Multi-View Geometry: Find Corresponding Points (New book: Ch7.4, 7.5, 7.6 Old book: 11.3-11.5)

Guido Gerig
CS-GY 6643, Spring 2016
gerig@nyu.edu

Credit for materials: Trevor Darrell, Berkeley, C280, Marc Pollefeys, UNC/ETH-Z, CS6320 S012, Andrew Zisserman, MVG Book

Excellent Website: http://vision.middlebury.edu/stereo/

vision.middlebury.edu

stereo • mview • MRF • flow • color

Stereo

Evaluation • Datasets • Code • Submit

Daniel Scharstein • Richard Szeliski

Welcome to the Middlebury Stereo Vision Page, formerly located at www.middlebury.edu/stereo. This website accompanies our taxonomy and comparison of two-frame stereo correspondence algorithms [1]. It contains:

- An on-line evaluation of current algorithms
- Many stereo datasets with ground-truth disparities
- Our stereo correspondence software
- An on-line submission script that allows you to evaluate your stereo algorithm in our framework

How to cite the materials on this website:
We grant permission to use and publish all images and numerical results on this website. If you report performance results, we request that you cite our paper [1]. Instructions on how to cite our datasets are listed on the datasets page. If you want to cite this website, please use the URL "**vision.middlebury.edu/stereo/**".

References:

[1] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1/2/3):7-42, April-June 2002. Microsoft Research Technical Report MSR-TR-2001-81, November 2001.

Support for this work was provided in part by NSF CAREER grant 9984485 and NSF grant IIS-0413169. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Stereo reconstruction: main steps

– Calibrate cameras

– Rectify images

– Compute disparity

– Estimate depth

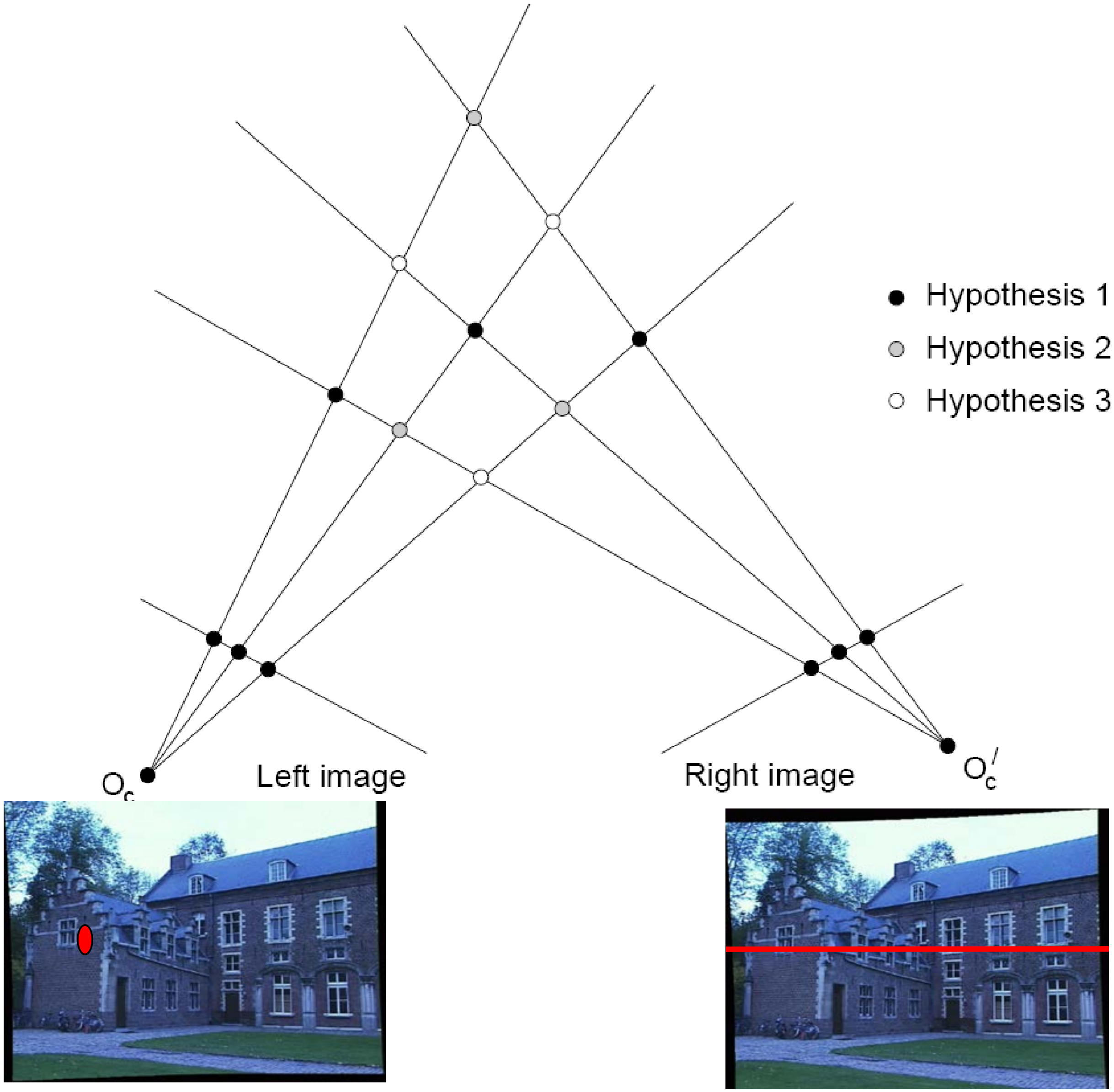
Grauman

Stereo reconstruction: main steps

- Calibrate cameras
- Rectify images
- Compute disparity
- Estimate depth

Grauman

Correspondence problem



Multiple match hypotheses satisfy epipolar constraint, but which is correct?

Figure from Gee & Cipolla 1999

Grauman

Correspondence problem

- Beyond the hard constraint of epipolar geometry, there are “soft” constraints to help identify corresponding points
  - Similarity
  - Uniqueness
  - Ordering
  - Disparity gradient

Grauman

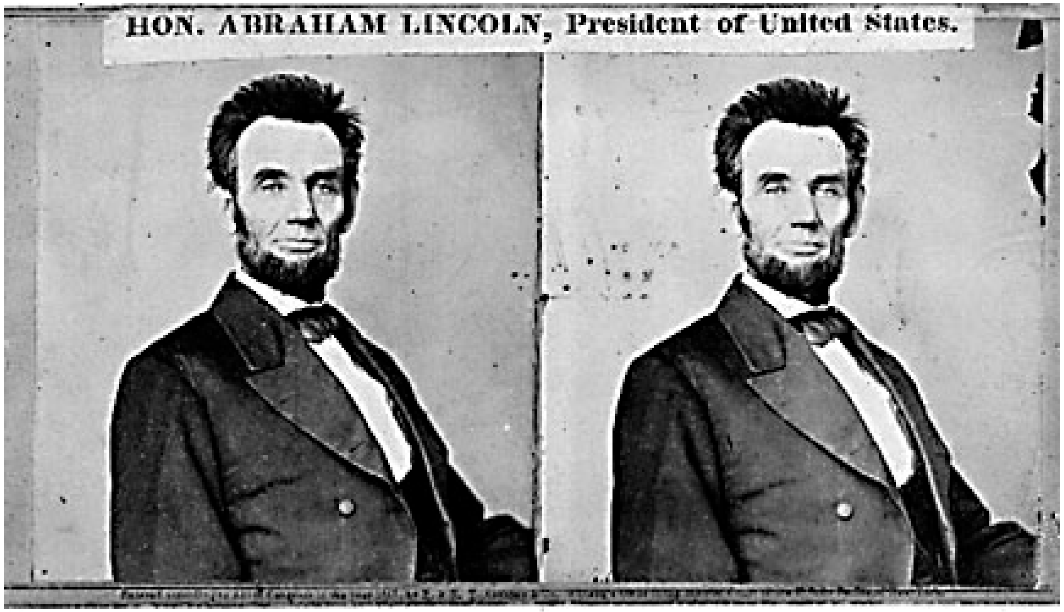
Correspondence problem

- Beyond the hard constraint of epipolar geometry, there are “soft” constraints to help identify corresponding points
  - Similarity
  - Uniqueness
  - Ordering
  - Disparity gradient

- To find matches in the image pair, we will assume
  - Most scene points visible from both views
  - Image regions for the matches are similar in appearance

Grauman

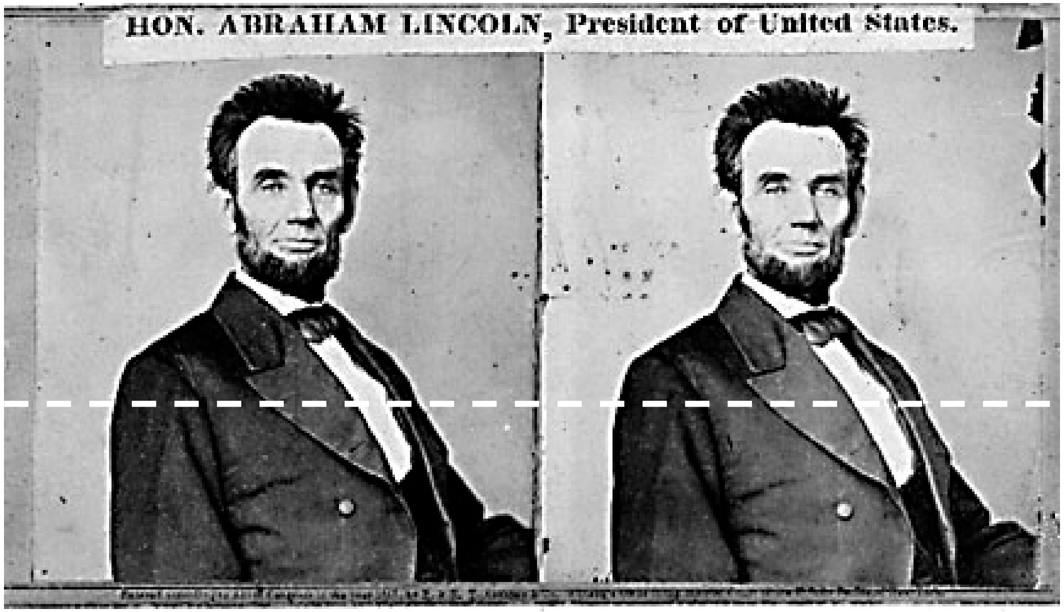
Your basic stereo algorithm



Adapted from Li Zhang

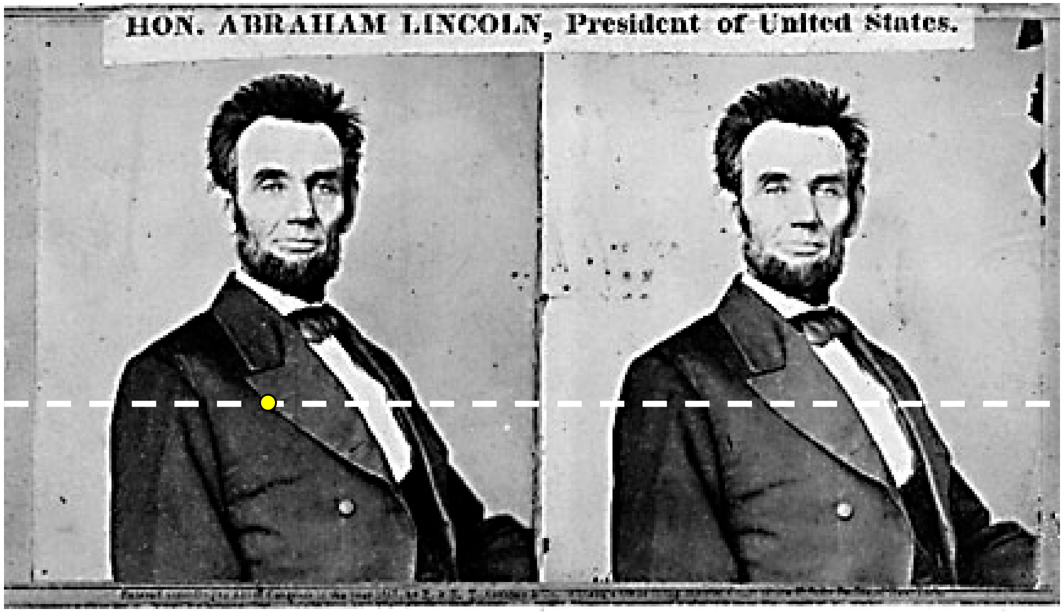
Grauman

Your basic stereo algorithm



For each epipolar line:

Your basic stereo algorithm

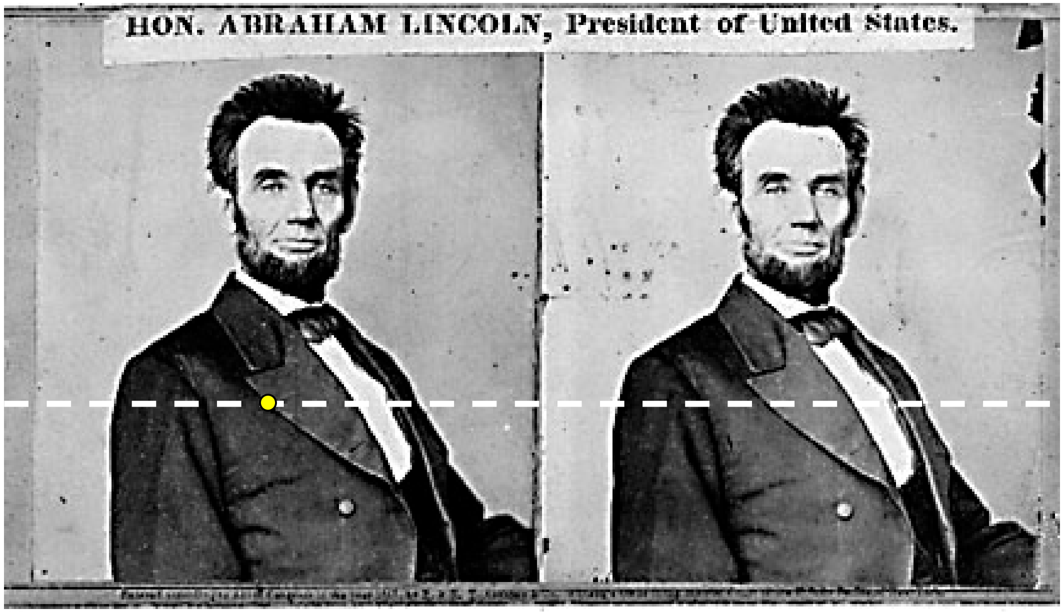


For each epipolar line:

For each pixel in the left image

Adapted from Li Zhang

Your basic stereo algorithm

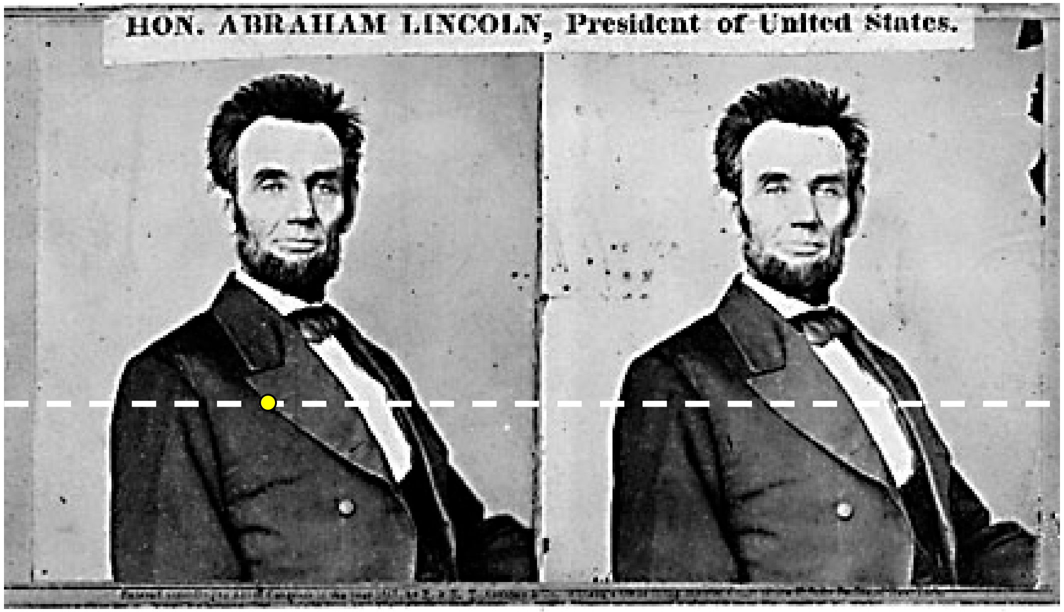


For each epipolar line:

For each pixel in the left image

- compare with every pixel on same epipolar line in right image

Your basic stereo algorithm



For each epipolar line:

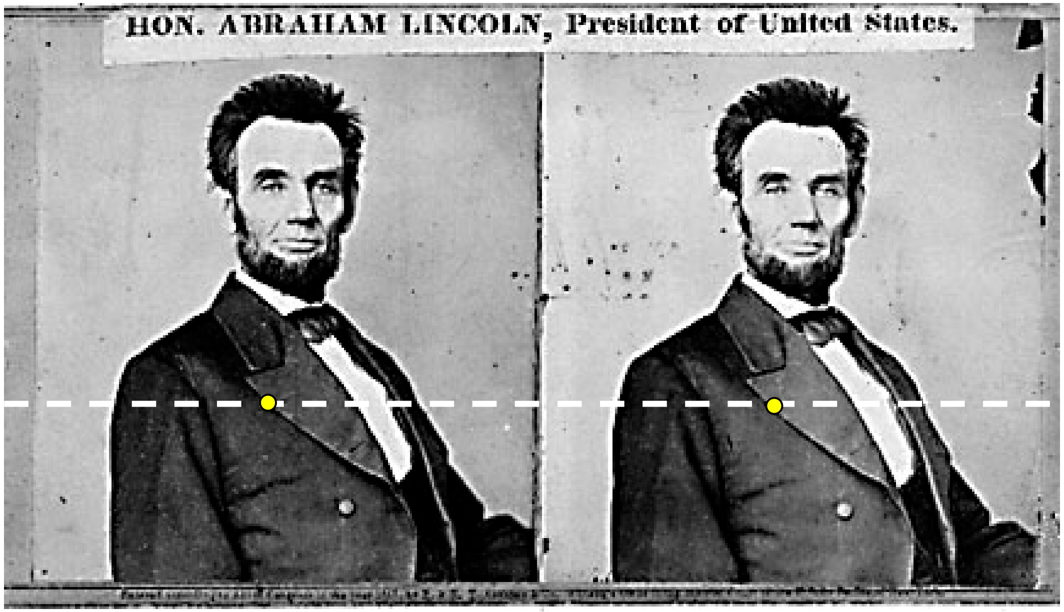
For each pixel in the left image

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

Adapted from Li Zhang

Grauman

Your basic stereo algorithm



For each epipolar line:

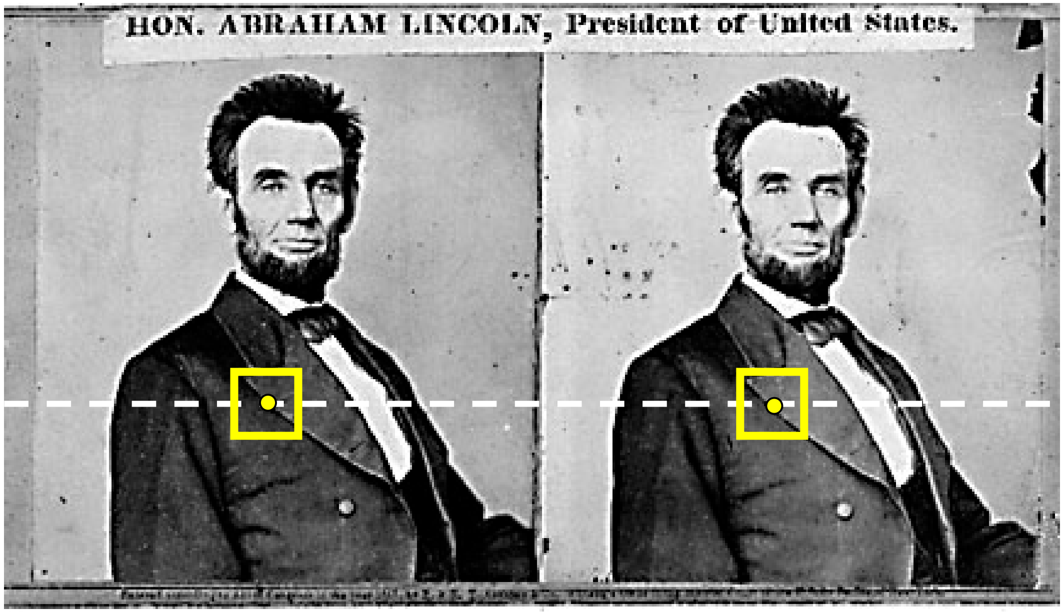
For each pixel in the left image

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

Adapted from Li Zhang

Grauman

Your basic stereo algorithm



For each epipolar line:

For each pixel in the left image

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

Improvement: match ***windows***

- This should look familiar...
- E.g. SSD, correlation etc.

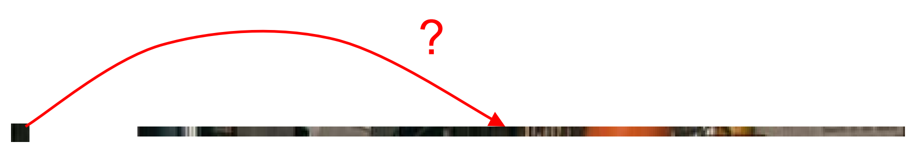
Adapted from Li Zhang

Grauman

Stereo matching

- Search is limited to epipolar line (1D)
- Look for “most similar pixel”

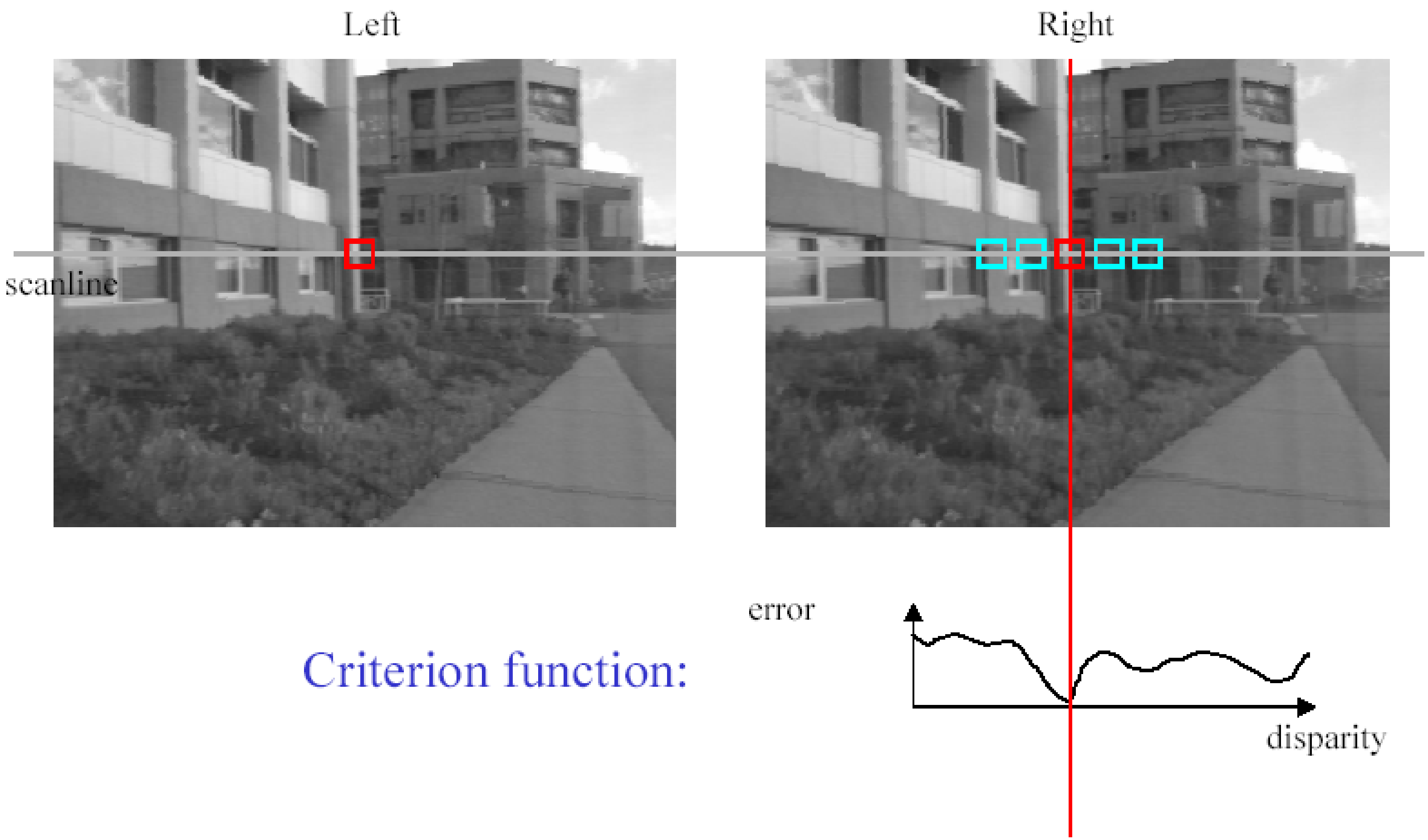
```
for x=1:w,
  for y=1:h,
    bestdist=inf;
    for i=-dr:0,
      if (dist(pix(x,y),pix(x+i,y))<bestdist)
         d(x,y)=i; best=sim(pix(x,y),pix(x+i,y)); end
    end
  end
end
```



Stereo matching algorithms

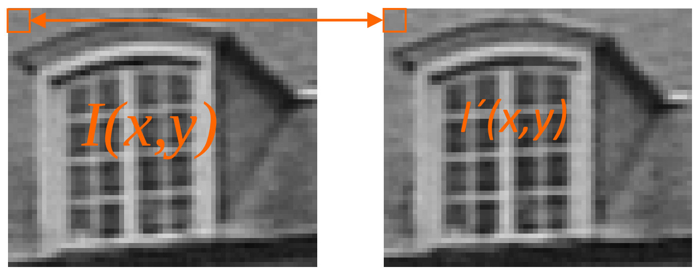
- Match Pixels in Conjugate Epipolar Lines
  - Assume brightness constancy
  - This is a tough problem
  - Numerous approaches
    - dynamic programming [Baker 81,Ohta 85]
    - smoothness functionals
    - more images (trinocular, N-ocular) [Okutomi 93]
    - graph cuts [Boykov 00]
  - A good survey and evaluation:
  - http://vision.middlebury.edu/stereo/

Correspondence using Discrete Search



Comparing image regions

Compare intensities pixel-by-pixel



Similarity measures

Census

$$C_I(i,j) = (I(x+i, y+j) > I(x,y))$$

125	126	125
127	128	130
129	132	135

→

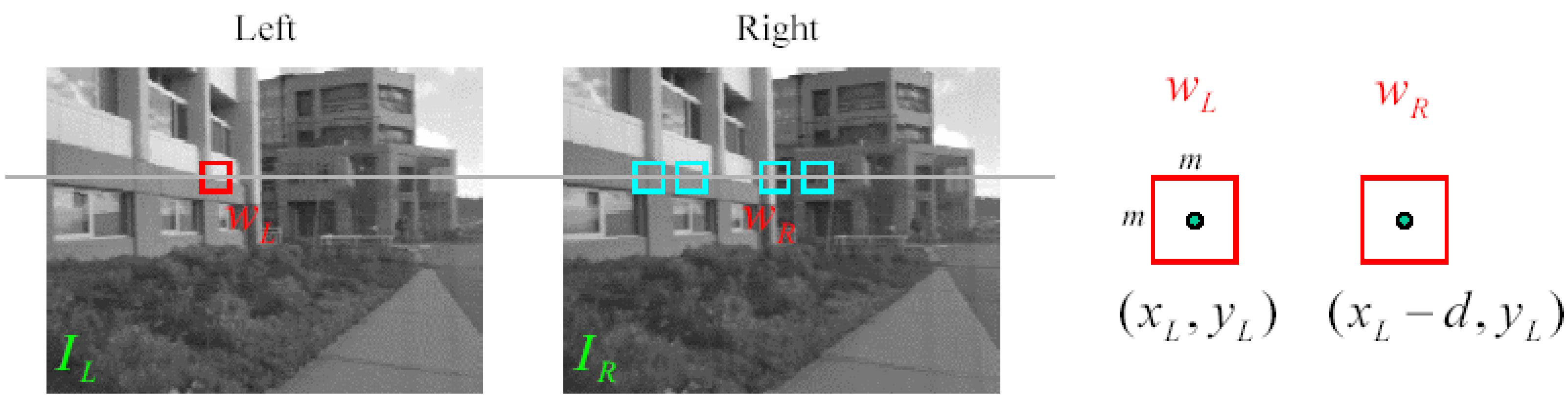
0	0	0
0		1
1	1	1

→ [00001111]

only compare bit signature

(Real-time chip from TYZX based on Census)

Sum of Squared Differences (SSD)



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 :

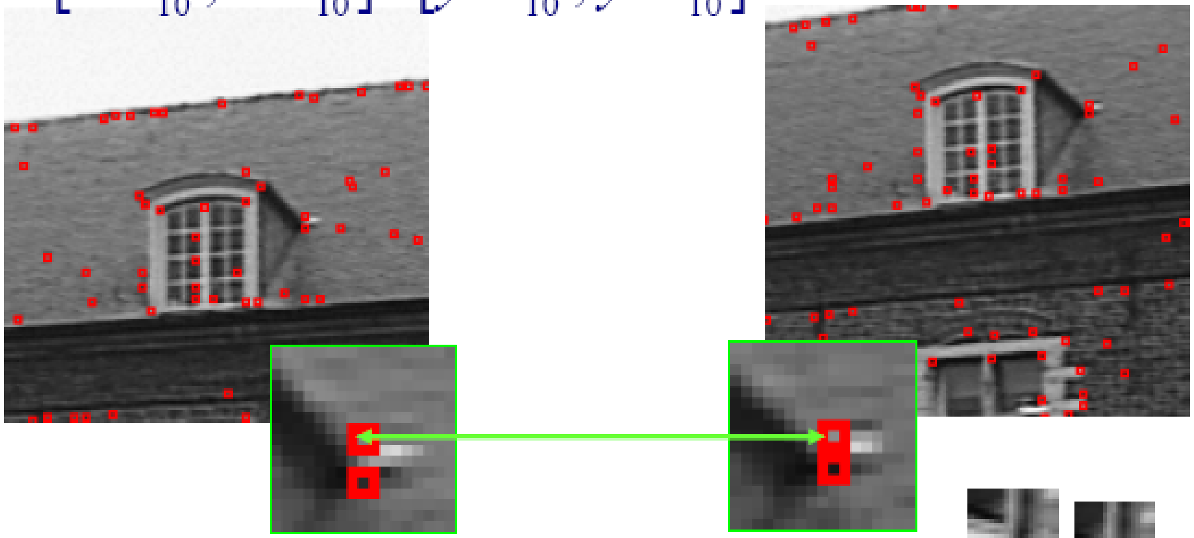
$$C_r(x,y,d) = \sum_{(u,v)\in W_m(x,y)} [I_L(u,v) - I_R(u-d,v)]^2$$

Example

Feature Matching

Evaluate NCC for all features with similar coordinates

$$\text{e.g. } (x', y') \in \left[x - \tfrac{w}{10}, x + \tfrac{w}{10}\right] \times \left[y - \tfrac{h}{10}, y + \tfrac{h}{10}\right]$$

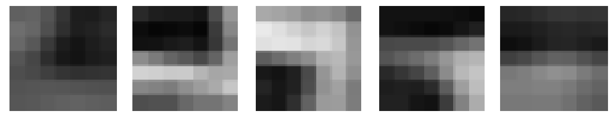


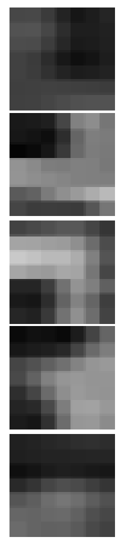
Keep mutual best matches

Still many wrong matches!

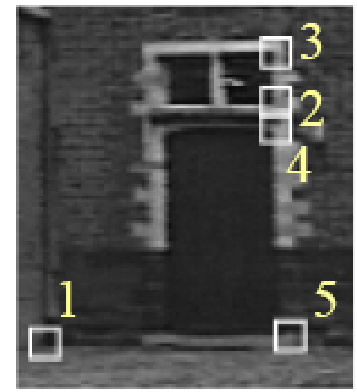
Example ctd

Feature Example





0.96	-0.40	-0.16	-0.39	0.19
-0.05	**0.75**	-0.47	0.51	0.72
-0.18	-0.39	**0.73**	0.15	-0.75
-0.27	0.49	0.16	**0.79**	0.21
0.08	0.50	-0.45	0.28	**0.99**



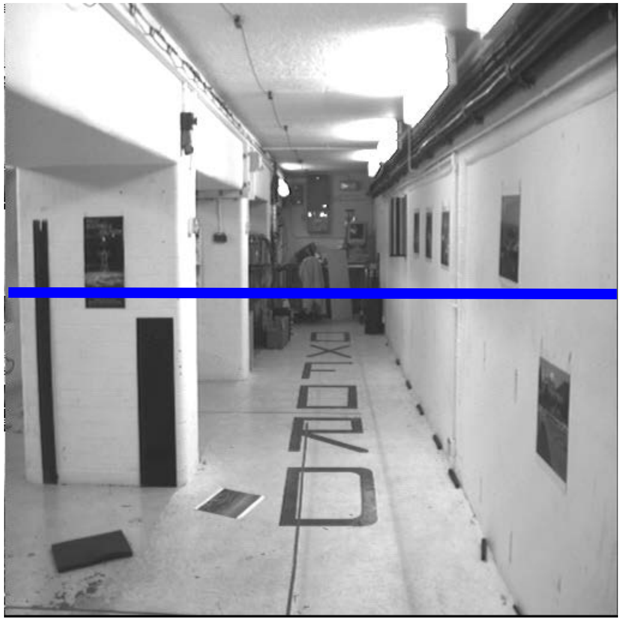
Gives satisfying results
for small image motions

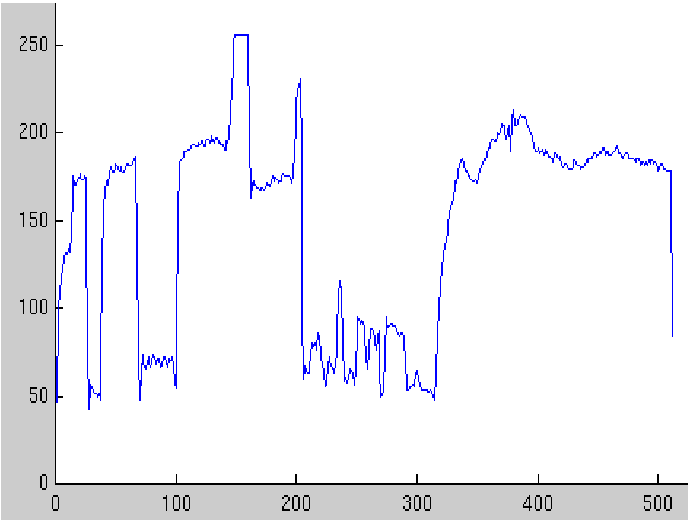
Example image pair – parallel cameras

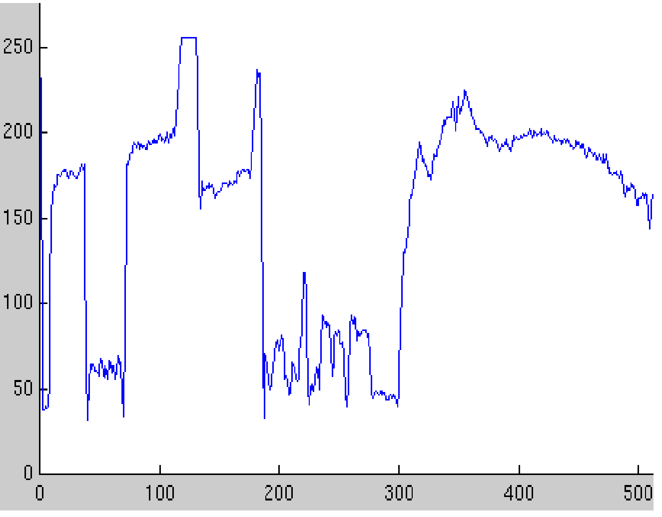
First image

Second image

Intensity profiles







- Clear correspondence between intensities, but also noise and ambiguity

Dense correspondence algorithm

Parallel camera example – epipolar lines are corresponding rasters

Dense correspondence algorithm

Parallel camera example – epipolar lines are corresponding rasters

Dense correspondence algorithm

Parallel camera example – epipolar lines are corresponding rasters

Dense correspondence algorithm

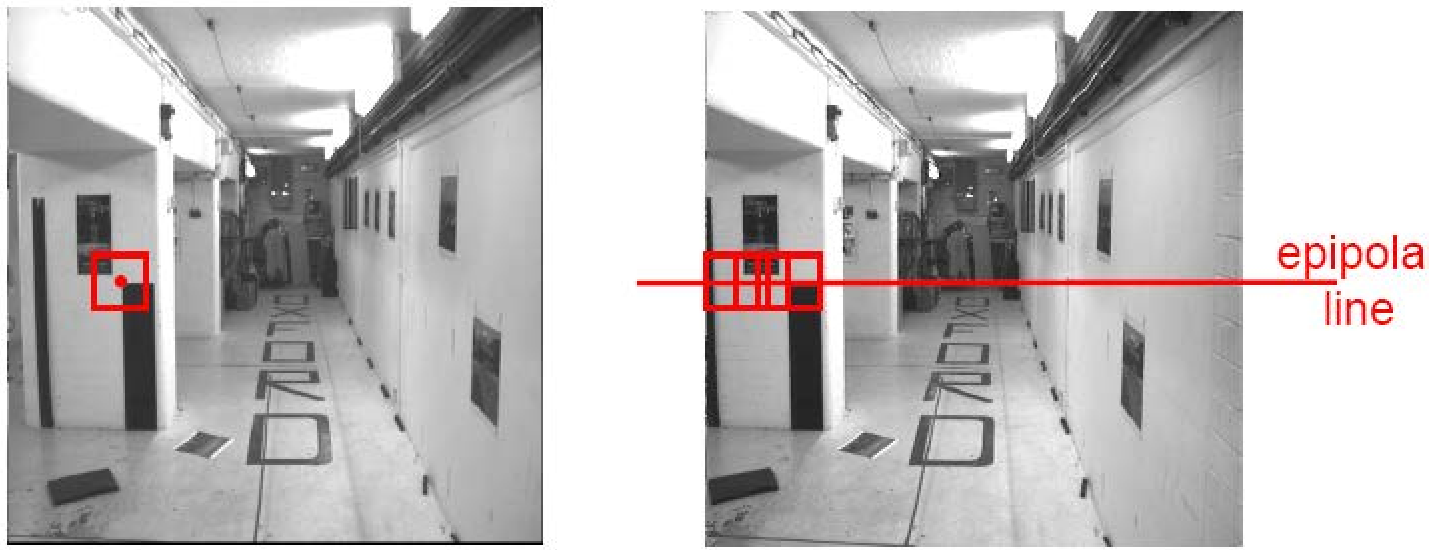
Parallel camera example – epipolar lines are corresponding rasters

Search problem (geometric constraint): for each point in the left image, the corresponding point in the right image lies on the epipolar line (1D ambiguity)

Disambiguating assumption (photometric constraint): the intensity neighbourhood of corresponding points are similar across images

Measure similarity of neighbourhood intensity by cross-correlation

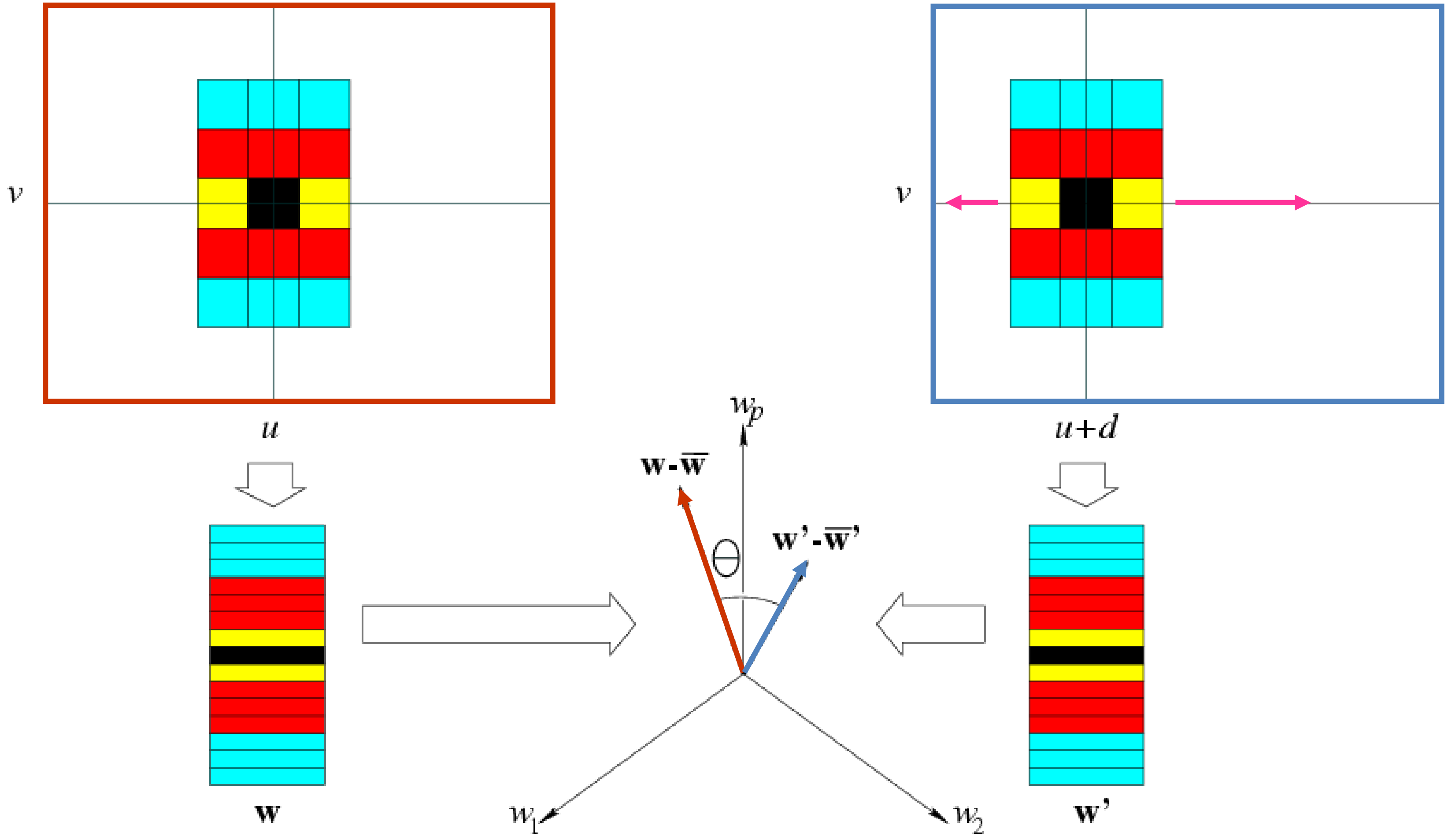
Correspondence problem



Neighborhood of corresponding points are similar in intensity patterns.

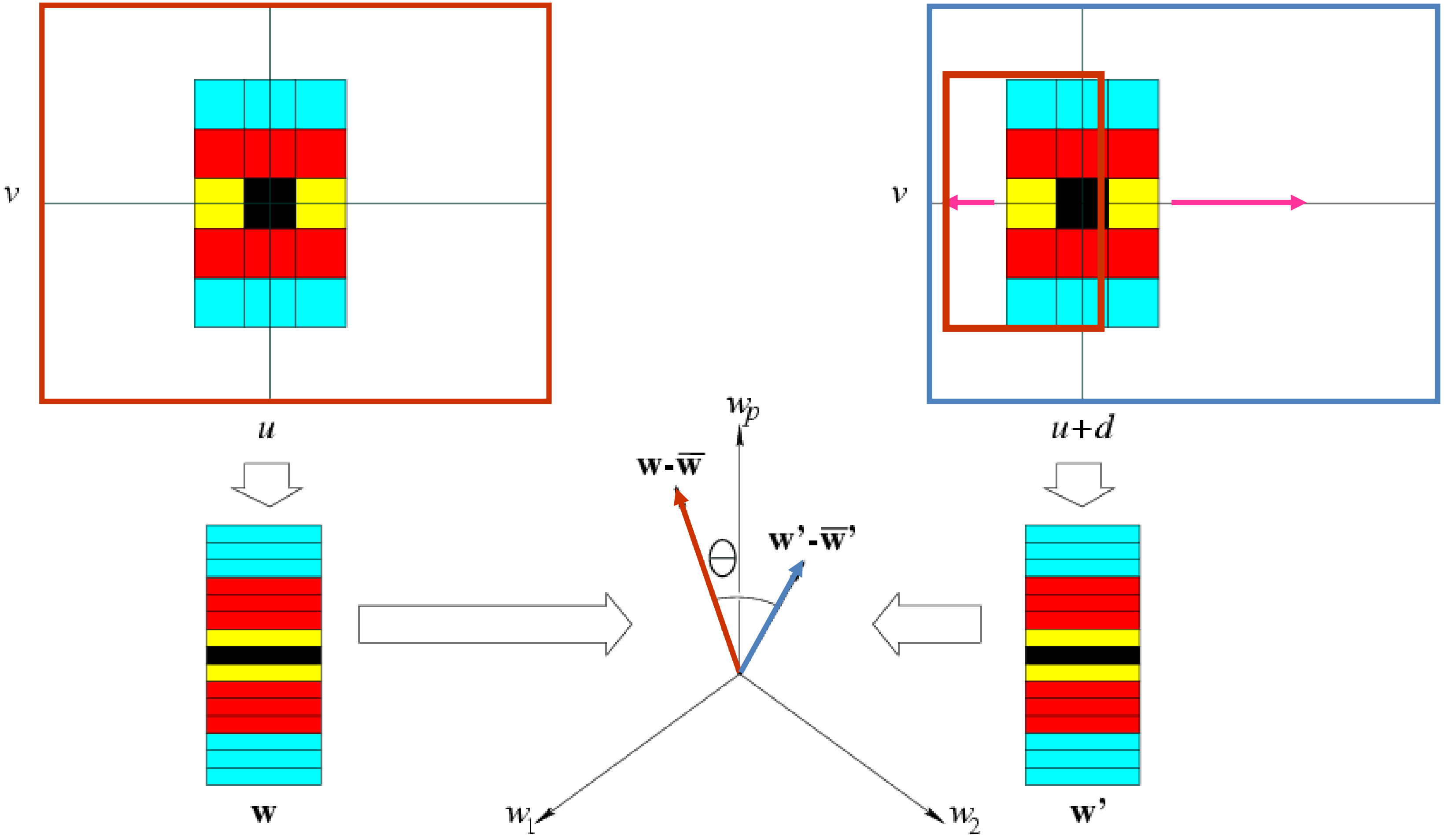
Source: Andrew Zisserman

Correlation Methods (1970--)   F&P book new: 7.4, old 11.3



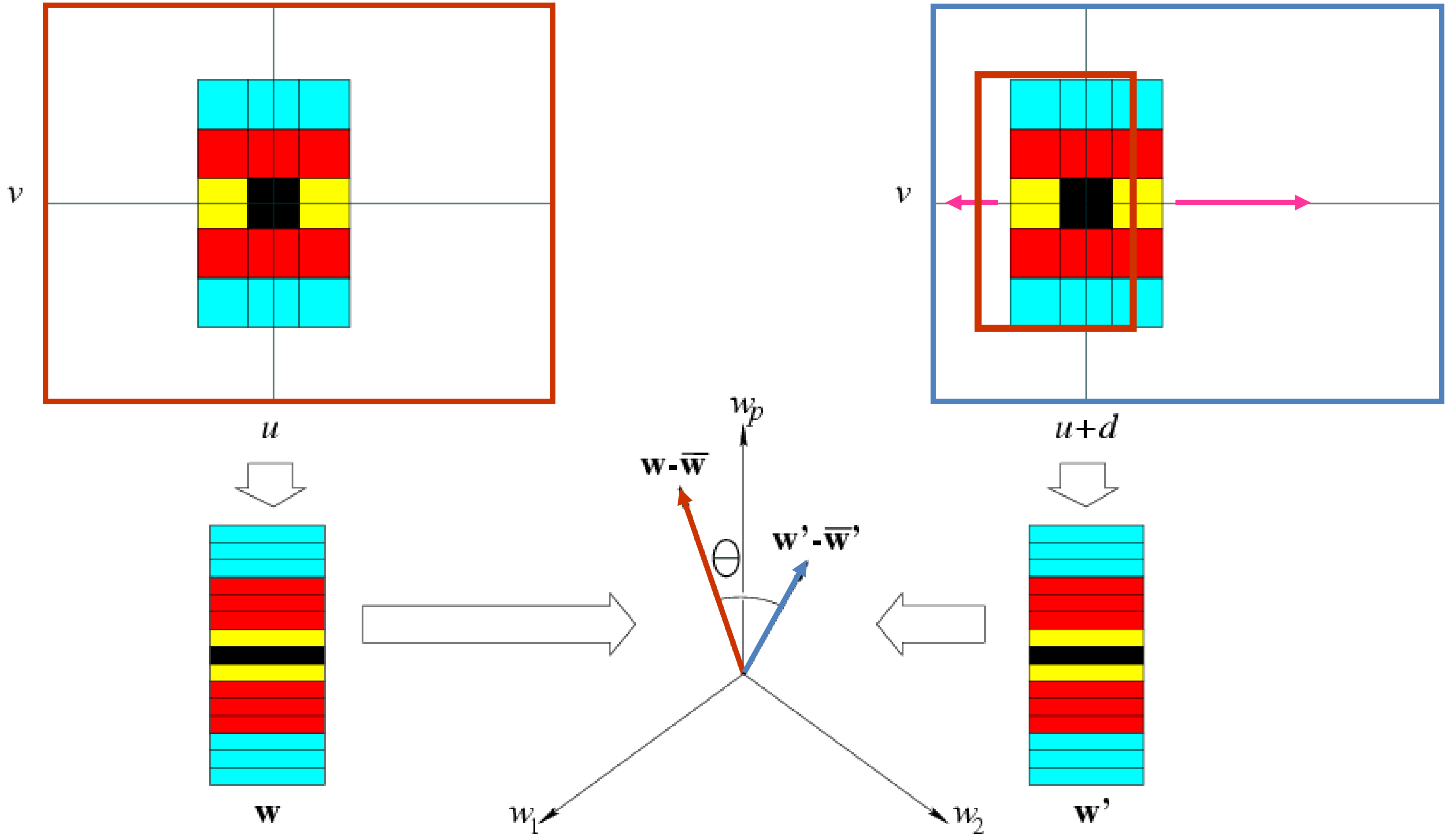
Slide the window along the epipolar line until *w.w'* is maximized.

Correlation Methods (1970--) F&P book new: 7.4, old 11.3



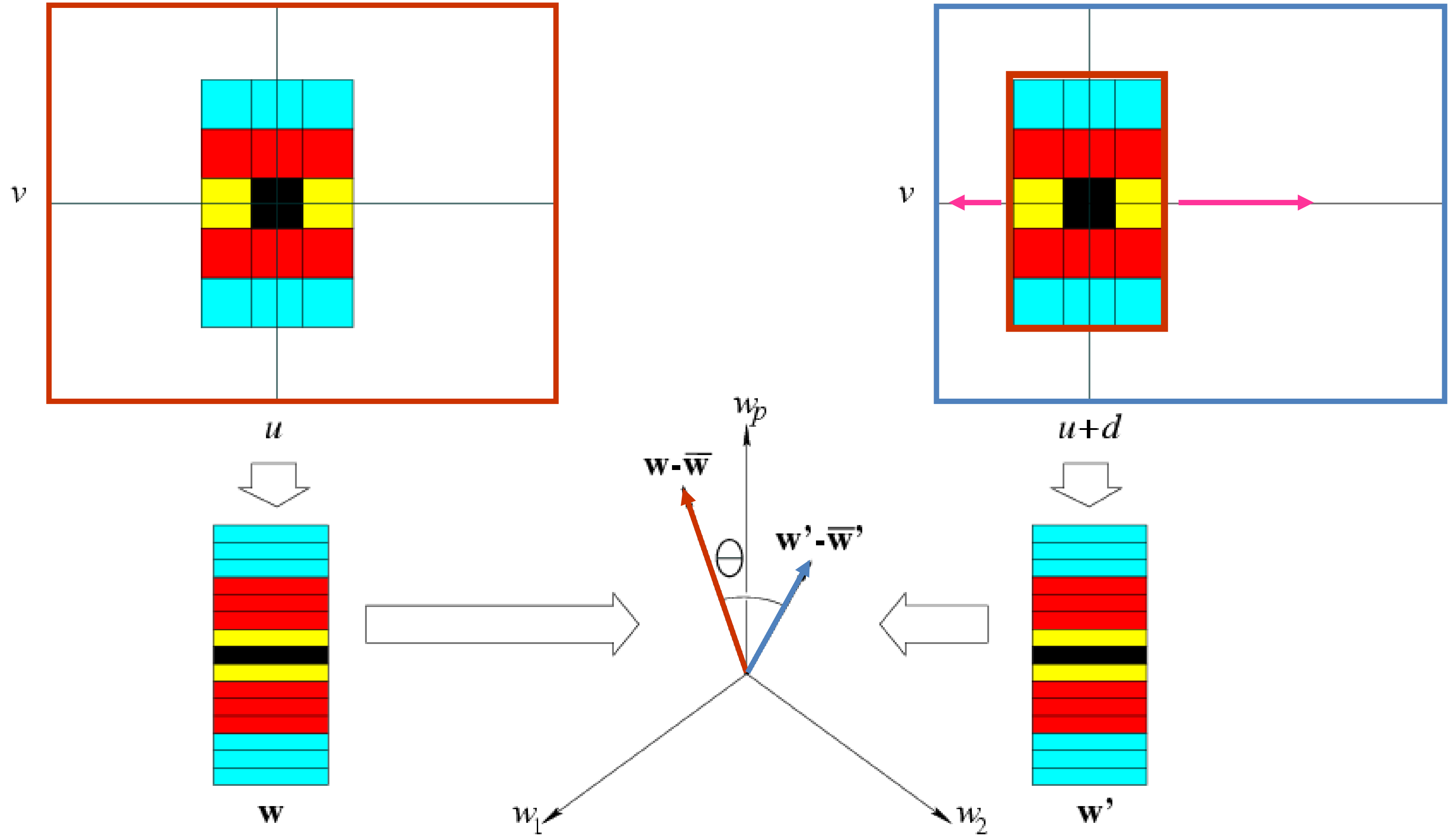
Slide the window along the epipolar line until *w.w'* is maximized.

Correlation Methods (1970--)   F&P book new: 7.4, old 11.3



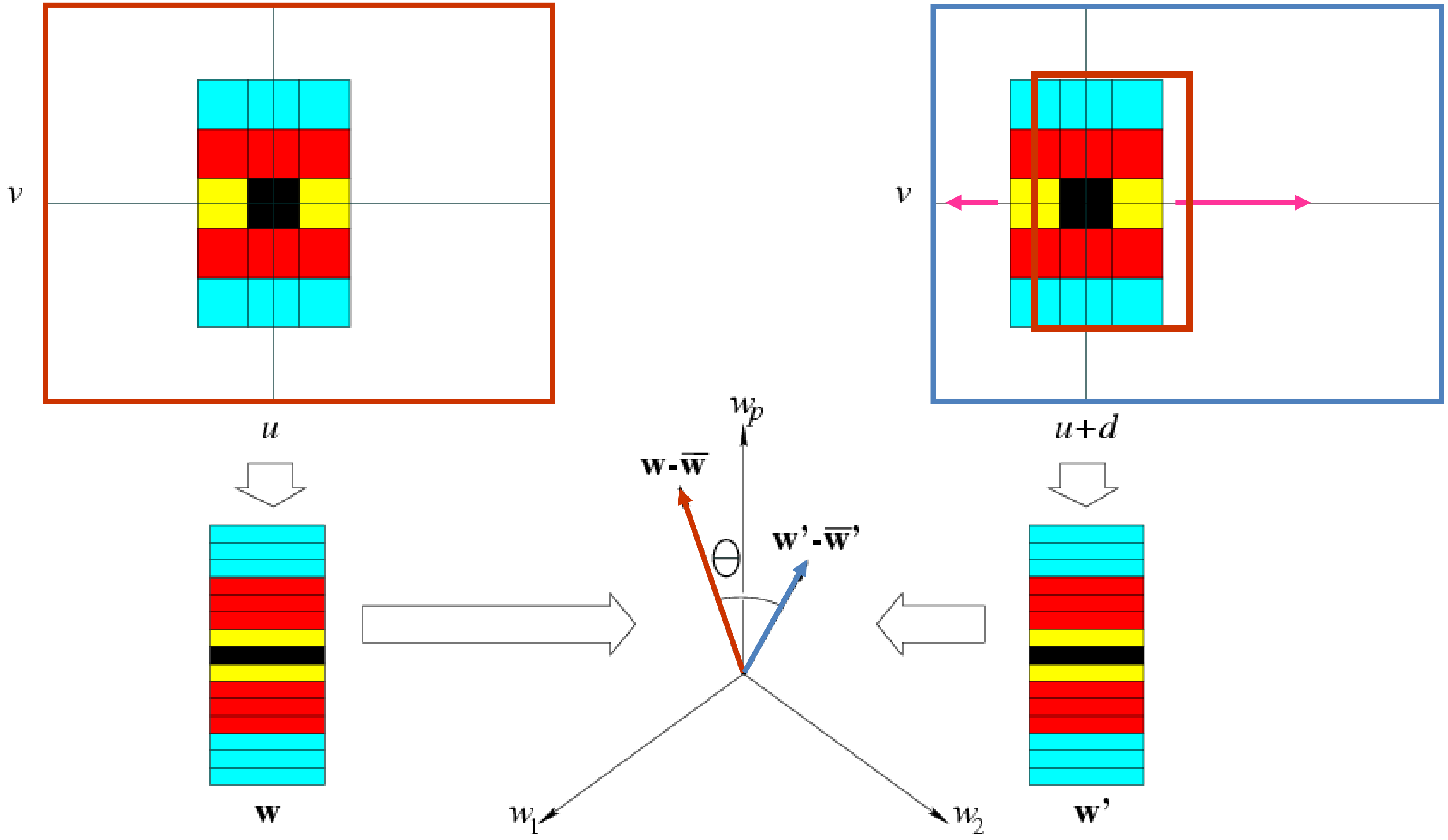
Slide the window along the epipolar line until *w.w'* is maximized.

Correlation Methods (1970--) F&P book new: 7.4, old 11.3



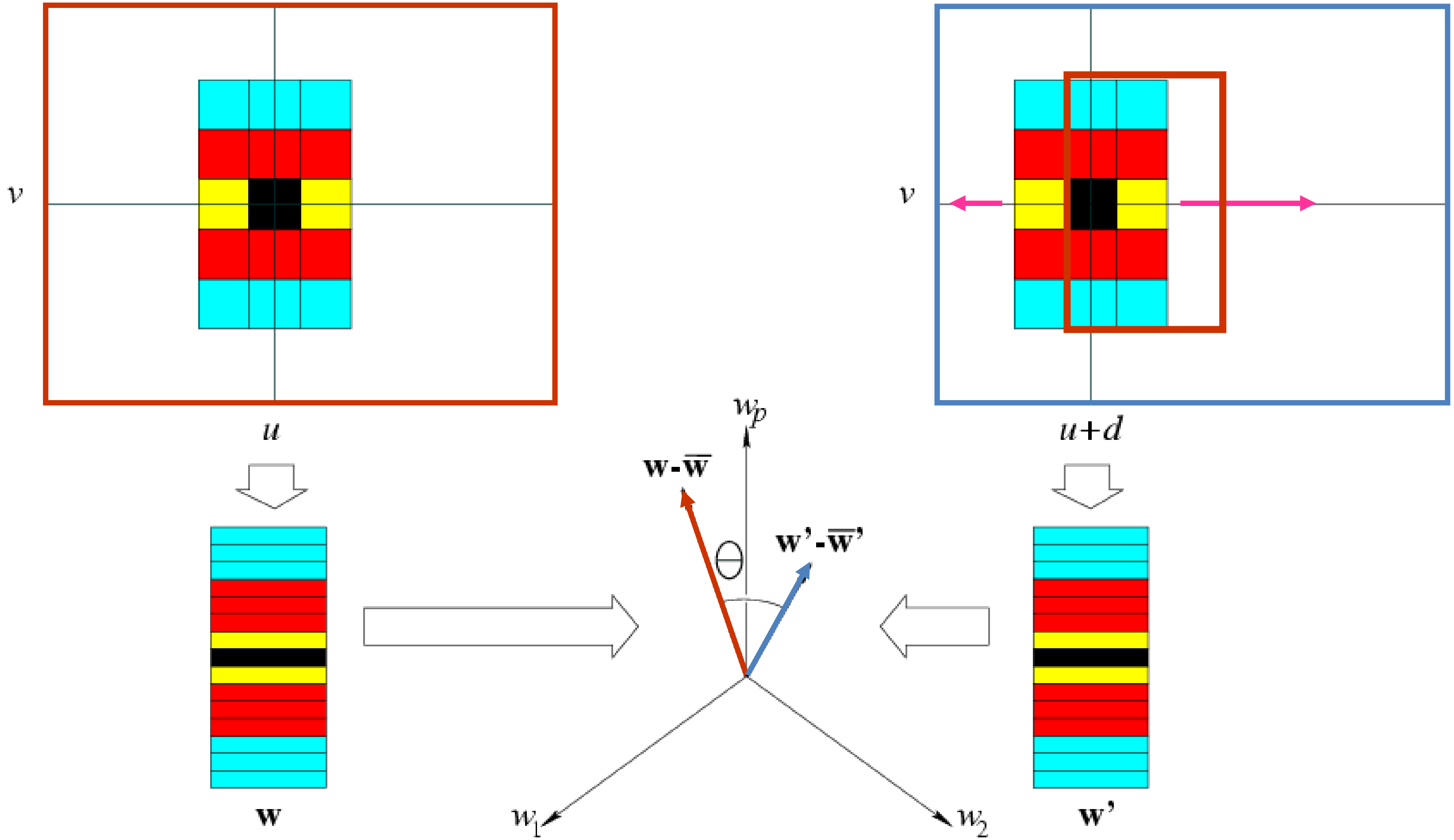
Slide the window along the epipolar line until *w.w'* is maximized.

Correlation Methods (1970--)   F&P book new: 7.4, old 11.3



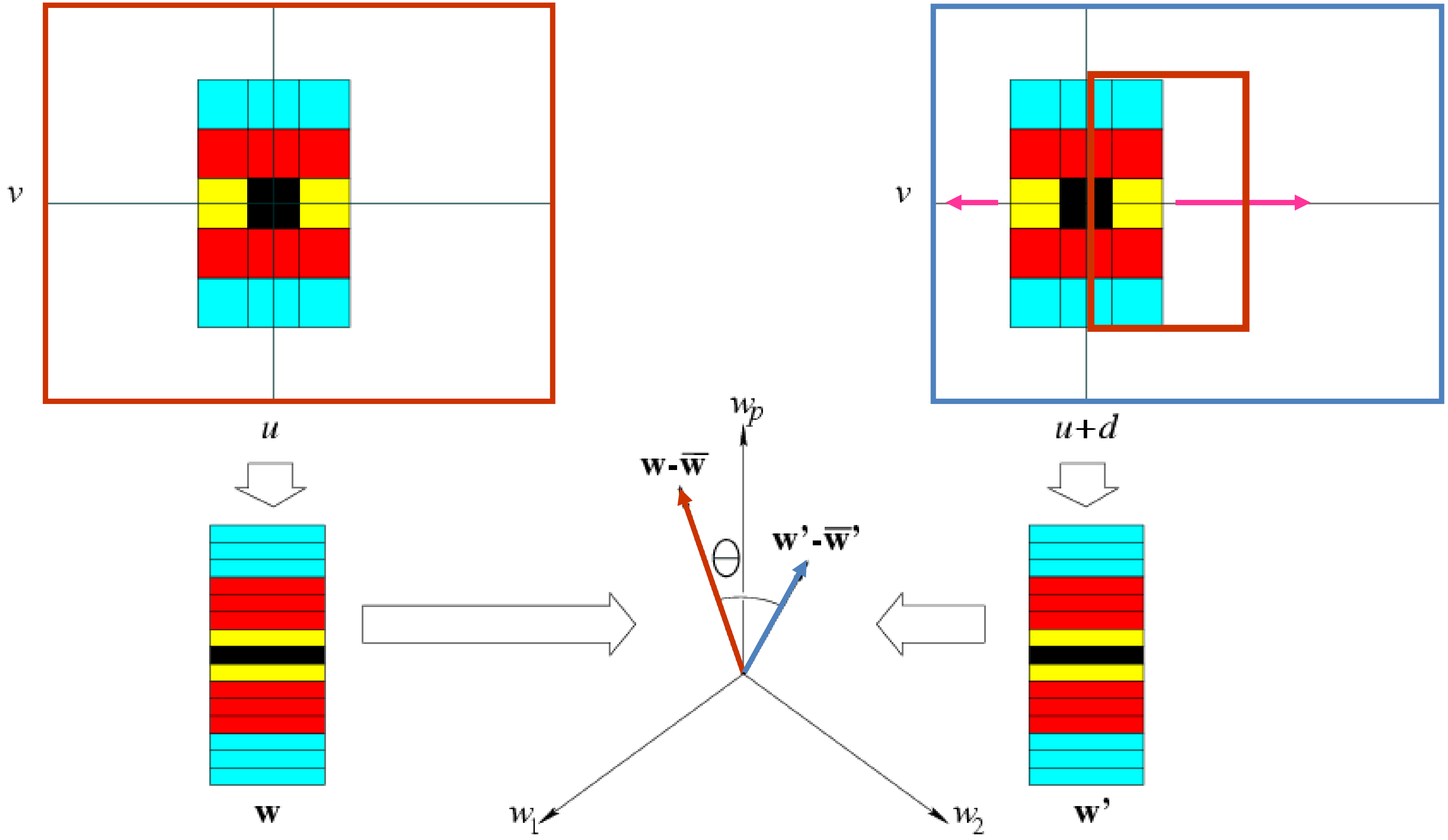
Slide the window along the epipolar line until *w.w'* is maximized.

Correlation Methods (1970--)   F&P book new: 7.4, old 11.3



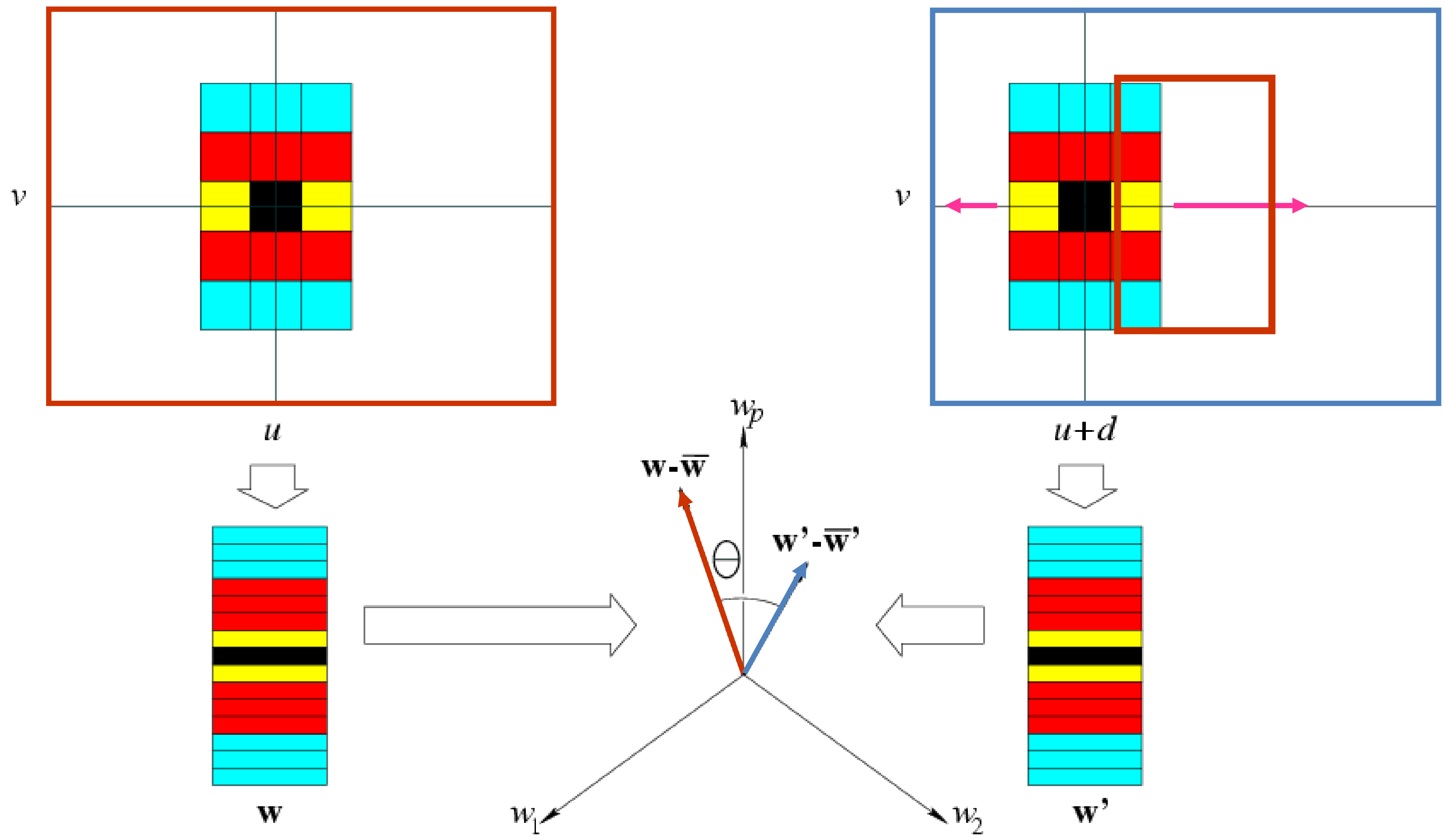
Slide the window along the epipolar line until *w.w'* is maximized.

Correlation Methods (1970--) F&P book new: 7.4, old 11.3



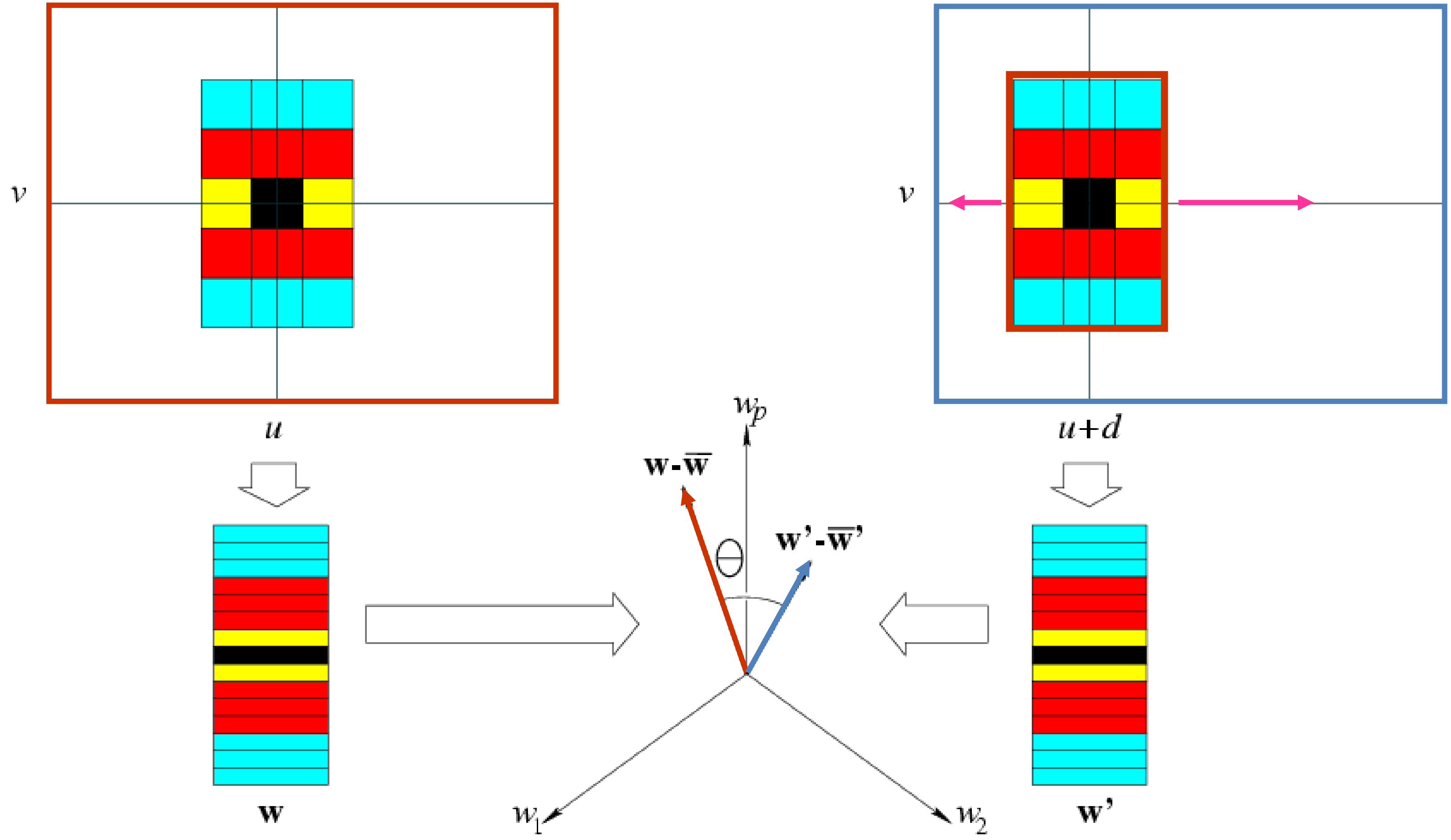
Slide the window along the epipolar line until *w.w'* is maximized.

Correlation Methods (1970--) F&P book new: 7.4, old 11.3



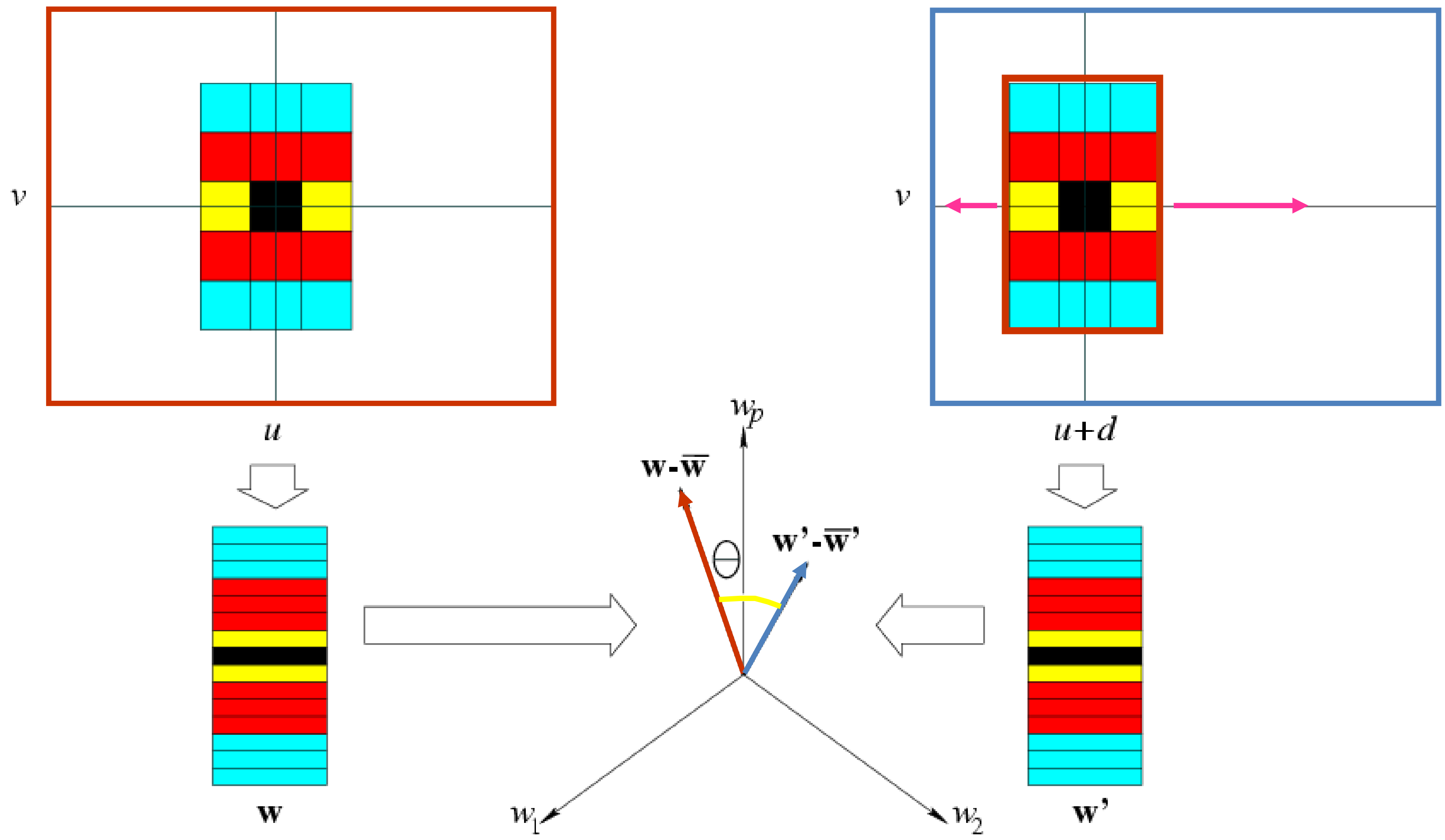
Slide the window along the epipolar line until *w.w'* is maximized.

Correlation Methods (1970--) F&P book new: 7.4, old 11.3



Slide the window along the epipolar line until *w.w'* is maximized.

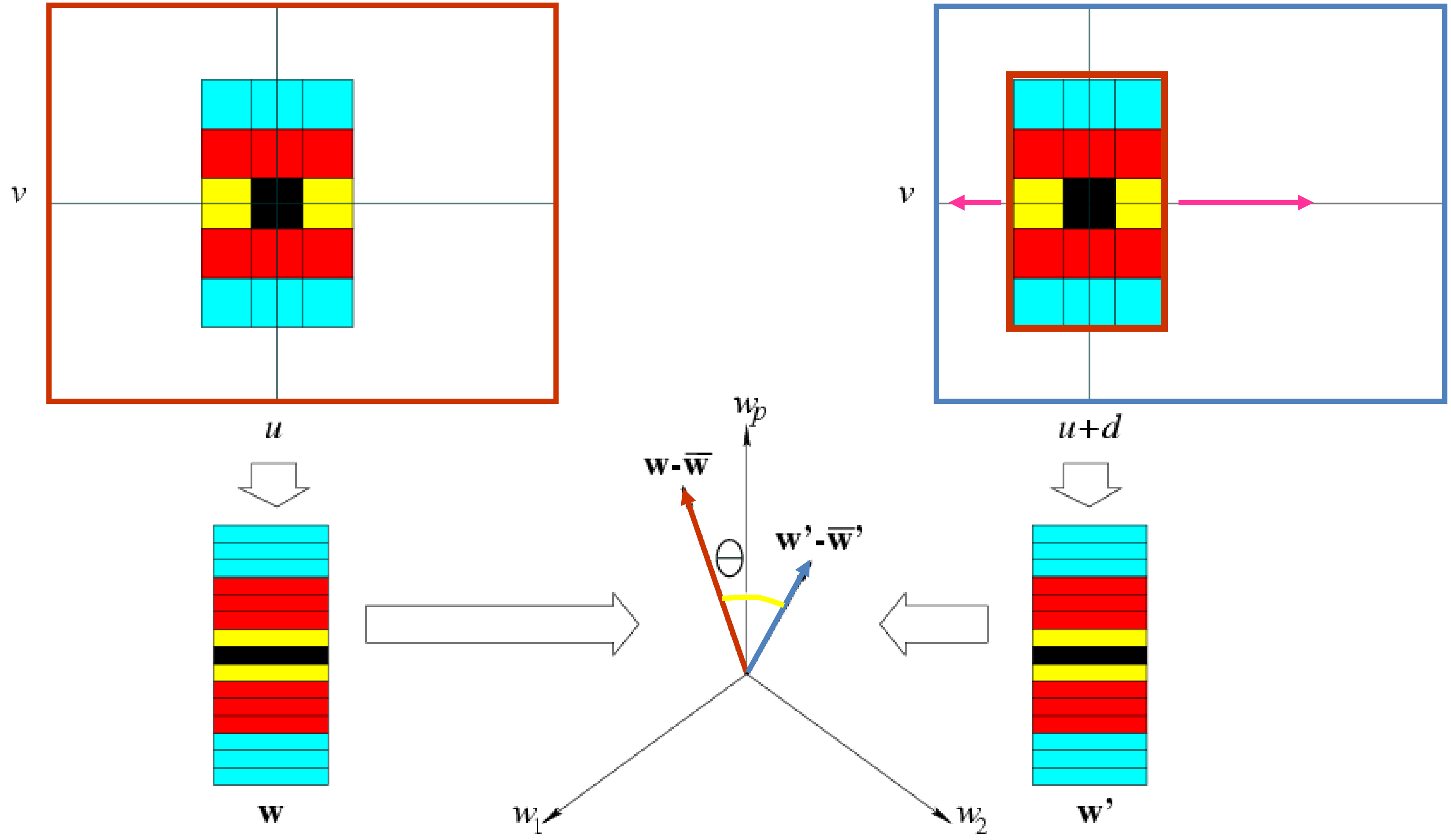
Correlation Methods (1970--) F&P book new: 7.4, old 11.3



Slide the window along the epipolar line until *w.w'* is maximized.

Normalized Correlation: minimize θ instead.

Correlation Methods (1970--)   F&P book new: 7.4, old 11.3



Slide the window along the epipolar line until *w.w'* is maximized.

Normalized Correlation: minimize θ instead. ⟺ Minimize $|w-w'|^2$.

Cross-correlation of neighbourhood regions

- left and right windows encoded as vectors w and w'
- zero-mean vectors $(w - \overline{w})$ and $(w' - \overline{w}')$
- Normalized cross-correlation:

$$C(d) = \frac{1}{||\boldsymbol{w} - \bar{\boldsymbol{w}}||} \frac{1}{||\boldsymbol{w}' - \bar{\boldsymbol{w}}'||} [(\boldsymbol{w} - \bar{\boldsymbol{w}}) \cdot (\boldsymbol{w}' - \bar{\boldsymbol{w}}')].$$

- Advantage: Invariant to intensity differences: Invariant to affine intensity transformation $I' = \alpha I + \mu$

Cross-correlation of neighbourhood regions

- left and right windows encoded as vectors w and w'
- zero-mean vectors $(w - \overline{w})$ and $(w' - \overline{w}')$
- Normalized cross-correlation:

$$C(d) = \frac{1}{||\mathbf{w} - \bar{\mathbf{w}}||} \frac{1}{||\mathbf{w}' - \bar{\mathbf{w}}'||} [(\mathbf{w} - \bar{\mathbf{w}}) \cdot (\mathbf{w}' - \bar{\mathbf{w}}')].$$

- Advantage: Invariant to intensity differences: Invariant to affine intensity transformation $I' = \alpha I + \mu$

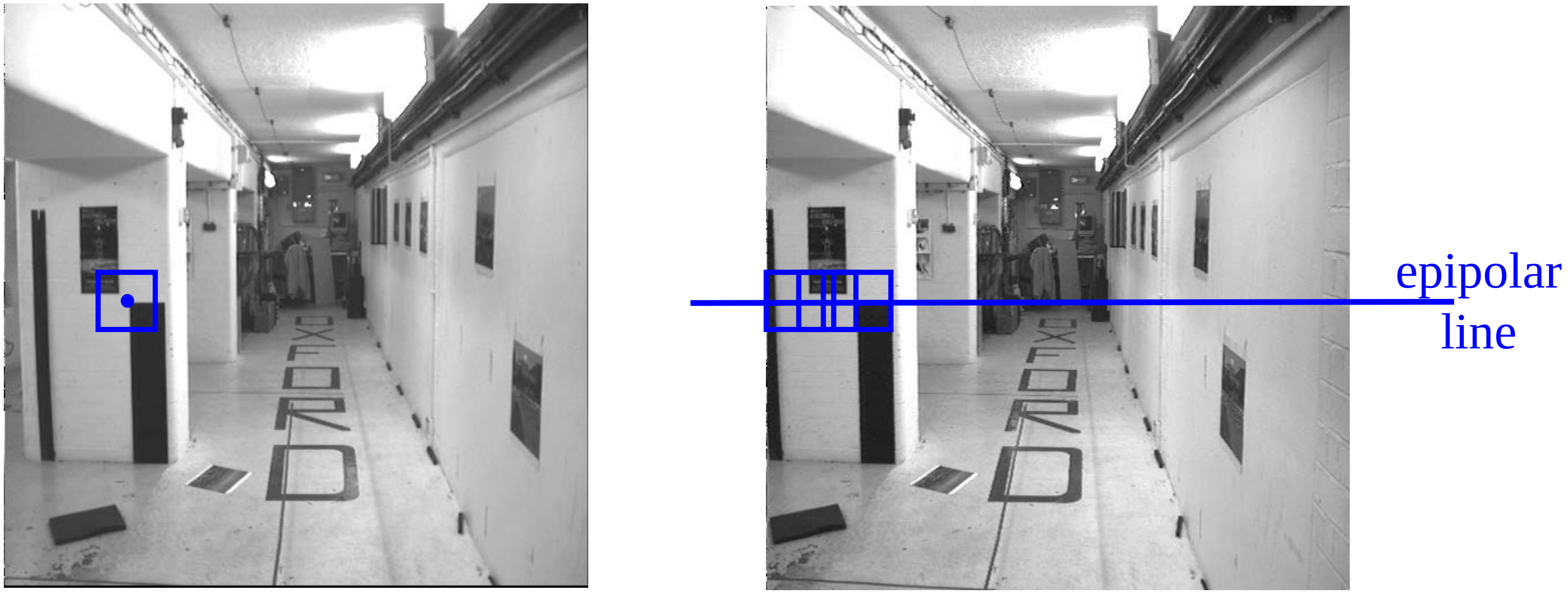
Cross-correlation of neighbourhood regions

- left and right windows encoded as vectors w and w'
- zero-mean vectors $(w - \overline{w})$ and $(w' - \overline{w}')$
- Normalized cross-correlation:

$$C(d) = \frac{1}{||w - \bar{w}||} \frac{1}{||w' - \bar{w}'||} [(w - \bar{w}) \cdot (w' - \bar{w}')].$$

- Advantage: Invariant to intensity differences: Invariant to affine intensity transformation $I' = \alpha I + \mu$

Cross-correlation of neighbourhood regions



- left and right windows encoded as vectors w and w'
- zero-mean vectors $(w - \overline{w})$ and $(w' - \overline{w}')$
- Normalized cross-correlation:

$$C(d) = \frac{1}{||w - \bar{w}||} \frac{1}{||w' - \bar{w}'||} [(w - \bar{w}) \cdot (w' - \bar{w}')],$$

- Advantage: Invariant to intensity differences: Invariant to affine intensity transformation $I' = \alpha I + \mu$

Cross-correlation of neighbourhood regions

- left and right windows encoded as vectors w and w'
- zero-mean vectors $(w - \overline{w})$ and $(w' - \overline{w}')$
- Normalized cross-correlation:

$$C(d) = \frac{1}{||w - \bar{w}||} \frac{1}{||w' - \bar{w}'||} [(w - \bar{w}) \cdot (w' - \bar{w}')].$$

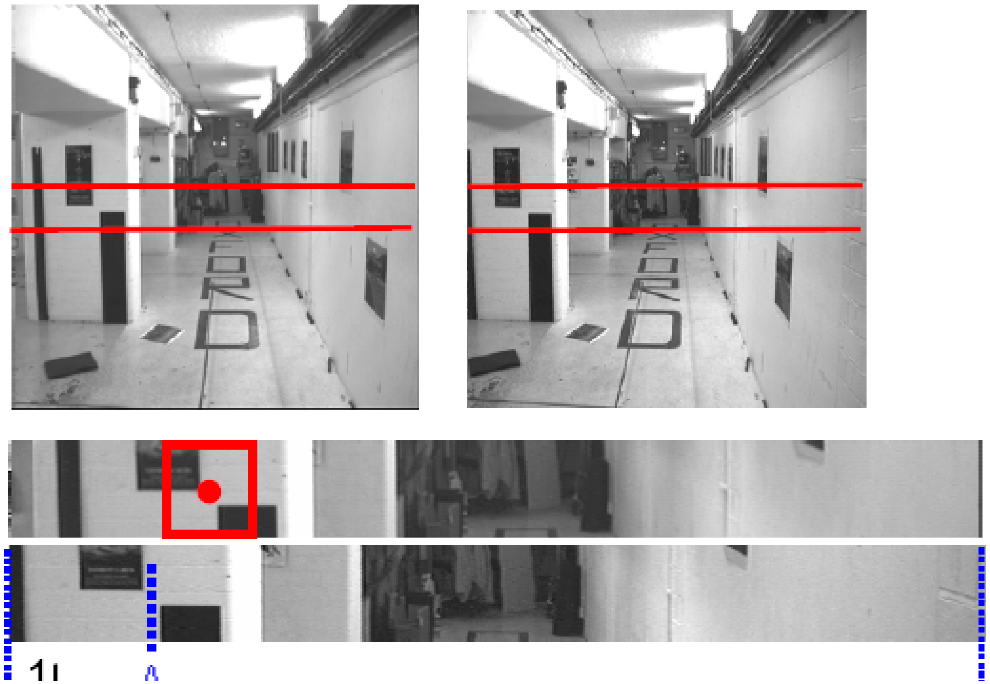
- Advantage: Invariant to intensity differences: Invariant to affine intensity transformation $I' = \alpha I + \mu$

Correlation-based window matching

left image band (x)

Source: A

Correlation-based window matching

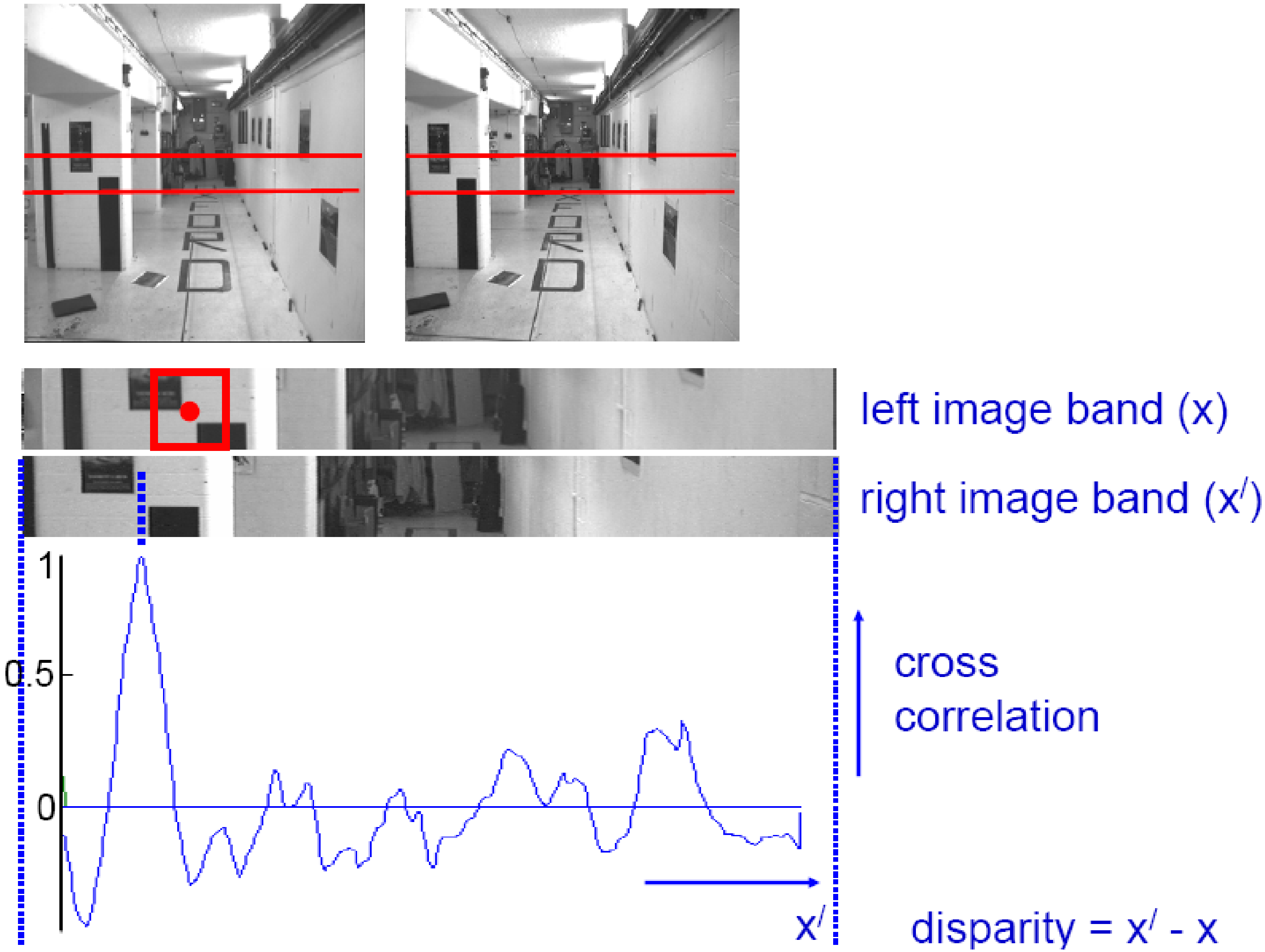


left image band (x)

right image band (x')

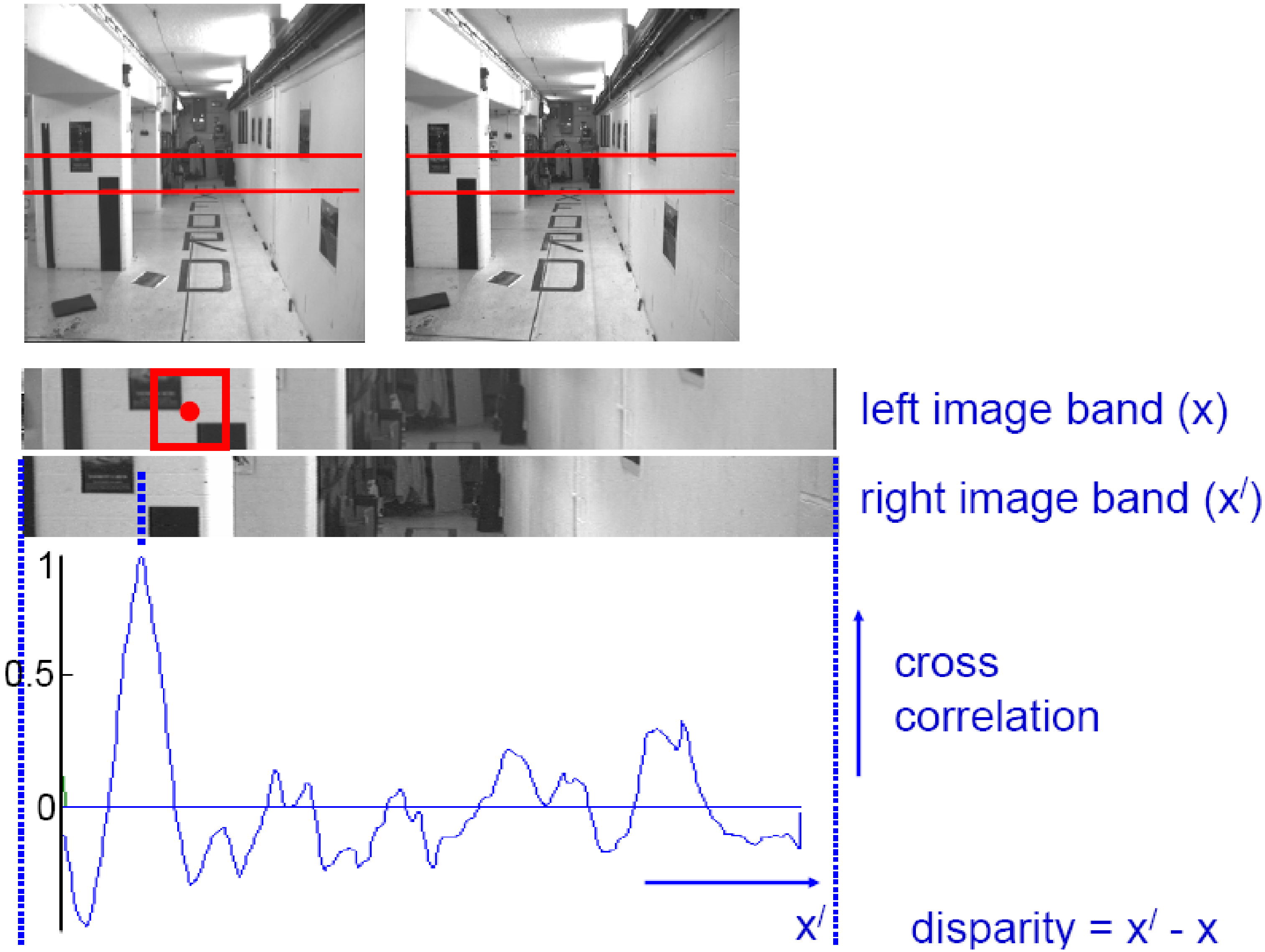
Source: A

Correlation-based window matching



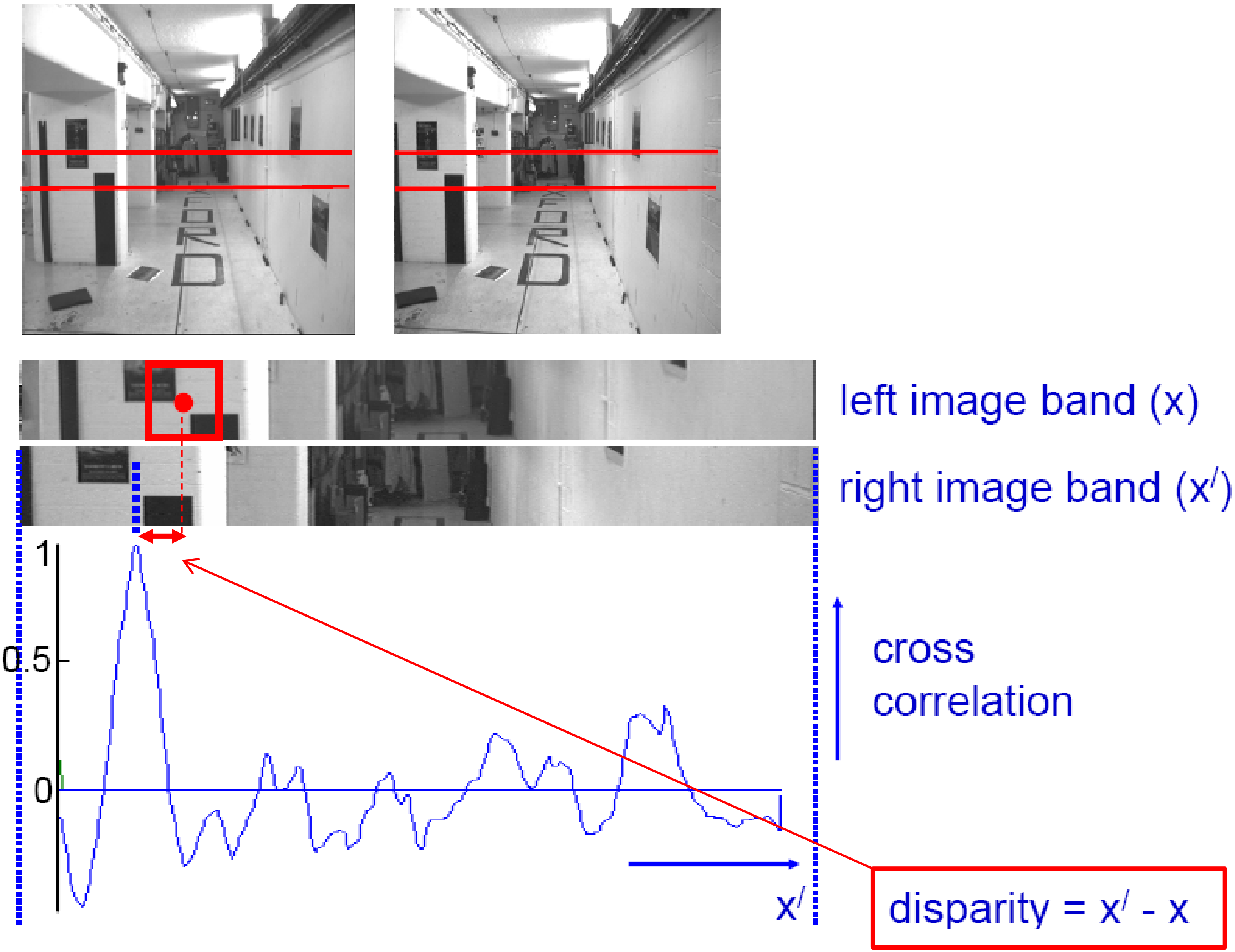
Source: Andrew Zisserman

Correlation-based window matching



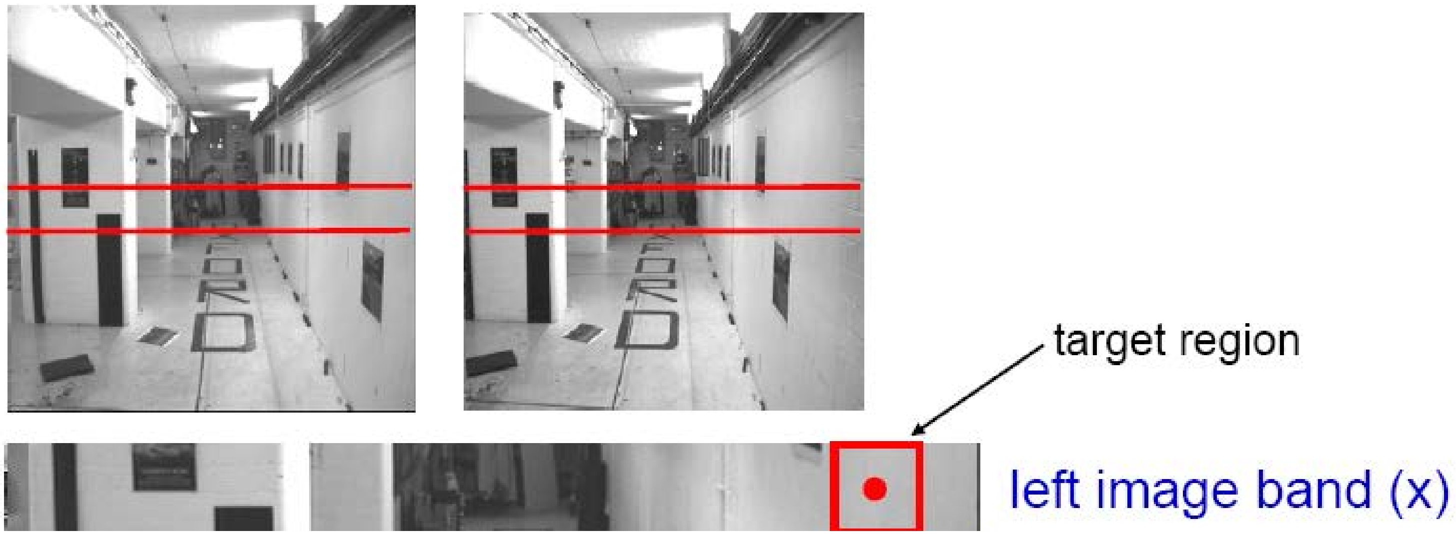
Source: Andrew Zisserman

Correlation-based window matching



Source: Andrew Zisserman

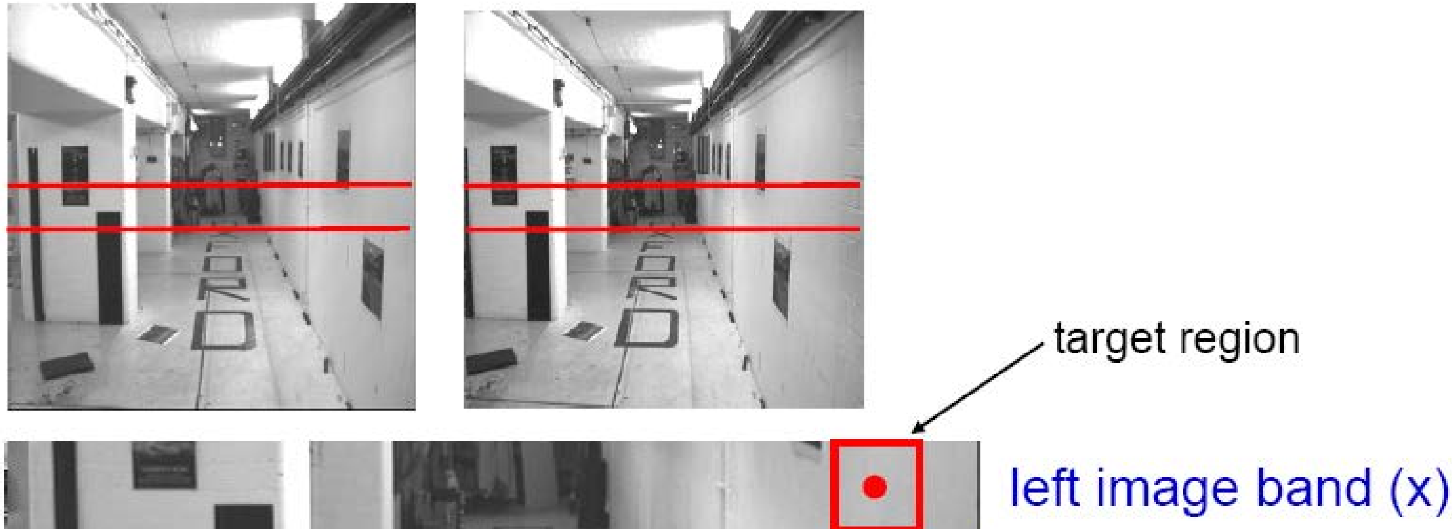
Textureless regions



Source: Andrew Zisserman

Grauman

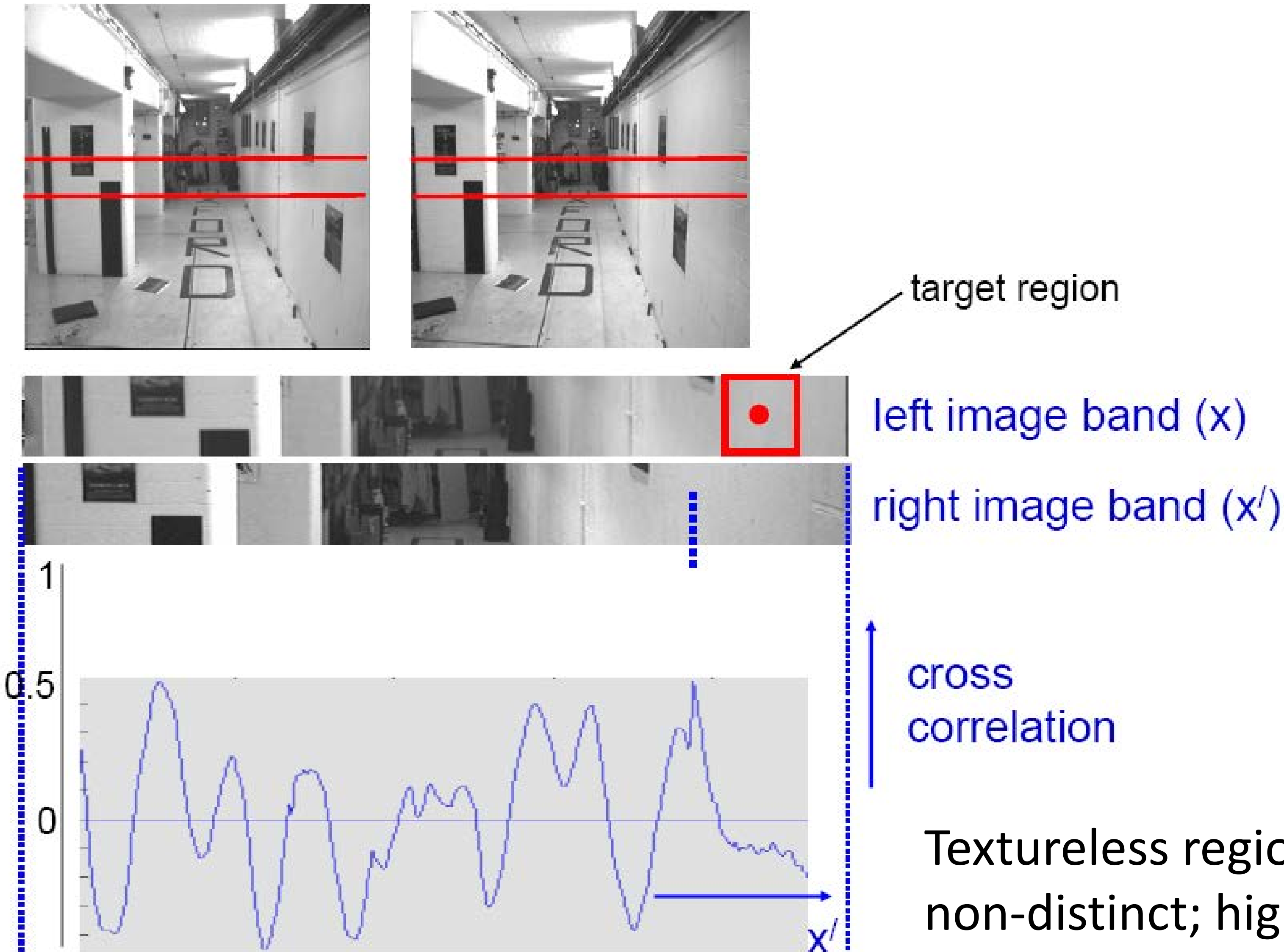
Textureless regions



Source: Andrew Zisserman

Grauman

Textureless regions

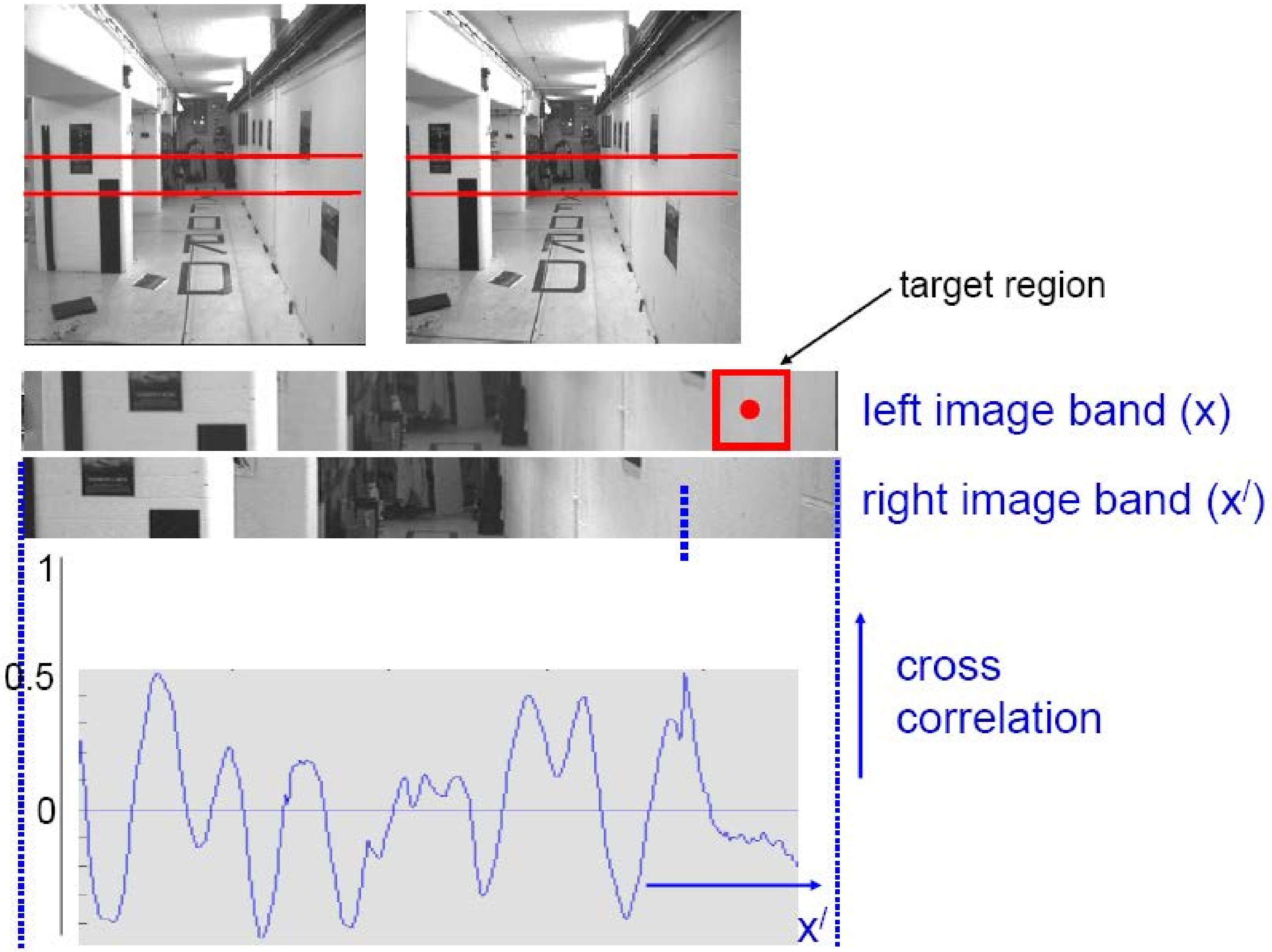


Textureless regions are non-distinct; high ambiguity for matches.

Source: Andrew Zisserman

Grauman

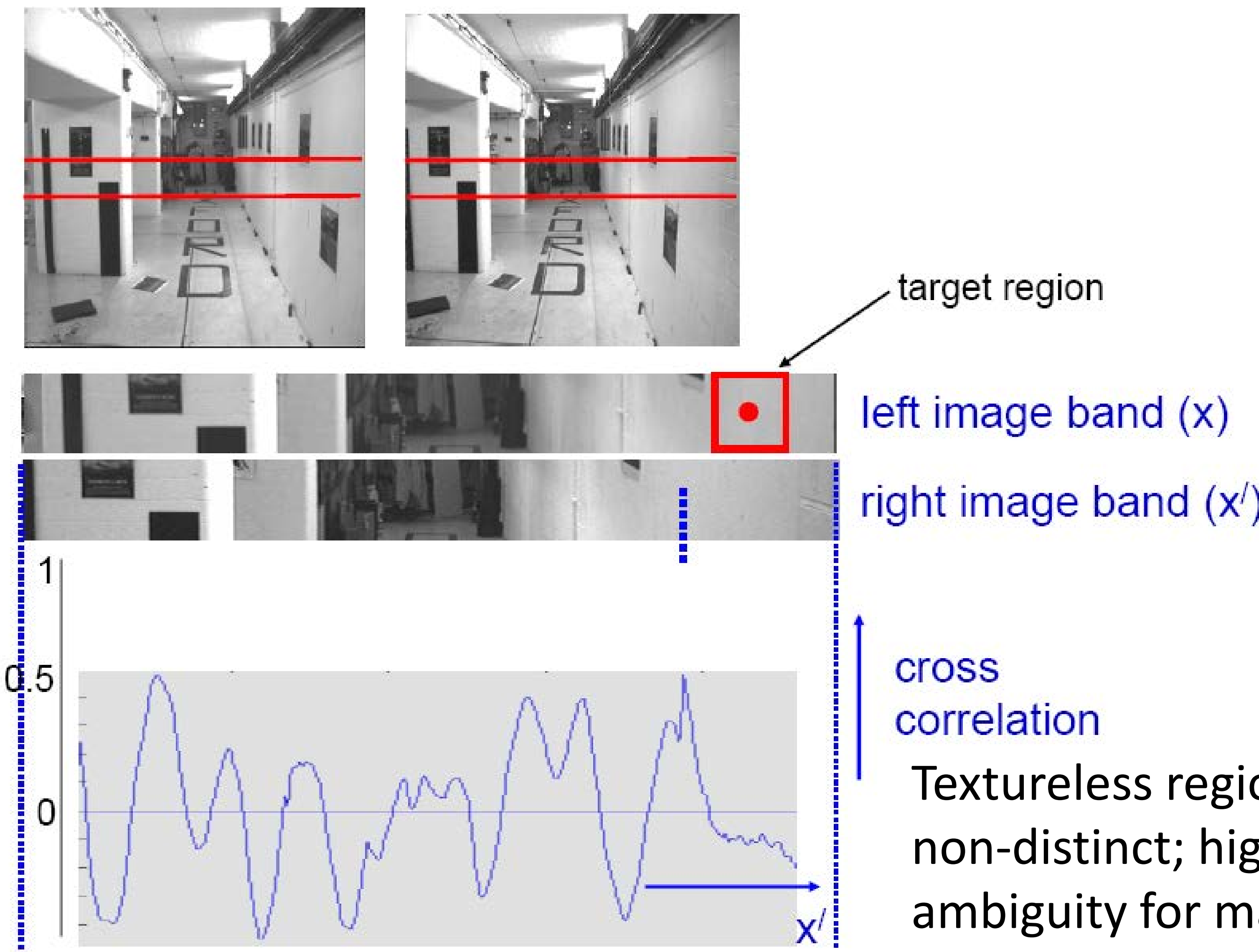
Textureless regions



Source: Andrew Zisserman

Grauman

Textureless regions



Textureless regions are non-distinct; high ambiguity for matches, → wrong matches

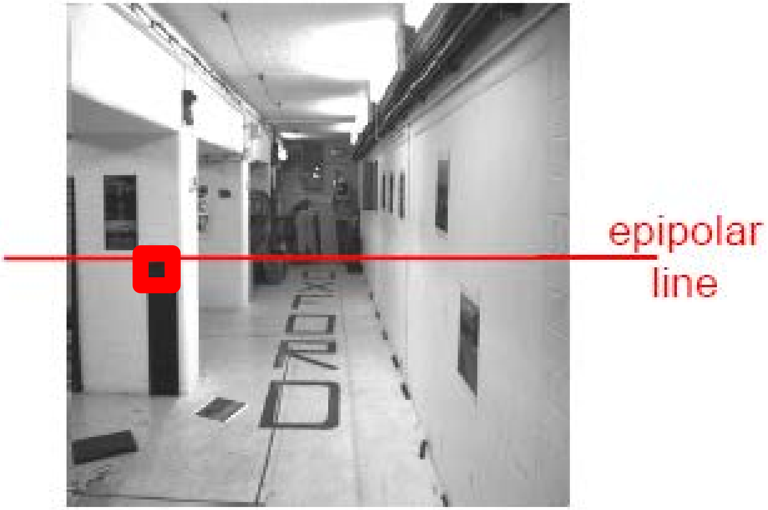
Source: Andrew Zisserman

Grauman

Effect of window size

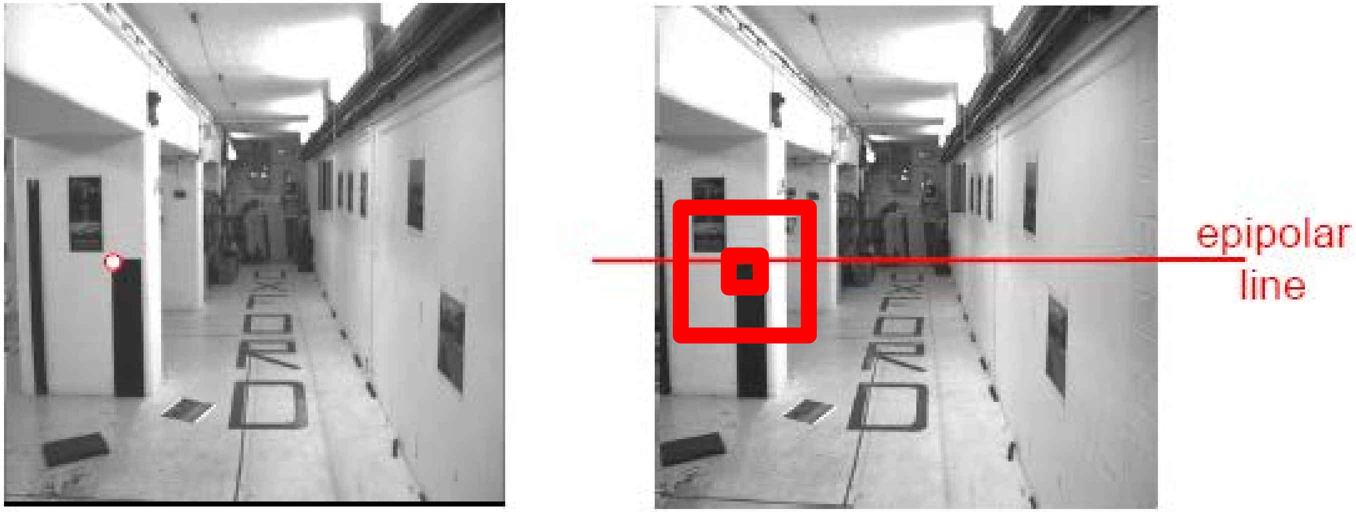
Source: Andrew Zisserman

Effect of window size



Source: Andrew Zisserman

Effect of window size



Source: Andrew Zisserman

Problems with window matching

Patch too small?

Patch too large?

Can try variable patch size [Okutomi and Kanade], or arbitrary window shapes [Veksler and Zabih]

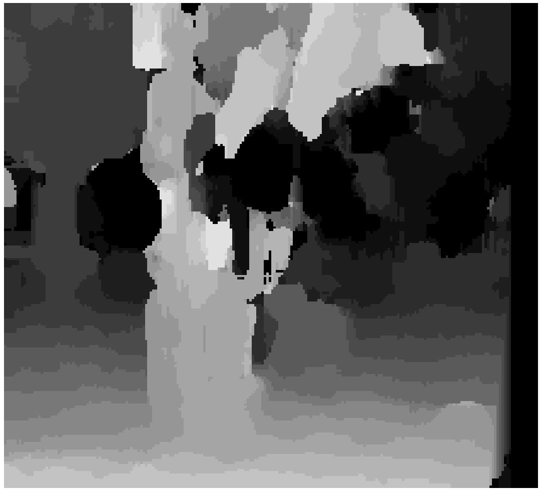
Effect of window size

Want window large enough to have sufficient intensity variation, yet small enough to contain only pixels with about the same disparity.

Figures from Li Zhang

Effect of window size

W = 3



W = 20

Want window large enough to have sufficient intensity variation, yet small enough to contain only pixels with about the same disparity.

Figures from Li Zhang

Grauman

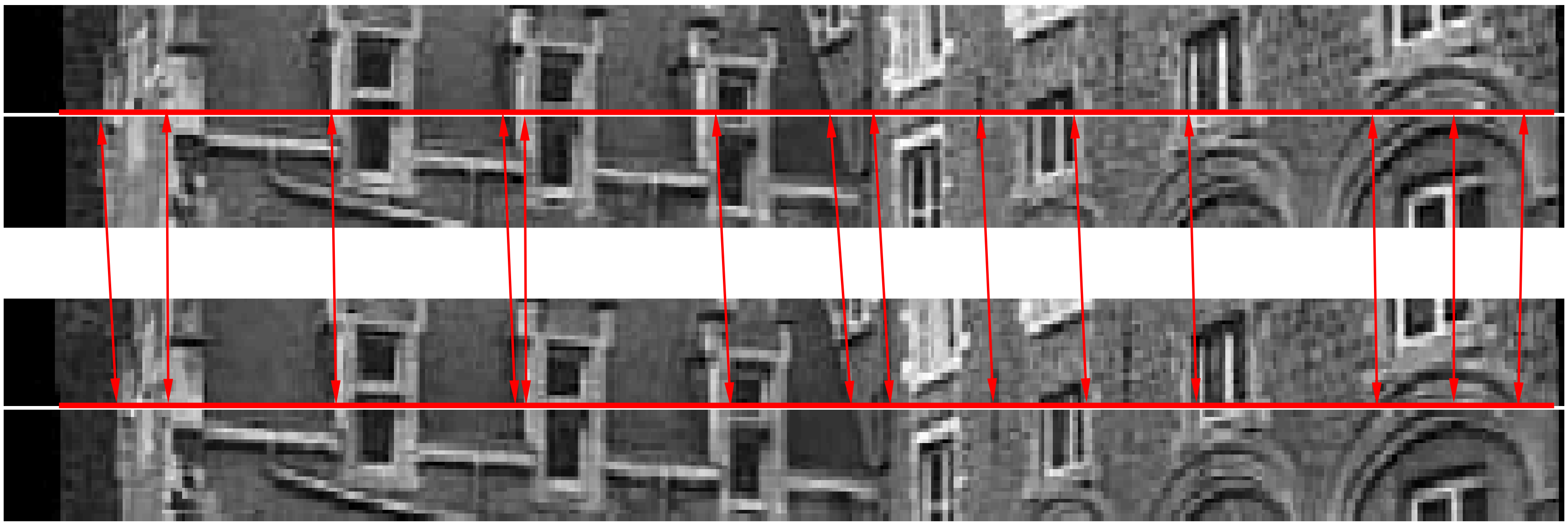
Problems?

- Ordering
- Occlusion
- Foreshortening

Solutions:

- Formulate Constraints
- Use more than two views
- Smart solutions vs. “brute force” searches with statistics

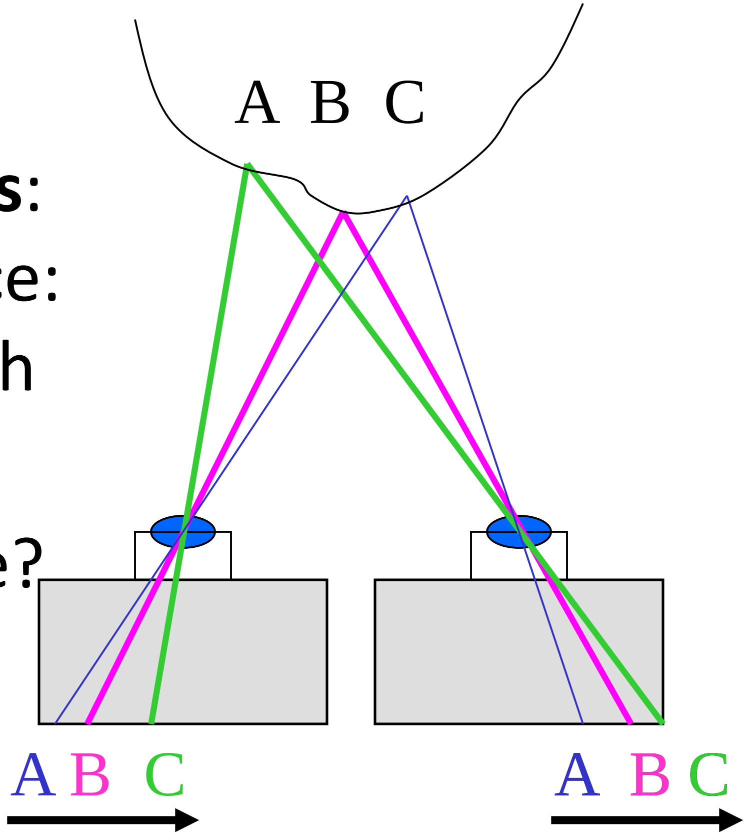
Exploiting scene constraints



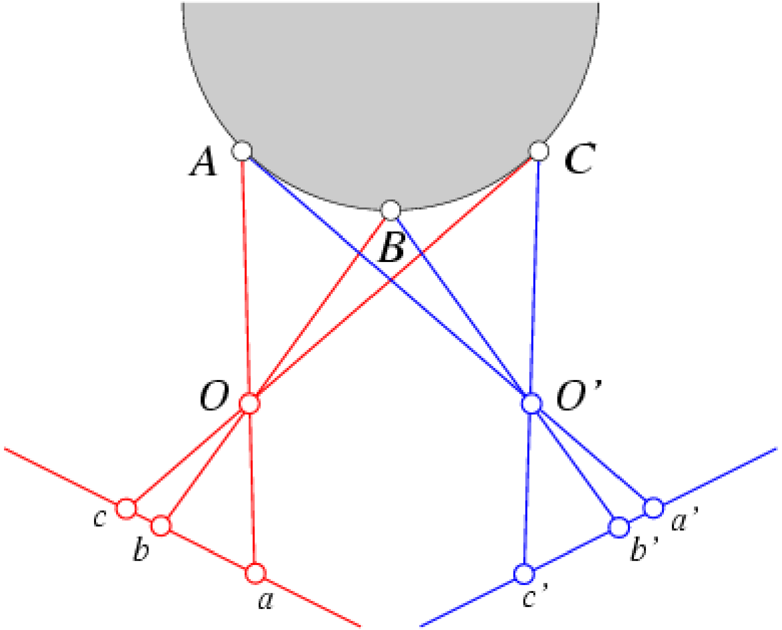
Additional geometric constraints for correspondence

[Faugeras, pp. 321]

- **Ordering of points**: Continuous surface: same order in both images.
- Is that always true?

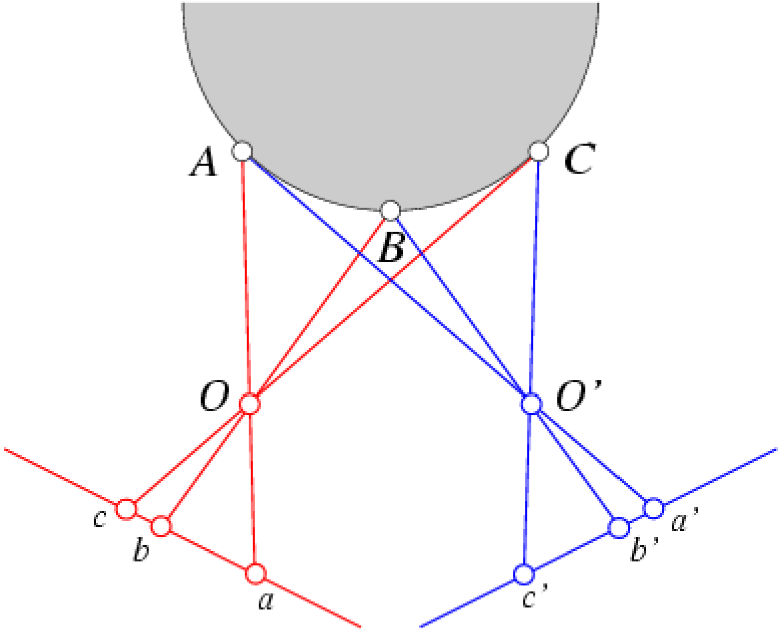


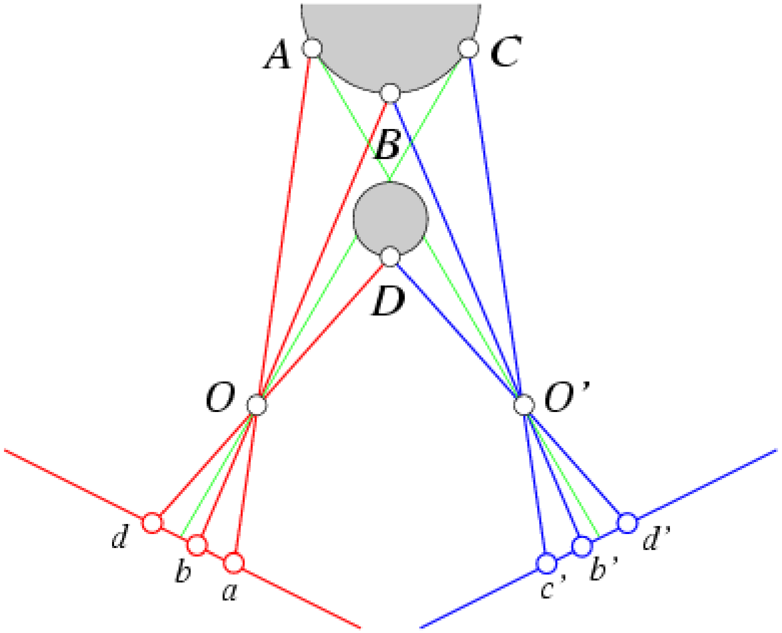
The Ordering Constraint



In general the points are in the same order on both epipolar lines.

The Ordering Constraint

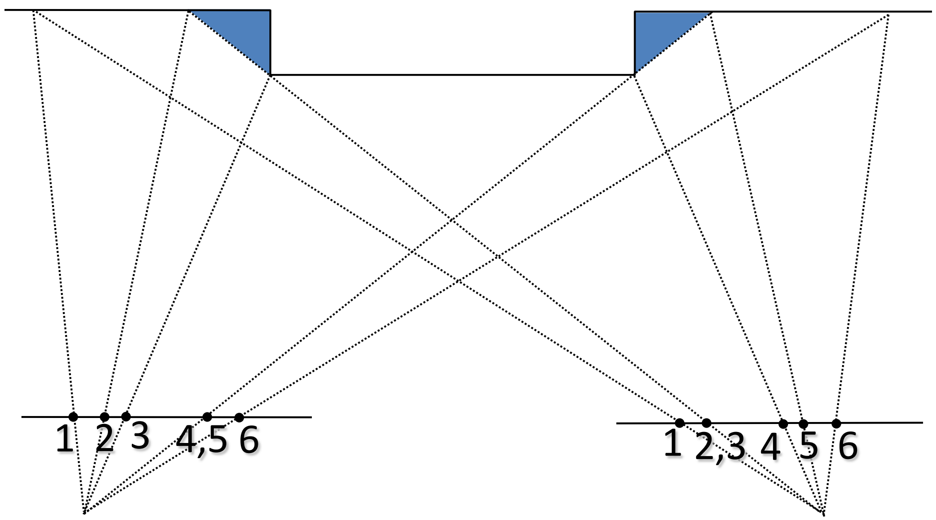




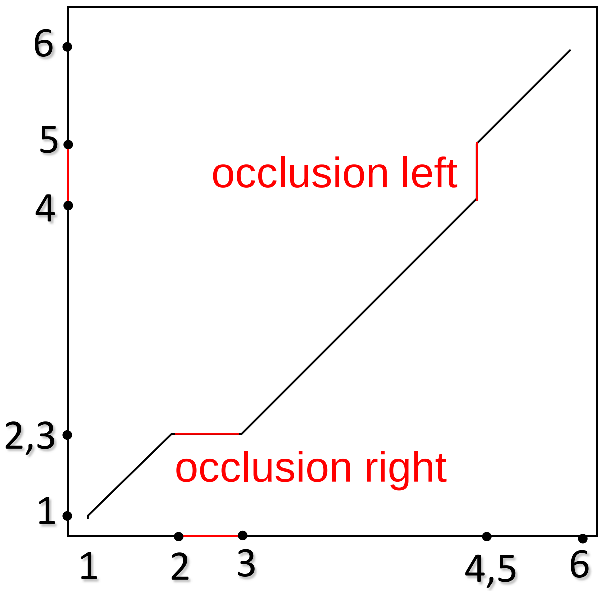
But it is not always the case..

Ordering constraint

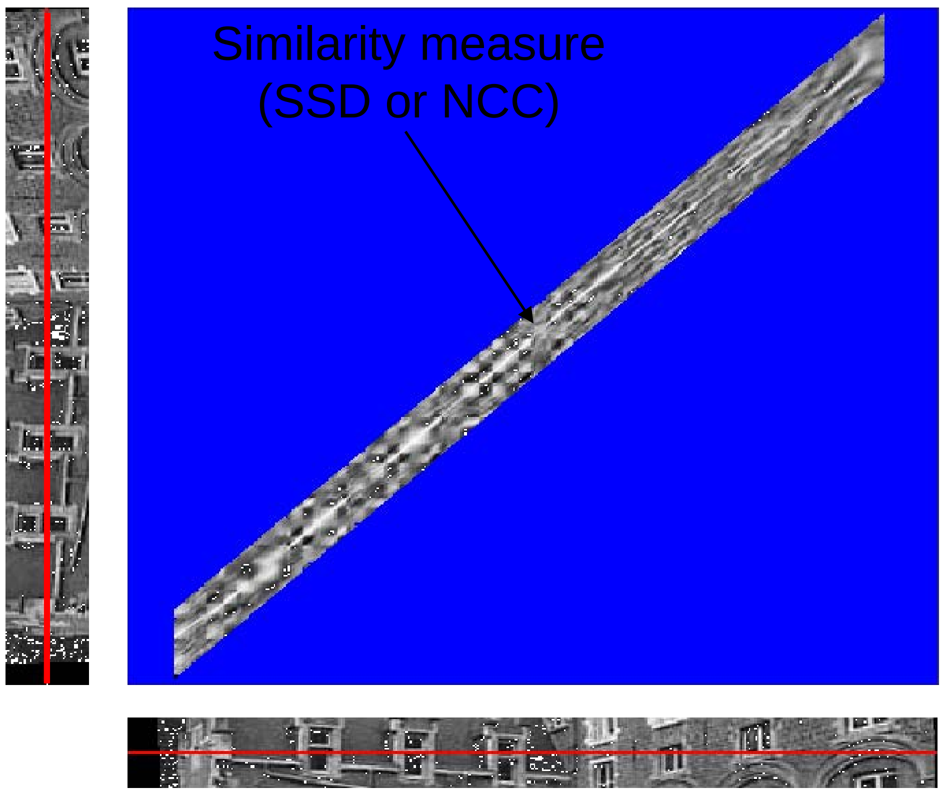
surface slice



surface as a path



Stereo matching



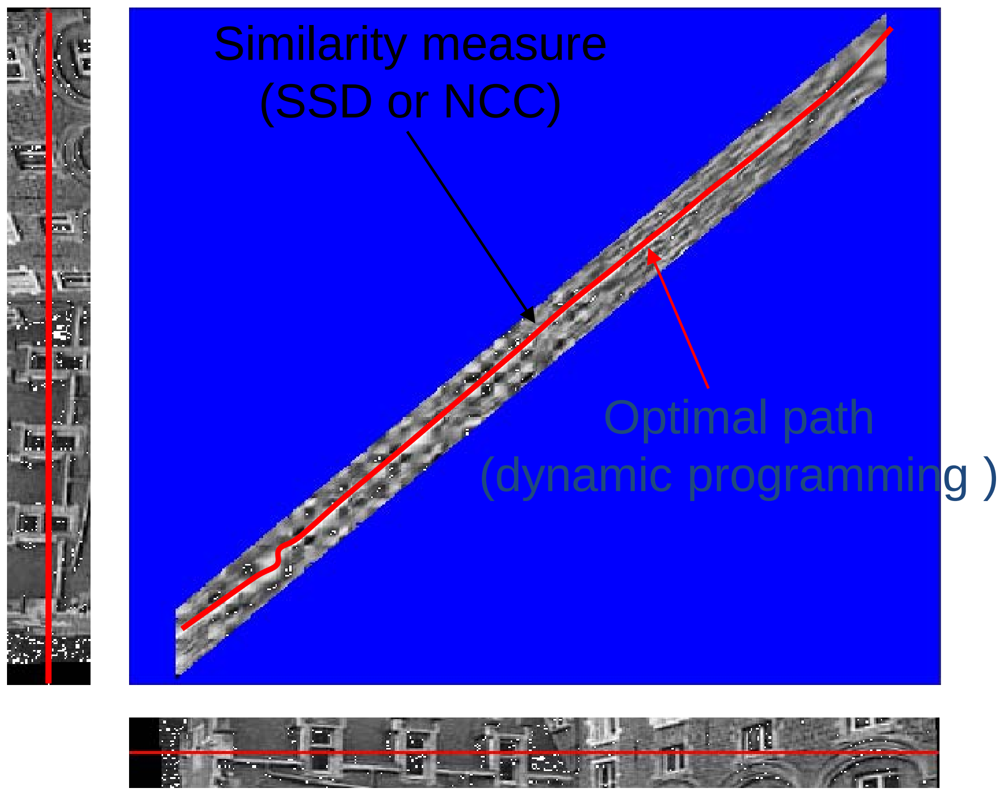
Constraints
- epipolar
- ordering
- uniqueness
- disparity limit

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

Consider all paths that satisfy the constraints

pick best using dynamic programming

Stereo matching



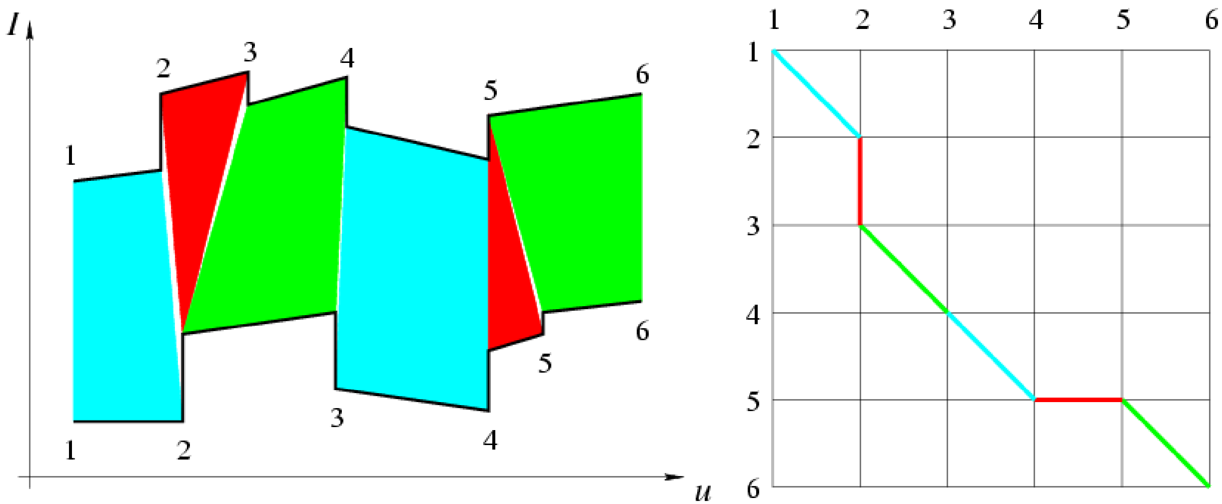
Constraints
- epipolar
- ordering
- uniqueness
- disparity limit

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

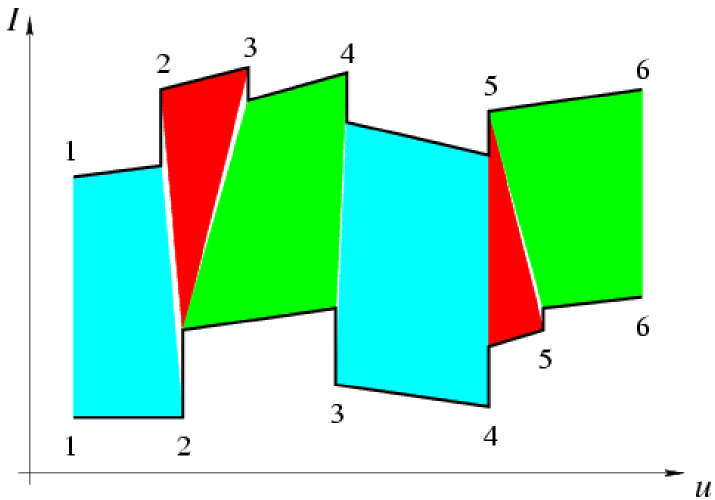
Consider all paths that satisfy the constraints

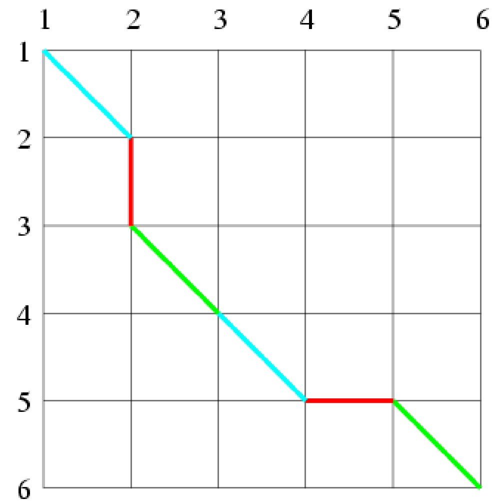
pick best using dynamic programming

Dynamic Programming (Baker and Binford, 1981)



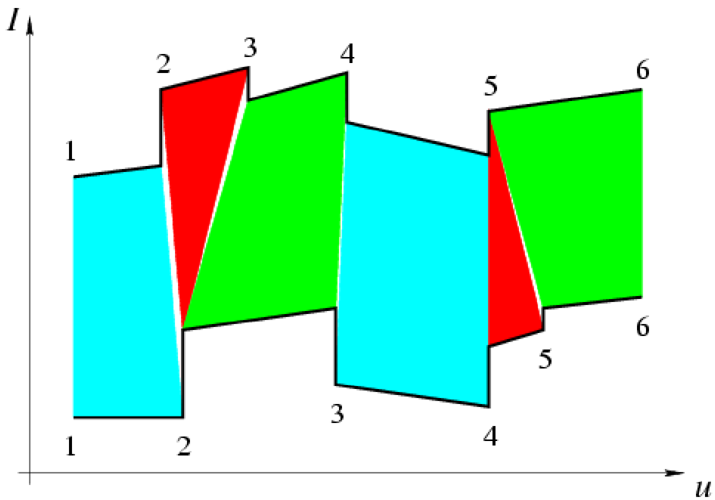
Dynamic Programming (Baker and Binford, 1981)

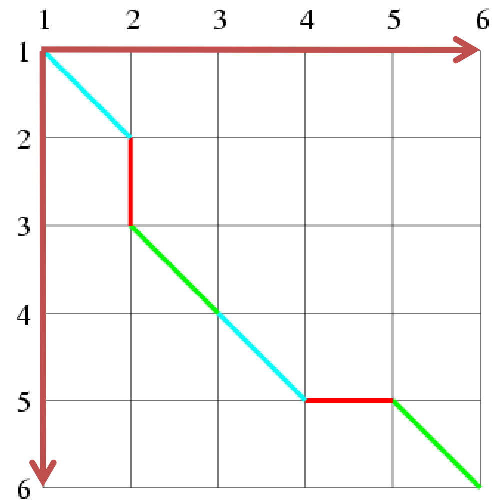





```
% Loop over all nodes (k, l) in ascending order.
for k = 1 to m do
  for l = 1 to n do
    % Initialize optimal cost C(k, l) and backward pointer B(k, l).
    C(k, l) ← +∞; B(k, l) ← nil;
    % Loop over all inferior neighbors (i, j) of (k, l).
    for (i, j) ∈ Inferior − Neighbors(k, l) do
      % Compute new path cost and update backward pointer if necessary.
      d ← C(i, j) + Arc − Cost(i, j, k, l);
      if d < C(k, l) then C(k, l) ← d; B(k, l) ← (i, j) endif;
      endfor;
    endfor;
  endfor;
% Construct optimal path by following backward pointers from (m, n).
P ← {(m, n)}; (i, j) ← (m, n);
while B(i, j) ≠ nil do (i, j) ← B(i, j); P ← {(i, j)} ∪ P endwhile.
```

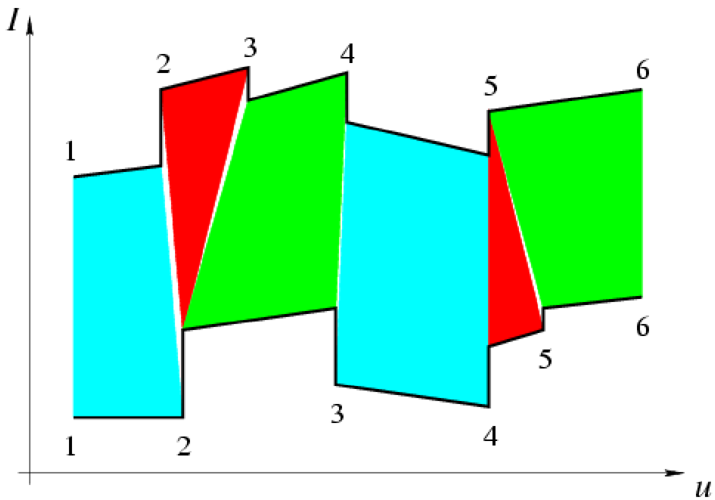
Dynamic Programming (Baker and Binford, 1981)

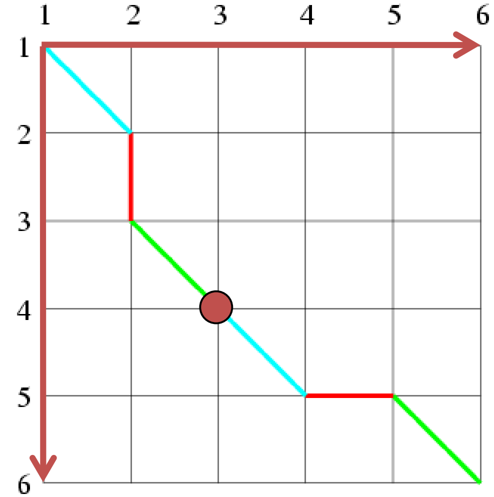




% Loop over all nodes (k, l) in ascending order.
for $k = 1$ to m do
  for $l = 1$ to n do
    % Initialize optimal cost $C(k, l)$ and backward pointer $B(k, l)$.
    $C(k, l) \leftarrow +\infty$; $B(k, l) \leftarrow$ nil;
    % Loop over all inferior neighbors (i, j) of (k, l).
    for $(i, j) \in$ Inferior $-$ Neighbors(k, l) do
      % Compute new path cost and update backward pointer if necessary.
      $d \leftarrow C(i, j) +$ Arc $-$ Cost(i, j, k, l);
      if $d < C(k, l)$ then $C(k, l) \leftarrow d$; $B(k, l) \leftarrow (i, j)$ endif;
      endfor;
    endfor;
  endfor;
% Construct optimal path by following backward pointers from (m, n).
$P \leftarrow \{(m, n)\}$; $(i, j) \leftarrow (m, n)$;
while $B(i, j) \neq$ nil do $(i, j) \leftarrow B(i, j)$; $P \leftarrow \{(i, j)\} \cup P$ endwhile.

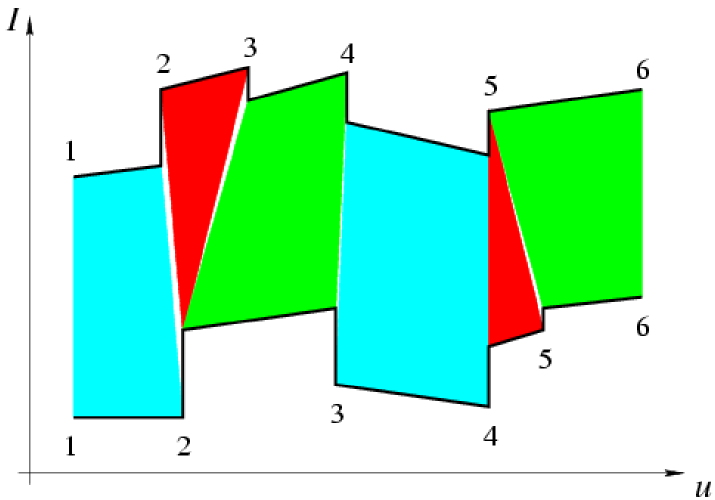
Dynamic Programming (Baker and Binford, 1981)

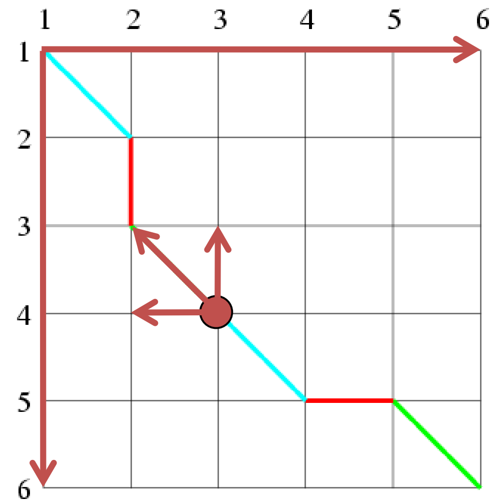




% Loop over all nodes (k, l) in ascending order.
for $k = 1$ to m do
  for $l = 1$ to n do
    % Initialize optimal cost $C(k, l)$ and backward pointer $B(k, l)$.
    $C(k, l) \leftarrow +\infty$; $B(k, l) \leftarrow$ nil;
    % Loop over all inferior neighbors (i, j) of (k, l).
    for $(i, j) \in$ Inferior – Neighbors(k, l) do
      % Compute new path cost and update backward pointer if necessary.
      $d \leftarrow C(i, j) +$ Arc – Cost(i, j, k, l);
      if $d < C(k, l)$ then $C(k, l) \leftarrow d$; $B(k, l) \leftarrow (i, j)$ endif;
      endfor;
    endfor;
  endfor;
% Construct optimal path by following backward pointers from (m, n).
$P \leftarrow \{(m, n)\}$; $(i, j) \leftarrow (m, n)$;
while $B(i, j) \neq$ nil do $(i, j) \leftarrow B(i, j)$; $P \leftarrow \{(i, j)\} \cup P$ endwhile.

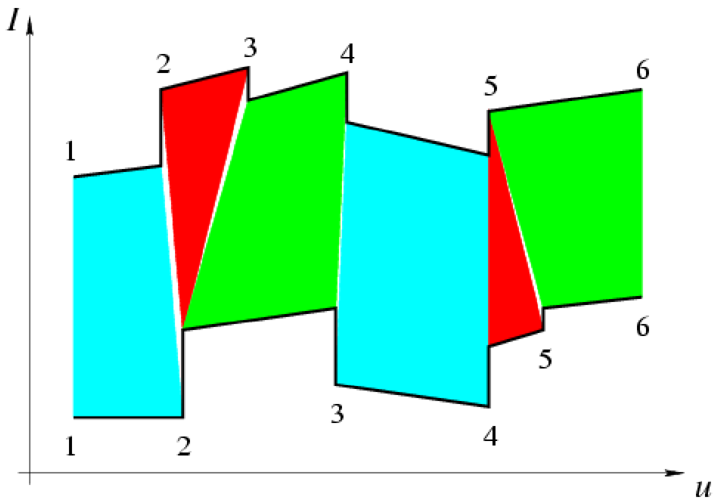
Dynamic Programming (Baker and Binford, 1981)

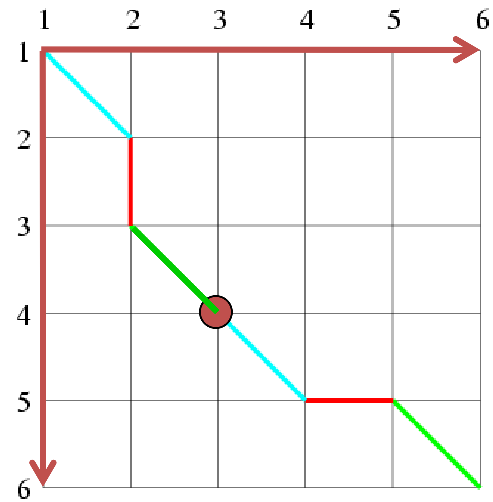




% Loop over all nodes (k, l) in ascending order.
for $k = 1$ to m do
  for $l = 1$ to n do
    % Initialize optimal cost $C(k, l)$ and backward pointer $B(k, l)$.
    $C(k, l) \leftarrow +\infty$; $B(k, l) \leftarrow$ nil;
    % Loop over all inferior neighbors (i, j) of (k, l).
    for $(i, j) \in$ Inferior – Neighbors(k, l) do
      % Compute new path cost and update backward pointer if necessary.
      $d \leftarrow C(i, j) +$ Arc – Cost(i, j, k, l);
      if $d < C(k, l)$ then $C(k, l) \leftarrow d$; $B(k, l) \leftarrow (i, j)$ endif;
      endfor;
    endfor;
  endfor;
% Construct optimal path by following backward pointers from (m, n).
$P \leftarrow \{(m, n)\}$; $(i, j) \leftarrow (m, n)$;
while $B(i, j) \neq$ nil do $(i, j) \leftarrow B(i, j)$; $P \leftarrow \{(i, j)\} \cup P$ endwhile.

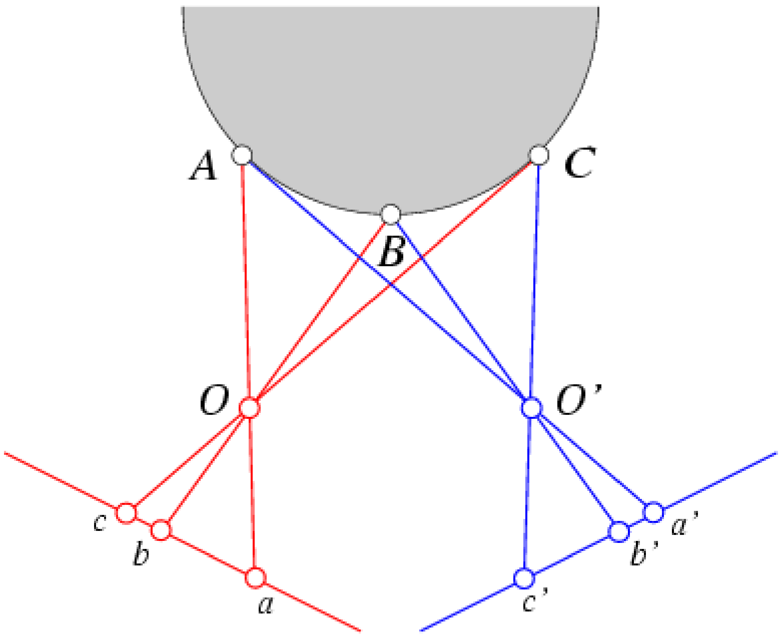
Dynamic Programming (Baker and Binford, 1981)





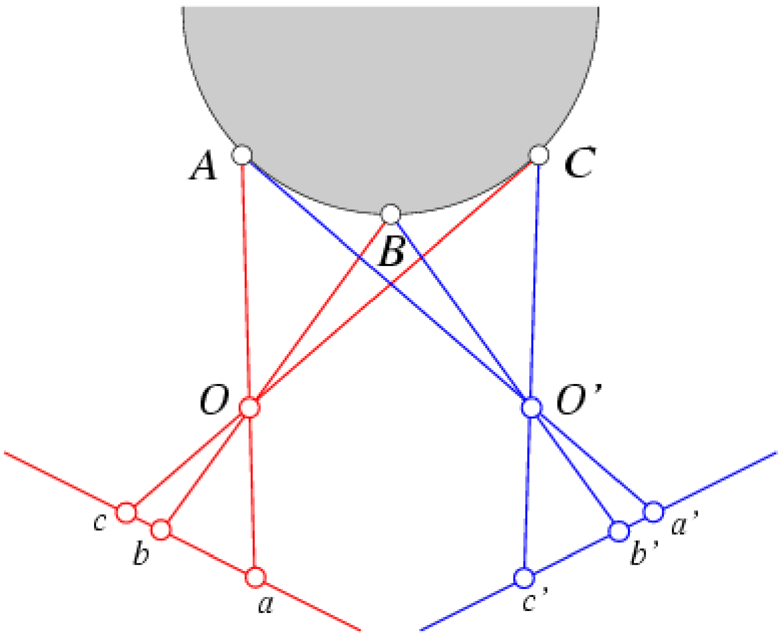
% Loop over all nodes (k, l) in ascending order.
for $k = 1$ to m do
  for $l = 1$ to n do
    % Initialize optimal cost $C(k, l)$ and backward pointer $B(k, l)$.
    $C(k, l) \leftarrow +\infty$; $B(k, l) \leftarrow$ nil;
    % Loop over all inferior neighbors (i, j) of (k, l).
    for $(i, j) \in$ Inferior $-$ Neighbors(k, l) do
      % Compute new path cost and update backward pointer if necessary.
      $d \leftarrow C(i, j) +$ Arc $-$ Cost(i, j, k, l);
      if $d < C(k, l)$ then $C(k, l) \leftarrow d$; $B(k, l) \leftarrow (i, j)$ endif;
      endfor;
    endfor;
  endfor;
% Construct optimal path by following backward pointers from (m, n).
$P \leftarrow \{(m, n)\}$; $(i, j) \leftarrow (m, n)$;
while $B(i, j) \neq$ nil do $(i, j) \leftarrow B(i, j)$; $P \leftarrow \{(i, j)\} \cup P$ endwhile.

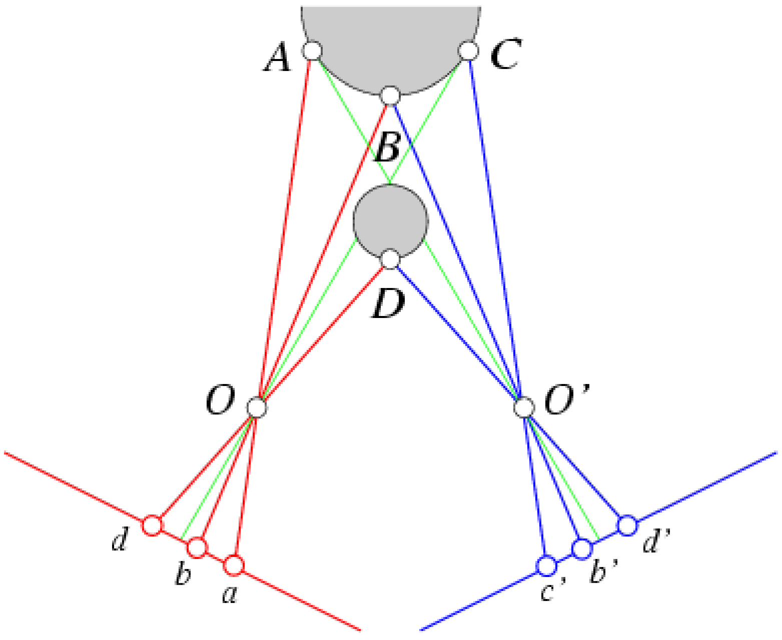
The Ordering Constraint



In general the points are in the same order on both epipolar lines.

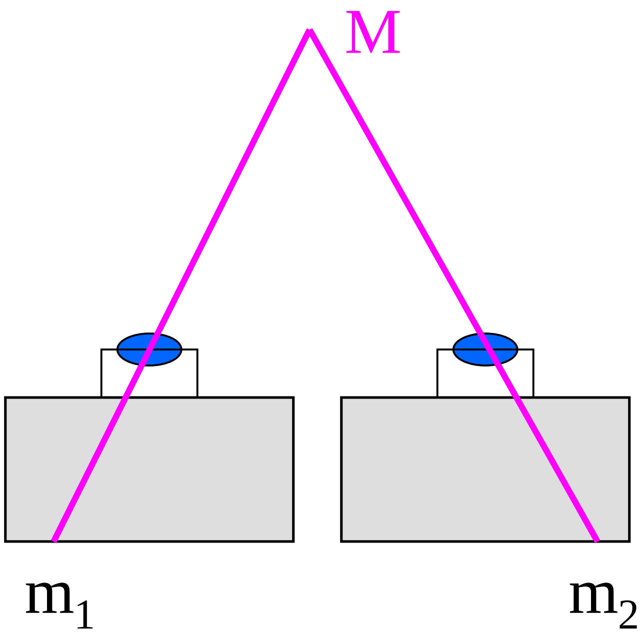
The Ordering Constraint



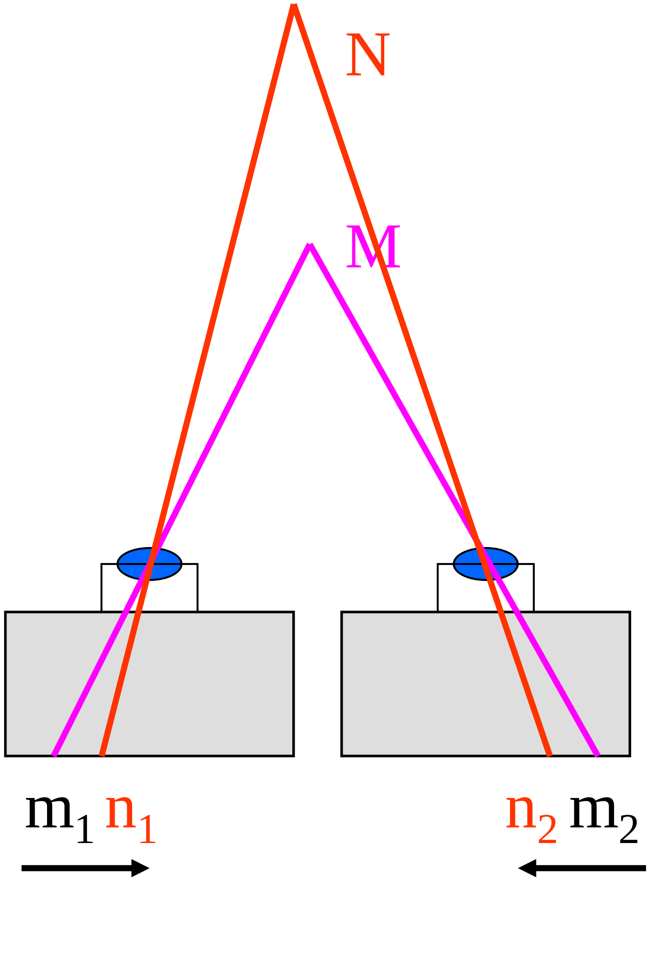


But it is not always the case..

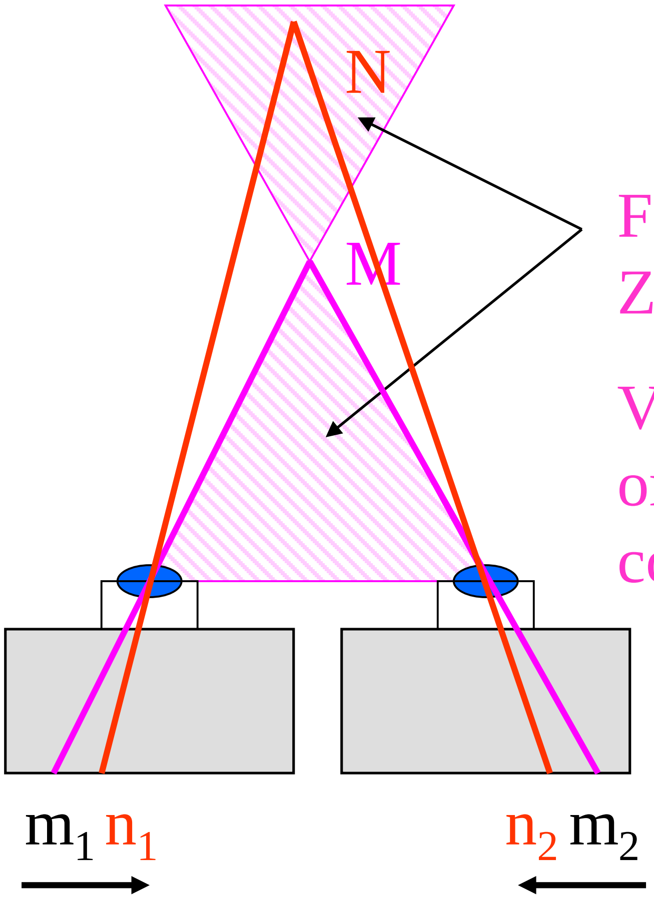
Forbidden Zone



Forbidden Zone



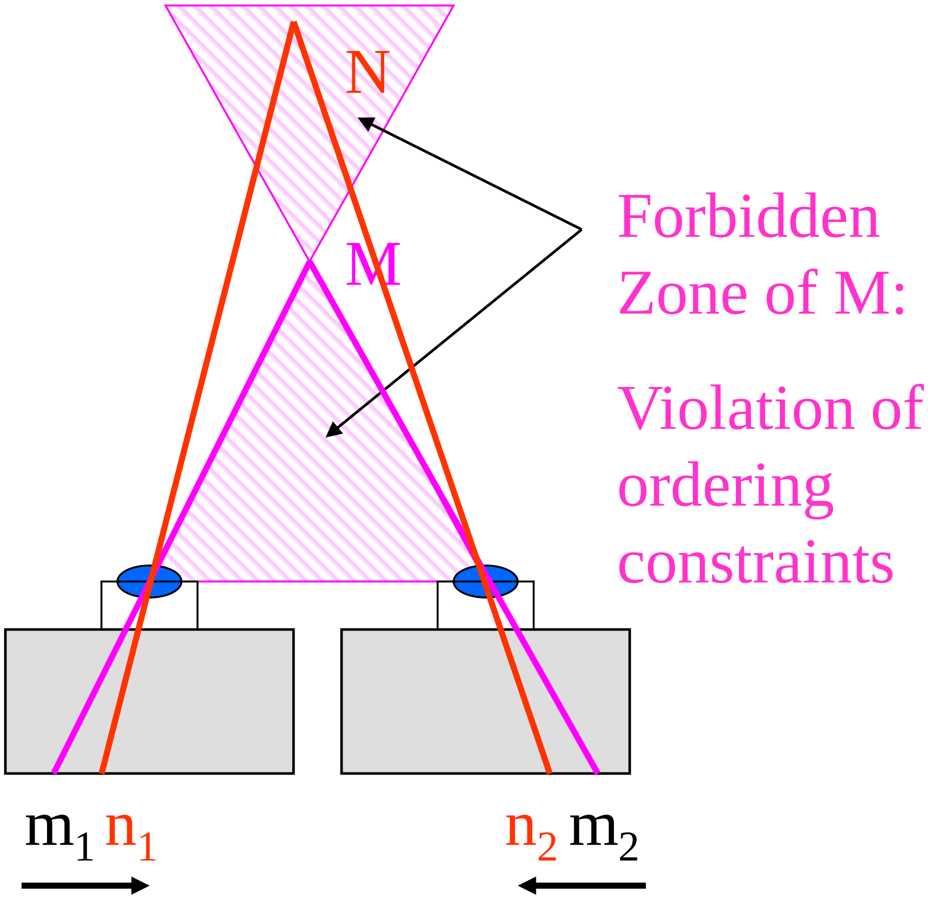
Forbidden Zone



Forbidden Zone of M:

Violation of ordering constraints

Forbidden Zone



Practical applications:

- Object bulges out: ok
- In general: ordering across whole image is not reliable feature
- Use ordering constraints for neighbors of M within small neighborhood only

Disparity map

image I(x,y)

Disparity map D(x,y)

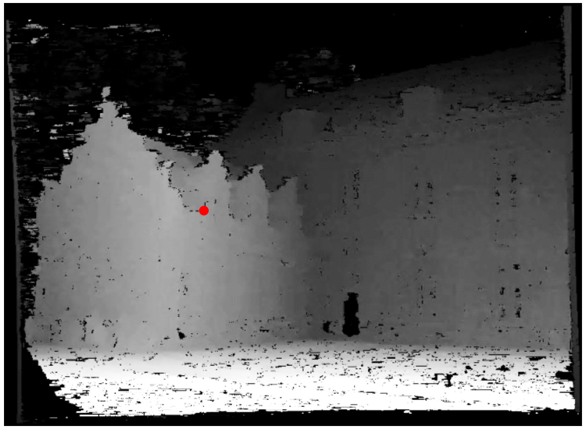


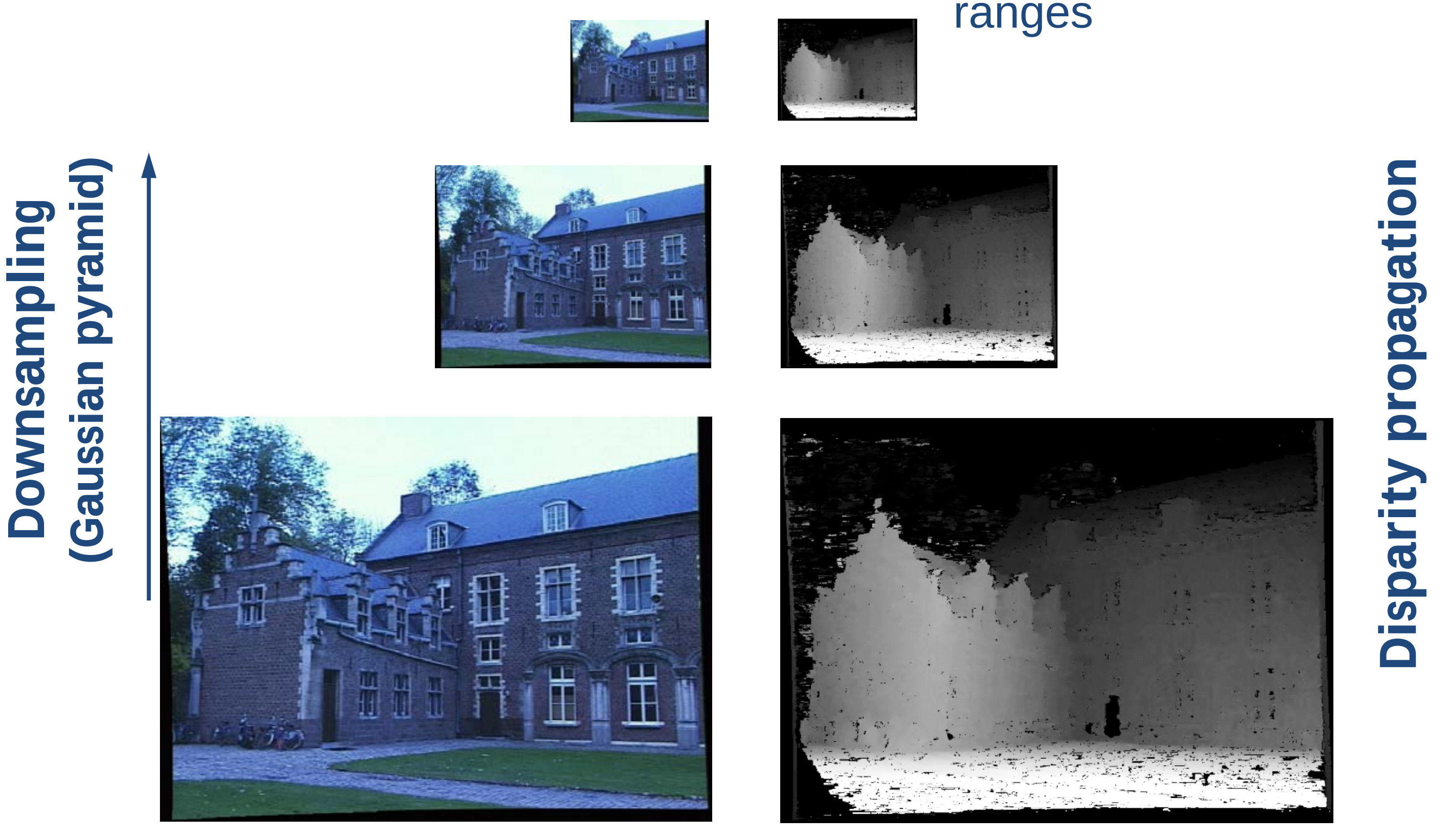
image I´(x´,y´)

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

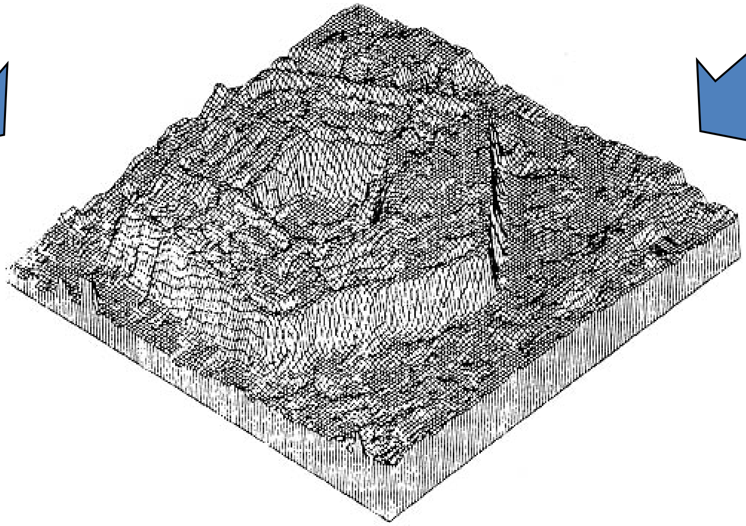
Hierarchical stereo matching

Allows faster computation

Deals with large disparity ranges



Dynamic Programming (Ohta and Kanade, 1985)

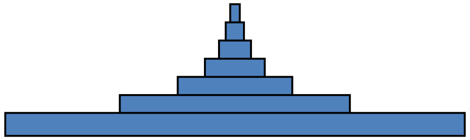


Reprinted from “Stereo by Intra- and Intet-Scanline Search,” by Y. Ohta and T. Kanade, IEEE Trans. on Pattern Analysis and Machine Intelligence, 7(2):139-154 (1985). © 1985 IEEE.

Real-time stereo on graphics hardware

Ruigang Yang and Marc Pollefeys, UNC

- Computes Sum-of-Square-Differences
- Hardware mip-map generation used to aggregate results over support region
- Trade-off between small and large support window



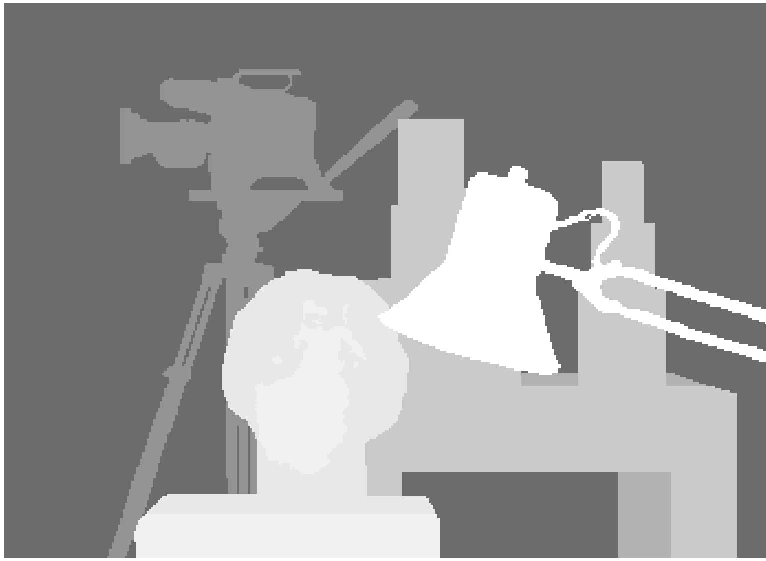
Shape of a kernel
for summing up 6 levels

140M disparity hypothesis/sec on Radeon 9700pro
e.g. 512x512x20disparities at 30Hz

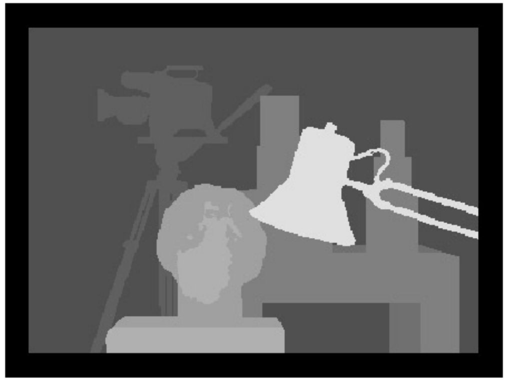
Stereo results

– Data from University of Tsukuba
– Similar results on other images without ground truth

Scene



Ground truth

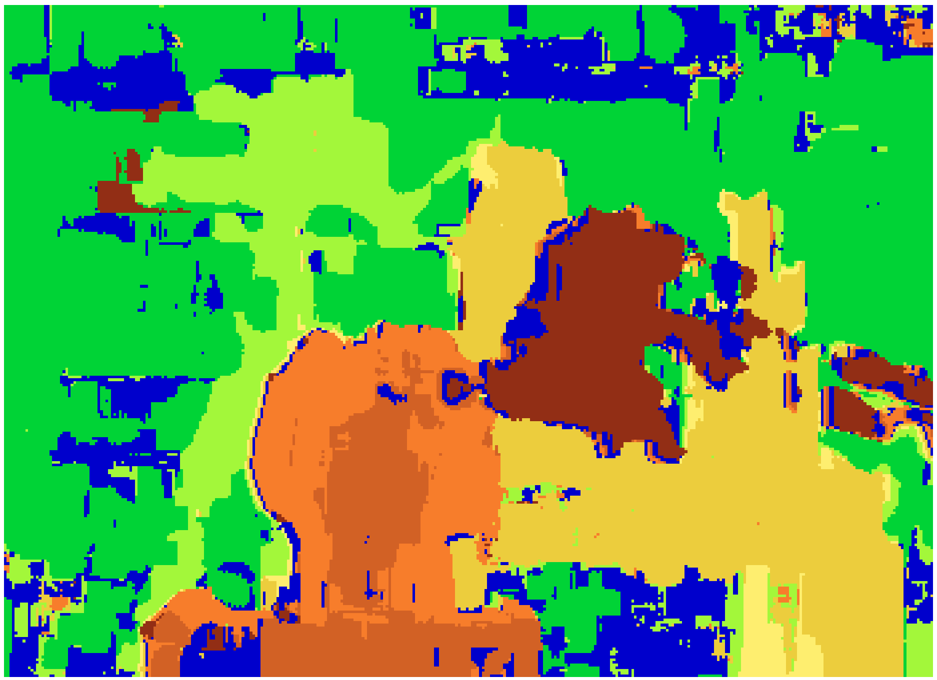


True disparities

16 – Fast Correlation

*1 – SSD+MF

Results with window correlation



Window-based matching
(best window size)

Ground truth

Results with better method

State of the art method

Boykov et al., Fast Approximate Energy Minimization via Graph Cuts, International Conference on Computer Vision, September 1999.

Ground truth

Material I

- http://vision.middlebury.edu/stereo/

- (online stereo pairs and truth (depth maps)
- Stereo correspondence software: e.g. http://vision.middlebury.edu/stereo/data/scenes2001/data/imagehtml/tsukuba.html
- CVonline compendium: http://homepages.inf.ed.ac.uk/rbf/CVonline/

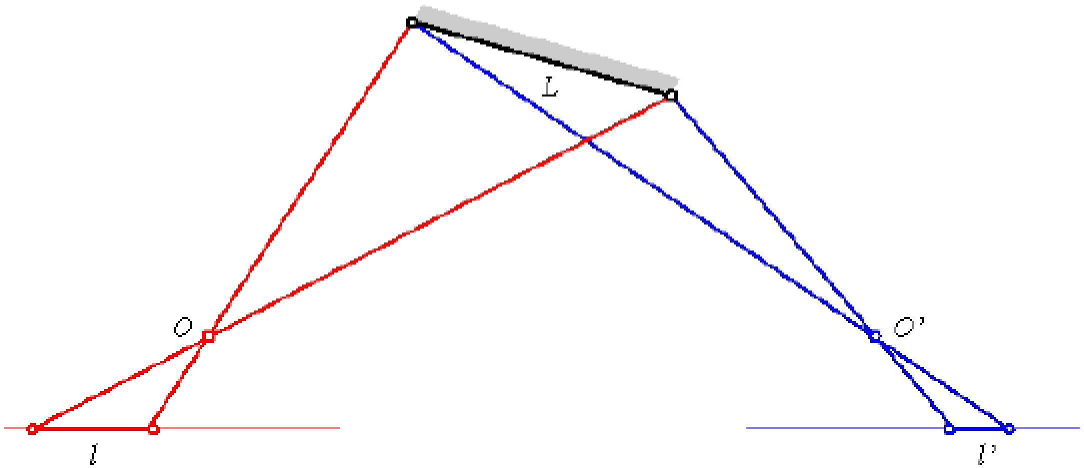
Material II

- Epipolar Geometry, Rectification:
- http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/FUSIELLO2/rectif_cvol.html
- and: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT11/node11.html
- Stereo:
- http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT11/lect11.html
- 3D Reconstruction:
- http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT11/node8.html

Additional Materials

Problem: Foreshortening

Window methods assume fronto-parallel surface at 3-D point.

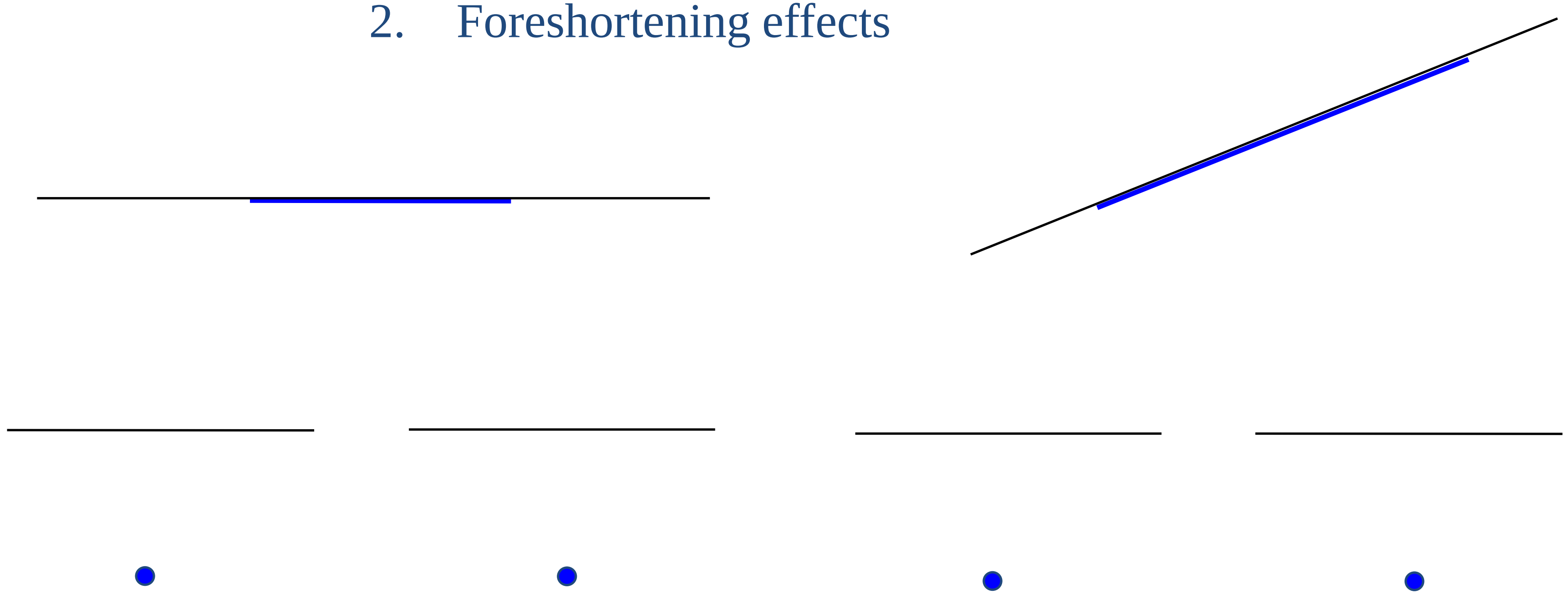


Initial estimates of the disparity can be used to warp the correlation windows to compensate for unequal amounts of foreshortening in the two pictures [Kass, 1987; Devernay and Faugeras, 1994].

Why is cross-correlation such a poor measure in the second case?

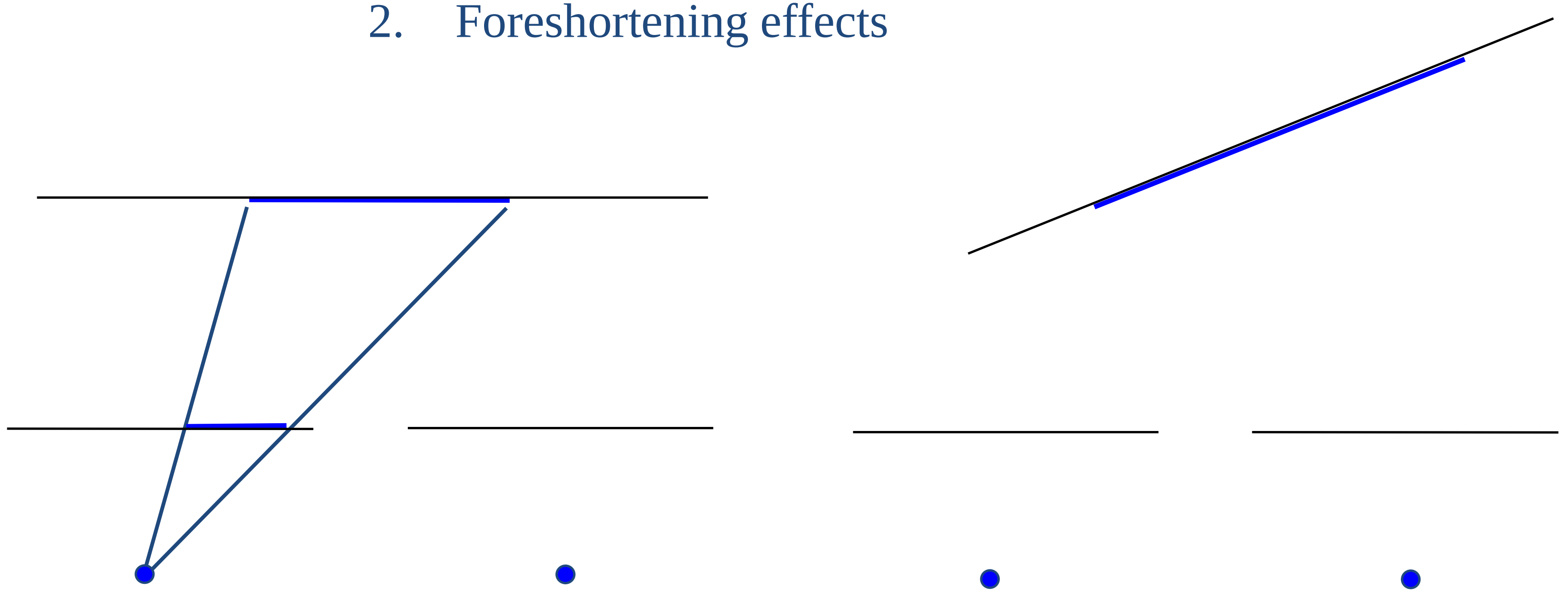
1. The neighbourhood region does not have a "distinctive" spatial intensity distribution

2. Foreshortening effects



Why is cross-correlation such a poor measure in the second case?

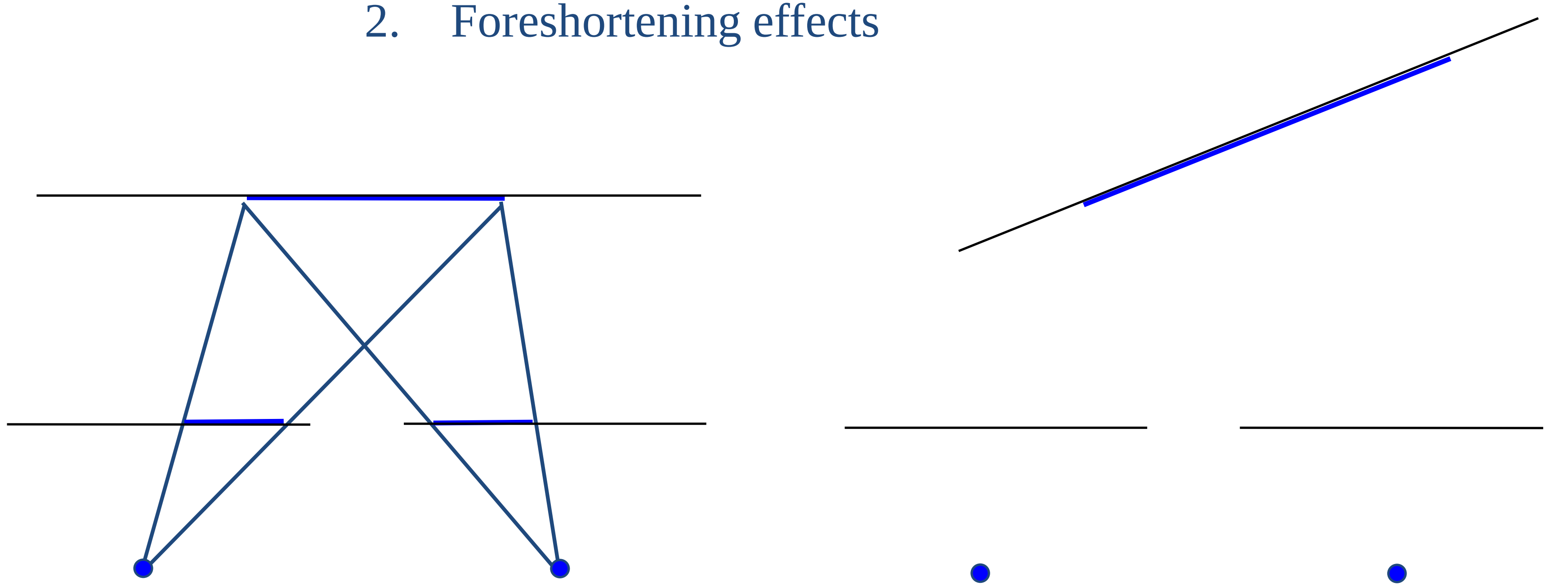
1. The neighbourhood region does not have a “distinctive” spatial intensity distribution
2. Foreshortening effects



Why is cross-correlation such a poor measure in the second case?

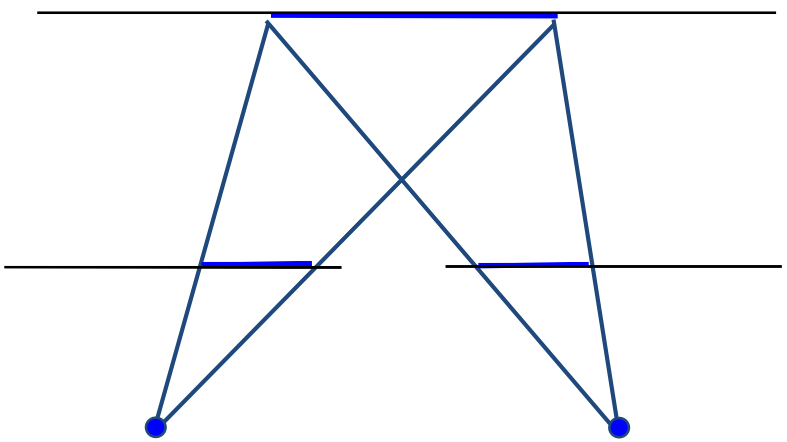
1. The neighbourhood region does not have a “distinctive” spatial intensity distribution

2. Foreshortening effects



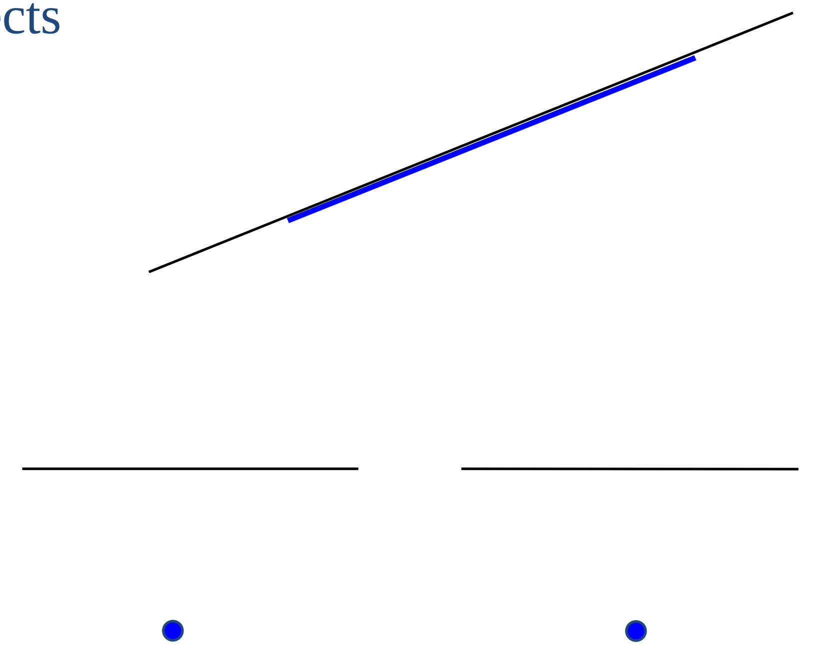
Why is cross-correlation such a poor measure in the second case?

1. The neighbourhood region does not have a “distinctive” spatial intensity distribution
2. Foreshortening effects



fronto-parallel surface

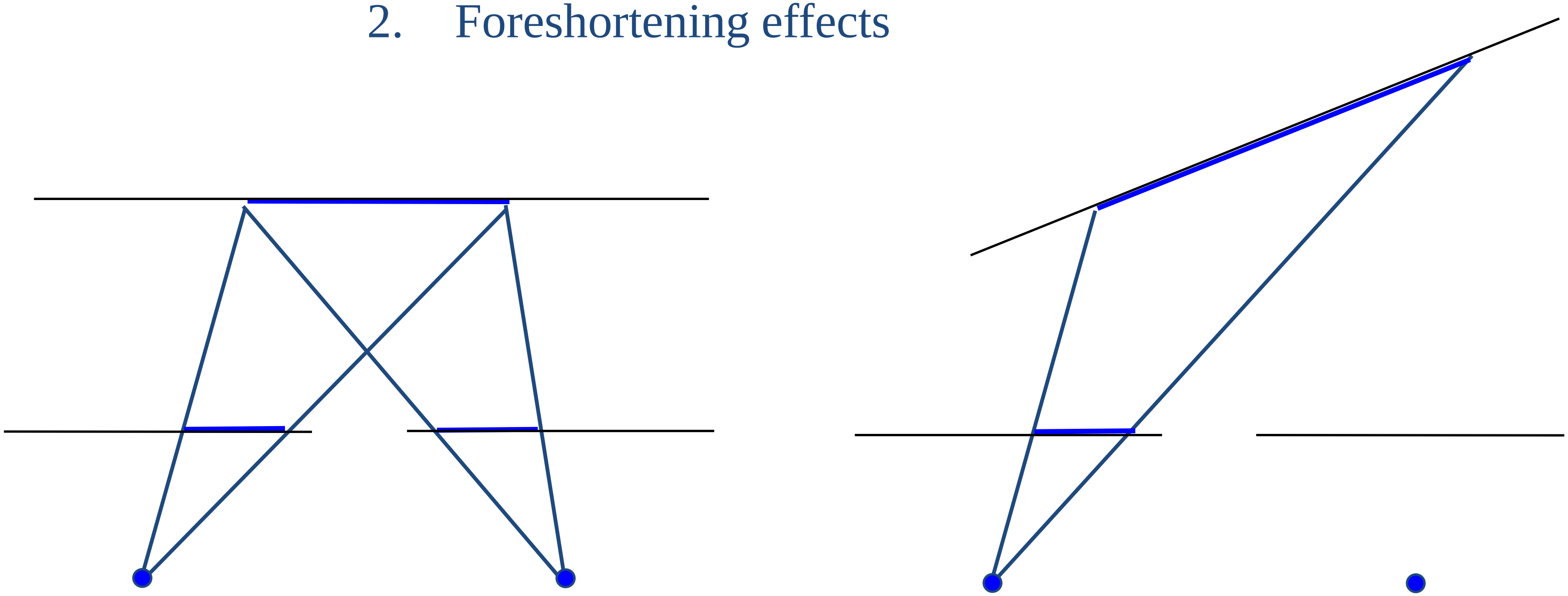
imaged length the same



Why is cross-correlation such a poor measure in the second case?

1. The neighbourhood region does not have a “distinctive” spatial intensity distribution

2. Foreshortening effects



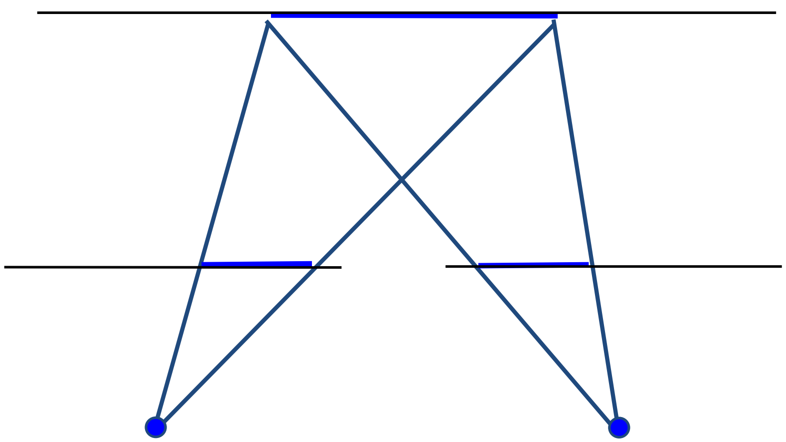
fronto-parallel surface

imaged length the same

Why is cross-correlation such a poor measure in the second case?

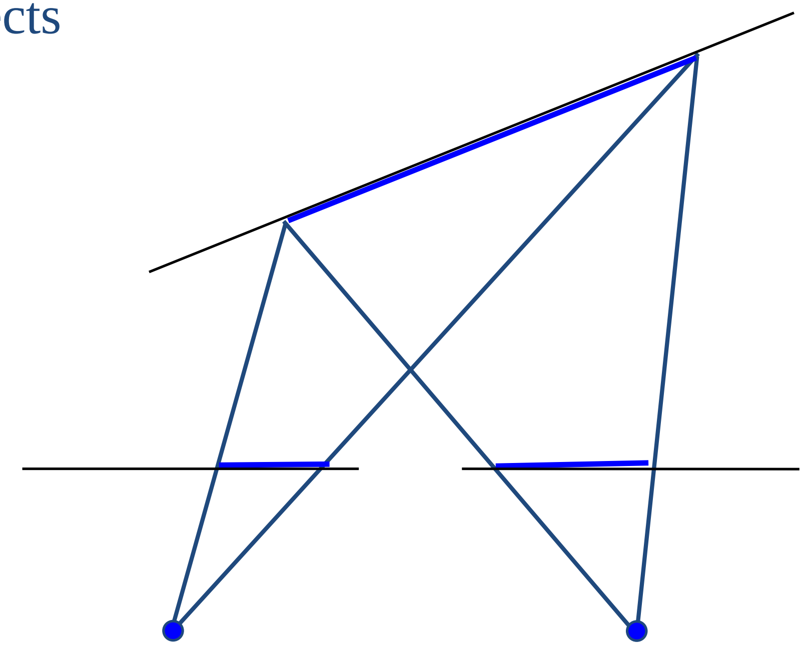
1. The neighbourhood region does not have a "distinctive" spatial intensity distribution

2. Foreshortening effects



fronto-parallel surface

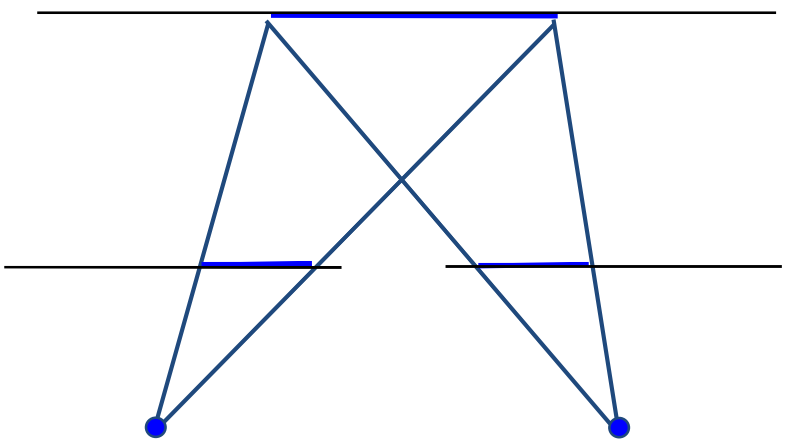
imaged length the same



Why is cross-correlation such a poor measure in the second case?

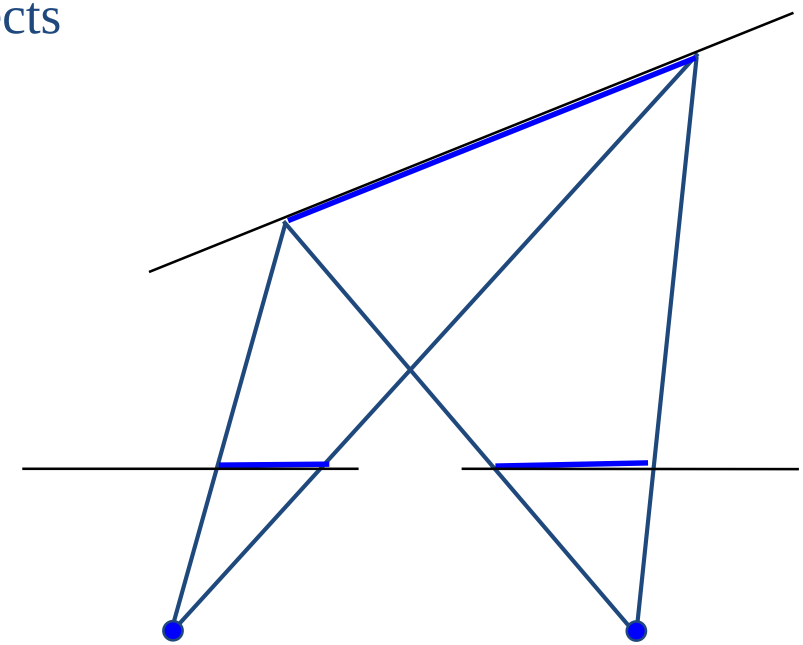
1. The neighbourhood region does not have a "distinctive" spatial intensity distribution

2. Foreshortening effects



fronto-parallel surface

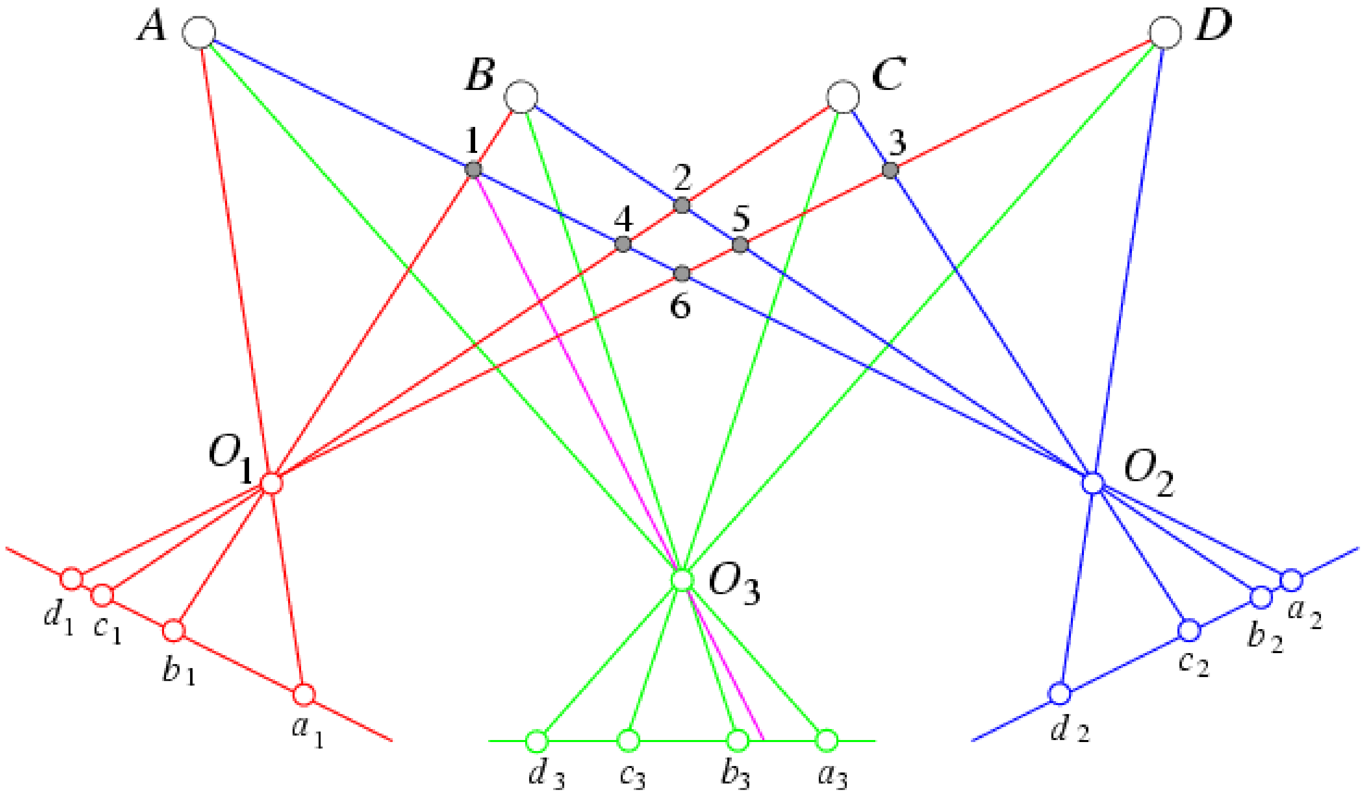
imaged length the same



slanting surface

imaged lengths differ

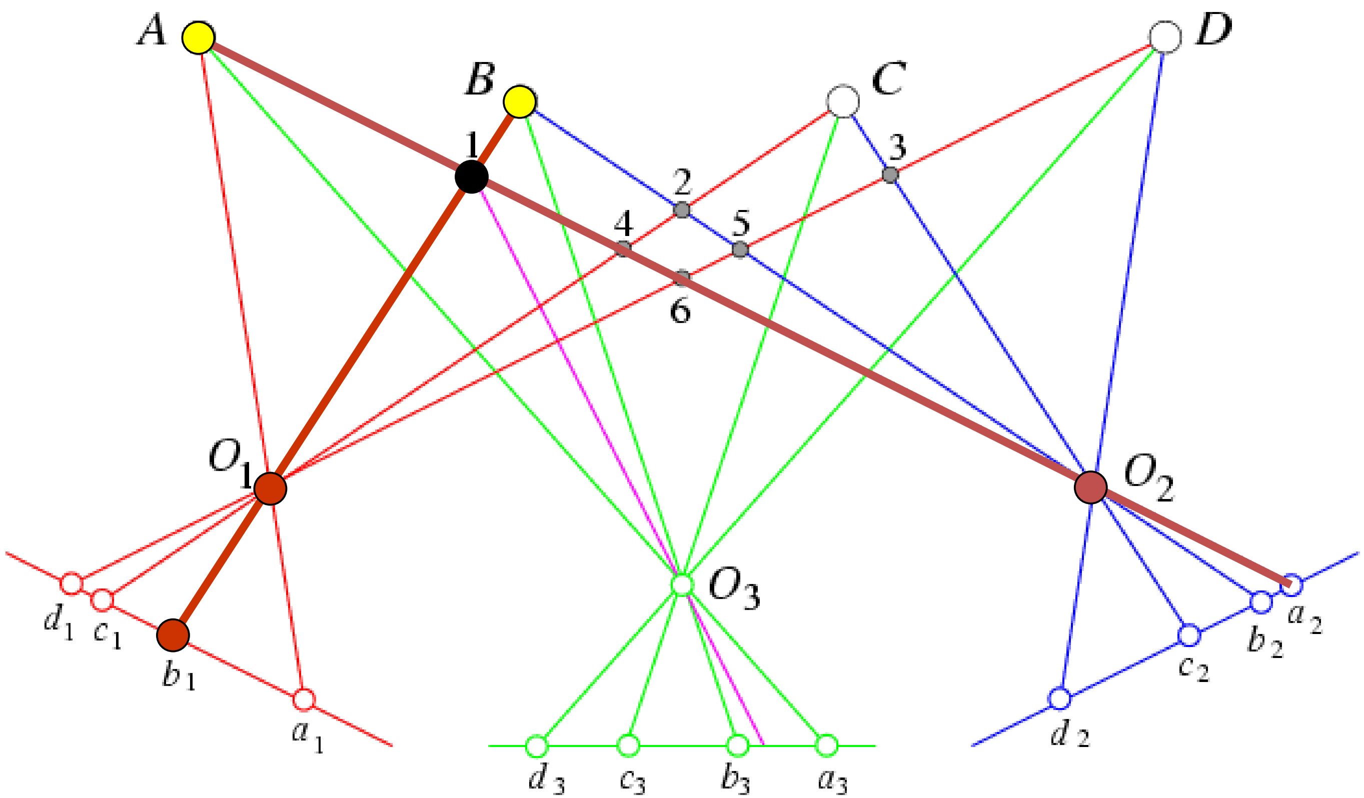
Three Views



The third eye can be used for verification..

[Demo epipolar geometry]()

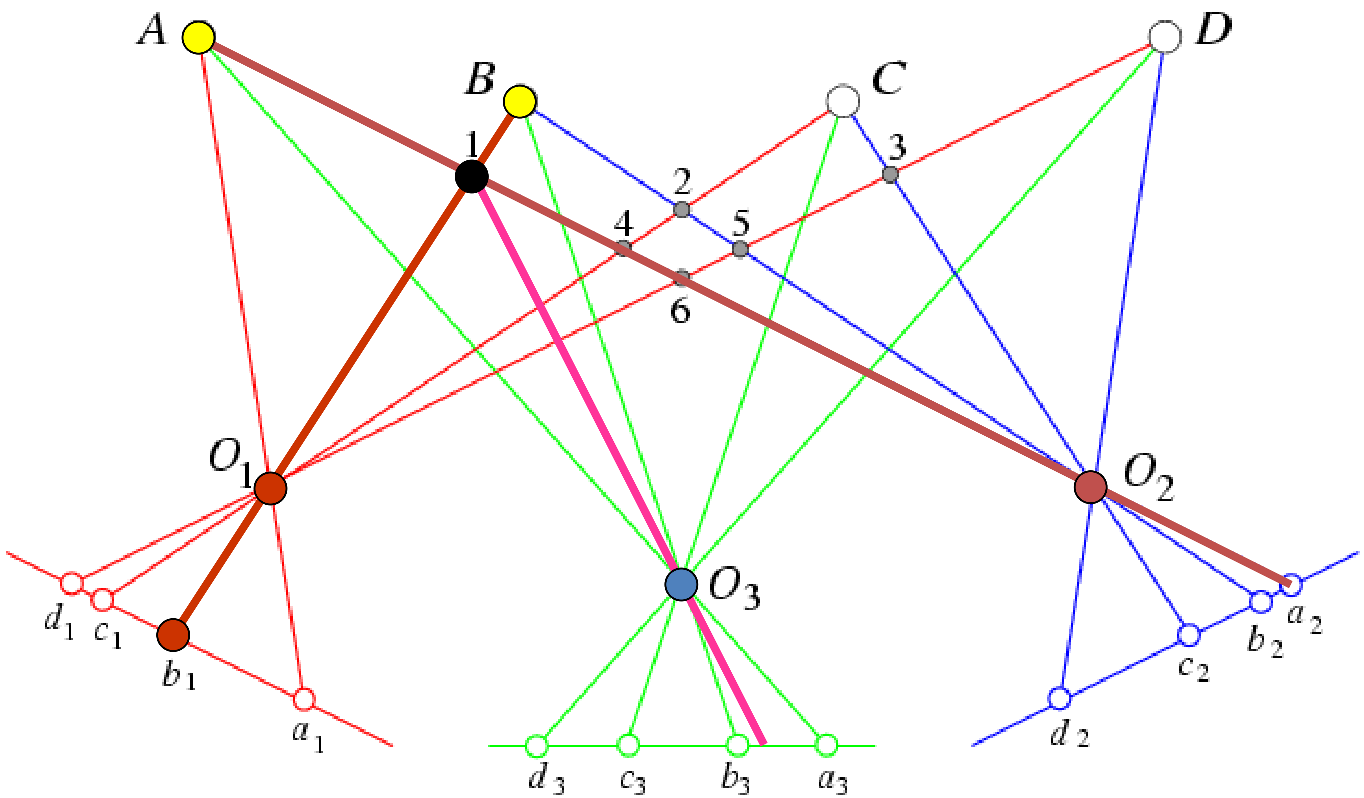
Three Views



The third eye can be used for verification..

[Demo epipolar geometry]()

Three Views

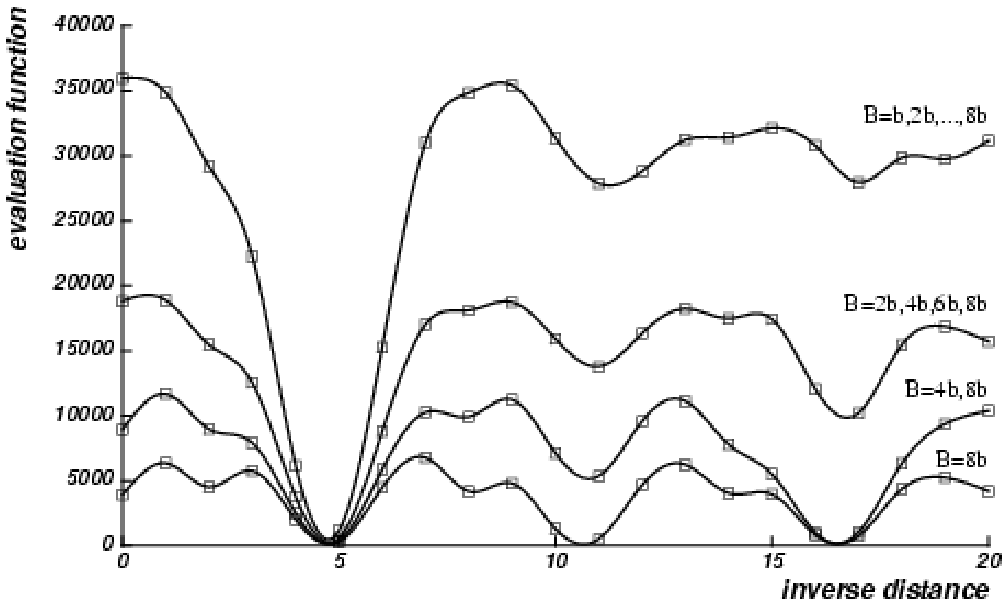


The third eye can be used for verification..

[Demo epipolar geometry]()

More Views (Okutami and Kanade, 1993)

New book: Ch7.6 p. 215: Pick a reference image, and slide the corresponding window along the corresponding epipolar lines of all other images, using inverse depth (Z^{-1}) relative to the first image as the search parameter.



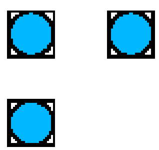
Reprinted from "A Multiple-Baseline Stereo System," by M. Okutami and T. Kanade, IEEE Trans. on Pattern Analysis and Machine Intelligence, 15(4):353-363 (1993). \copyright 1993 IEEE.

Use the sum of correlation scores to rank matches: SSD used as global evaluation function: Find Z^{-1} that minimizes SSD.

Multi-camera configurations

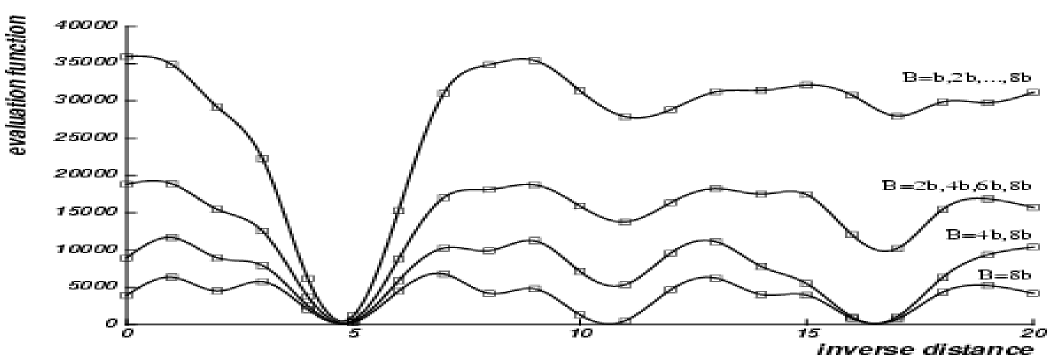
3 cameras give both robustness and precision

4 cameras give additional redundancy



3 cameras in a T arrangement allow the system to see vertical lines.

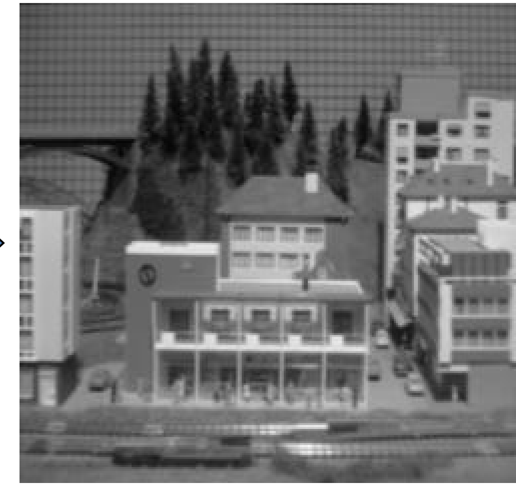
(illustration from Pascal Fua)



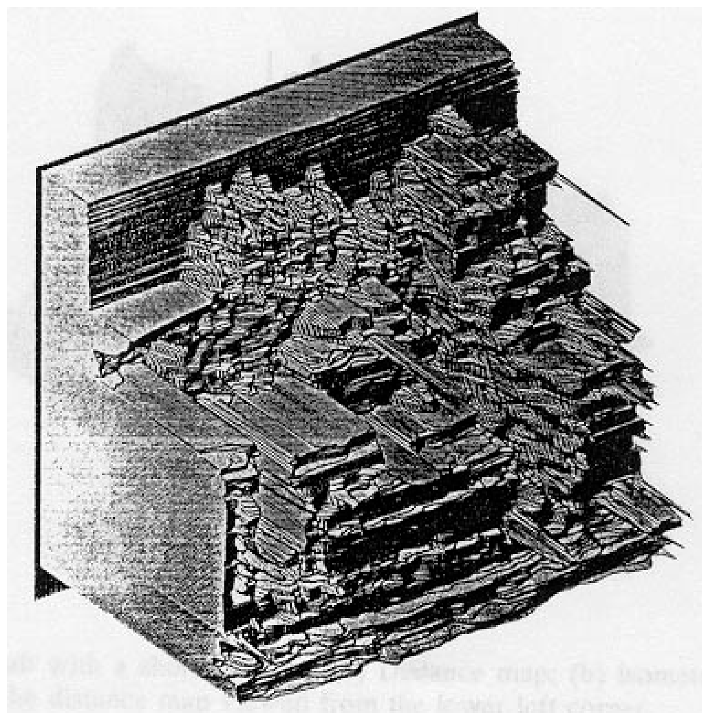
Okutami and Kanade

I1

I2



I10



Reprinted from “A Multiple-Baseline Stereo System,” by M. Okutami and T. Kanade, IEEE Trans. on Pattern Analysis and Machine Intelligence, 15(4):353-363 (1993). \copyright 1993 IEEE.

Normalized cross correlation

subtract mean: $A \leftarrow A - < A >, B \leftarrow B - < B >$

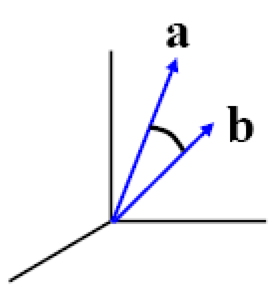
$$\mathrm{NCC} = \frac{\sum_i \sum_j A(i,j) B(i,j)}{\sqrt{\sum_i \sum_j A(i,j)^2} \sqrt{\sum_i \sum_j B(i,j)^2}}$$

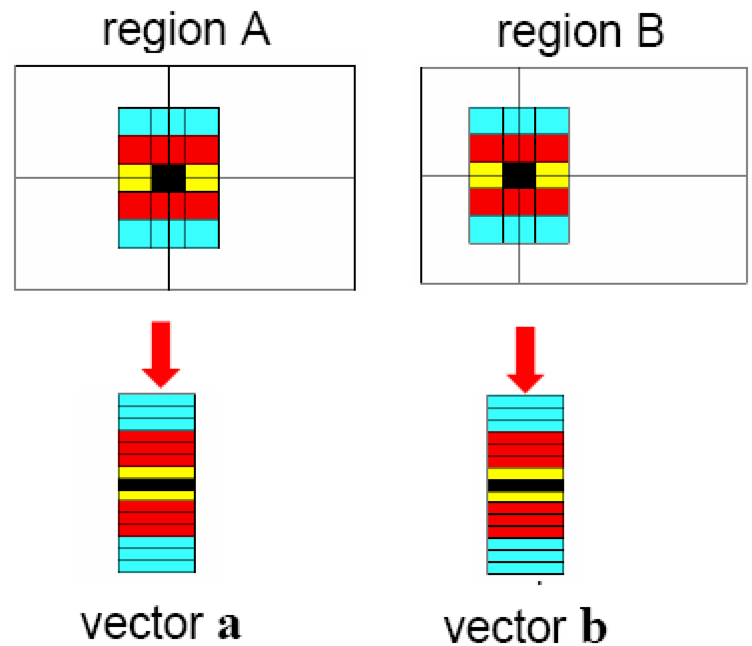
Write regions as vectors

$$\mathtt{A} \rightarrow \mathbf{a}, \;\; \mathtt{B} \rightarrow \mathbf{b}$$

$$\mathrm{NCC} = \frac{\mathbf{a}.\mathbf{b}}{|\mathbf{a}||\mathbf{b}|}$$

$$-1 \leq \mathrm{NCC} \leq 1$$





Source: Andrew Zisserman

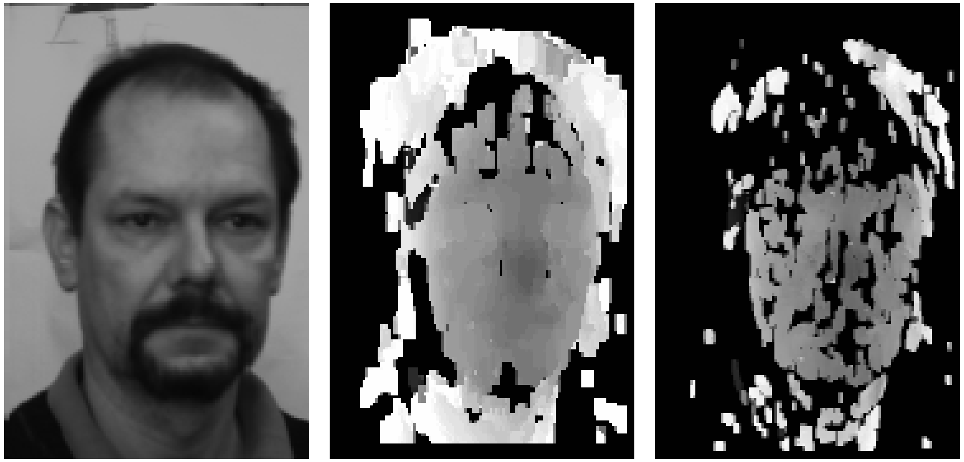
Aggregation window sizes

Small windows

- disparities similar
- more ambiguities
- accurate when correct

Large windows

- larger disp. variation
- more discriminant
- often more robust
- use shiftable windows to deal with discontinuities



14x14 7x7

(Illustration from Pascal Fua)