Li and Richard Hartley

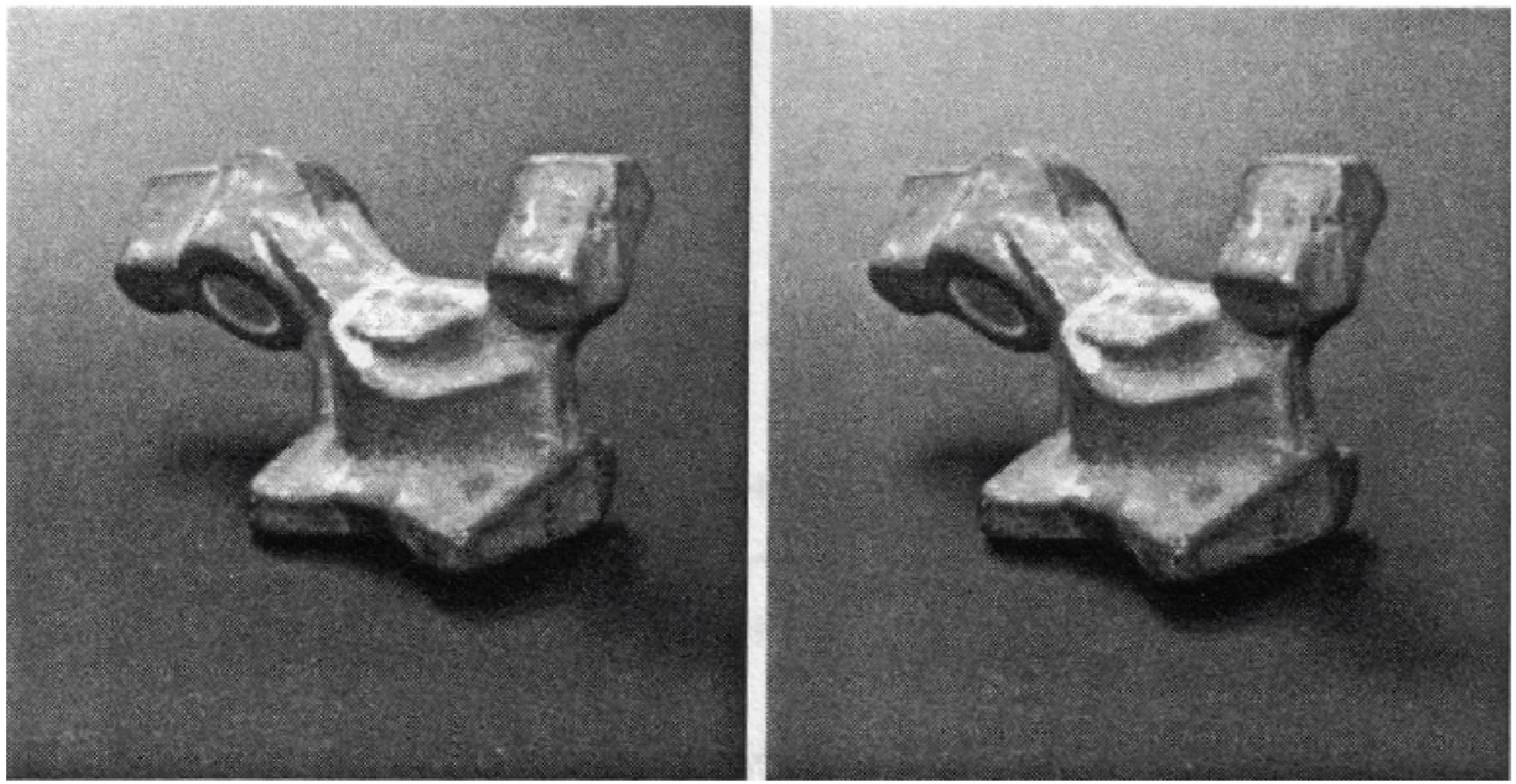
RSISE, The Australian National University.
Canberra Research Labs, National ICT Australia.*
Email: firstname.lastname@anu.edu.au

Stereo

1. Stereo Image Acquisition

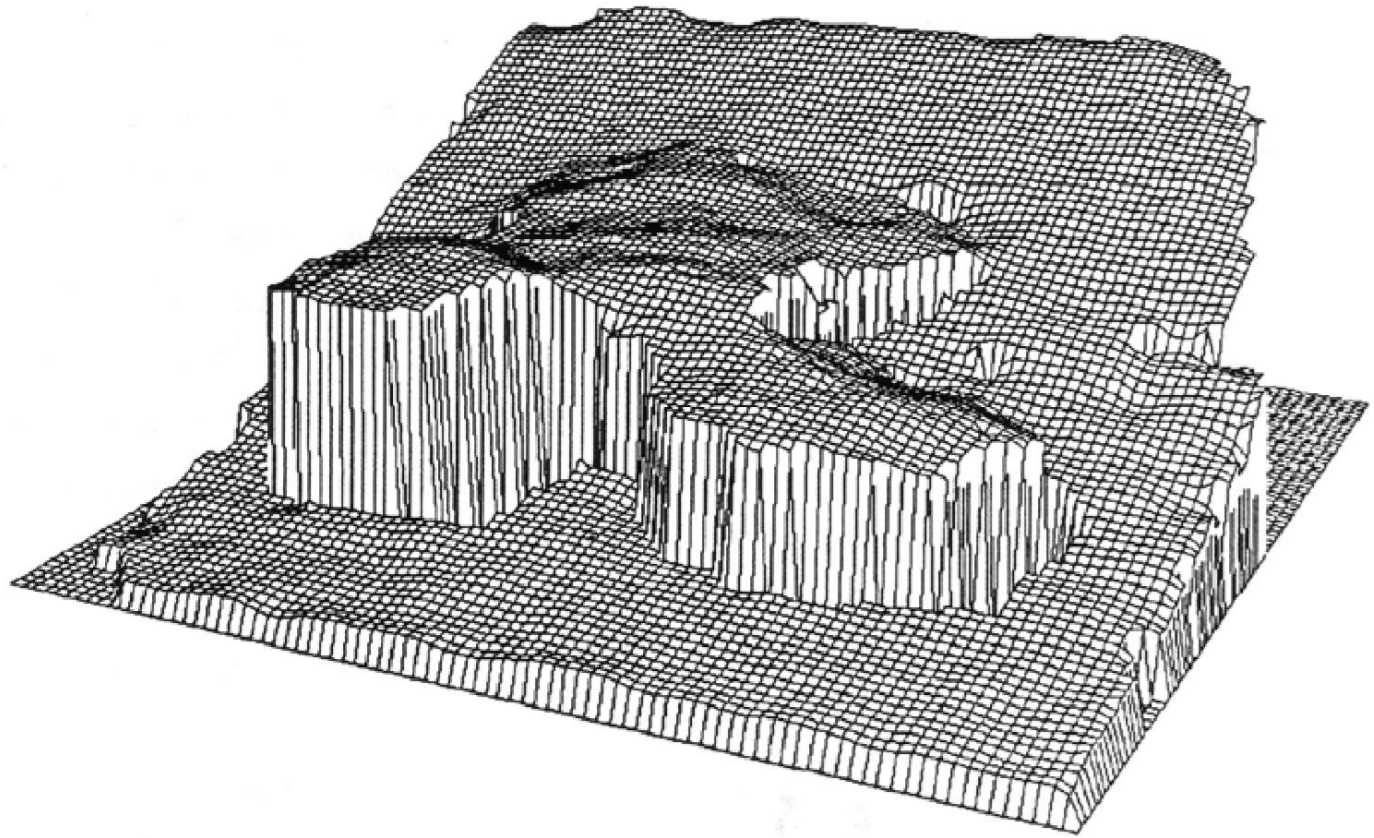
- As the name implies
- Capturing two images with a very specific camera geometry

Example Stereo Pair



Left Camera **Right Camera**

Obtained Depth Map in 3D



2. Camera Modelling

- Related to image acquisition
- For accurate depth results the camera parameters must be known
  - Intrinsic parameters
  - Focal length, pixel skew/shape, distortion, etc
- And the relationship between the two cameras must be known
  - Extrinsic parameters
  - Rotation, translation

Stereo Vision

Richard Hartley, Hongdong Li
Australian National University; National ICT Australia

November 23, 2018

Synonyms. Stereo

Related Concepts. Structure and motion; Euclidean recontruction;

Description. Stereo vision refers to extraction of the 3-dimensional structure of a scene from a pair of images.

1 Introduction

Description of the problem, and geometry. Stereo reconstruction involves the analysis of images from two cameras, and it is usually assumed that these are accurately modelled as pinhole cameras. In this case, a point $\mathbf{X}$ in the world is mapped to points $\mathbf{x}^R$ and $\mathbf{x}^L$ in the two images, according to the projection equations

$$\mathbf{x}^L = \mathtt{P}^L\mathbf{X} \;\;;\;\; \mathbf{x}^R = \mathtt{P}^R\mathbf{X}\,. \tag{1}$$

In this equation, the world point $\mathbf{X}$ and image points $\mathbf{x}^L, \mathbf{x}^R$ are represented as *homogeneous vectors*, so $\mathbf{X}$ is a 4-vector and $\mathbf{x}^L$ and $\mathbf{x}^R$ are 3-vectors. The matrices $\mathtt{P}^L$ and $\mathtt{P}^R$ are the projection matrices for the *left* and *right* cameras respectively, and have dimension 3×4. The general form for a camera matrix is $\mathtt{P} = \mathtt{K}[\mathtt{R} \mid -\mathbf{t}]$, where $\mathtt{K}$ is an upper-triangular *calibration matrix*, and $\mathtt{R}$ is a rotation matrix, representing the orientation of the camera with respect to some world-coordinate system. The camera centre is located at the point $\mathtt{R}^{-1}\mathbf{t}$. Assuming that the *principal point* of the camera is known (often the centre of the image) and image coordinates are relative to the principal point, then the calibration matrix is a diagonal matrix $\mathtt{K}_f = \mathrm{diag}(f, f, 1)$ where f is the focal length of the camera, expressed in pixels (in other words, the focal length of the camera is equal to f times the pixel pitch).

It is convenient to assume that a world coordinate frame is chosen to correspond with the centre of the left camera, oriented so that its principal axis points along the Z-axis. In this case, $\mathtt{P}^L = \mathtt{K}_f[\mathtt{I} \mid \mathbf{0}]$.

In the ideal situation the two cameras have the same orientation, and are placed side-by-side, with the right camera located at the point $(b, 0, 0)$, hence displaced from the left camera, by a distance b along the X-axis. In this case, the line between the two camera centres is known as the *base-line* and b is the *camera displacement*. (The length b is normally expressed in the scale of the world-coordinated – not in pixel units.) In this ideal situation, the principal axes (the pointing directions) of the two camera are parallel, and perpendicular to the base-line. A pair of cameras geometrically placed in this way are said to be *rectified*. Generally speaking, commercially available stereo cameras are rectified. The two corresponding images are known as a rectified stereo image pair.

The pair of camera matrices for a rectified pair of cameras (assuming equal focal lengths f) are given by

$$\begin{aligned} \mathtt{P}^L &= \mathtt{K}_f[\mathtt{I} \mid \mathbf{0}] \\ \mathtt{P}^R &= \mathtt{K}_f[\mathtt{I} \mid -\mathbf{b}] \end{aligned} \tag{2}$$

where $\mathbf{b}_R = (b, 0, 0)^\mathsf{T}$ is the centre of the right camera.

Note: for these matrices to express the correct mapping from world-to-image pixel coordinates, image coordinates must be expressed in pixels relative to their distance from the principal point.

Now consider a world point, expressed in homogeneous coordinates as $\mathbf{X} = (\mathtt{X}, \mathtt{Y}, \mathtt{Z}, 1)$. One computes that

$$\begin{aligned} \mathbf{x}^L &= \mathtt{K}_f[\mathtt{I} \mid \mathbf{0}](\mathtt{X}, \mathtt{Y}, \mathtt{Z}, 1)^\mathsf{T} = (f\mathtt{X}, f\mathtt{Y}, \mathtt{Z})^\mathsf{T} \\ \mathbf{x}^R &= \mathtt{K}_f[\mathtt{I} \mid (-b, 0, 0)^\mathsf{T}](\mathtt{X}, \mathtt{Y}, \mathtt{Z}, 1)^\mathsf{T} = (f(\mathtt{X} - b), f\mathtt{Y}, \mathtt{Z})^\mathsf{T} \end{aligned}$$

Transforming homogeneous coordinates to Euclidean coordinates (dehomogenizing) gives

$$\begin{aligned} \mathbf{x}^L &= (x^L, y^L) = (f\mathtt{X}/\mathtt{Z}, f\mathtt{Y}/\mathtt{Z}) \\ \mathbf{x}^R &= (x^R, y^R) = (f(\mathtt{X} - b)/\mathtt{Z}), f\mathtt{Y}/\mathtt{Z}) \end{aligned} \tag{3}$$

Description of the problem, and geometry. Stereo reconstruction involves the analysis of images from two cameras, and it is usually assumed that these are accurately modelled as pinhole cameras. In this case, a point $\mathbf{X}$ in the world is mapped to points $\mathbf{x}^R$ and $\mathbf{x}^L$ in the two images, according to the projection equations

$$\mathbf{x}^L = \mathtt{P}^L\mathbf{X} \;\; ; \;\; \mathbf{x}^R = \mathtt{P}^R\mathbf{X} \; . \tag{1}$$

In this equation, the world point $\mathbf{X}$ and image points $\mathbf{x}^L, \mathbf{x}^R$ are represented as *homogeneous vectors*, so $\mathbf{X}$ is a 4-vector and $\mathbf{x}^L$ and $\mathbf{x}^R$ are 3-vectors. The matrices $\mathtt{P}^L$ and $\mathtt{P}^R$ are the projection matrices for the *left* and *right* cameras respectively, and have dimension 3×4. The general form for a camera matrix is $\mathtt{P} = \mathtt{K}[\mathtt{R} \mid -\mathbf{t}]$, where $\mathtt{K}$ is an upper-triangular *calibration matrix*, and $\mathtt{R}$ is a rotation matrix, representing the orientation of the camera with respect to some world-coordinate system. The camera centre is located at the point $\mathtt{R}^{-1}\mathbf{t}$. Assuming that the *principal point* of the camera is known (often the centre of the image) and image coordinates are relative to the principal point, then the calibration matrix is a diagonal matrix $\mathtt{K}_f = \mathrm{diag}(f, f, 1)$ where f is the focal length of the camera, expressed in pixels (in other words, the focal length of the camera is equal to f times the pixel pitch).

$$x^L - x^R = d = \frac{fb}{\mathrm{Z}} \ , \tag{4}$$

known as the *disparity*. Expressed the other way round

$$\mathrm{Z} = \frac{fb}{d} \ . \tag{5}$$

Hence, the depth Z of a point is inversely proportional to the disparity; knowing f and b, the depth of a point may be directly computed from measured disparity. The complete coordinates of the 3-D point can be computed knowing $\mathbf{x}^L$ and the disparity d according to

$$\mathbf{X} = (\mathrm{X}, \mathrm{Y}, \mathrm{Z})^\mathsf{T} = \frac{b}{d}(x^L, y^L, f)^\mathsf{T} \ . \tag{6}$$

It is possible to derive (4) geometrically, as shown in figure 1

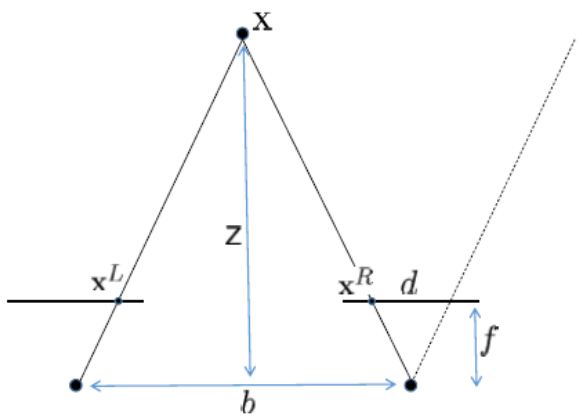
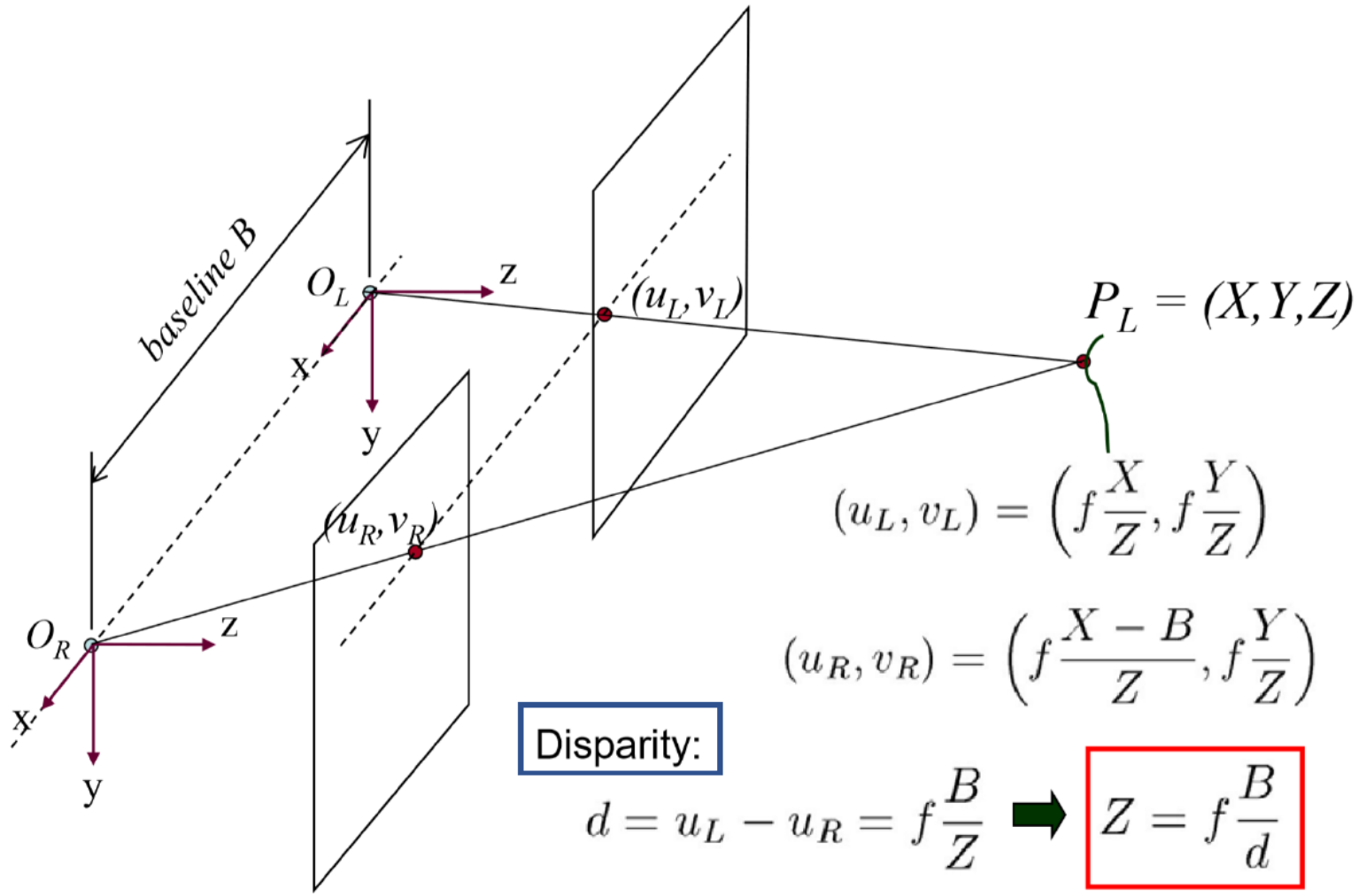


Figure 1: *Geometric derivation of the disparity formula (5). The triangles shown are similar, so* $\mathrm{Z}/b = f/d$*, from which (5) follows.*

Disparity Computation

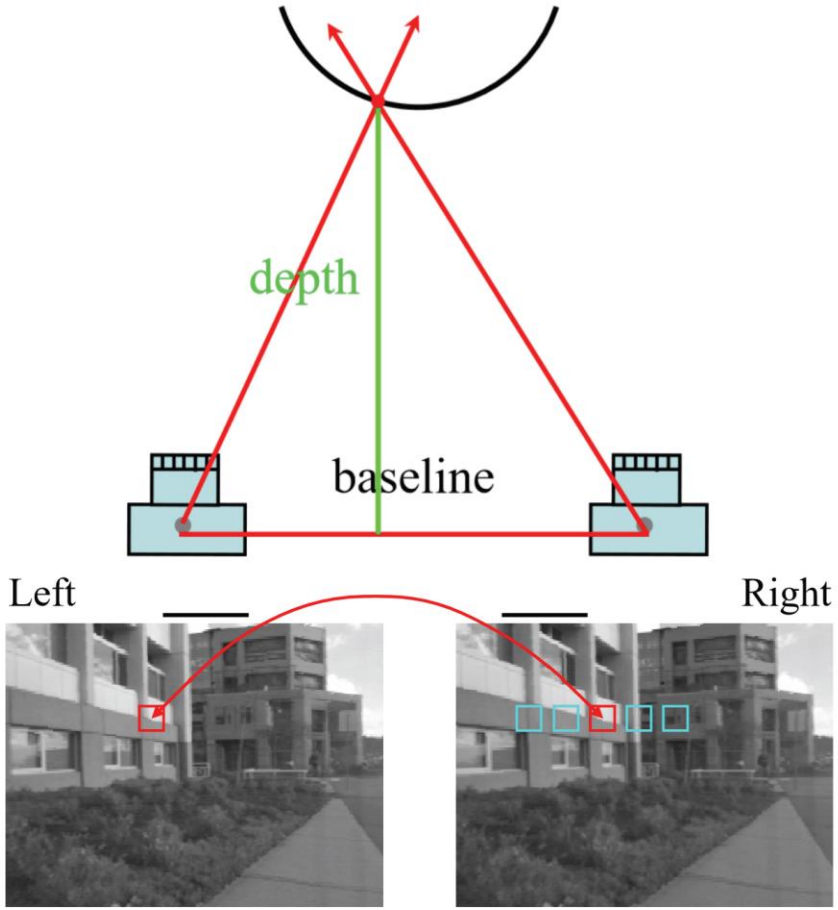


Accuracy

depth (m)	disparity(degrees/minutes)	error
1 m	$3.6°$	$\pm$ 2.3 mm
10 m	$22'$	$\pm$ 233mm
100 m	$2.2'$	$\pm$ 24 m

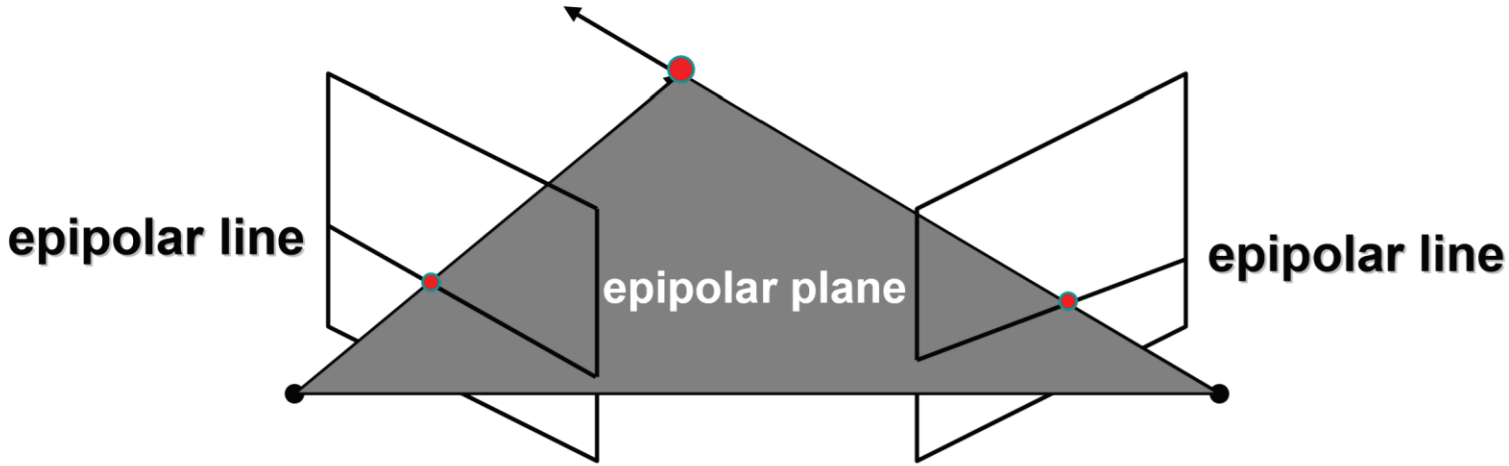
Table 1: *Assuming a baseline of* 64 *milimetres (mm), approximately the distance between human eyes, and angular resolution of* $\Delta d = \pm 0.5$ *minutes of arc (approximately human eye resolution), this table shows accuracy of stereo depth resolution for points at different depths measured in metres (m). Disparity represents the angle between rays from the two eyes to a point at the given depth. Depth error is inversely proportional to focal length* f *and baseline* b.

The Correspondence Problem



- Triangulate on two images of the same point to recover depth
  - Feature matching across views
  - Calibrated cameras
- Matching correlation windows across scan lines

General Epipolar Constraint



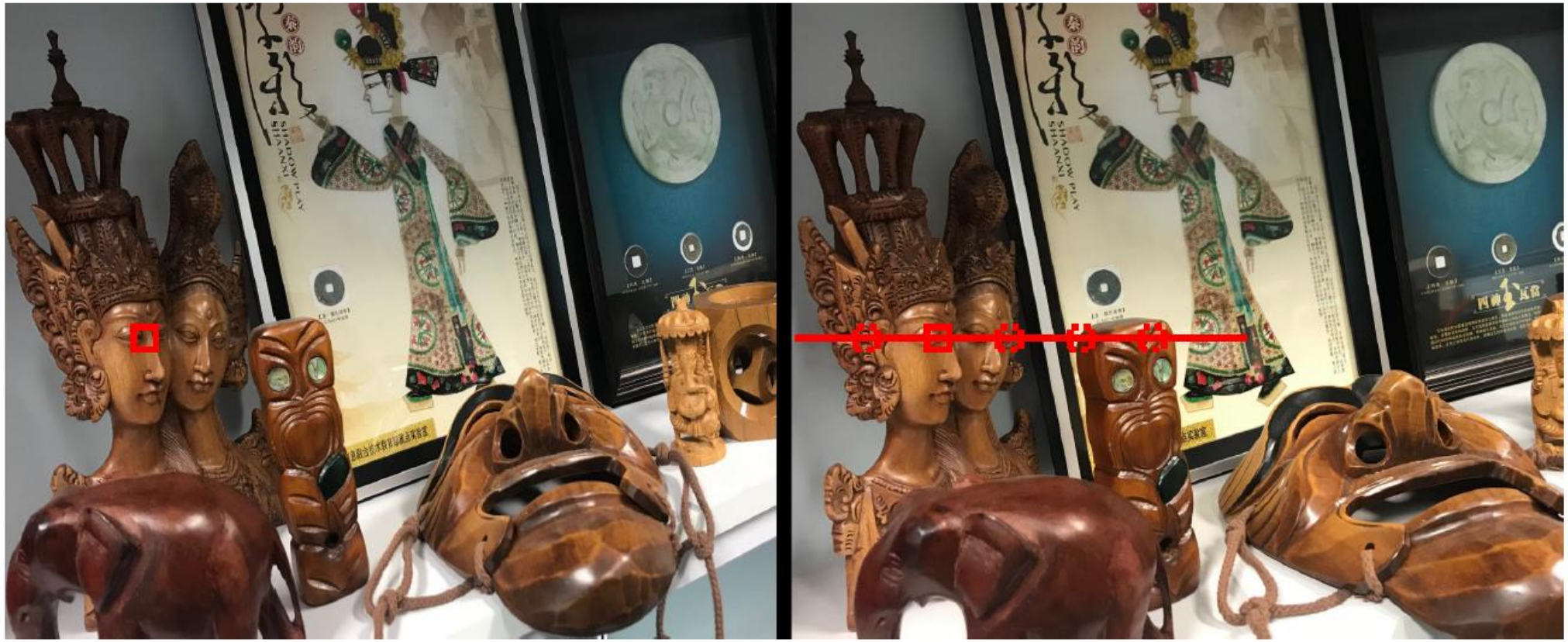
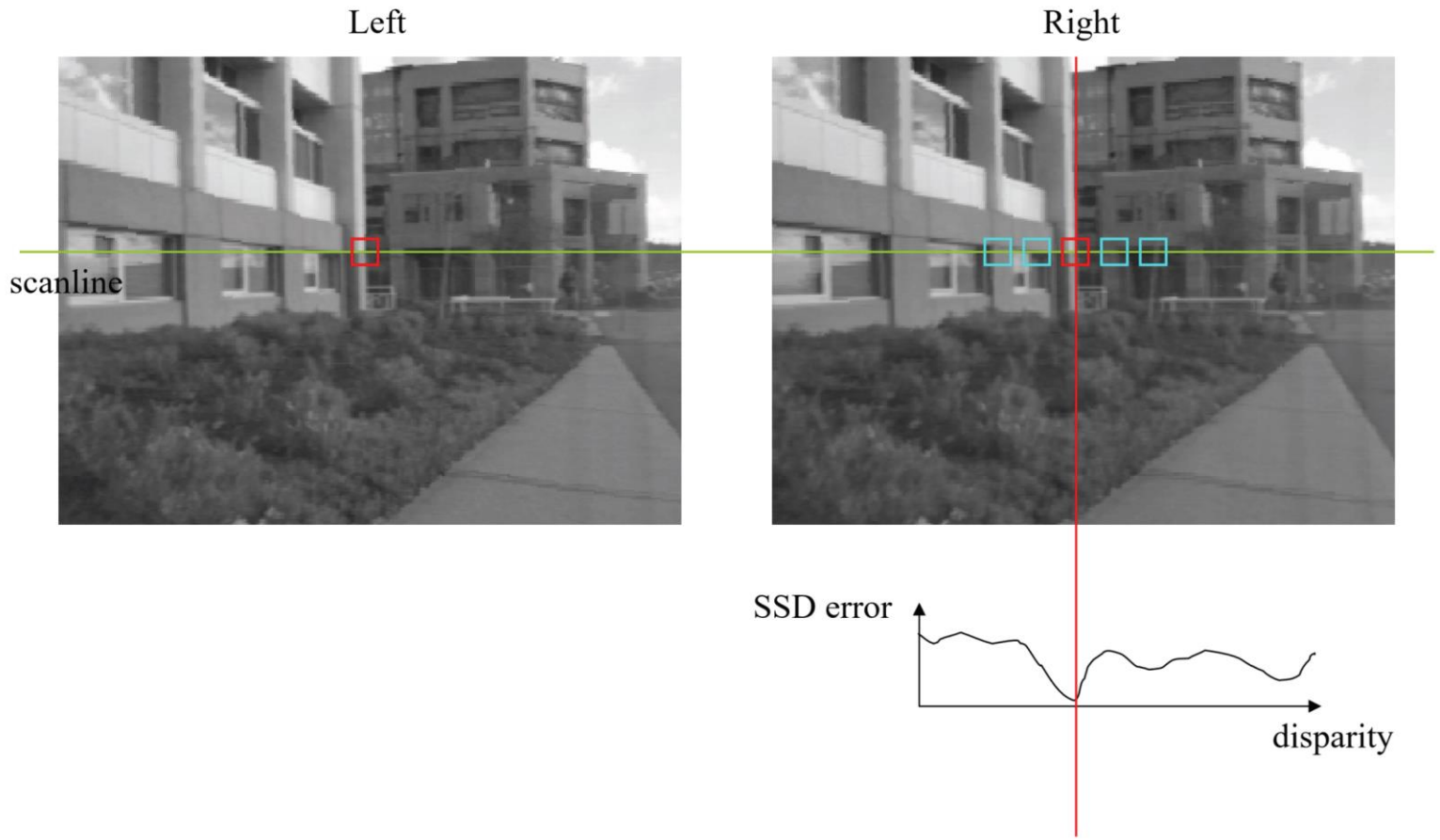
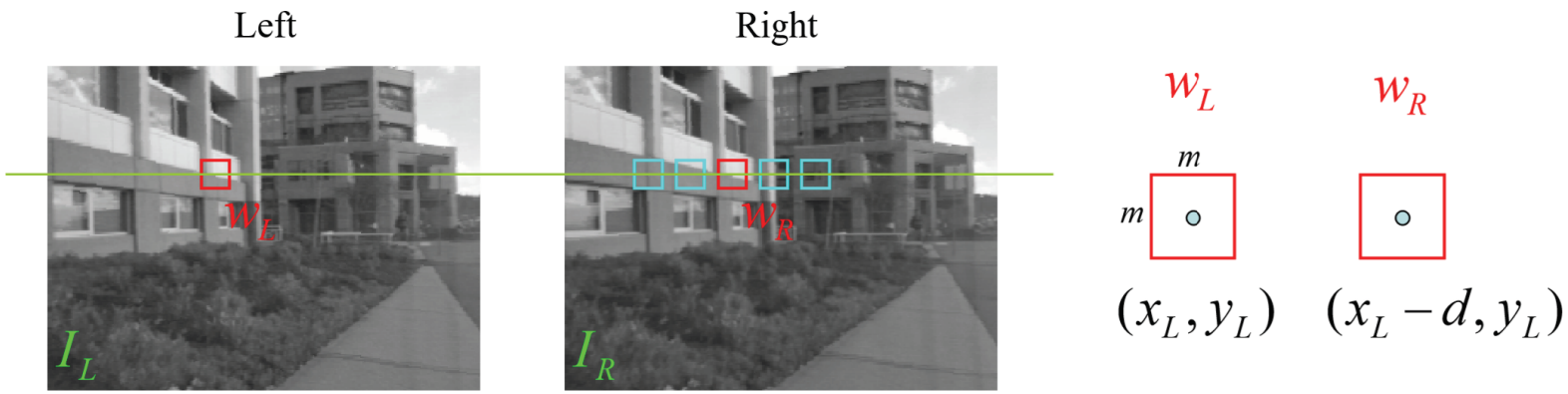


Figure 2: *If the images are rectified, a point in the left image is matched to a point lying on the same horizontal scan line. A point may potentially match any other point on the line. (Here and elsewhere, rectified images are shown with the left-image on the right, to allow for viewing using cross-fusion – view left-image with right eye, and vice versa).*

The Correspondence Problem: Correspondence Using Correlation



Sum of Squared (Pixel) Differences



w_L and w_R are corresponding m by m windows of pixels.

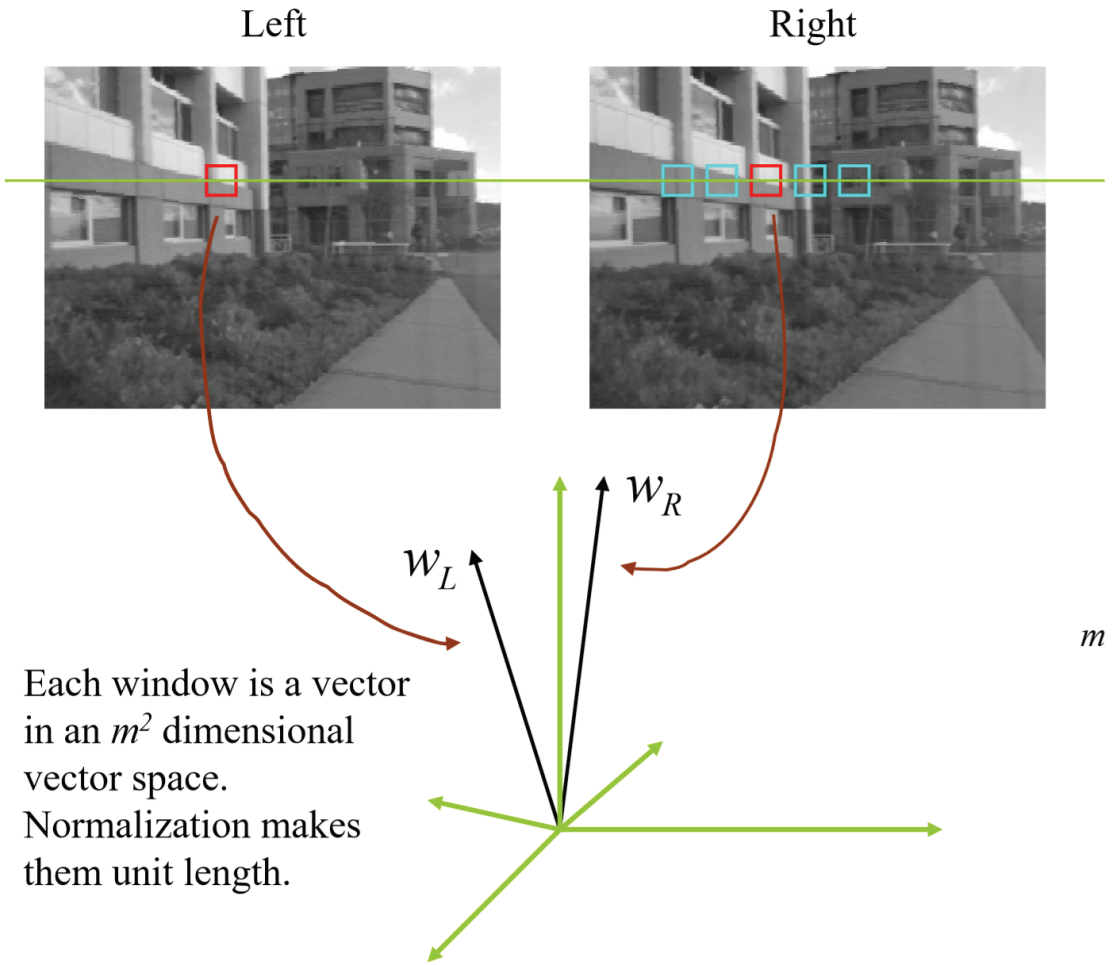
We define the window function :

$$W_m(x,y) = \{u,v \mid x - \tfrac{m}{2} \le u \le x + \tfrac{m}{2}, y - \tfrac{m}{2} \le v \le y + \tfrac{m}{2}\}$$

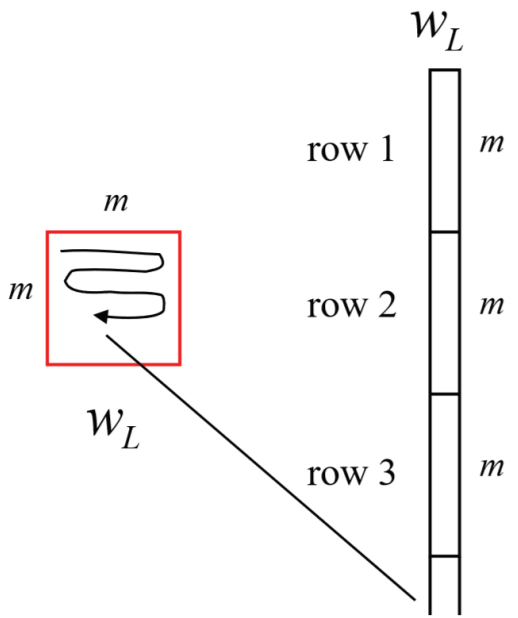
The SSD cost measures the intensity difference as a function of disparity :

$$C_r(x,y,d) = \sum_{(u,v) \in W_m(x,y)} [I_L(u,v) - I_R(u-d,v)]^2$$

Images as Vectors



"Unwrap" image to form vector, using raster scan order



2.1 Window based similarity

Since a single pixel by itself rarely has enough information to allow accurate matching, image-to-image matching is commonly carried out by comparing windows around points to be compared. Let $\mathbf{x}^L$ and $\mathbf{x}^R$ be two points, and denote by W_L and W_R windows (for example, of dimension 7×7) centred on these two points. Various measures of similarity between two windows have been proposed. It is convenient to consider vectors $\mathbf{w}^L$ and $\mathbf{w}^R$ representing all the pixels lying in the windows $\mathbf{W}^L$ and $\mathbf{W}_R$.[1] Various similarity measures between vectors $\mathbf{w}^L$ and $\mathbf{w}^R$ are as follows. For simplicity, $\mathbf{w}^L$ will be represented as $\mathbf{x}$ and $\mathbf{w}^R$ as $\mathbf{y}$. Different similarity measures used are

- Sum of Squares Difference (SSD), defined as

$$\mathrm{SSD}(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|^2 = \sum_{i=1}^{n} (x_i - y_i)^2 \ .$$

  This is just the squared Euclidean (L_2) distance between the vectors.

- Sum of Absolute Difference (SAD), defined as

$$\mathrm{SAD}(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_1 = \sum_{i=1}^{n} |x_i - y_i|$$

  measures the L_1-norm distance. Both SSD and SAD are sensitive to lighting or intensity changes between the right and left images. If, for instance, one image is brighter than the other, then both the SAD and SSD distances may be large, even though $(\mathbf{x}, \mathbf{y})$ represents a correct match. The two windows are compared, based more on their overall intensity, than details of intensity changes in the windows. The SAD is a more robust measure of similarity than SSD, since it is less sensitive to noise. If a single pixel differs greatly between left and right, then the SSD distance will be large, because the difference is squared; the SAD distance will be less affected.

[1] In the case of 7×7 windows, these could be vectors of dimension 49 consisting of the pixel intensities. In the case of colour images, one may use 3 intensity values in some colour space, such as RGB, resulting in vectors of dimension 147.

- Sum of Squares Difference (SSD), defined as

$$\mathrm{SSD}(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|^2 = \sum_{i=1}^{n}(x_i-y_i)^2 \ .$$

  This is just the squared Euclidean (L_2) distance between the vectors.

- Sum of Absolute Difference (SAD), defined as

$$\mathrm{SAD}(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|_1 = \sum_{i=1}^{n}|x_i-y_i|$$

- Normalized similarity measures: If the left and right images have different intensities, one may account for this by scaling, resulting in normalized SSD and normalized SAD measures.

$$\begin{aligned}\mathrm{NSSD}(\mathbf{x},\mathbf{y}) &= \mathrm{SSD}(\mathbf{x}/\|\mathbf{x}\|,\mathbf{y}/\|\mathbf{y}\|) = 2(1-\cos(\theta))\\ \mathrm{NSAD}(\mathbf{x},\mathbf{y}) &= \mathrm{SAD}(\mathbf{x}/\|\mathbf{x}\|,\mathbf{y}/\|\mathbf{y}\|)\ ,\end{aligned}$$

  where θ represents the angle between the vectors $\mathbf{x}$ and $\mathbf{y}$. NSSD is closely related to the negative Normalized Cross Correlation (NCC), defined as

$$-\mathrm{NCC}(\mathbf{x},\mathbf{y}) = -\frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}} = -\frac{\langle \mathbf{x},\mathbf{y}\rangle}{\|\mathbf{x}\|\,\|\mathbf{y}\|} = -\cos(\theta).$$

- ZNCC etc: If one of $\mathbf{x}$ or $\mathbf{y}$ is perturbed by adding a constant value to each component (corresponding to adding a constant to intensities in one of the images), then the value of the similarity measures above will be changed. One may address this by subtracting the mean value $\bar{\mathbf{x}}$ or $\bar{\mathbf{y}}$ from each component before computing the NCC, SSD or SAD values. For instance, a suitable similarity measure is

$$-\mathrm{ZNCC}(\mathbf{x},\mathbf{y}) = -\frac{\langle \mathbf{x}-\mathbf{1}\bar{x},\mathbf{y}-\mathbf{1}\bar{y}\rangle}{\|x-\mathbf{1}\bar{x}\|\,\|\mathbf{y}-\mathbf{1}\bar{y}\|} = \frac{n\bar{x}\bar{y}-\langle \mathbf{x},\mathbf{y}\rangle}{\sigma_x\sigma_y}\ .$$

- Sum of Squares Difference (SSD), defined as

$$\mathrm{SSD}(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|^2 = \sum_{i=1}^{n}(x_i-y_i)^2 \ .$$

  This is just the squared Euclidean (L_2) distance between the vectors.

- Sum of Absolute Difference (SAD), defined as

$$\mathrm{SAD}(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|_1 = \sum_{i=1}^{n}|x_i-y_i|$$

- Normalized similarity measures: If the left and right images have different intensities, one may account for this by scaling, resulting in normalized SSD and normalized SAD measures.

$$\begin{aligned}\mathrm{NSSD}(\mathbf{x},\mathbf{y}) &= \mathrm{SSD}(\mathbf{x}/\|\mathbf{x}\|,\mathbf{y}/\|\mathbf{y}\|) = 2(1-\cos(\theta))\\ \mathrm{NSAD}(\mathbf{x},\mathbf{y}) &= \mathrm{SAD}(\mathbf{x}/\|\mathbf{x}\|,\mathbf{y}/\|\mathbf{y}\|)\ ,\end{aligned}$$

  where θ represents the angle between the vectors $\mathbf{x}$ and $\mathbf{y}$. NSSD is closely related to the negative Normalized Cross Correlation (NCC), defined as

$$-\mathrm{NCC}(\mathbf{x},\mathbf{y}) = -\frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}} = -\frac{\langle \mathbf{x},\mathbf{y}\rangle}{\|\mathbf{x}\|\,\|\mathbf{y}\|} = -\cos(\theta).$$

- ZNCC etc: If one of $\mathbf{x}$ or $\mathbf{y}$ is perturbed by adding a constant value to each component (corresponding to adding a constant to intensities in one of the images), then the value of the similarity measures above will be changed. One may address this by subtracting the mean value $\bar{\mathbf{x}}$ or $\bar{\mathbf{y}}$ from each component before computing the NCC, SSD or SAD values. For instance, a suitable similarity measure is

$$-\mathrm{ZNCC}(\mathbf{x},\mathbf{y}) = -\frac{\langle \mathbf{x}-\mathbf{1}\bar{x},\mathbf{y}-\mathbf{1}\bar{y}\rangle}{\|x-\mathbf{1}\bar{x}\|\,\|\mathbf{y}-\mathbf{1}\bar{y}\|} = \frac{n\bar{x}\bar{y}-\langle \mathbf{x},\mathbf{y}\rangle}{\sigma_x\sigma_y}\ .$$

- Sum of Squares Difference (SSD), defined as

$$\mathrm{SSD}(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|^2 = \sum_{i=1}^{n}(x_i - y_i)^2 \ .$$

  This is just the squared Euclidean (L_2) distance between the vectors.

- Sum of Absolute Difference (SAD), defined as

$$\mathrm{SAD}(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|_1 = \sum_{i=1}^{n}|x_i - y_i|$$

- Normalized similarity measures: If the left and right images have different intensities, one may account for this by scaling, resulting in normalized SSD and normalized SAD measures.

$$\begin{aligned}\mathrm{NSSD}(\mathbf{x},\mathbf{y}) &= \mathrm{SSD}(\mathbf{x}/\|\mathbf{x}\|, \mathbf{y}/\|\mathbf{y}\|) = 2(1-\cos(\theta)) \\ \mathrm{NSAD}(\mathbf{x},\mathbf{y}) &= \mathrm{SAD}(\mathbf{x}/\|\mathbf{x}\|, \mathbf{y}/\|\mathbf{y}\|) \ ,\end{aligned}$$

  where θ represents the angle between the vectors $\mathbf{x}$ and $\mathbf{y}$. NSSD is closely related to the negative Normalized Cross Correlation (NCC), defined as

$$-\mathrm{NCC}(\mathbf{x},\mathbf{y}) = -\frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}} = -\frac{\langle \mathbf{x},\mathbf{y}\rangle}{\|\mathbf{x}\|\,\|\mathbf{y}\|} = -\cos(\theta).$$

- ZNCC etc: If one of $\mathbf{x}$ or $\mathbf{y}$ is perturbed by adding a constant value to each component (corresponding to adding a constant to intensities in one of the images), then the value of the similarity measures above will be changed. One may address this by subtracting the mean value $\bar{\mathbf{x}}$ or $\bar{\mathbf{y}}$ from each component before computing the NCC, SSD or SAD values. For instance, a suitable similarity measure is

$$-\mathrm{ZNCC}(\mathbf{x},\mathbf{y}) = -\frac{\langle \mathbf{x}-\mathbf{1}\bar{x}, \mathbf{y}-\mathbf{1}\bar{y}\rangle}{\|x-\mathbf{1}\bar{x}\|\,\|\mathbf{y}-\mathbf{1}\bar{y}\|} = \frac{n\bar{x}\bar{y} - \langle \mathbf{x},\mathbf{y}\rangle}{\sigma_x \sigma_y} \ .$$

- Sum of Squares Difference (SSD), defined as

$$\mathrm{SSD}(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|^2 = \sum_{i=1}^{n}(x_i - y_i)^2 \ .$$

  This is just the squared Euclidean (L_2) distance between the vectors.

- Sum of Absolute Difference (SAD), defined as

$$\mathrm{SAD}(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|_1 = \sum_{i=1}^{n}|x_i - y_i|$$

- Normalized similarity measures: If the left and right images have different intensities, one may account for this by scaling, resulting in normalized SSD and normalized SAD measures.

$$\begin{aligned}\mathrm{NSSD}(\mathbf{x},\mathbf{y}) &= \mathrm{SSD}(\mathbf{x}/\|\mathbf{x}\|, \mathbf{y}/\|\mathbf{y}\|) = 2(1-\cos(\theta)) \\ \mathrm{NSAD}(\mathbf{x},\mathbf{y}) &= \mathrm{SAD}(\mathbf{x}/\|\mathbf{x}\|, \mathbf{y}/\|\mathbf{y}\|) \ ,\end{aligned}$$

  where θ represents the angle between the vectors $\mathbf{x}$ and $\mathbf{y}$. NSSD is closely related to the negative Normalized Cross Correlation (NCC), defined as

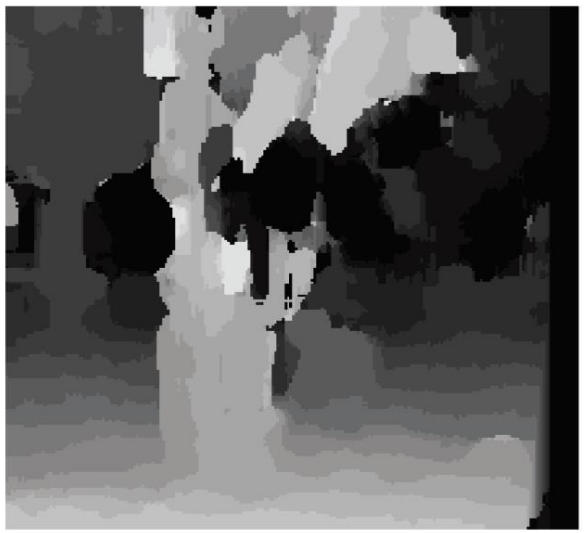
$$-\mathrm{NCC}(\mathbf{x},\mathbf{y}) = -\frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}} = -\frac{\langle \mathbf{x},\mathbf{y}\rangle}{\|\mathbf{x}\|\,\|\mathbf{y}\|} = -\cos(\theta).$$

- ZNCC etc: If one of $\mathbf{x}$ or $\mathbf{y}$ is perturbed by adding a constant value to each component (corresponding to adding a constant to intensities in one of the images), then the value of the similarity measures above will be changed. One may address this by subtracting the mean value $\bar{\mathbf{x}}$ or $\bar{\mathbf{y}}$ from each component before computing the NCC, SSD or SAD values. For instance, a suitable similarity measure is

$$-\mathrm{ZNCC}(\mathbf{x},\mathbf{y}) = -\frac{\langle \mathbf{x}-\mathbf{1}\bar{x}, \mathbf{y}-\mathbf{1}\bar{y}\rangle}{\|x-\mathbf{1}\bar{x}\|\,\|\mathbf{y}-\mathbf{1}\bar{y}\|} = \frac{n\bar{x}\bar{y} - \langle \mathbf{x},\mathbf{y}\rangle}{\sigma_x \sigma_y} \ .$$

Effect of Window Size

W = 3



W = 20

- Improvement: use an adaptive window size (try multiple sizes and select best match)

(Seitz)

3 Pyramid matching

Matching based solely on local appearance and epipolar search usually gives poor performance, because there may be many points $\mathbf{x}^R$ that match a given point $\mathbf{x}^L$ closely. This is particularly true with when images have repeated structures, such as windows on a building facade, where local matching is insufficient to determine which window matches which. This is illustrated in ??figure **??**. Moreover, natural variations between left and right images, such as slight lighting

Figure 3: *Local matching methods are not able to distinguish the correct match among repeated structures. In a small window, the two corners look very similar. Looking further in a larger window shows that they are a false match.*

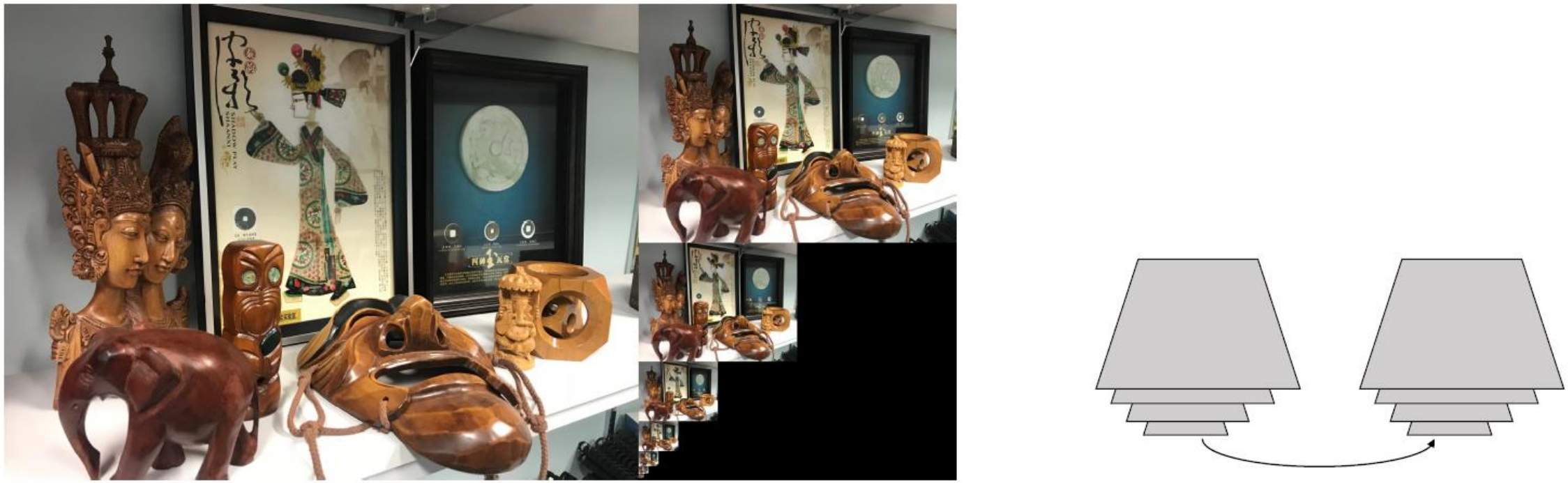


Figure 4: *An image pyramid (on left) is build from both images. Then, images are matched from the bottom up, propagating estimated disparities up the pyramid.*

Monotonicity and regularization

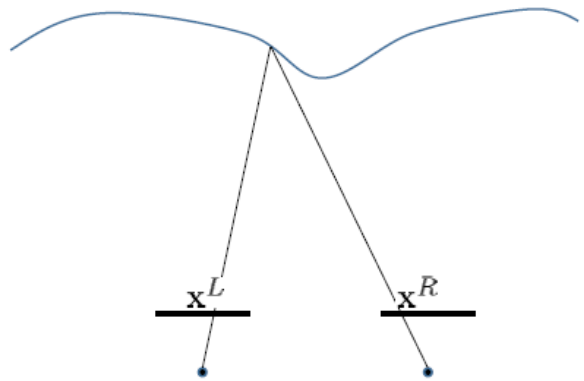


Figure 5: *Illustration of monotonicity. As a point* $\mathbf{x}^L$ *moves from left to right in the left image, the corresponding point* $\mathbf{x}^R = m(\mathbf{x}^L)$ *also moves from left to right. The function* m *is monotonically increasing.*

An important property of the matching function m is the following.

Lemma 1 Monotonicity condition. *If there are no occlusions, and* m *is a continuous matching function for a non-transparent scene, then it is monotone-increasing.*

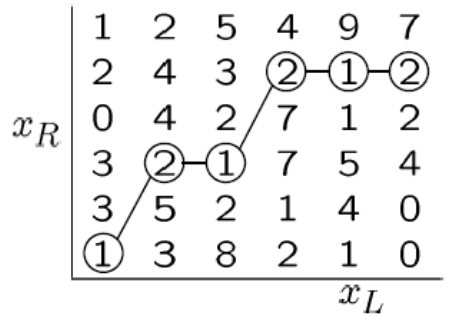


Figure 6: *Dynamic programming to find the least-cost match. Given an array of cost values, the task is to find the monotonically increasing (or non-decreasing) path through the table. Note that there are non-monotonic paths of lesser cost.*

```
1: procedure DynamicProgramming
2:     for (j = 1, ..., n) do S_0j = 0
3:     for i = 1, ..., m do    // forward sweep
4:         for j = 1, ..., n do
5:             S_ij = C_ij + min_{j'≤j} S_{i-1,j'}
6:     j*_n = arg min_j S_nj
7:     for i = n − 1, ..., 1 do    // backward sweep
8:         j*_i = arg min_{j≤j*_{i+1}} S_ij
```

Figure 7: *Dynamic programming enforcing monotonicity. At termination, $j_i^* = x^R$ is the match for $i = x_i^L$.*

Monotonicity with occlusions. The monotonicity condition will hold under more general conditions than continuity of the matching function. A general condition under which the monotonicity condition will hold is where you cannot see round the back of objects, in the sense shown in figure 8.

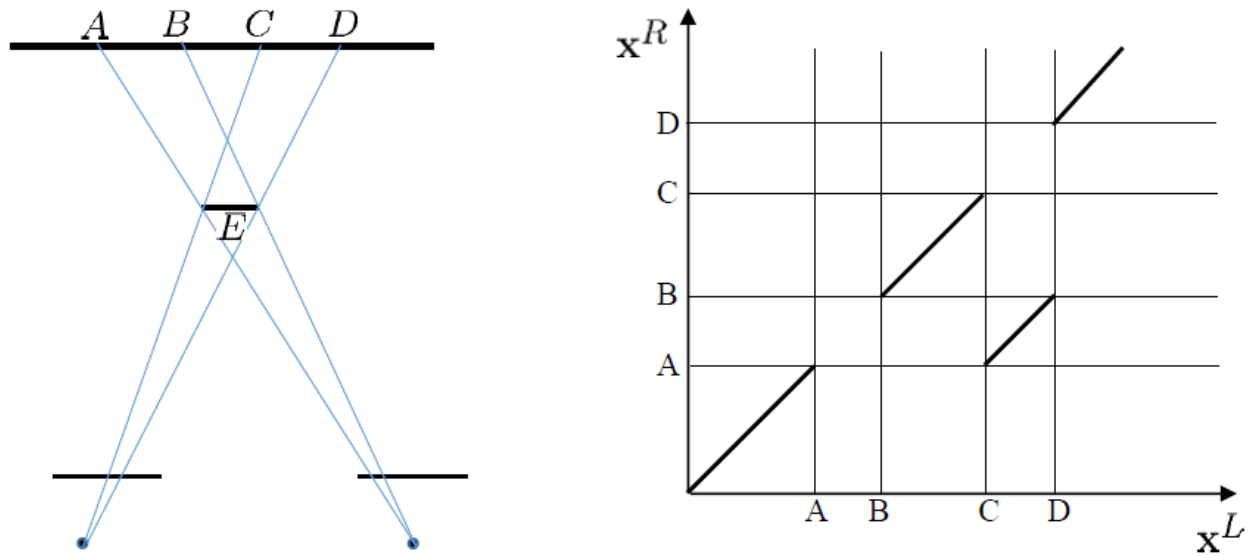


Figure 8: *If it is possible to see "round the back" of an object, as shown in the figure, then monotonicity fails. For instance, point E is seen to the right of point B in the left view, but to the left of B in the right view.*

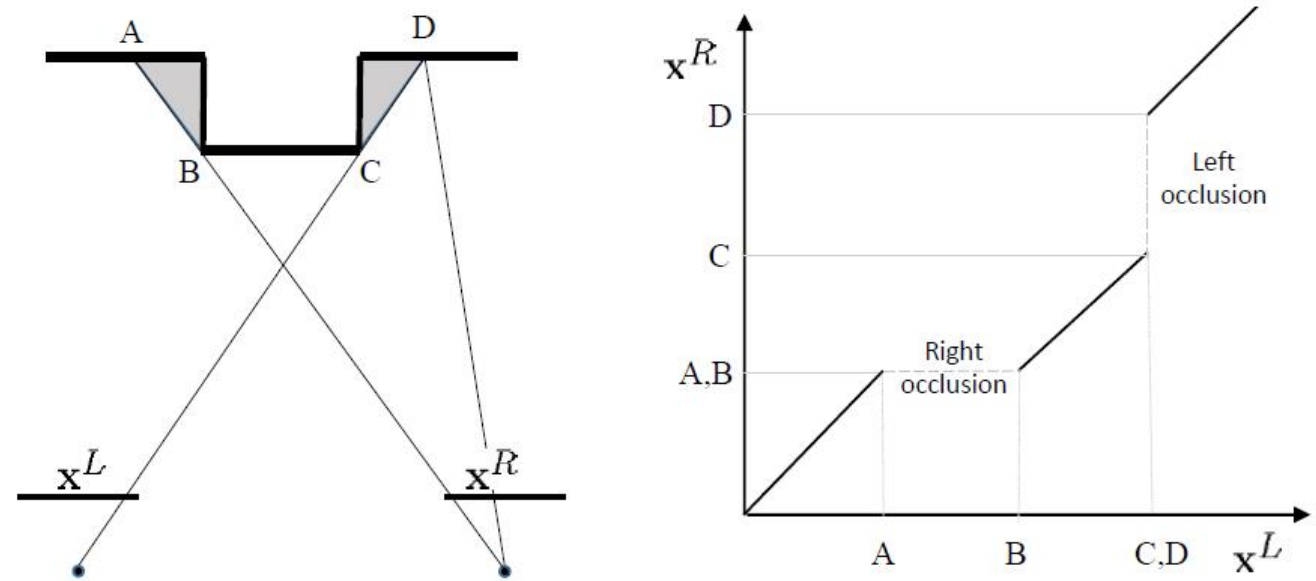
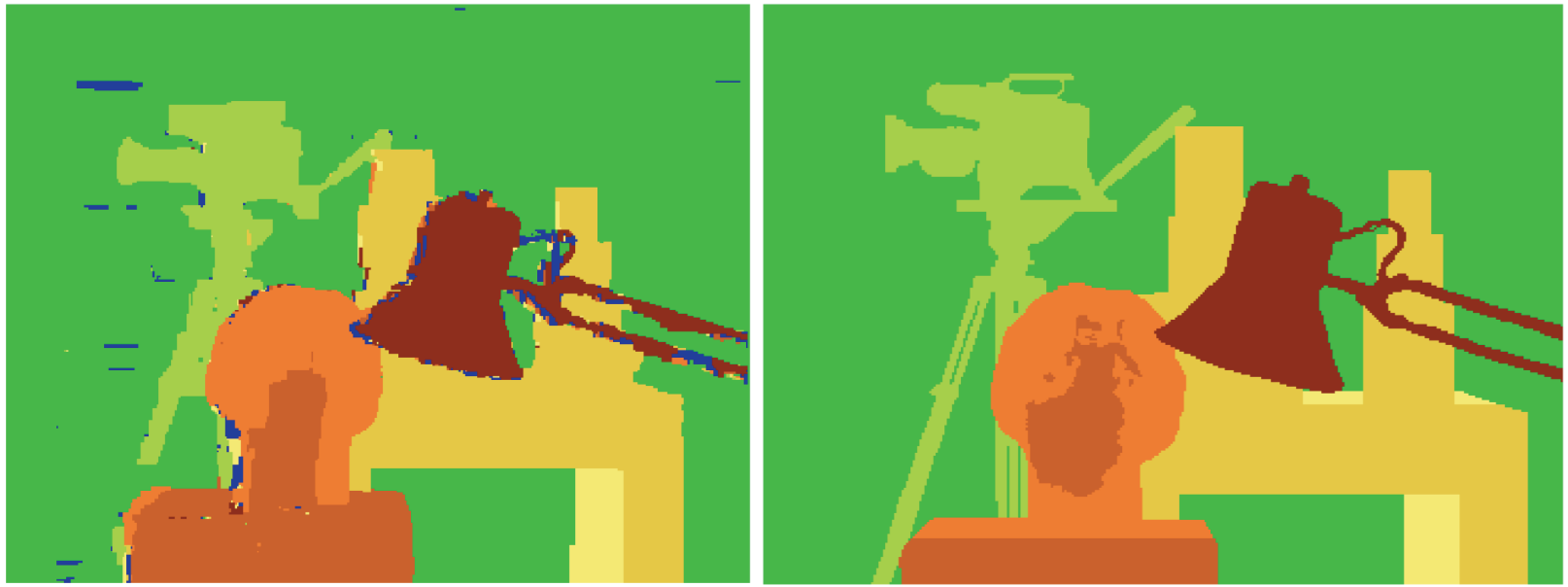


Figure 9: *Monotonicity may still hold if there are occlusions. In this figure, the greyed areas are occluded from either the left of right views. Hence, no matching points are found, leading to gaps in the graph of the matching function (right figure). However, it is possible to fill in the gaps (with horizontal or vertical lines) such that the graph is monotonic non-decreasing. Right and left occlusions are indicated when the graph is horizontal or vertical.*

Graph-cut-based Stereo Matching



State of the art method: Graph cuts

Ground truth

(Seitz)

Semi-global matching (SGM) Semi-global matching (SGM) [7] is a method for rapidly computing a good, but approximate, solution minimizing the MRF. The algorithm figure 7, with the modified forward sweep equation given by (7), will minimize the energy on an MRF consisting of a single scan-line. (This algorithm can be easily modified to work on trees as well.); A slightly different version of this algorithm is given in figure 10, involving identical sweeps in two different directions. The SGM algorithm extends this approach by carrying out sweeps in both horizontal and vertical direction, as well as diagonal directions across the image; it recommends sweeps in 16 directions, indexed by $r = 1, \ldots, R$.

Line 13 of the algorithm in figure 10 is then modified to

$$S_i^*(\lambda) = \sum_{r=1}^{R} S_i^r(\lambda) + E_i(\lambda) \, .$$

1: **procedure** MINMARGINAL
2: **for** $(\lambda \in \mathcal{L})$ **do** // Initialization
3: $S_1^R(\lambda) = 0$
4: $S_{n,\lambda}^L = 0$
5: **for** $i = 2, \ldots, n$ **do** // right sweep
6: **for** $\lambda \in \mathcal{L}$ **do**
7: $S_i^R(\lambda) = \min_{\mu \in \mathcal{L}} \left(S_{i-1}^R(\mu) + E_{i-1,i}(\mu, \lambda) \right)$
8: **for** $i = n-1, \ldots, 1$ **do** // left sweep
9: **for** $\lambda \in \mathcal{L}$ **do**
10: $S_i^L(\lambda) = \min_{\mu \in \mathcal{L}} \left(S_{i+1}^L(\mu) + E_{i,i+1}(\lambda, \mu) \right)$
11: **for** $i = 1, \ldots, n$ **do** // Summing
12: **for** $\lambda \in \mathcal{L}$ **do**
13: $S_i^*(\lambda) = S_i^L(\lambda) + S_i^R(\lambda) + E_i(\lambda)$
14: **for** $i = 1, \ldots, n$ **do** // Finding min-marginal
15: $d_i = \operatorname{argmin}_{\lambda \in \mathcal{L}} S_i^*(\lambda)$

Figure 10: *Sweep-algorithm for finding min-marginals. The value of $S_i^*(\lambda)$ found in this algorithm is equal to the min-marginal, $\min_{\{\mathbf{d} \mid d_i = \lambda\}} E(\mathbf{d})$, in other words, the minimum value of $E(\mathbf{d})$ among those $\mathbf{d}$ with $d_i = \lambda$. Hence, $\min_{\lambda \in \mathcal{L}} S_i^*(\lambda) = \min_{\mathbf{d}} E(\mathbf{d})$ (see line 15). The algorithm gives the optimal label d_i for each i, provided that $\operatorname{argmin}_{\lambda \in \mathcal{L}} S_i^*(\lambda)$ is unique.*

Rectification

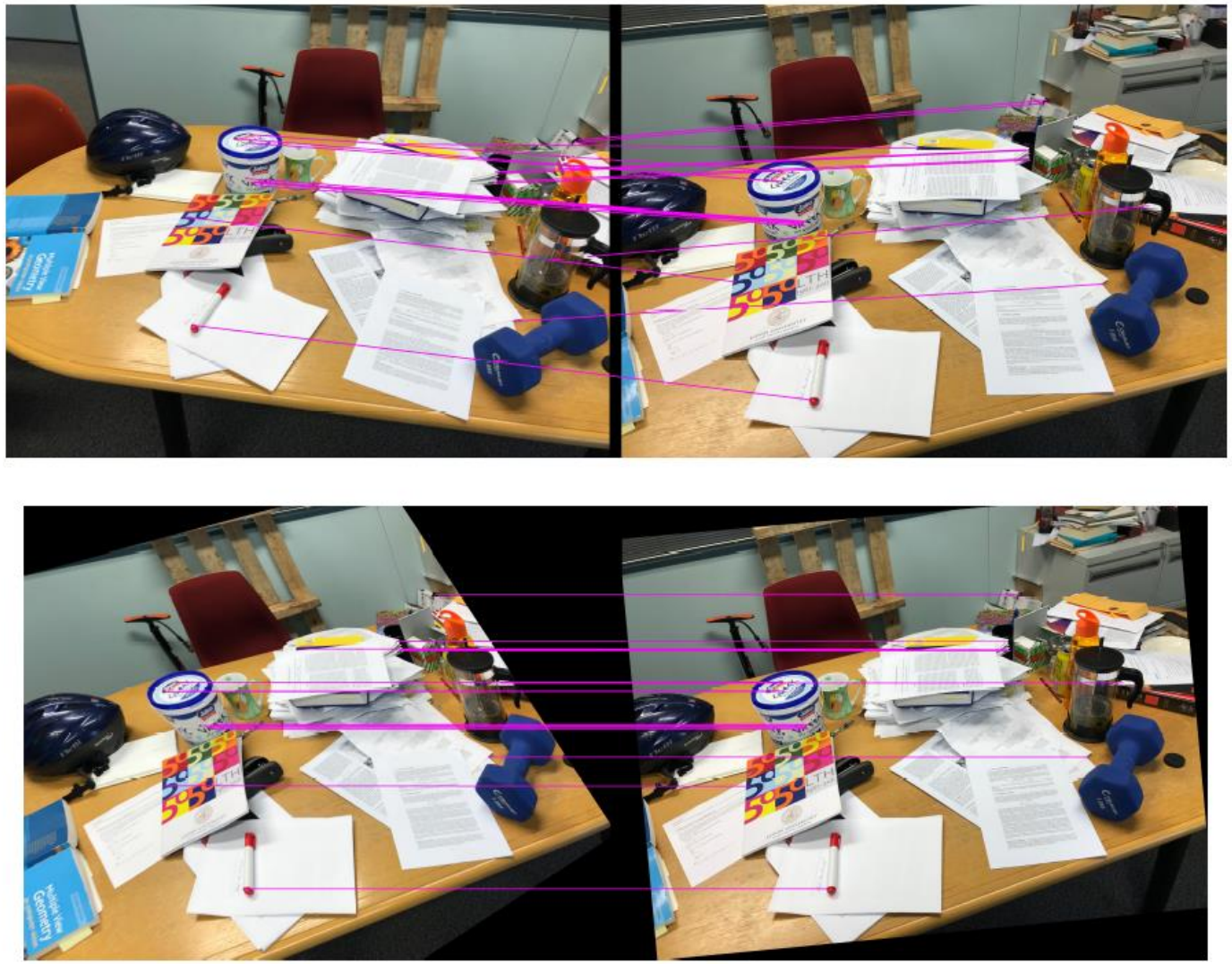


Figure 11: *Pair of images before and after rectification. Lines indicate matching points. After rectification, matching points lie on the same horizontal scan line in the two images.*

State-of-the-art (before 2015)

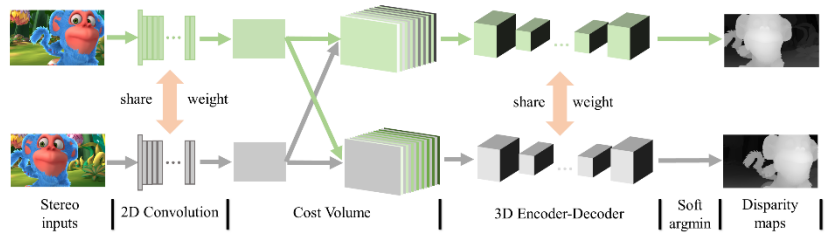


Figure 16: *Network structure of GC-Net [9]. Features are extracted from both images, and are then combined at varying disparity offsets to create a 4-dimensional cost volume of features. The disparities are extracted from the cost volumes using an encoder-decoder network, followed by softmin to find the minimal-cost disparities.*

PSMnet ([5]) further extended the idea of GC-Net by using a multi-scale spatial pyramid pooling module in the feature extraction stage and replacing the 3D-CNN with a stacked hourglass architecture.

7.4 Self-supervised deep stereo matching

All the above-mentioned deep-learning based stereo matching methods directly use ground-truth disparity maps as supervision. Their performance relies on the availability of a large number of training stereo images with corresponding ground-truth disparity maps. However, getting accurate disparity maps is an expensive and laborious task (e.g., KITTI, Middlebury stereo). In contrast, most traditional stereo matching methods do not need ground-truth disparity maps. Instead, they find the best result simply by minimizing a predefined global energy function (such as MRF/CRF energy function). This suggests simply using the energy function as the loss function to train a deep neural network. The deep network works as a mathematical optimizer.

In one approach (sSMNet, [23]) a disparity-prediction network takes a pair of stereo images I^L and I^R as input, and outputs a disparity map $d_R = g(I^R, I^L)$. The left image is then warped using this resulting disparity map to the right viewpoint to obtain a warped new right image $\tilde{I}^R(u,v) = I^L(u+d,v)$. The photometric error between $\tilde{I}^R$ and the original right image I^R can be used as the loss function which is minimized through training, using back- propagation.

No external ground-truth depth map is needed, and any input stereo image can be used for supervision. The network can be trained in an end-to-end fashion. Once the training is complete, the disparity-prediction network (i.e. $g(I^R, I^L)$) can be applied to new (unseen) stereo images, outputting disparity maps directly. Despite not seeing any ground-truth disparity map, the network's depth estimation accuracy is comparable with that of fully-supervised deep methods.

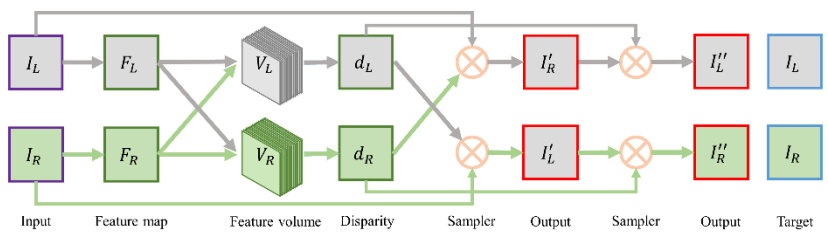


Figure 17: *Overall network structure of SsSMnet. The network first generates disparity maps directly from the left and right input images, in a way similar to figure 16. The disparity maps are then used to warp the input images back and forth. They are then compared with the original images to compute a training loss.*

This approach can be used to process stereo video sequences, adding a recurrent unit (using convolutional LSTMs) to leverage temporal continuity in stereo video matching [24].

Another approach, ([25]) to unsupervised deep-learning uses supervision signals generated by some sparse matches. The network is randomly initialized to produce a random disparity map. Then, a few 'good matches' are selected to drive

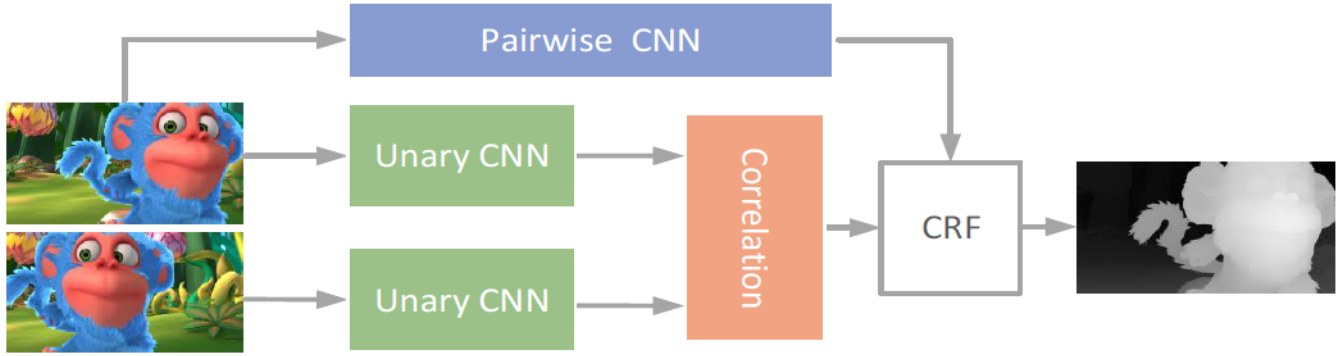


Figure 14: *Hybrid CNN-CRF Net for stereo [11]: The Unary-CNN computes deep features of the two images at each pixel. The features are compared using a correlation layer which produces the unary cost parameters for a CRF. The pairwise (edge) costs for the CRF are parametrized by edge weights, which are learned by the Pairwise-CNN.*

Figure 15 depicts the network architecture of a stereo regression network known as DispNet [17]. This is the first, and a representative deep network for stereo matching following the direct disparity regression approach. In training, the network is trained to produce a disparity field that minimizes the regression loss with respect to the ground-truth disparity map. To train this network, a large-scale computer-graphics simulated stereo dataset was created. Several following-up deep networks with different architecture or training strategies have been proposed in recent years, including Cascade residual learning (CRL) [18] and iResNet [15].

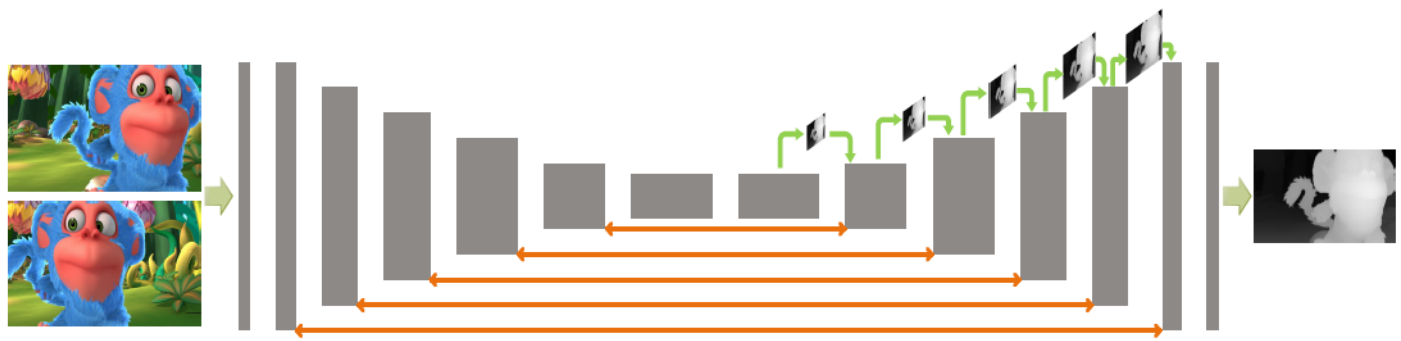
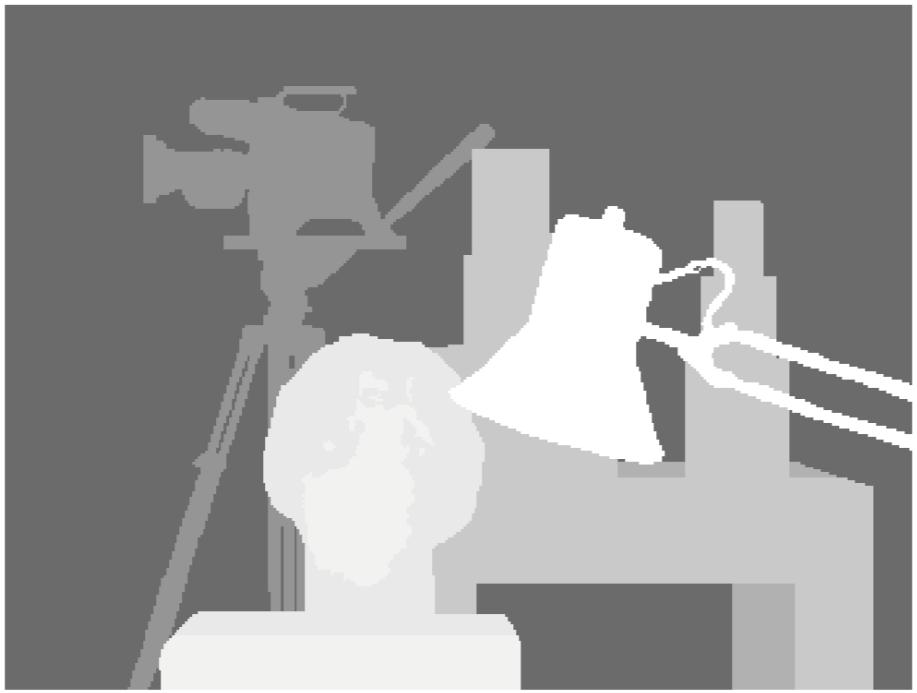


Figure 15: *Overall network structure of DispNet [17]. The network learns to predict disparities directly from the stereo image pair, using a feed-forward network. Because of the large amount of training data required for effective learning, generated computer-graphic images are used for training.*

A Taxonomy of Stereo Algorithms

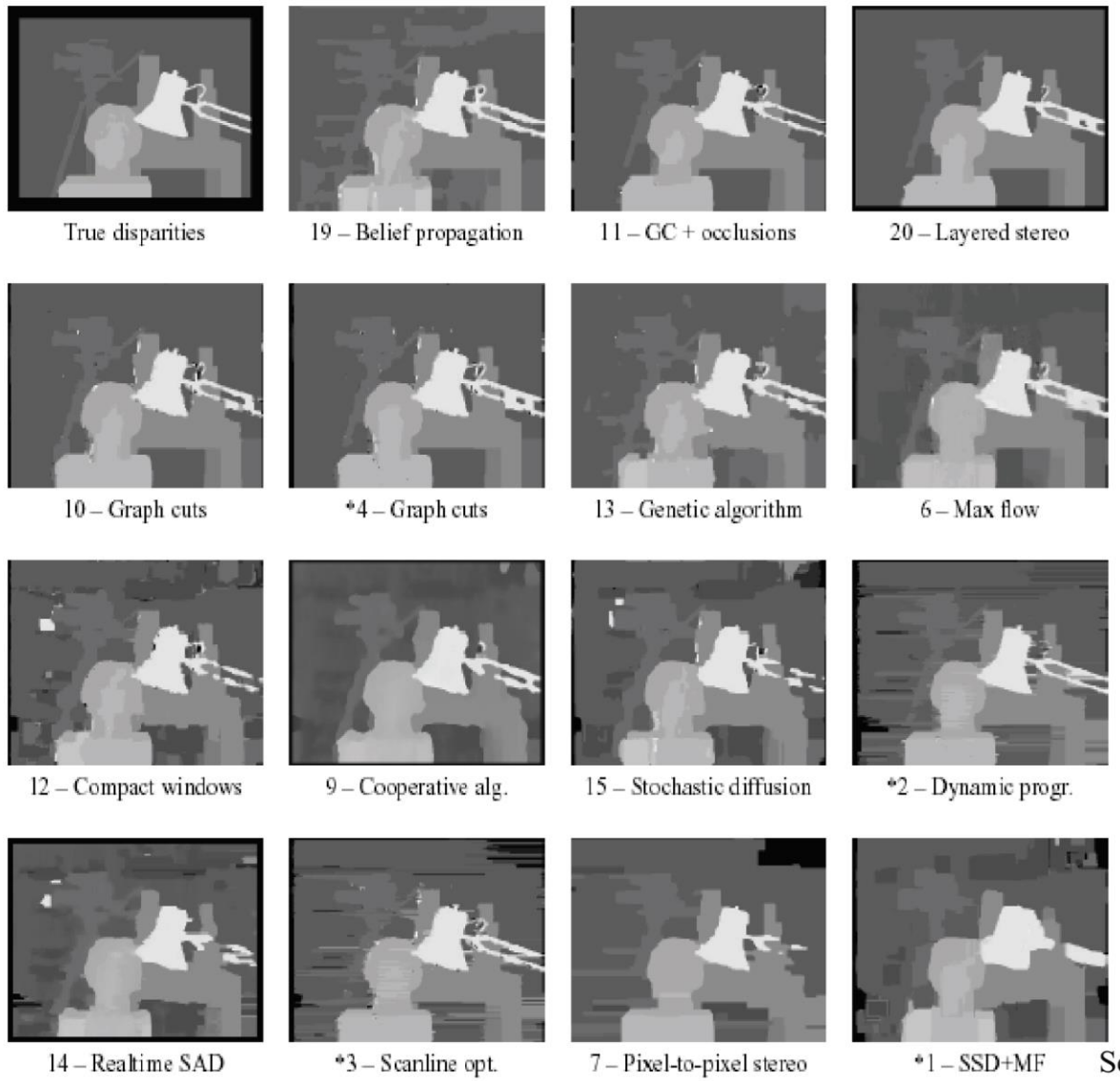
- D. Scharstein and R. Szeliski. "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms," *International Journal of Computer Vision*, 47 (2002), pp. 7-42.

Scene



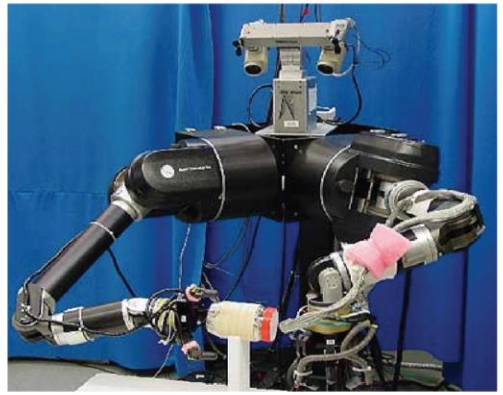
Ground truth

A Taxonomy of Stereo Algorithms



Scharstein and Szeliski

Applications of Stereo Vision in Robotics



References

[1] Thalaiyasingam Ajanthan, Richard Hartley, Mathieu Salzmann, and Hongdong Li. Iteratively reweighted graph cut for multi-label MRFs with non-convex priors. *CVPR*, 2015.

[2] P. N. Belhumeur and D. Mumford. A bayesian treatment of the stereo correspondence problem using half-occluded regions. In *Proceedings 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 506–512, June 1992.

[3] Julian Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society*, 1974.

[4] P.J. Burt and E.H. Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31(4):532–540, 1983.

[5] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418, 2018.

[6] R.I. Hartley. Theory and practice of projective rectification. *International Journal on Computer Vision*, 35(2):115–127, 1999.

[7] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, Feb 2008.

[8] Hiroshi Ishikawa. Exact optimization for Markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1333–1336, 2003.

[9] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *Proc. IEEE Int. Conf. Comp. Vis.*, Oct 2017.

[10] Junhwan Kim, Kolmogorov, and Zabih. Visual correspondence using energy minimization and mutual information. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 1033–1040 vol.2, Oct 2003.

[11] Patrick Knbelreiter, Christian Reinbacher, Alexander Shekhovtsov, and Thomas Pock. End-to-End Training of Hybrid CNN-CRF Models for Stereo. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2017.

[12] Vladimir Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1568 – 1583, October 2006.

[13] Vladimir Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *PAMI*, 2006.

[14] Vladimir Kolmogorov and Ramin Zabin. What energy functions can be minimized via graph cuts? *PAMI*, 2004.

[15] Zhengfa Liang, Yiliu Feng, Yulan Guo, Hengzhu Liu, Wei Chen, Linbo Qiao, Li Zhou, and Jianfeng Zhang. Learning for disparity estimation through feature constancy. In *Computer Vision and Pattern Recognition*, 2018.

[16] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, September 1981.

[17] N. Mayer, E. Ilg, P. Husser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2016.

[18] Jiahao Pang, Wenxiu Sun, Jimmy SJ. Ren, Chengxi Yang, and Qiong Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *International Conf. on Computer Vision - Workshop on Geometry Meets Deep Learning (ICCVW)*, 2017.

[19] Lynn H. Quam. Hierarchical warp stereo. In Martin A. Fischler and Oscar Firschein, editors, *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, pages 80–86. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1987.

[20] Dmitrij Schlesinger and Boris Flach. *Transforming an arbitrary minsum problem into a binary one*. TU, Fak. Informatik, 2006.

[21] Olga Veksler. Multi-label moves for MRFs with truncated convex priors. *IJCV*, 2012.

[22] Jure Zbontar and Yann LeCun. Computing the stereo matching cost with a convolutional neural network. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2015.

[23] Yiran Zhong, Yuchao Dai, and Hongdong Li. Self-supervised learning for stereo matching with self-improving ability. In *arXiv:1709.00930*, 2017.

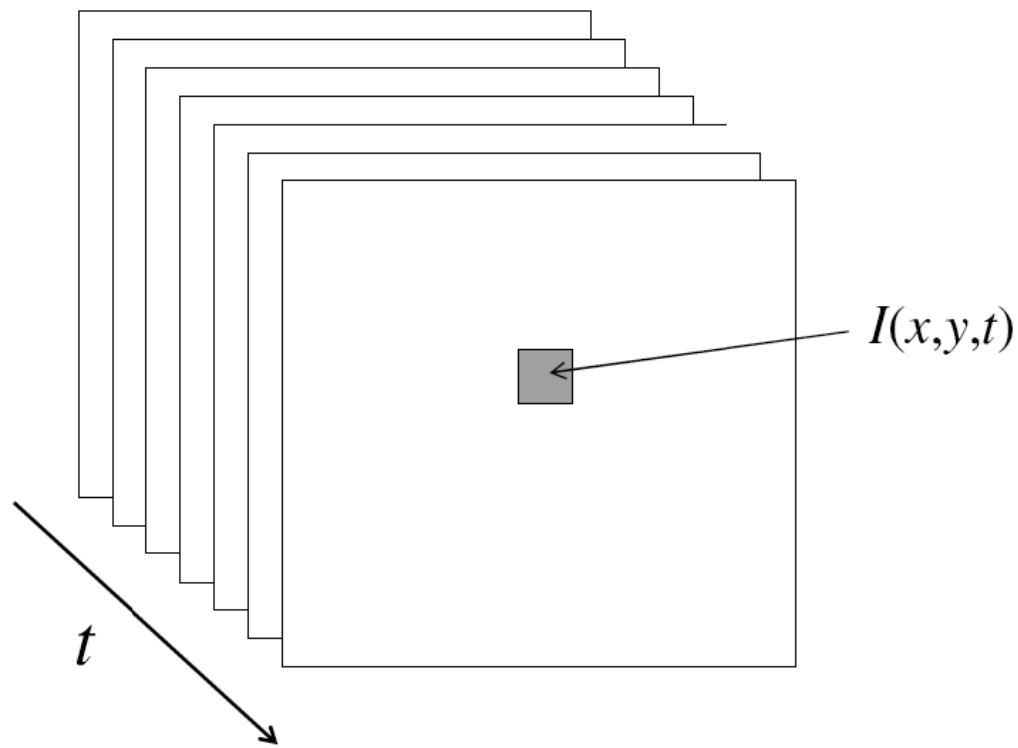
[24] Yiran Zhong, Yuchao Dai, and Hongdong Li. Open-world stereo video matching with deep rnn. In *Proc. Eur. Conf. Comp. Vis.*, 2018.

[25] Chao Zhou, Hong Zhang, Xiaoyong Shen, and Jiaya Jia. Unsupervised learning of stereo matching. In *Proc. IEEE Int. Conf. Comp. Vis.*, Oct 2017.

Optical Flow

Video

- A video is a sequence of frames captured over time
- Now our image data is a function of space (x, y) and time (t)



Credit: K. Grauman

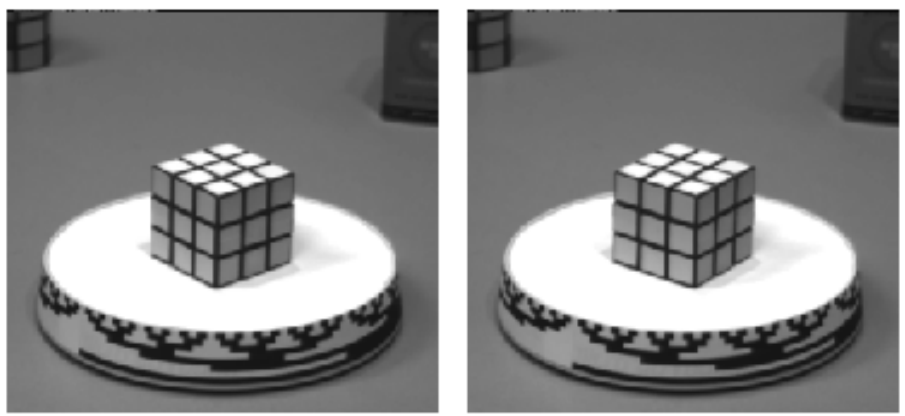
Uses of motion

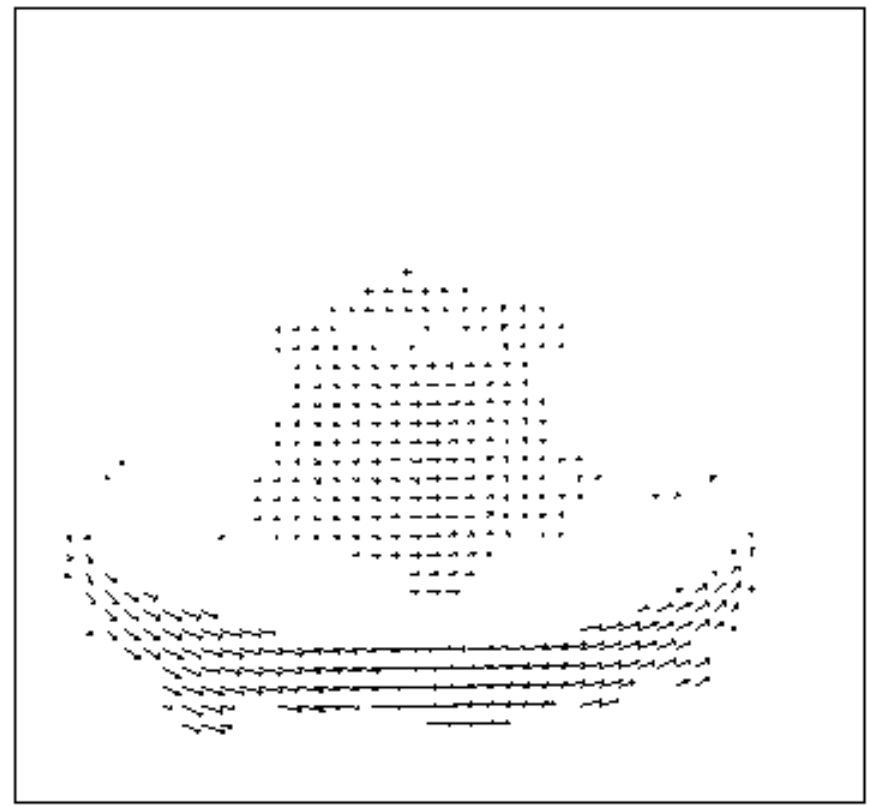
- Estimating 3D structure
- Segmenting objects based on motion cues
- Learning dynamical models
- Recognizing events and activities
- Improving video quality (motion stabilization)

Credit: K. Grauman

Motion field

- The motion field is the projection of the 3D scene motion into the image

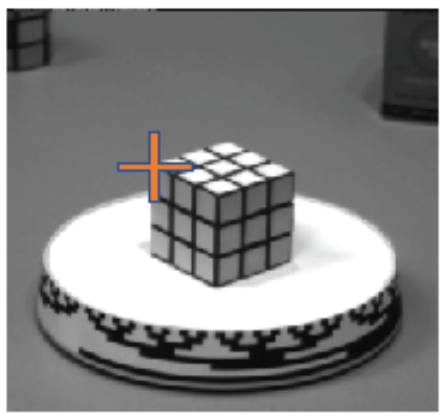




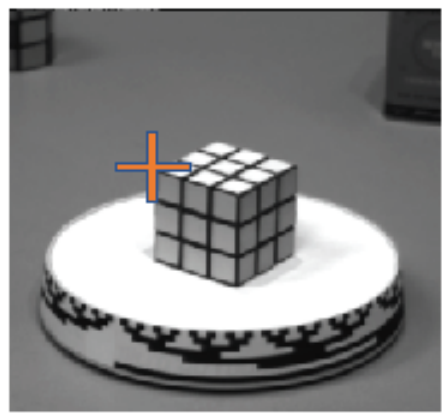
Credit: K. Grauman

Motion field

- The motion field is the projection of the 3D scene motion into the image

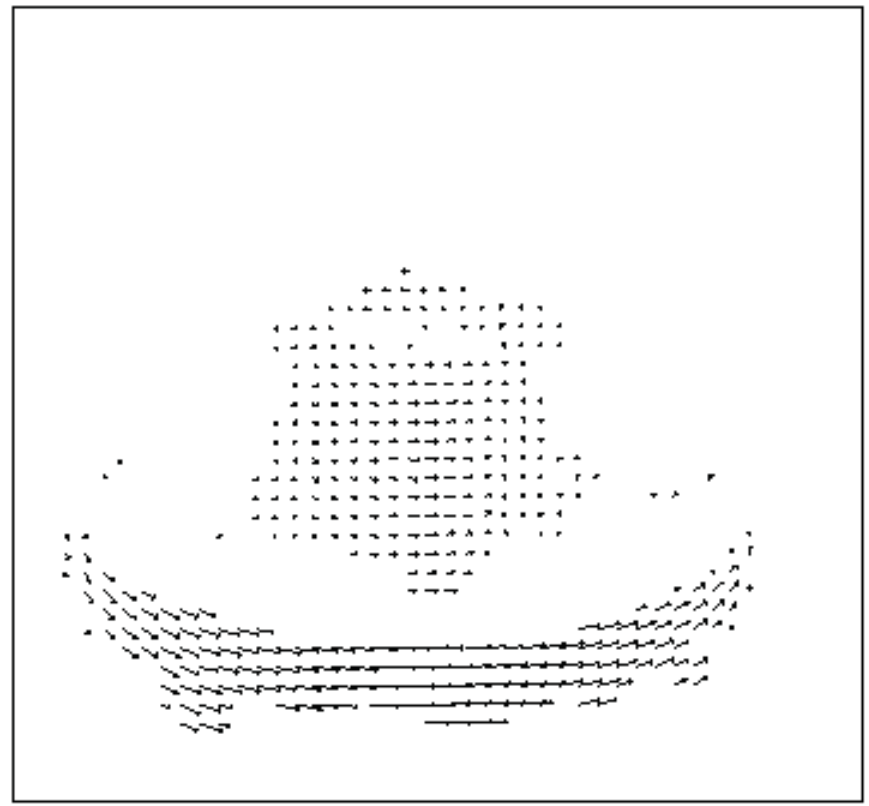


(100, 100)



(101, 102)

(u,v) = (1, 2)



Credit: K. Grauman

Motion field + camera motion

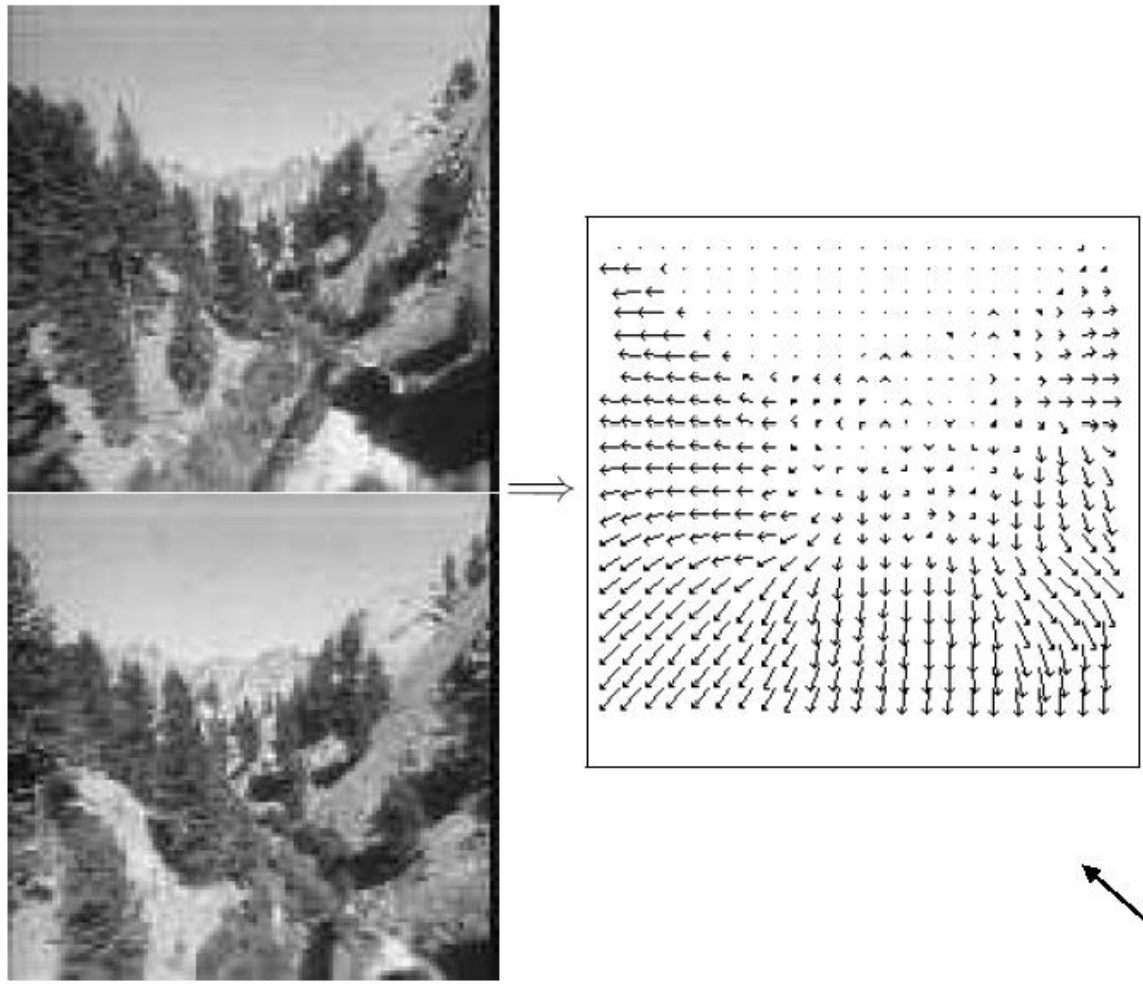


Figure 1.2: Two images taken from a helicopter flying through a canyon and the computed optical flow field.

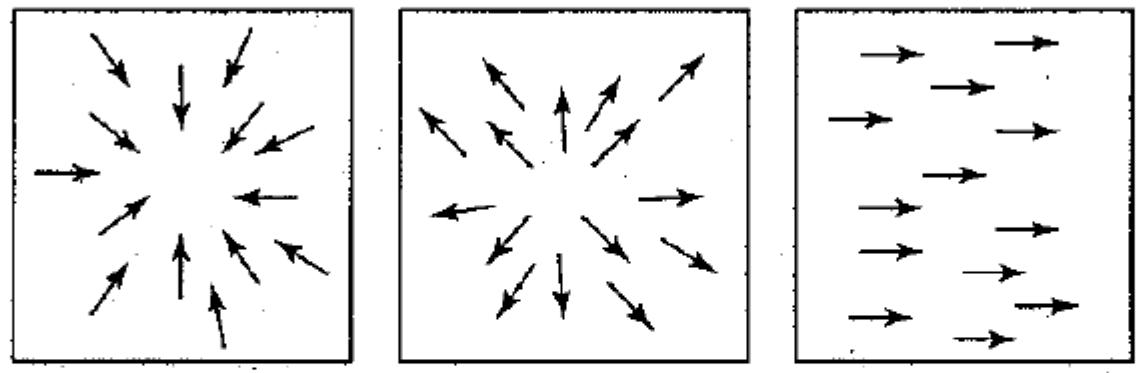
Length of flow vectors inversely proportional to depth Z of 3d point

points closer to the camera move more quickly across the image plane

Figure from Michael Black, Ph.D. Thesis

Credit: K. Grauman

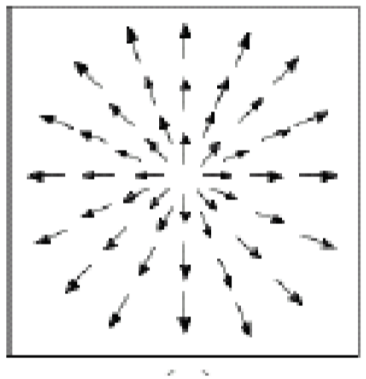
Motion field + camera motion



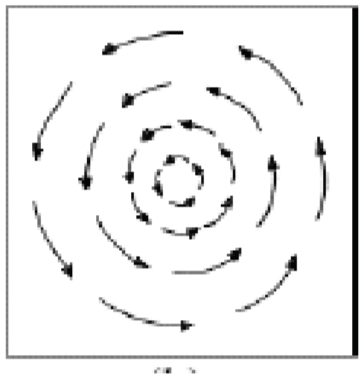
Zoom out Zoom in Pan right to left

Credit: K. Grauman

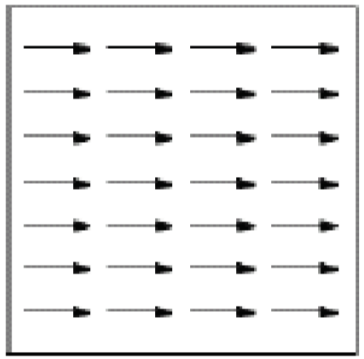
Some Simple Motion Fields



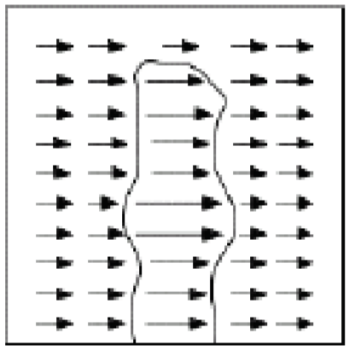
Forward motion



Rotation



Horizontal translation

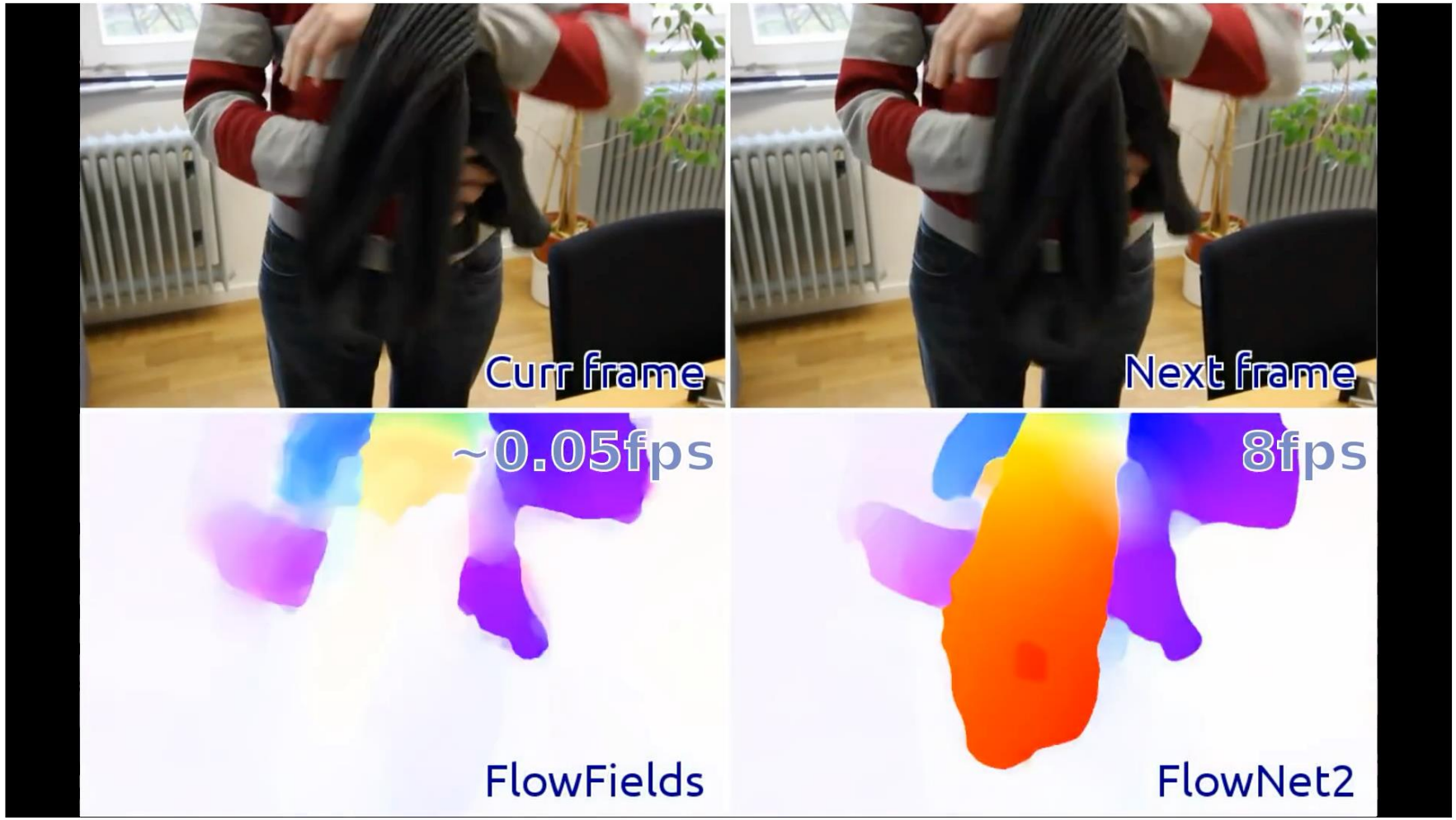


Closer objects appear to move faster!!

Optical flow

- Definition: optical flow is the *apparent* motion of brightness patterns in the image

Credit: K. Grauman



https://www.youtube.com/watch?v=JSzUdVBmQP4 FlowNet

Optical flow

- Definition: optical flow is the *apparent* motion of brightness patterns in the image
- Ideally, optical flow would be the same as the motion field
- Have to be careful: apparent motion can be caused by lighting changes without any actual motion

Credit: K. Grauman

Apparent motion != motion field

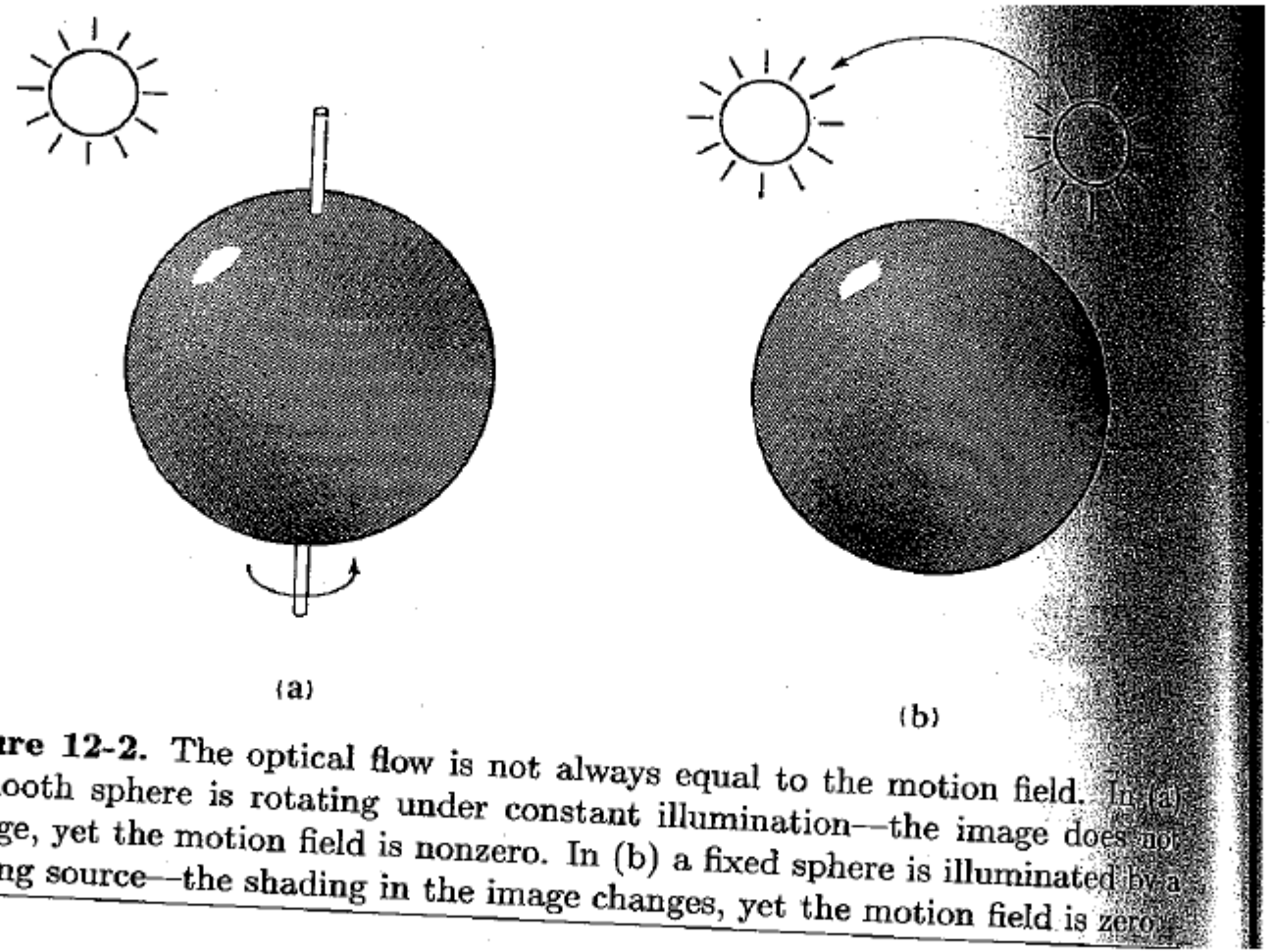
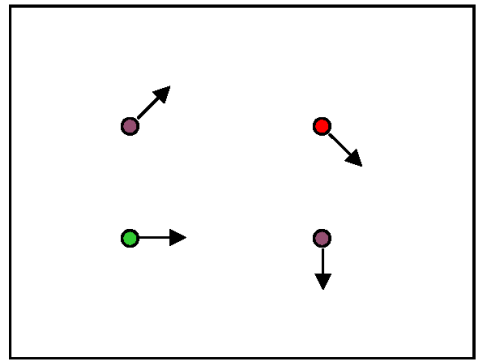


Figure 12-2. The optical flow is not always equal to the motion field. In (a) a smooth sphere is rotating under constant illumination—the image does not change, yet the motion field is nonzero. In (b) a fixed sphere is illuminated by a moving source—the shading in the image changes, yet the motion field is zero.

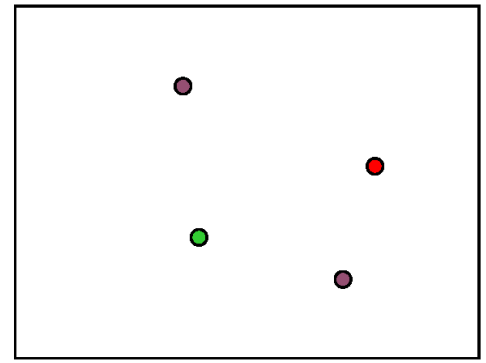
Figure from Horn book

Credit: K. Grauman

Problem definition: optical flow



$H(x, y)$



$I(x, y)$

- How to estimate pixel motion from image H to image I?
  - Solve pixel correspondence problem
    - given a pixel in H, look for nearby pixels of the same color in I

Key assumptions

- **color constancy:** a point in H looks the same in I
  - For grayscale images, this is **brightness constancy**
- **small motion**: points do not move very far

This is called the **optical flow** problem

Credit: K. Grauman

Color/brightness constancy

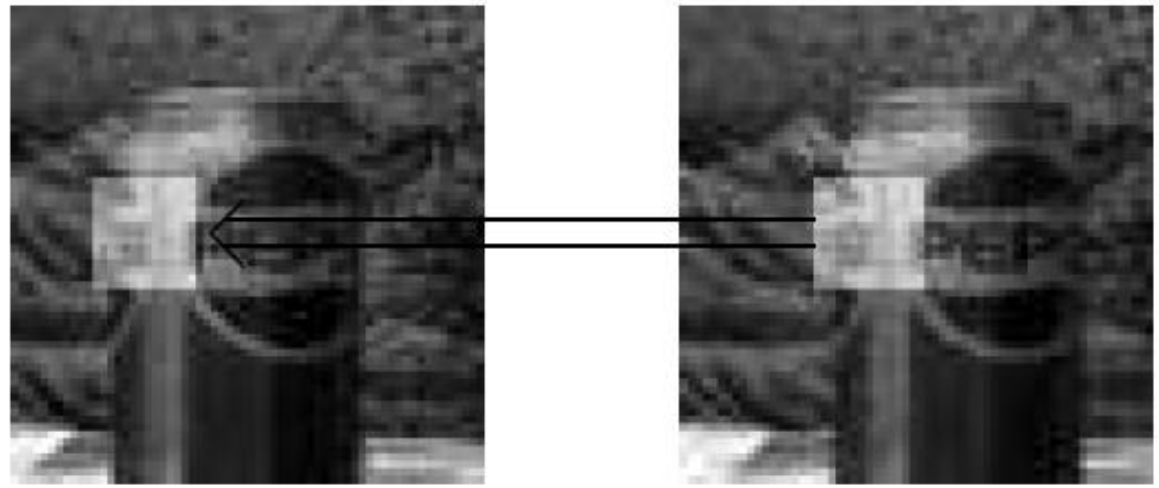
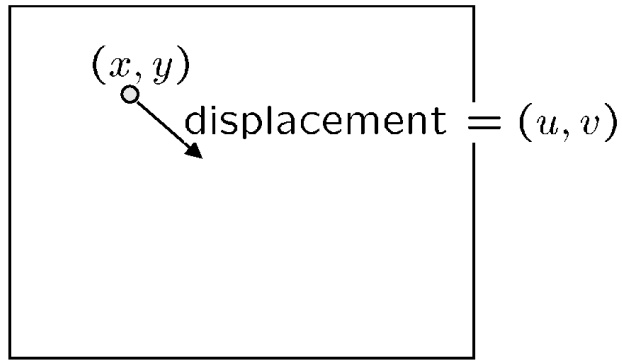


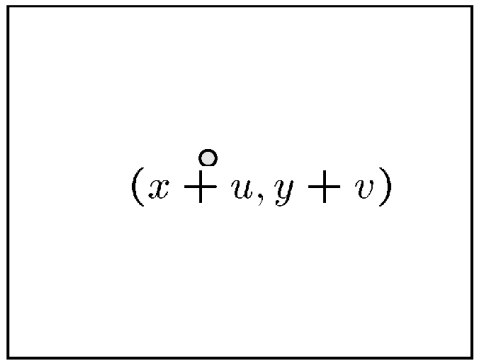
Figure 1.5: Data conservation assumption. The highlighted region in the right image looks roughly the same as the region in the left image, despite the fact that it has moved.

Figure by Michael Black

Optical flow constraints



$H(x, y)$



$I(x, y)$

- Let's look at these constraints more closely
  - brightness constancy: Q: what's the equation?

$$H(x,y) = I(x+u, y+v)$$

  - small motion:

$$I(x+u, y+v) = I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$

$$\approx I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

Optical flow equation

shorthand: $I_x = \frac{\partial I}{\partial x}$

- Combining these two equations

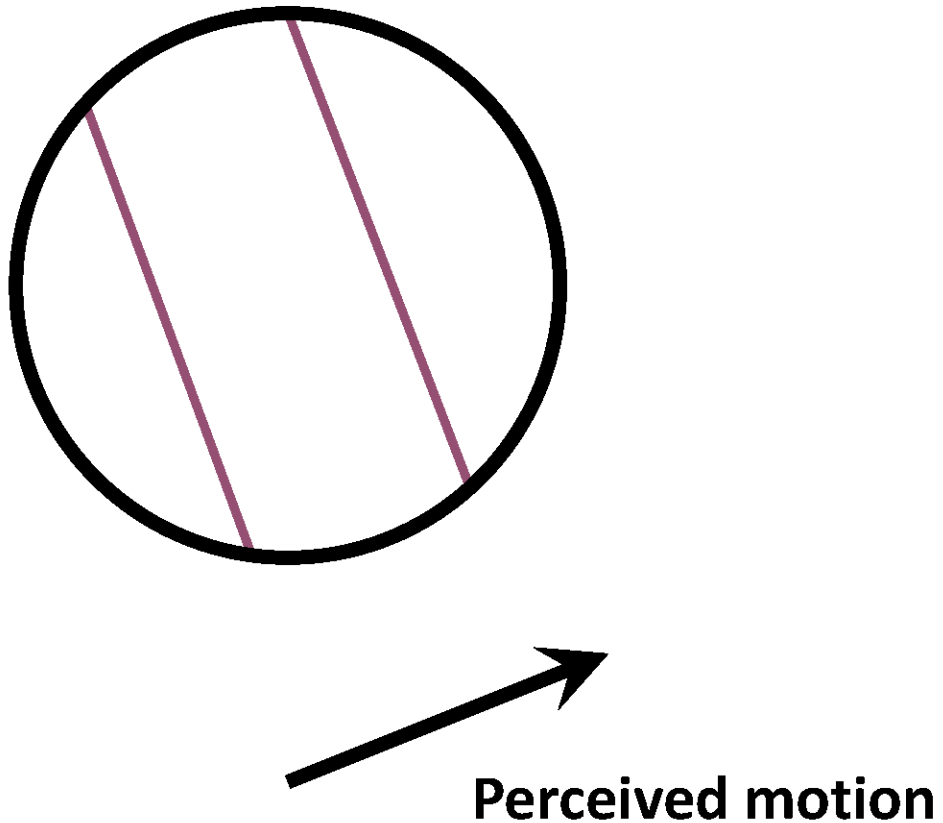
$$
\begin{aligned}
0 &= I(x+u, y+v) - H(x,y) \\
&\approx I(x,y) + I_x u + I_y v - H(x,y) \\
&\approx (I(x,y) - H(x,y)) + I_x u + I_y v \\
&\approx I_t + I_x u + I_y v \\
&\approx I_t + \nabla I \cdot [u \ v]
\end{aligned}
$$

Optical flow equation

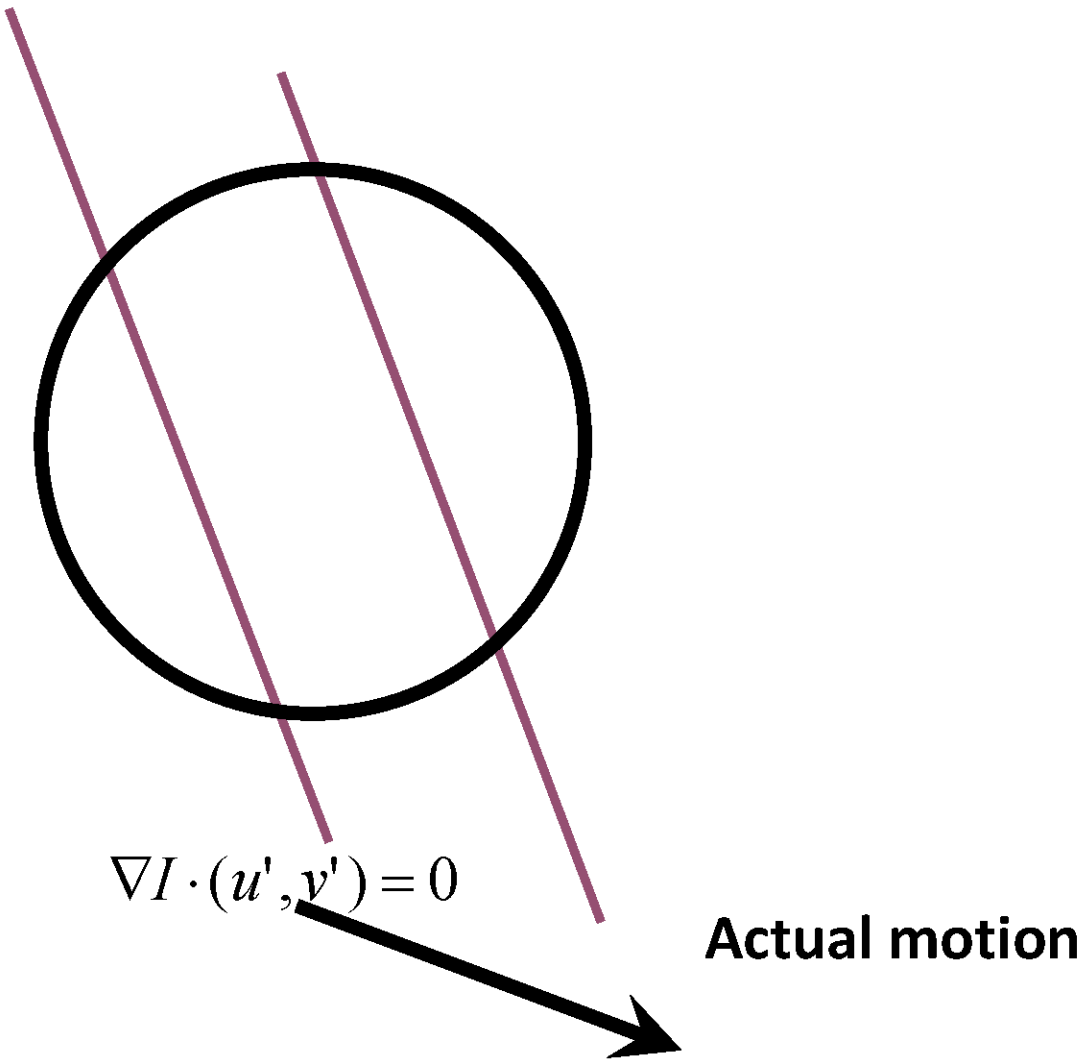
$$0 = I_t + \nabla I \cdot [u \ v]$$

- Q: how many unknowns and equations per pixel?

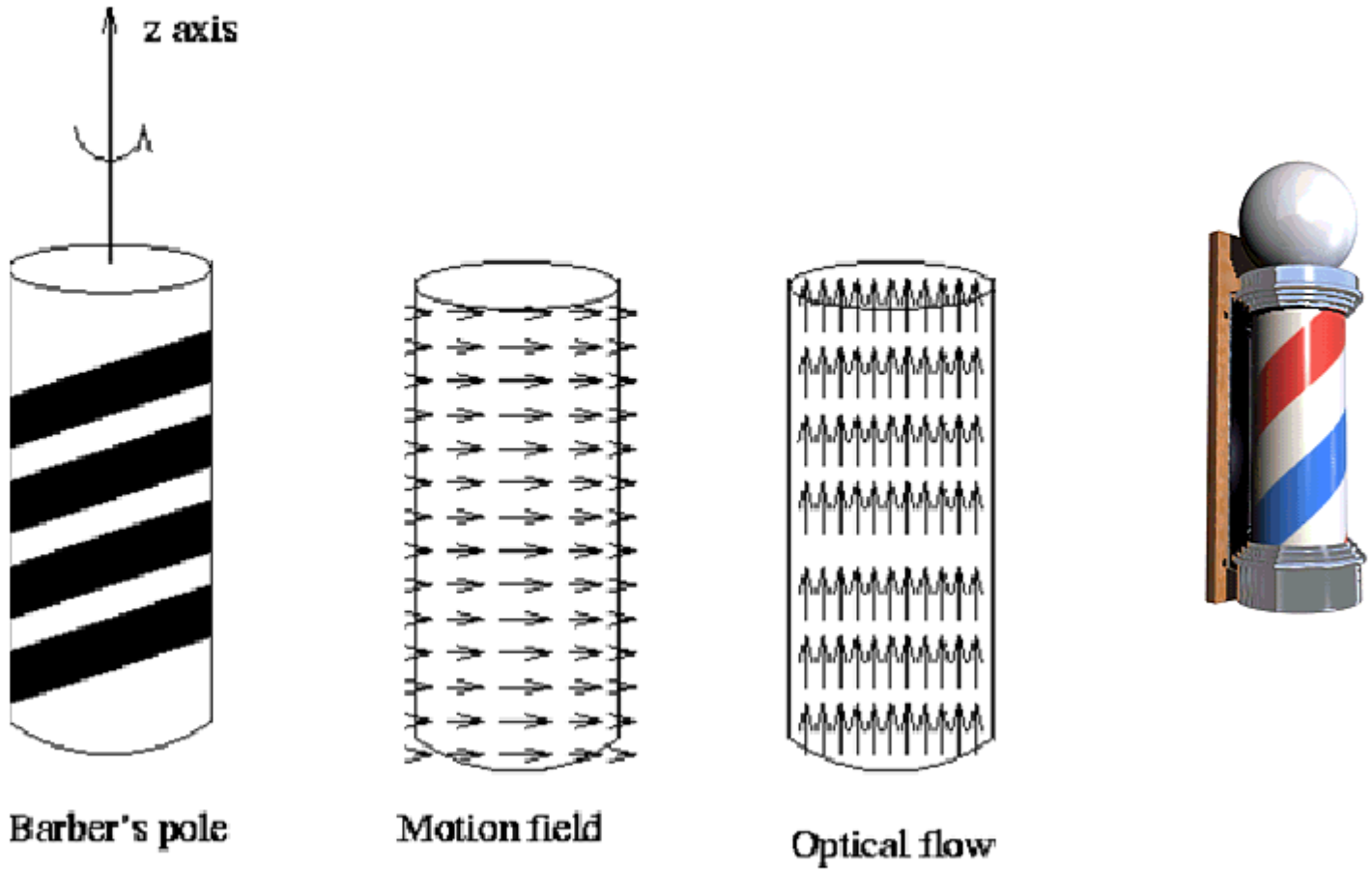
The aperture problem



The aperture problem



The barber pole illusion



http://en.wikipedia.org/wiki/Barberpole_illusion

Solving the aperture problem

- How to get more equations for a pixel?
- **Spatial coherence constraint:** pretend the pixel's neighbors have the same (u,v)

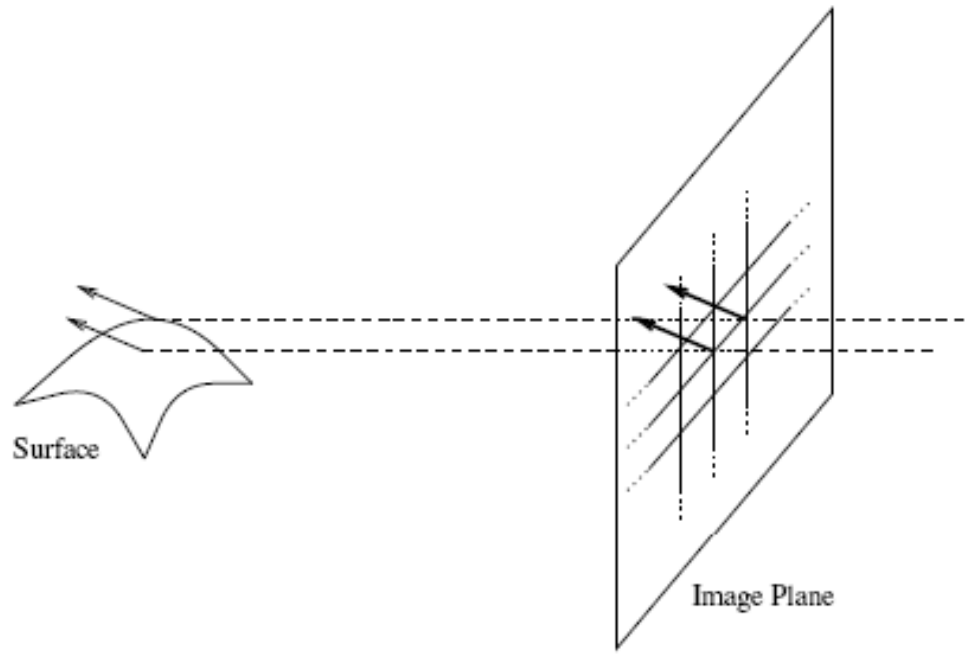


Figure 1.7: Spatial coherence assumption. Neighboring points in the image are assumed to belong to the same surface in the scene.

Figure by Michael Black

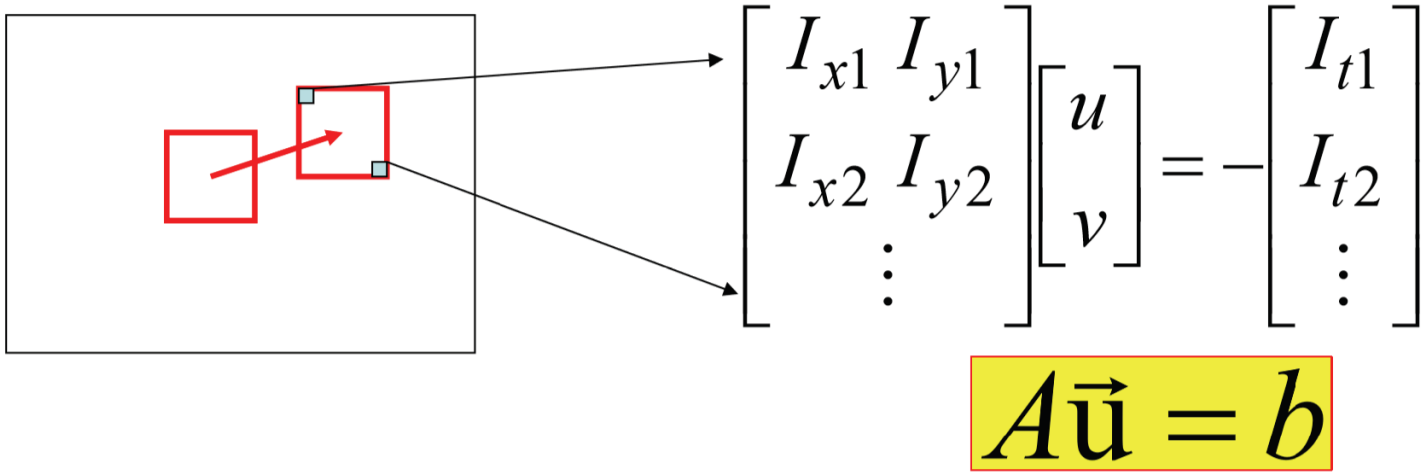
Solving the aperture problem

- How to get more equations for a pixel?
- **Spatial coherence constraint:** pretend the pixel's neighbors have the same (u,v)
  - If we use a 5x5 window, that gives us 25 equations per pixel $0 = I_t(\mathbf{p_i}) + \nabla I(\mathbf{p_i}) \cdot [u \ v]$

Local Patch Constant Motion: Lucas–Kanade optical flow algorithm

$$I_x u + I_y v = -I_t \quad \Rightarrow \quad \begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = -I_t$$

Assume constant motion (u,v) in small neighborhood



$$\begin{bmatrix} I_{x1} & I_{y1} \\ I_{x2} & I_{y2} \\ \vdots & \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_{t1} \\ I_{t2} \\ \vdots \end{bmatrix}$$

$$A\vec{u} = b$$

Lucas–Kanade Optical Flow Algorithm

Goal: Minimize $\left\| A\vec{u} - b \right\|^2$

Method: Least-Squares

$$A\vec{u} = b$$

$$\Downarrow$$

$$\underbrace{A^T A}_{2x2}\ \underbrace{\vec{u}}_{2x1} = \underbrace{A^T b}_{2x1}$$

$$\Downarrow$$

$$\vec{u} = \left(A^T A\right)^{-1} A^T b$$

Solving the aperture problem

$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

- The summations are over all pixels in the K x K window
- This technique was first proposed by Lucas & Kanade (1981)

Slide credit: Steve Seitz

Conditions for solvability

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

$A^T A$ $\qquad$ $A^T b$

When is this solvable?

- **A**^{T}**A** should be invertible
- **A**^{T}**A** should not be very small
  - eigenvalues λ_1 and λ_2 of **A**^{T}**A** should not be very small
- **A**^{T}**A** should be well-conditioned
  - λ_1 / λ_2 should not be too large (λ_1 = larger eigenvalue)

Slide by Steve Seitz, UW

Edge

– gradients very large or very small

– large λ_1, small λ_2

Low-texture region

– gradients have small magnitude

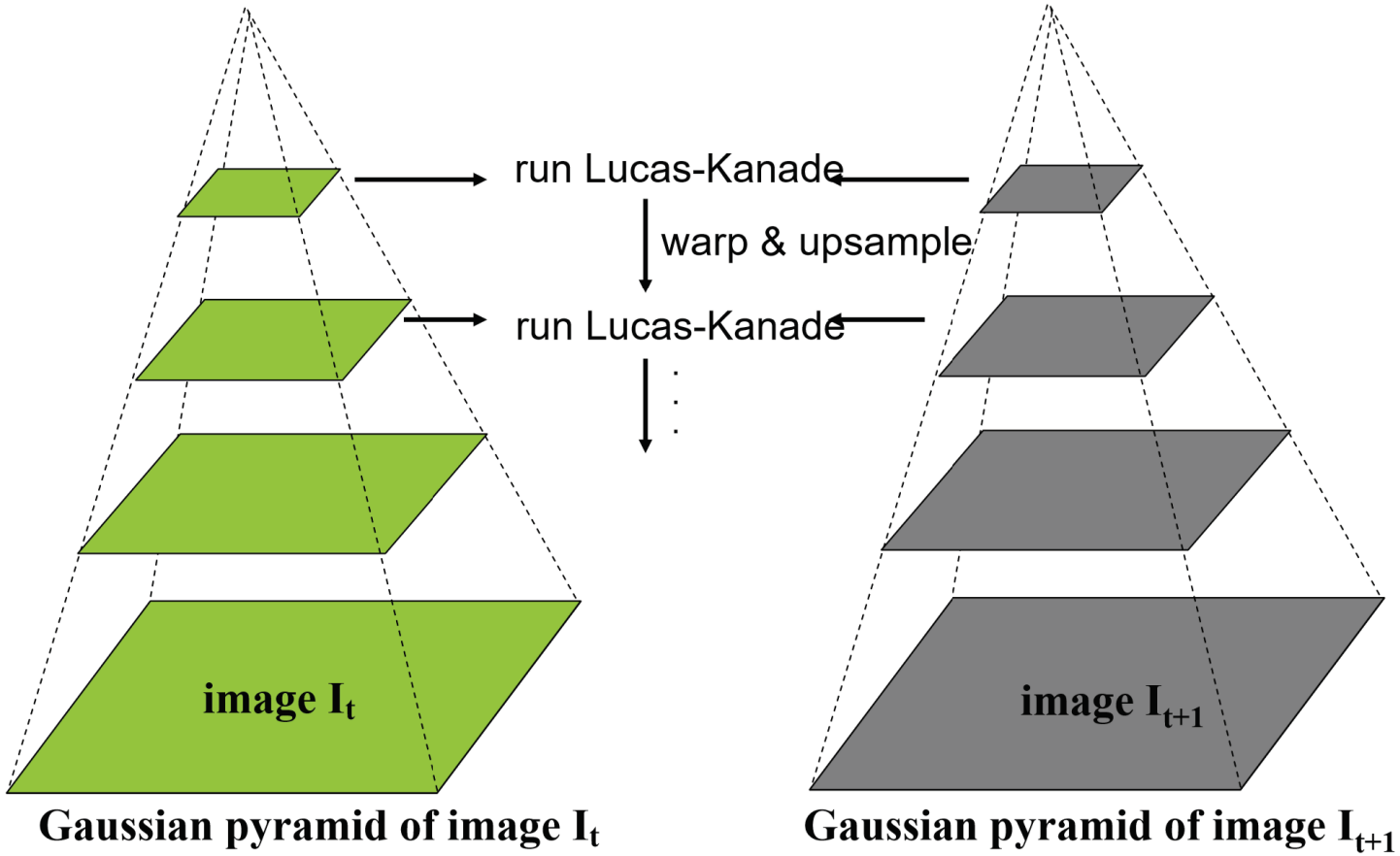
– small λ_1, small λ_2

High-texture region

– gradients are different, large magnitudes

– large λ_1, large λ_2

Multi-Scale Pyramid Estimation



Horn–Schunck Optical Flow Algorithm (1981)

- Regularisation: use a global smoothness term

Smoothness error: $$E_s = \iint_D \left(u_x^2 + u_y^2\right) + \left(v_x^2 + v_y^2\right) dx\, dy$$

Error in brightness constancy equation: $$E_c = \iint_D \left(I_x u + I_y v + I_t\right)^2 dx\, dy$$

Minimize: $E_c + \lambda E_s$

Solve by calculus of variations

Variational Minimization

The Euler-Lagrange equations :

$$F_u - \frac{\partial}{\partial x} F_{u_x} - \frac{\partial}{\partial y} F_{u_y} = 0$$

$$F_v - \frac{\partial}{\partial x} F_{v_x} - \frac{\partial}{\partial y} F_{v_y} = 0$$

In our case ,

$$F = (u_x^2 + u_y^2) + (v_x^2 + v_y^2) + \lambda (I_x u + I_y v + I_t)^2,$$

so the Euler-Lagrange equations are

$$\Delta u = \lambda (I_x u + I_y v + I_t) I_x,$$

$$\Delta v = \lambda (I_x u + I_y v + I_t) I_y,$$

$\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ is the Laplacian operator

Horn–Schunck Optical Flow Algorithm

1. Coupled PDEs solved using iterative methods and finite differences

$$\frac{\partial u}{\partial t} = \Delta u - \lambda(I_x u + I_y v + I_t)I_x,$$

$$\frac{\partial v}{\partial t} = \Delta v - \lambda(I_x u + I_y v + I_t)I_y,$$

2. More than two frames allow a better estimation of I_t

3. Information spreads from corner-type patterns

Optical Flow: Summary

- Optical flow definition

- Lucas–Kanade optical flow algorithm (KLT tracker)

B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 674–679, 1981.

- Horn–Schunck optical flow algorithm

B.K.P. Horn and B.G. Schunck, "Determining optical flow." *Artificial Intelligence, 1981*

SLAM
Simultaneous Localization And Mapping.

Acknowledgements

The images in this document were created by Paul Beardsley, Antonio Criminisi, Andrew Fitzgibbon, David Liebowitz, Luc Robert, and Phil Torr.