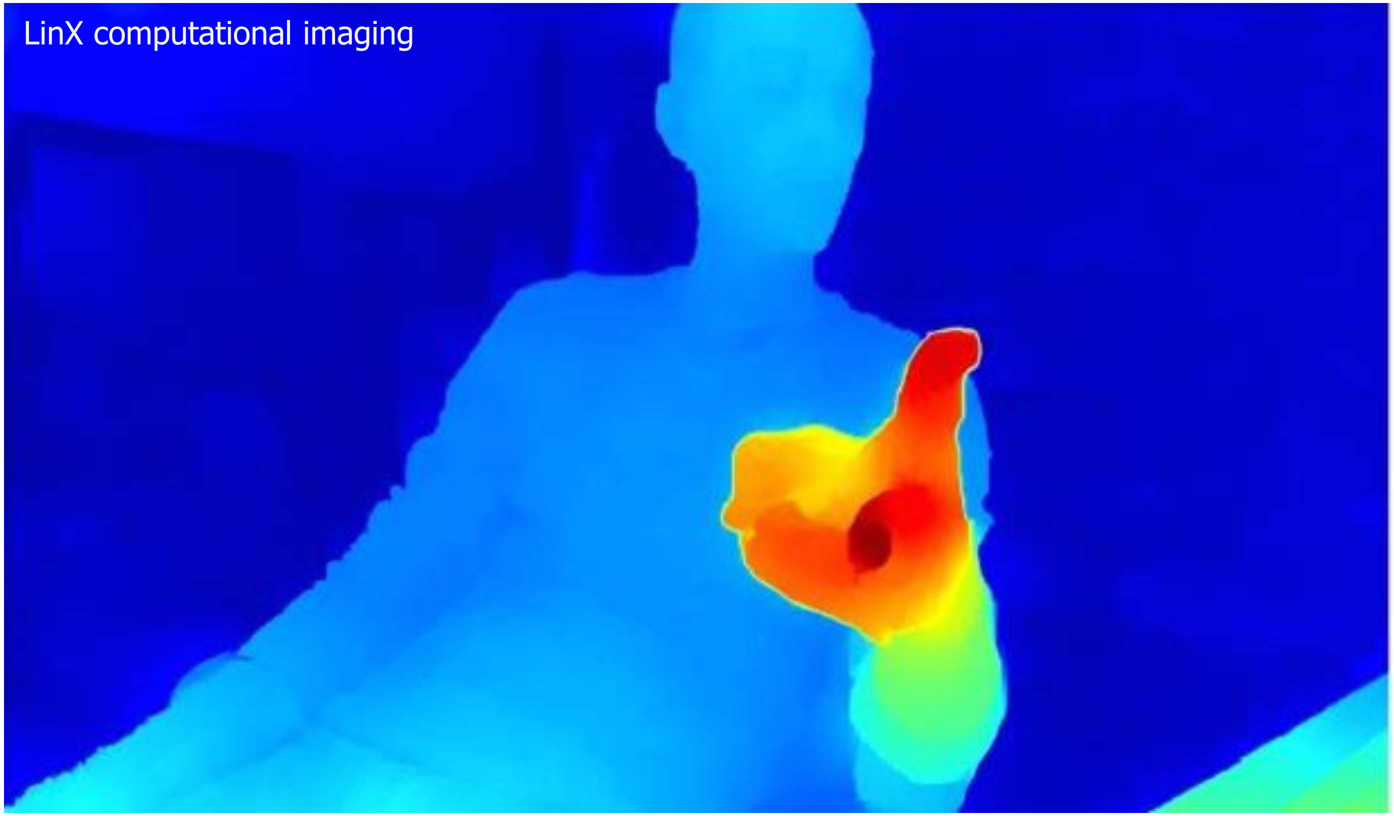
CS484/684 Computational Vision

Dense Stereo



Dense Stereo

towards **dense** 3D reconstruction

assumption: known camera motion, i.e. known epipolar lines

- (dense) stereo is an example of **dense correspondence**
- another example is dense motion estimation (*optical flow*)

But, **stereo is simpler** since the search for correspondences is restricted to 1D epipolar lines (versus 2D search for non-rigid motion)

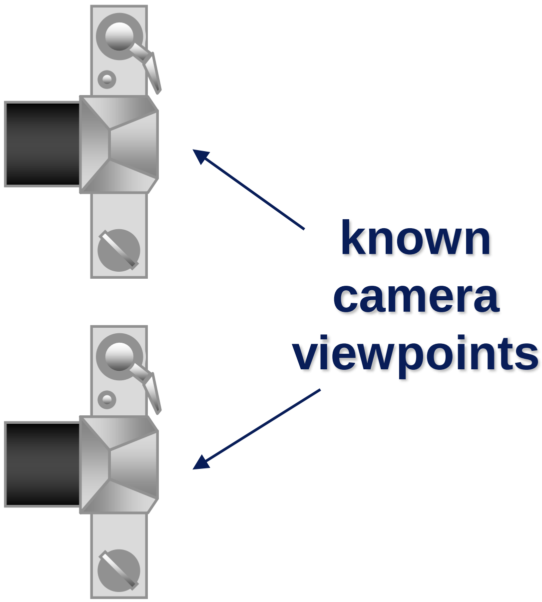
Dense Stereo Correspondence

- camera rectification for stereo pairs
- window-based (local) stereo
- scan-line stereo correspondence
  - optimization via DP, Viterbi, Dijkstra
- image-grid (global) stereo
  - optimization via graph cuts

examples of loss functions with spatial/geometric **regularization** of image labels

Szeliski, Chapter 11

Stereo vision



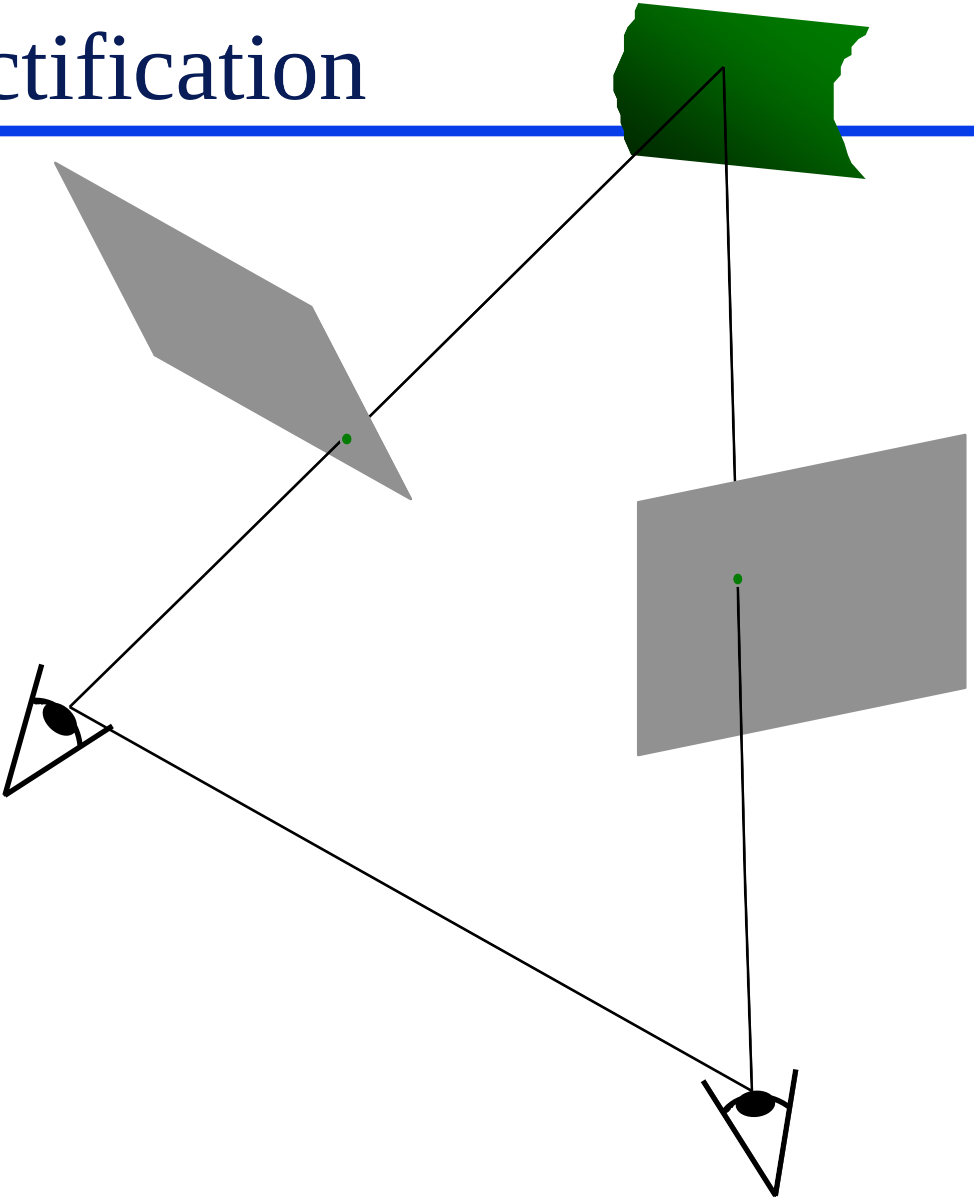
Two views of the same scene from slightly different point of view

Also called, narrow baseline stereo.

Motivation: - smaller difference in views allows to find **more matches** (Why?)

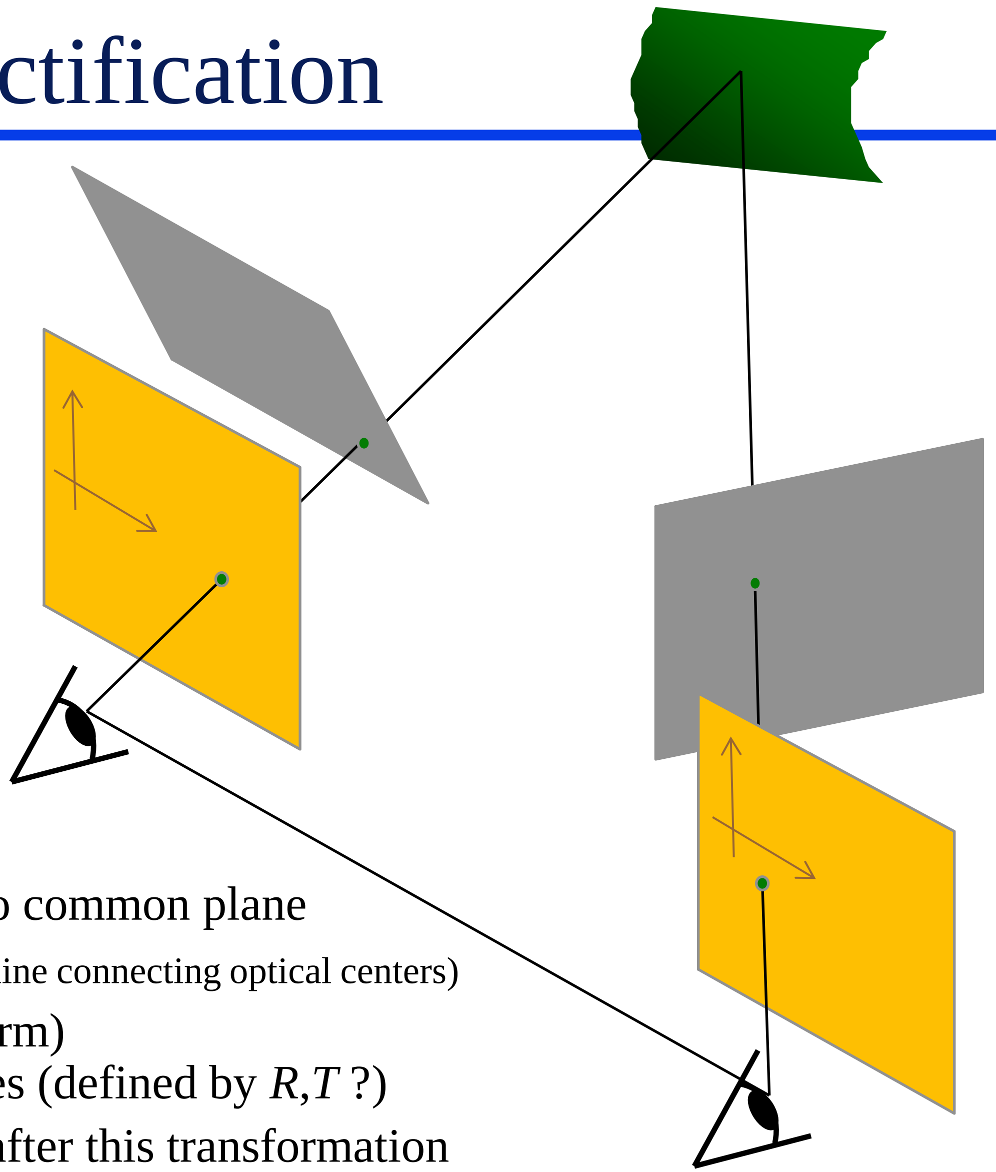
- scene reconstruction is simply represented via **depth map**

Stereo image rectification



Stereo image rectification

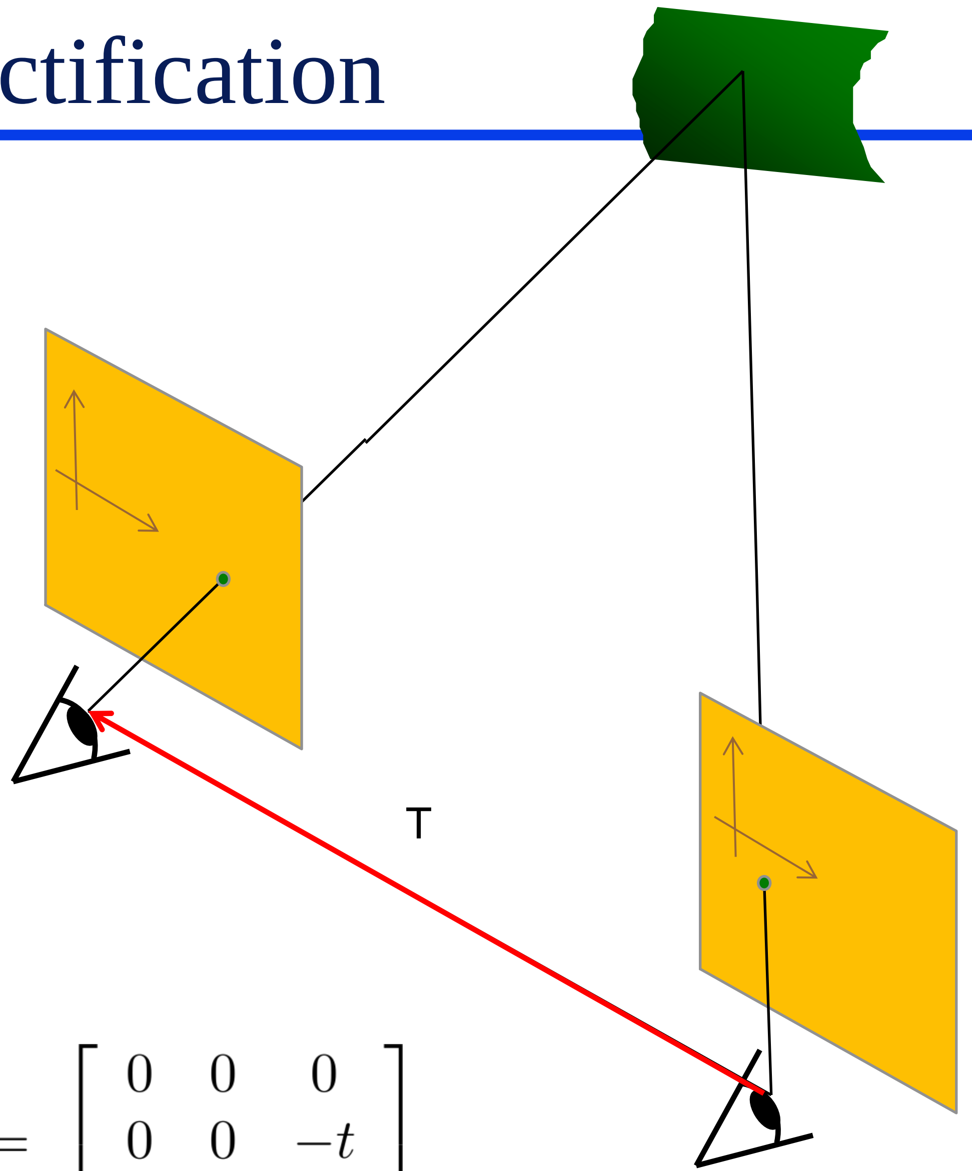
the corresponding (rectified) camera geometry is analogous to **"panning motion"**



- Image Reprojection
  - reproject image planes onto common plane parallel to the baseline (i.e. line connecting optical centers)
  - homographies (3x3 transform) applied to both input images (defined by R,T ?)
  - pixel motion is horizontal after this transformation
  - C. Loop and Z. Zhang. Computing Rectifying Homographies for Stereo Vision. IEEE Conf. Computer Vision and Pattern Recognition, 1999.

Stereo image rectification

the corresponding (rectified) camera geometry is analogous to **"panning motion"**



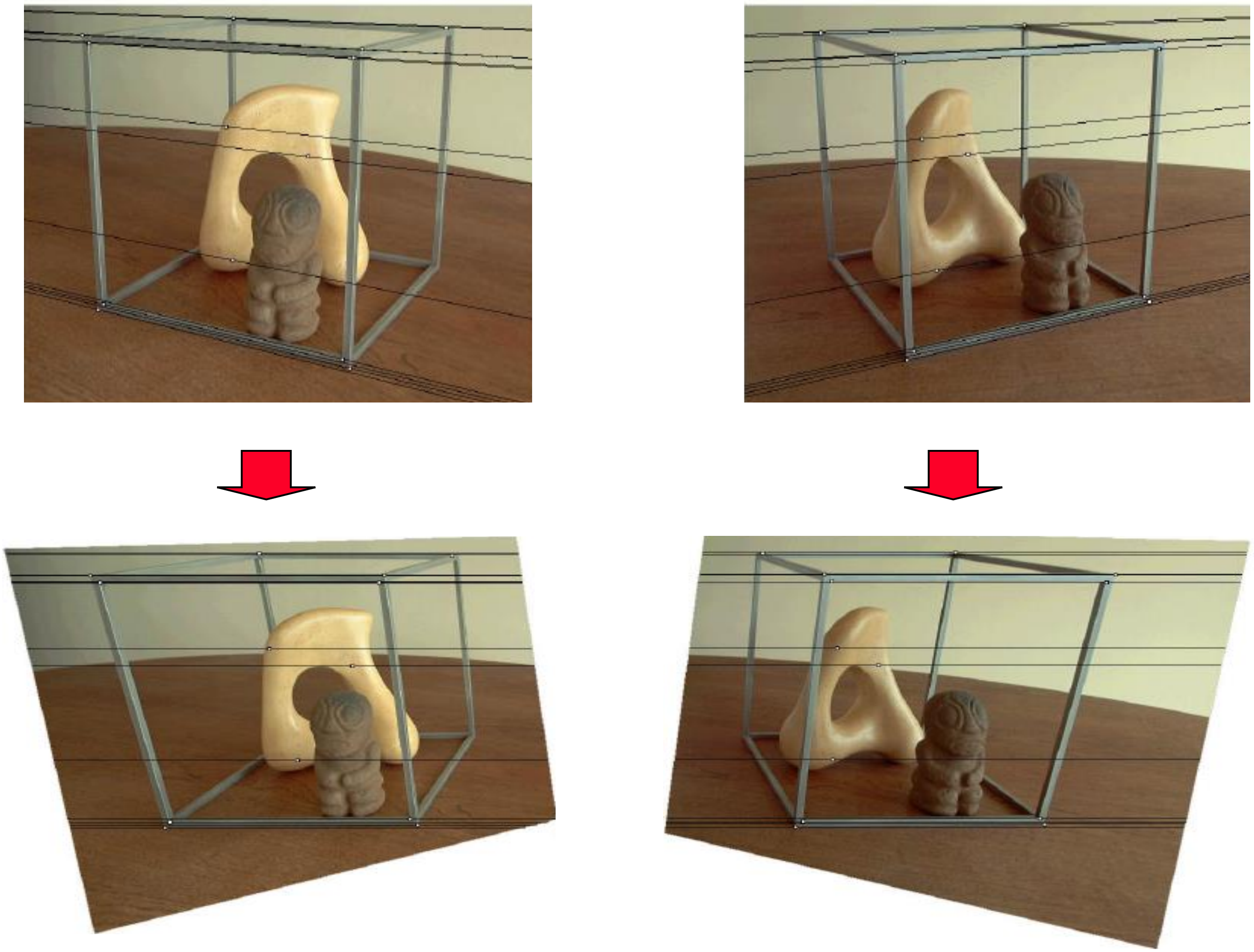
- Epipolar constraint:

$$R = I$$

$$T = \begin{bmatrix} t \\ 0 \\ 0 \end{bmatrix}$$

$$\Rightarrow E = [T]_\times R = [T]_\times = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -t \\ 0 & t & 0 \end{bmatrix}$$

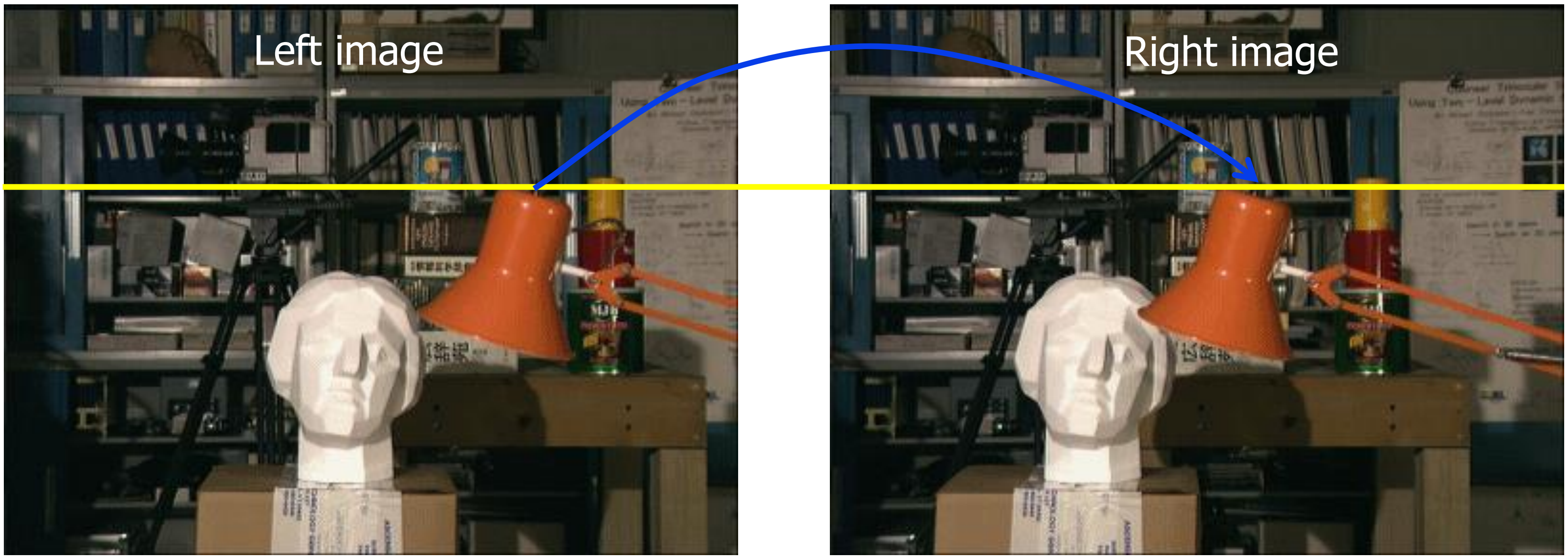
Stereo Rectification



Note projective distortion.
It will be much bigger
if images are taken from
very different viewpoints
(large baseline).

in this example the base line C_1C_2 is parallel to cube edges.

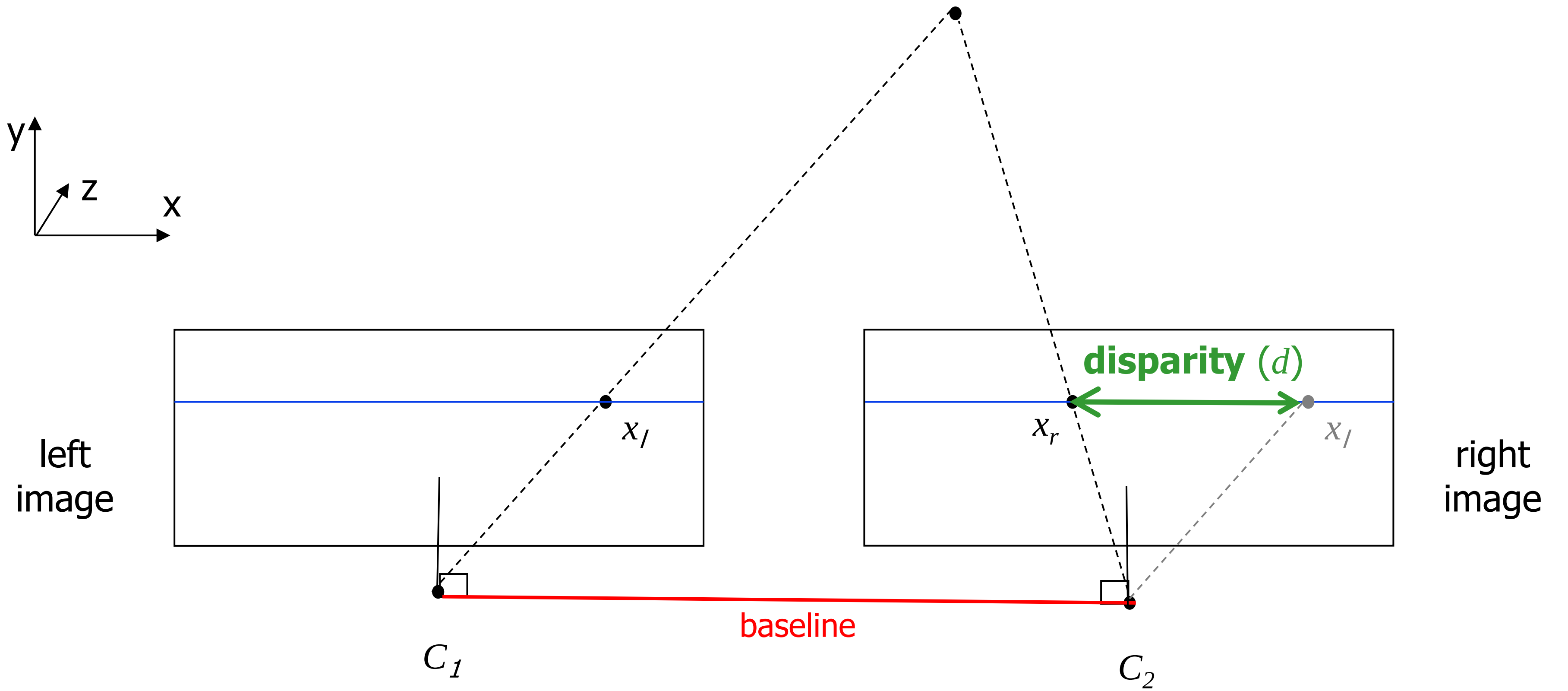
Stereo as a *correspondence* problem



(After rectification) all correspondences are along the same <u>horizontal scan lines</u>

(epipolar lines)

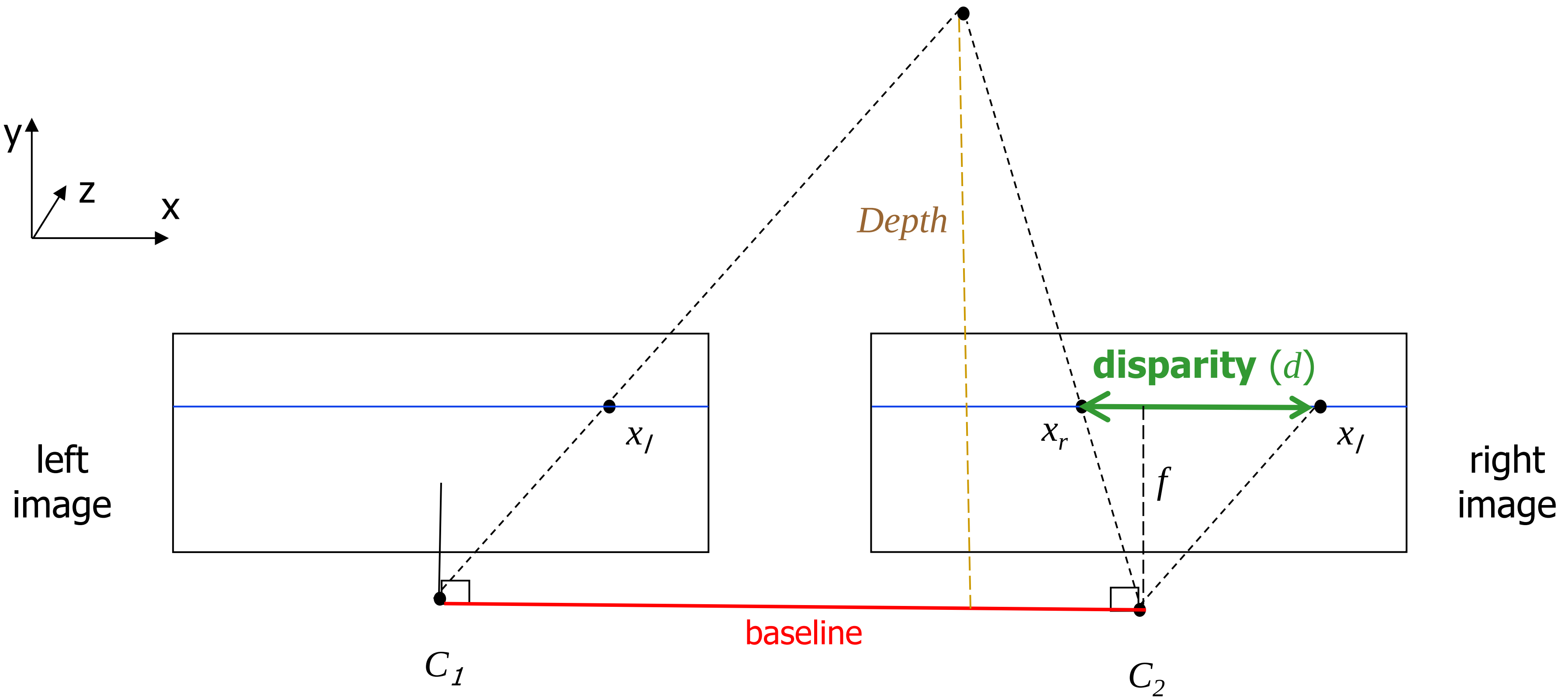
Rectified Cameras



epipolar lines are parallel to the x axis

difference between the x-coordinates of x_l and x_r is called the disparity

Rectified Cameras



$$\text{Depth} = |C_1C_2| \cdot f \,/\, d$$

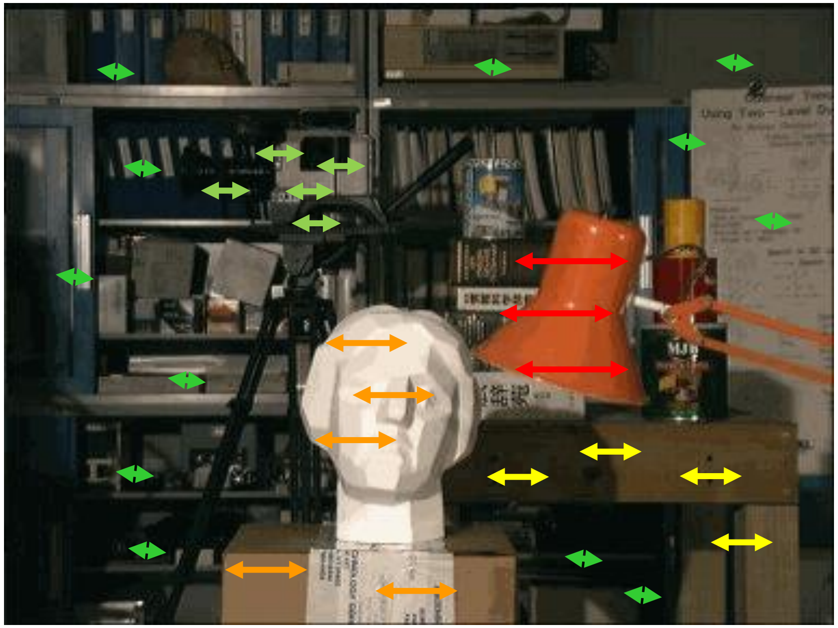
Stereo

Correspondences are described by shifts along horizontal scan lines (epipolar lines)

which can be represented by scalars (**disparities)**

Stereo

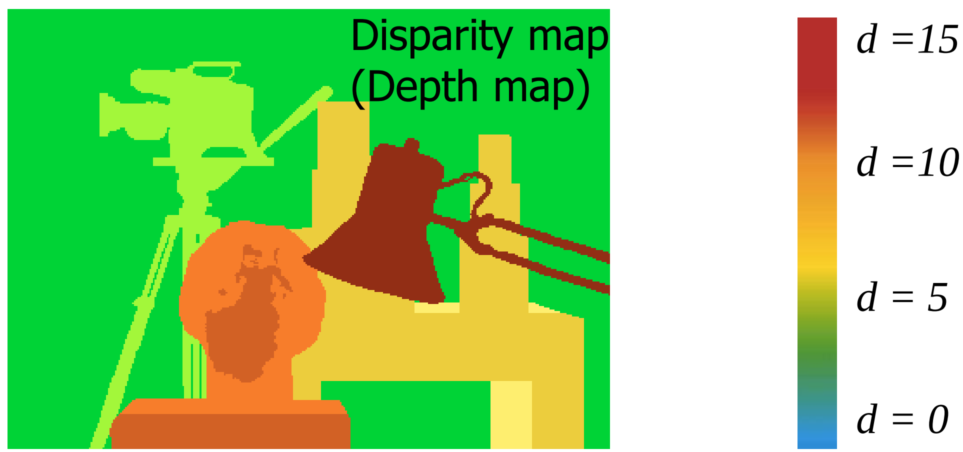
closer objects (smaller depths) correspond to **larger disparities**



Correspondences are described by shifts along horizontal scan lines (epipolar lines)

which can be represented by scalars (**disparities)**

Stereo

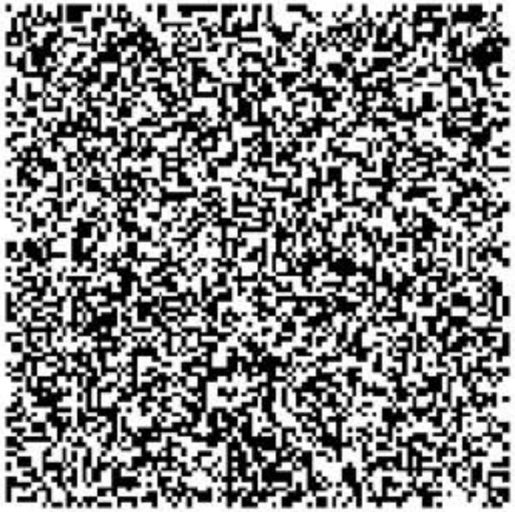


- If x-shifts (disparities) are known for all pixels in the left (or right) image then we can visualize them as a **disparity map** – scalar valued function $d(p)$
- larger disparities correspond to closer objects

Stereo Correspondence problem

- Human vision can solve it

  (even for "random dot" stereograms)



- Can computer vision solve it?

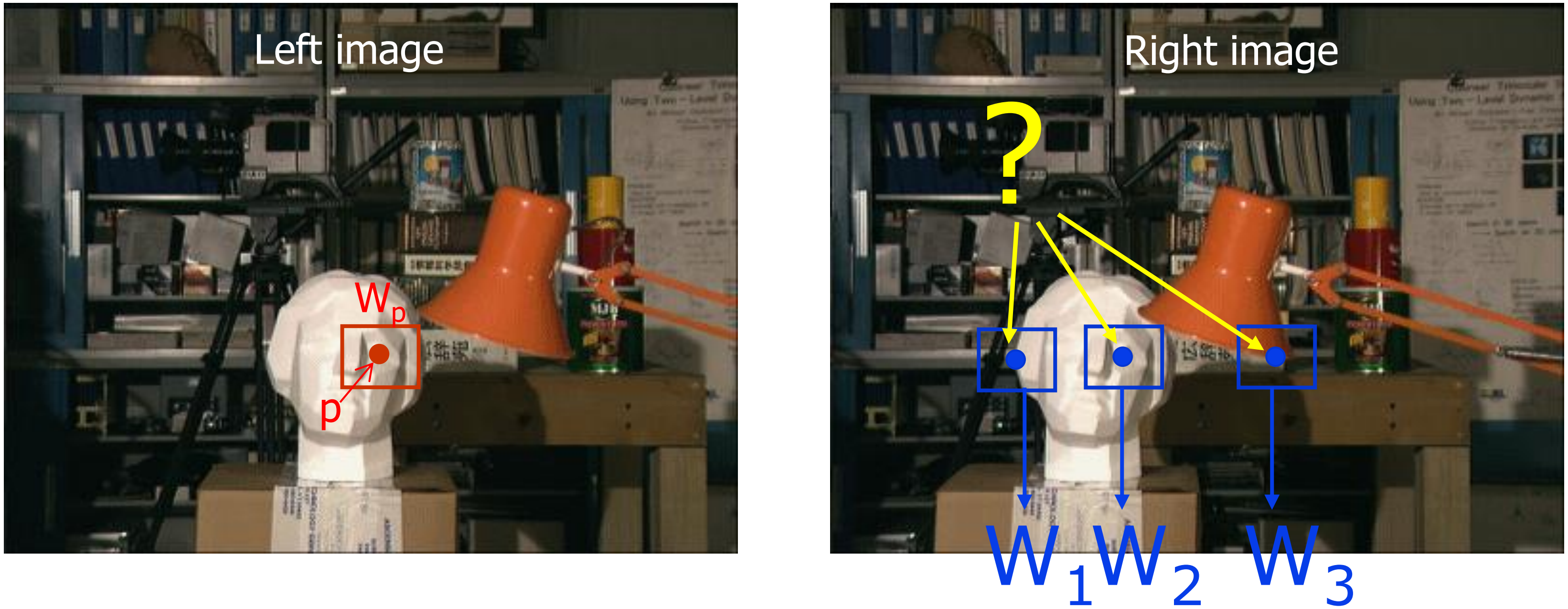
Maybe

see *Middlebury Stereo Database* for the state-of-the art results *http://cat.middlebury.edu/stereo/*

Stereo

- Window based
  - Matching rigid windows around each pixel
  - Each window is matched independently
- Scan-line based approach
  - Finding coherent correspondences for each scan-line
  - Scan-lines are independent
    - DP, *shortest paths*
- Global (muti-scan-line) approach
  - Finding coherent correspondences for all pixels (jointly)
    - spatial regularization over R^2 (grid), e.g. *graph cuts*

Stereo Correspondence problem
Window based approach

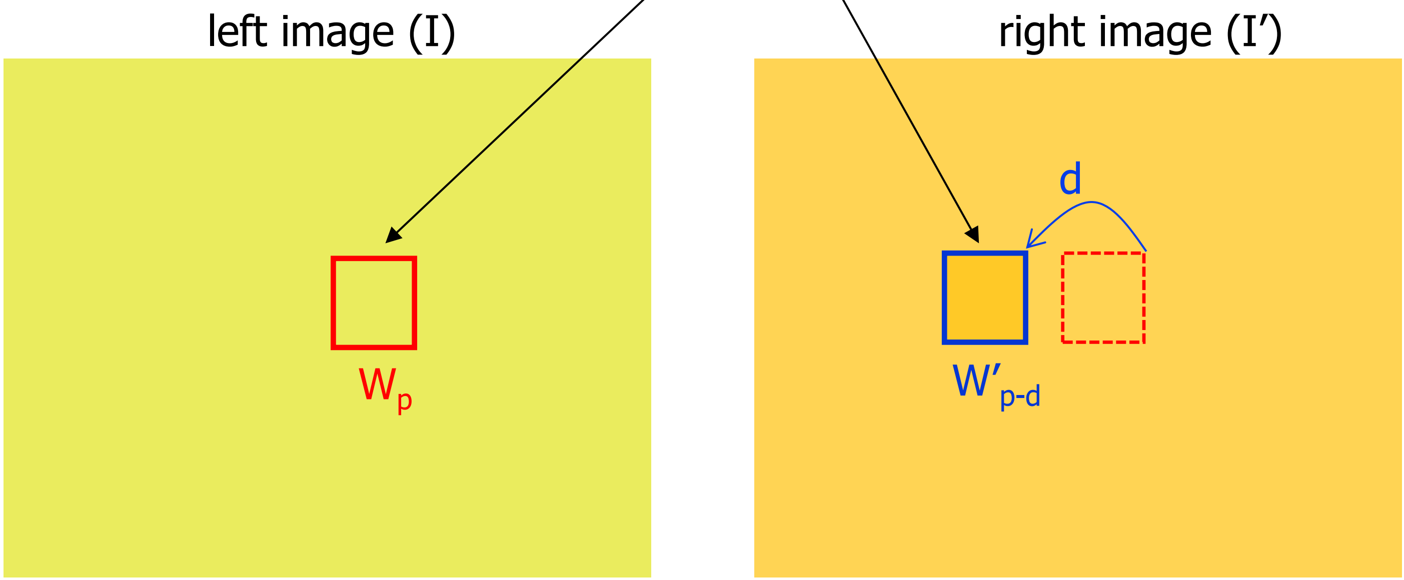


- For any given point p in left image consider window (or image patch) W_p around it
- Find **the best** matching window W_q on the same scan line in the right image that looks most similar to W_p

naïve implementation
has |I|*|d|*|W| operations

SSD (sum of squared differences) approach

computing SSD(p,d) $= \sum_{(x,y)\in W_p} (I(x,y) - I'(x-d,y))^2$

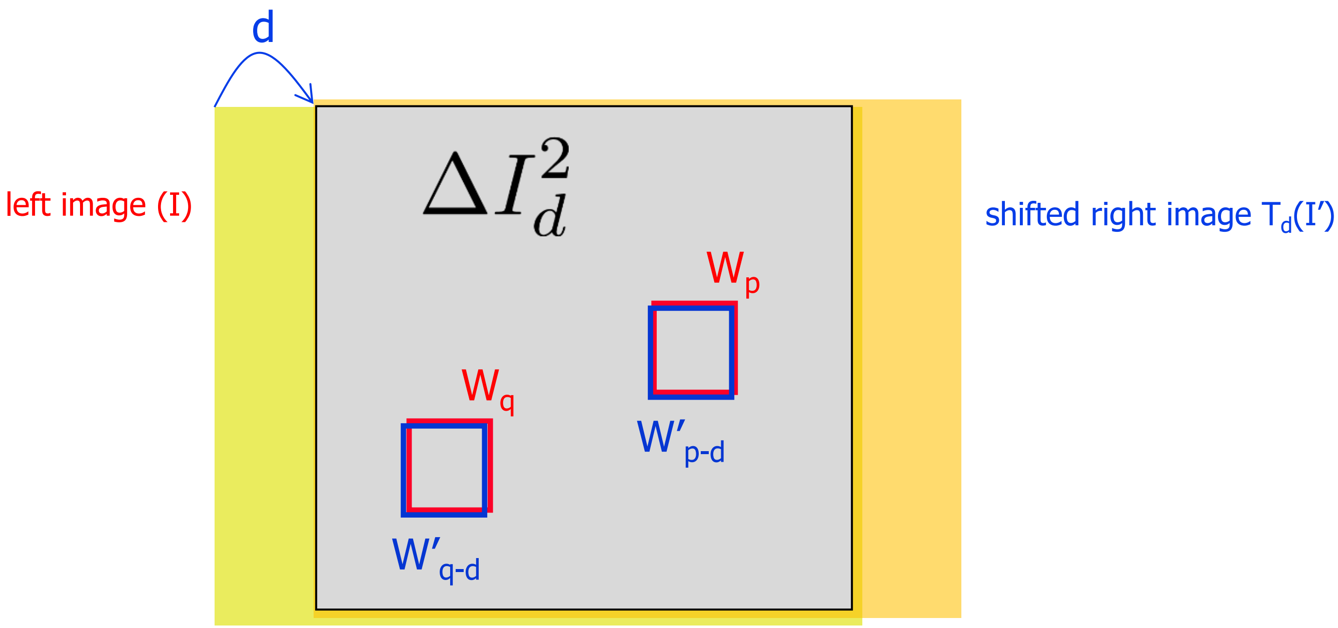


for any given pixel p compute SSD between windows W_p and W'_{p-d}
for all disparities d (in some interval [*min_d, max_d*])

the best disparity for p can be defined as $\hat{d}_p = \arg\min_d SSD(p,d)$

computing SSD efficiently

- For each fixed *d*, compute sq. differences image ΔI_d^2



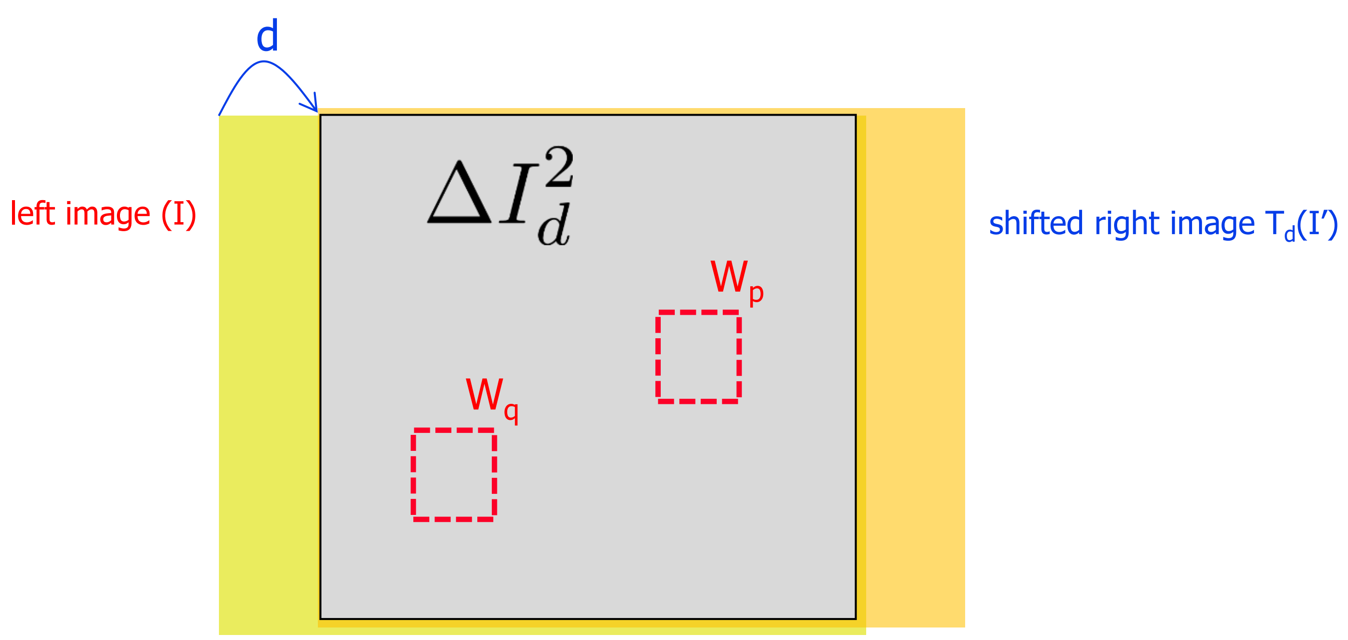
squared differences between the left image I and the shifted right image $T_d(I')$

$$\Delta I_d^2(x,y) \; := \; (I(x,y) - I'(x-d,y))^2$$

Then, SSD(p,d) between W_p and W'_{p-d} is $\sum_{(x,y)\in W_p} \Delta I_d^2(x,y)$

computing SSD efficiently

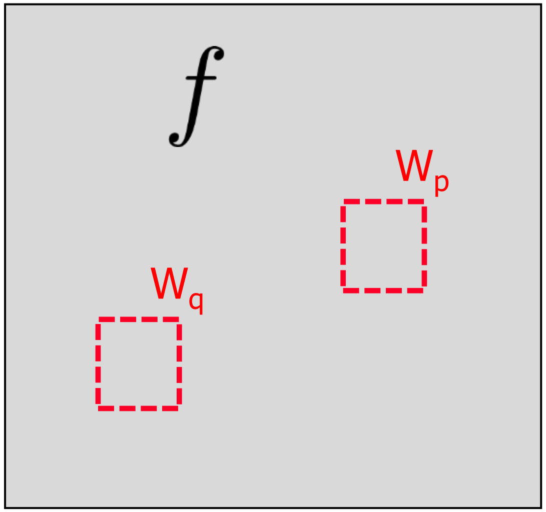
Need to sum pixel values at all possible windows



Then, SSD(p,d) between W_p and W'_{p-d} is $\sum_{(x,y)\in W_p} \Delta I_d^2(x,y)$

computing SSD efficiently

How to sum values at all possible rectangular windows □ in any given image f <u>efficiently</u> ?



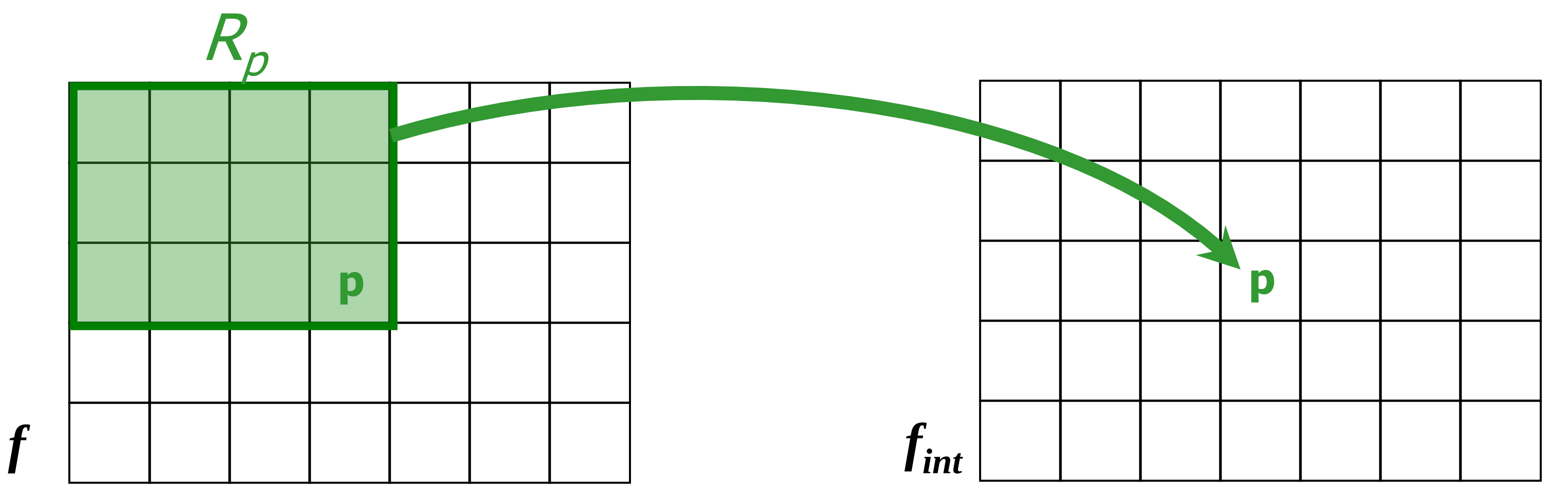
For SSD we just have special case $f(x,y) \equiv \Delta I_d^2(x,y)$

standard general trick:

"Integral Images"

$$f_{int}(p) := \sum_{q \in R_p} f(q)$$

- Define **integral image** $f_{int}(p)$ as the sum (integral) of image f over pixels in rectangle $R_p := \{q \mid \text{“}q \leq p\text{”}\}$



- Can compute $f_{int}(p)$ for all p in one or **two passes** over image f

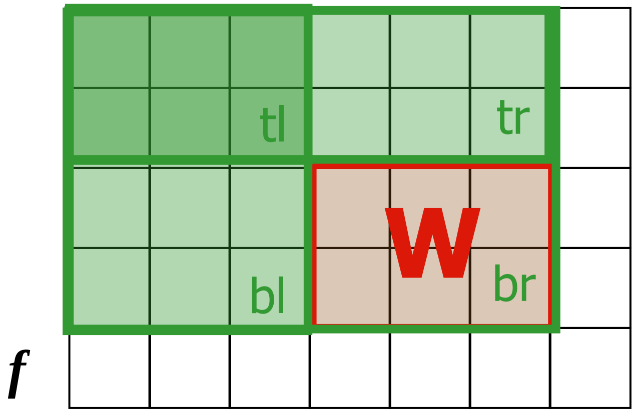
(How?)

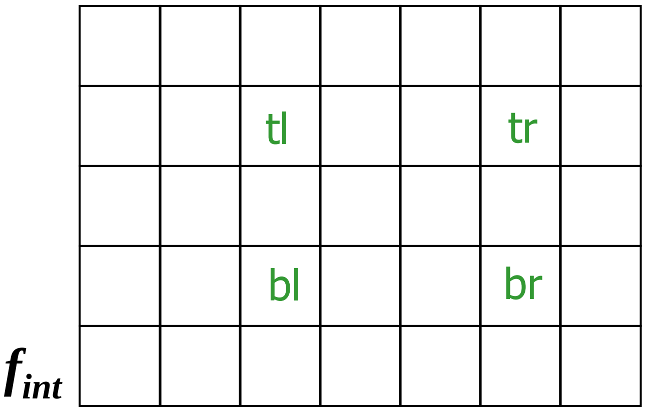
standard general trick:

"Integral Images"

$$f_{int}(p) := \sum_{q \in R_p} f(q)$$

- Define **integral image** $f_{int}(p)$ as the sum (integral) of image f over pixels in rectangle $R_p := \{q \mid \text{“}q \leq p\text{”}\}$

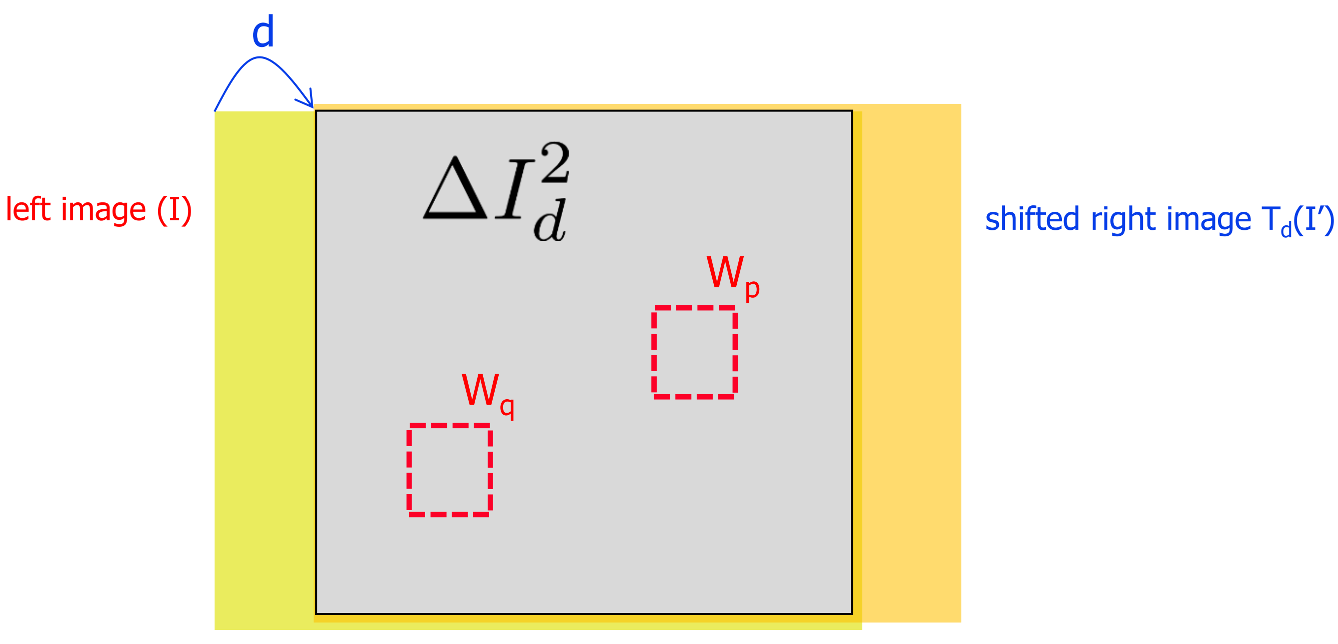




- Now, for any W the sum (integral) of f inside that window can be computed as $\sum_{q \in W} f(q) = f_{int}(\text{br}) - f_{int}(\text{bl}) - f_{int}(\text{tr}) + f_{int}(\text{tl})$

computing SSD efficiently

For SSD we have special case $f(x,y) \equiv \Delta I_d^2(x,y)$



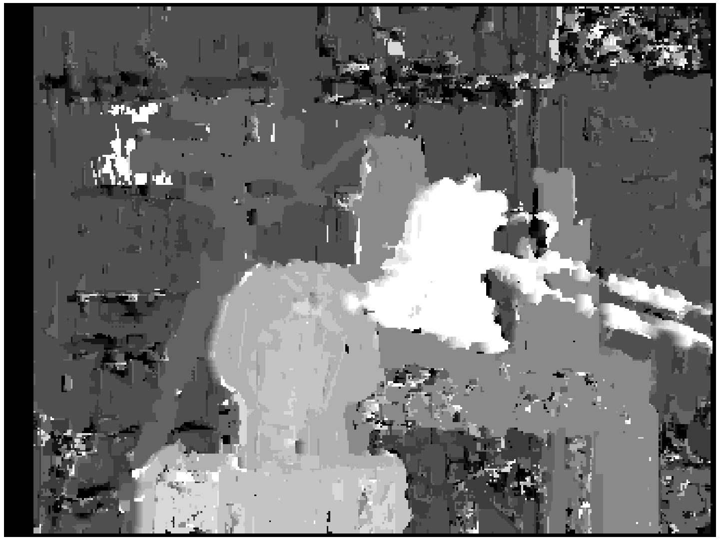
Now, the sum of ΔI_d^2 at any window ☐ takes 4 operations <u>independently of window size</u>

=> O(|I|*|d|) window-based stereo algorithm

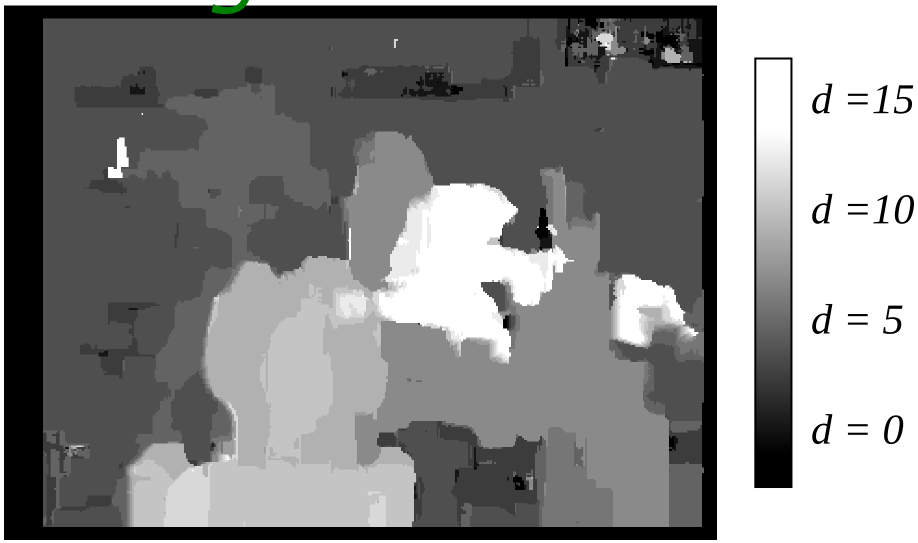
Problems with Fixed Windows

disparity maps $\hat{d}_p = \arg\min_d SSD(p,d)$ for:

small window



large window



- better at boundaries
- noisy in low texture areas

- better in low texture areas
- blurred boundaries

Q: what do we implicitly assume when using low SSD(d,p) at a window around pixel p as a criteria for "good" disparity d ?

window algorithms

- Maybe variable window size (pixel specific)?
  - What is the right window size?
  - Correspondences are still found <u>independently</u> at each pixel (no coherence)

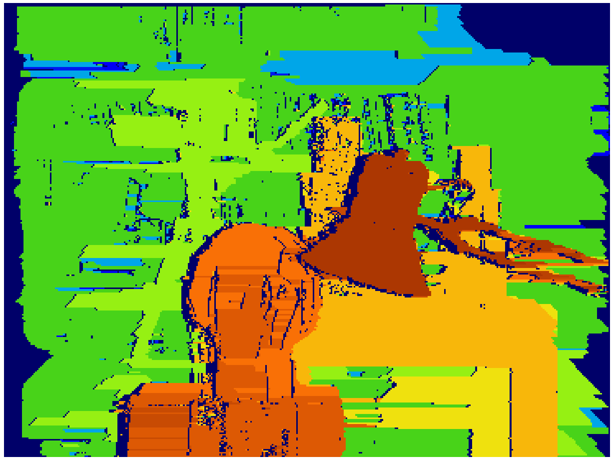
- All window-based solutions can be though of as "local" solutions - but very fast!

- How to go to "global" solutions?
  - use *objectives*, a.k.a. *energy* or *loss* functions
    - *surface regularization* or *spatial coherence*
  - optimization

need priors
to compensate
for local data ambiguity

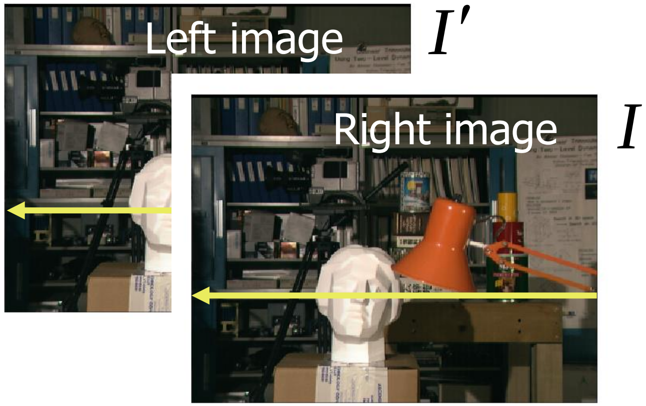
Stereo Correspondence problem
Scan-line approach

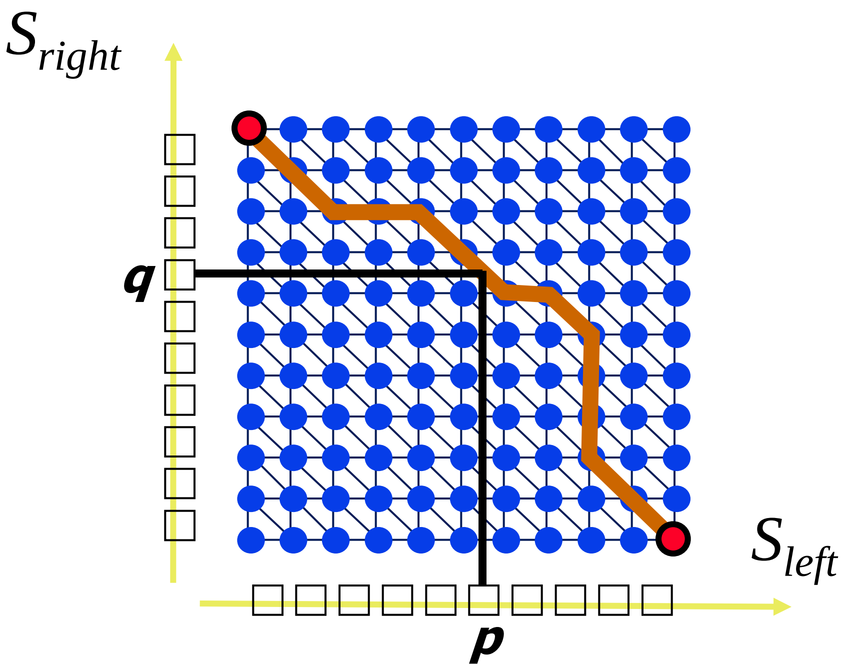
- Scan-line stereo
  - coherently match pixels in each scan line
  - DP or shortest paths work (easy 1D optimization)
  - Note: scan lines are still matched <u>independently</u>
    - – streaking artifacts



"Shortest paths" for Scan-line stereo

e.g. Ohta&Kanade'85, Cox at.al.'96

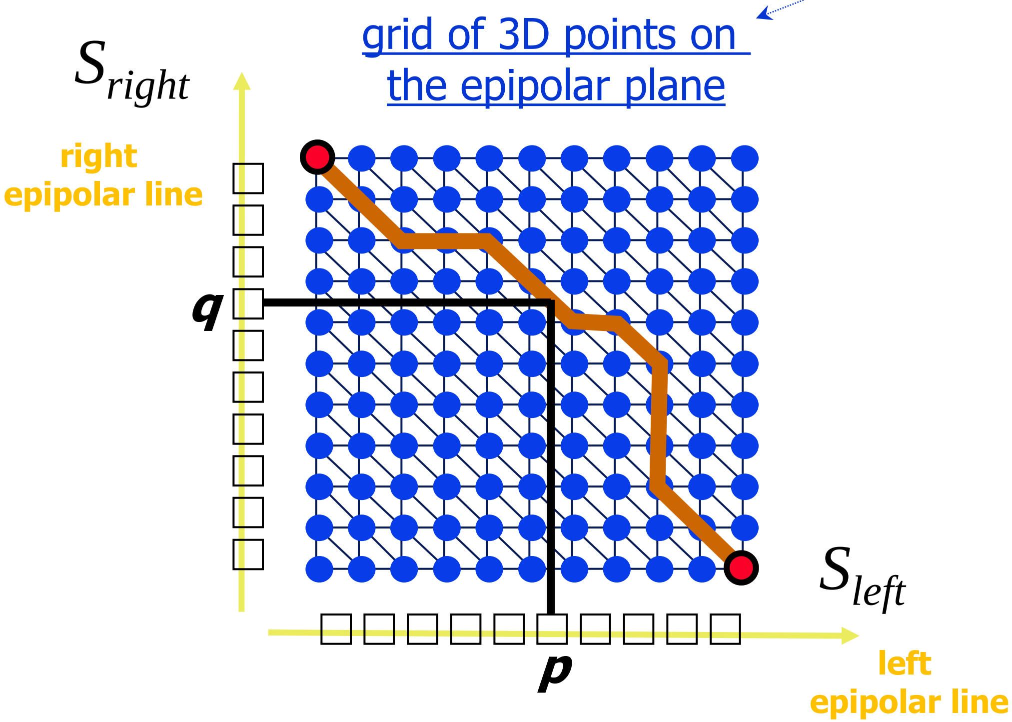


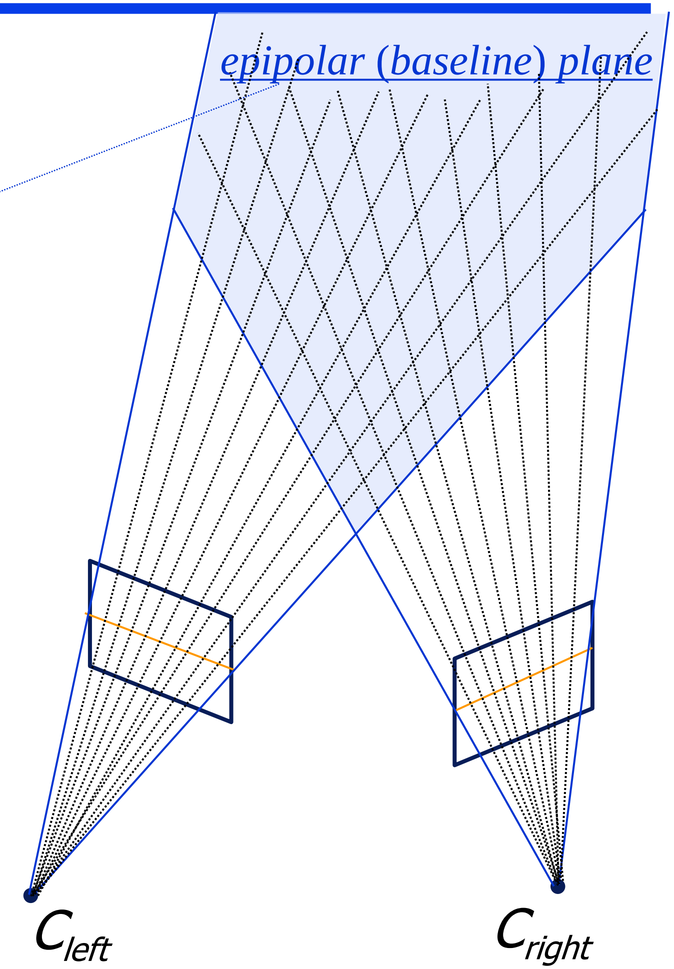


a **path** on this graph represents a matching function

"Shortest paths" for Scan-line stereo

3D interpretation:

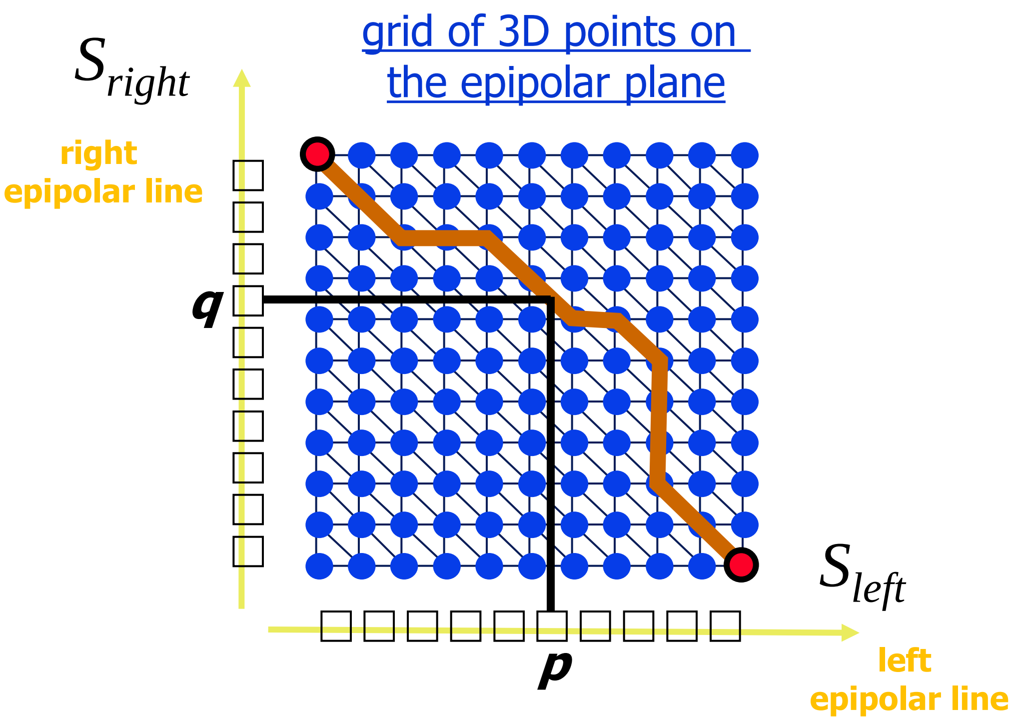




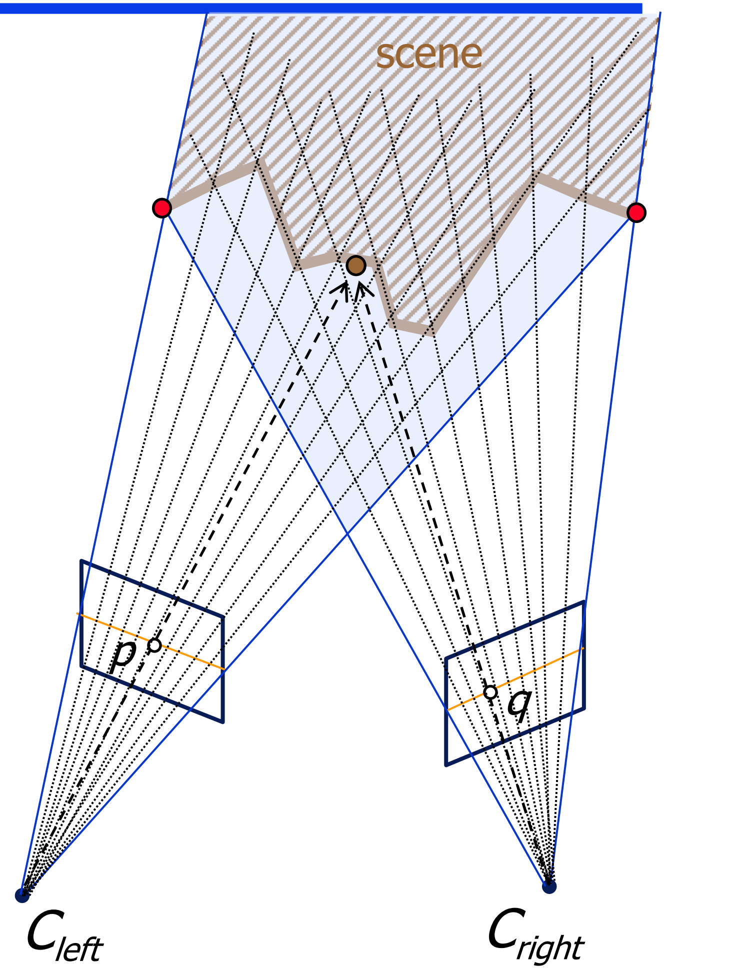
a **path** on this graph represents a matching function

“Shortest paths” for Scan-line stereo

3D interpretation:



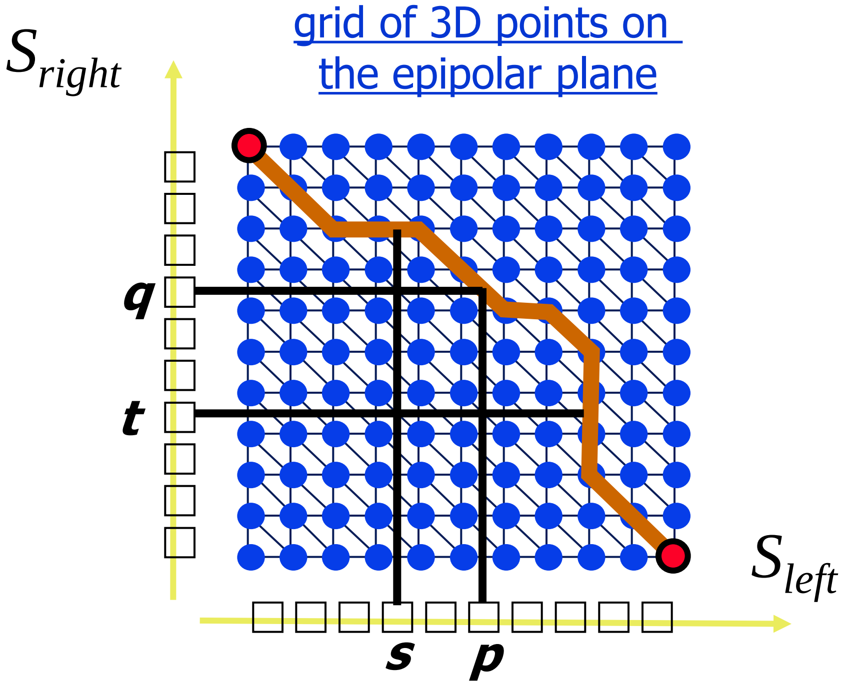
a **path** on this graph represents a matching function

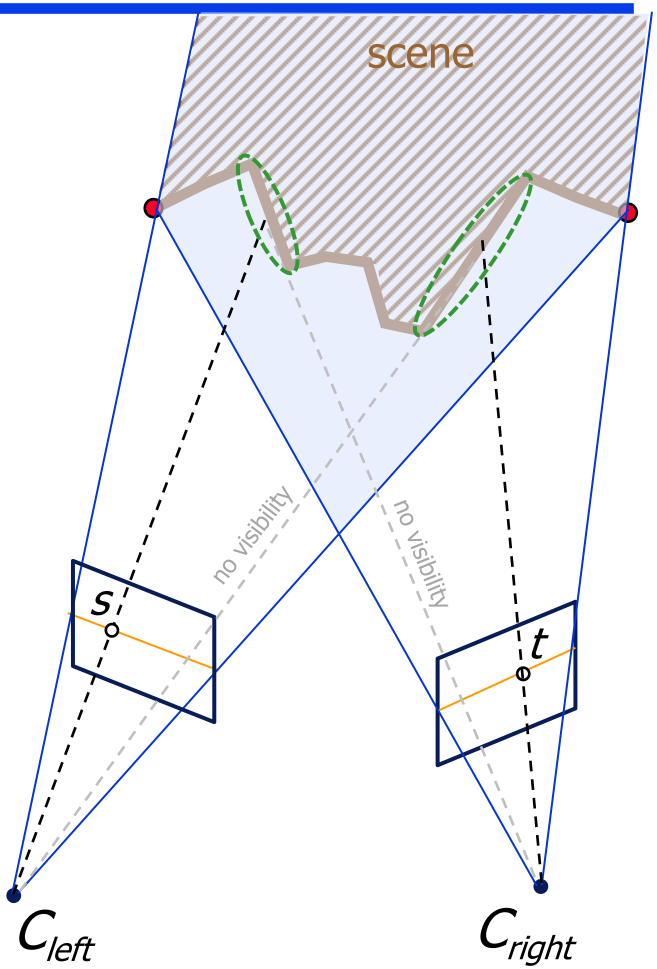


This path corresponds to an intersection of epipolar plane with 3D scene surface

"Shortest paths" for Scan-line stereo

3D interpretation:

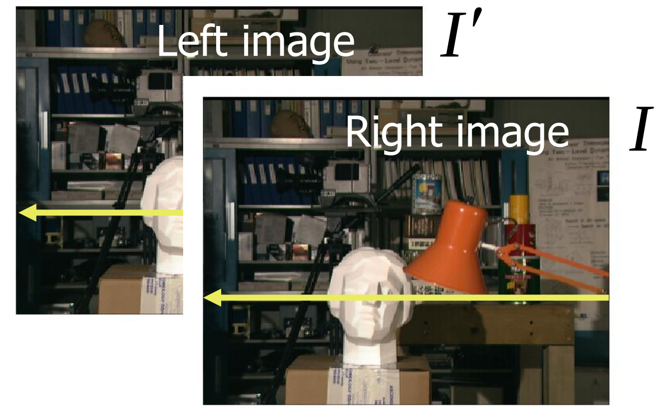


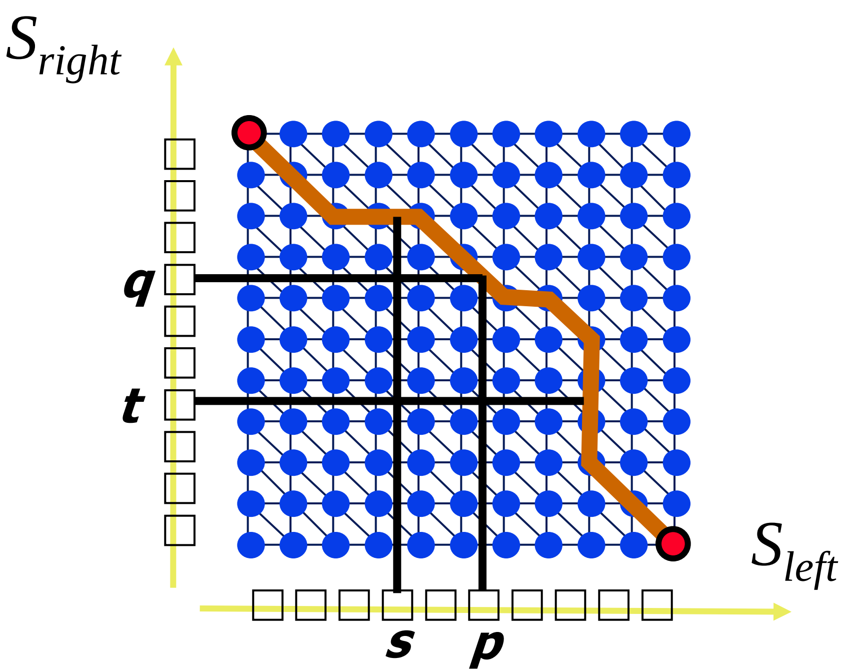


horizontal and vertical edges on the path imply "no correspondence" (*occlusion*)

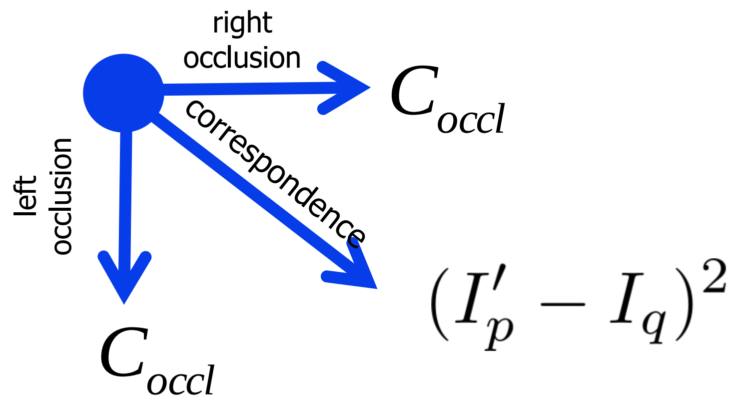
"Shortest paths" for Scan-line stereo

e.g. Ohta&Kanade'85, Cox at.al.'96



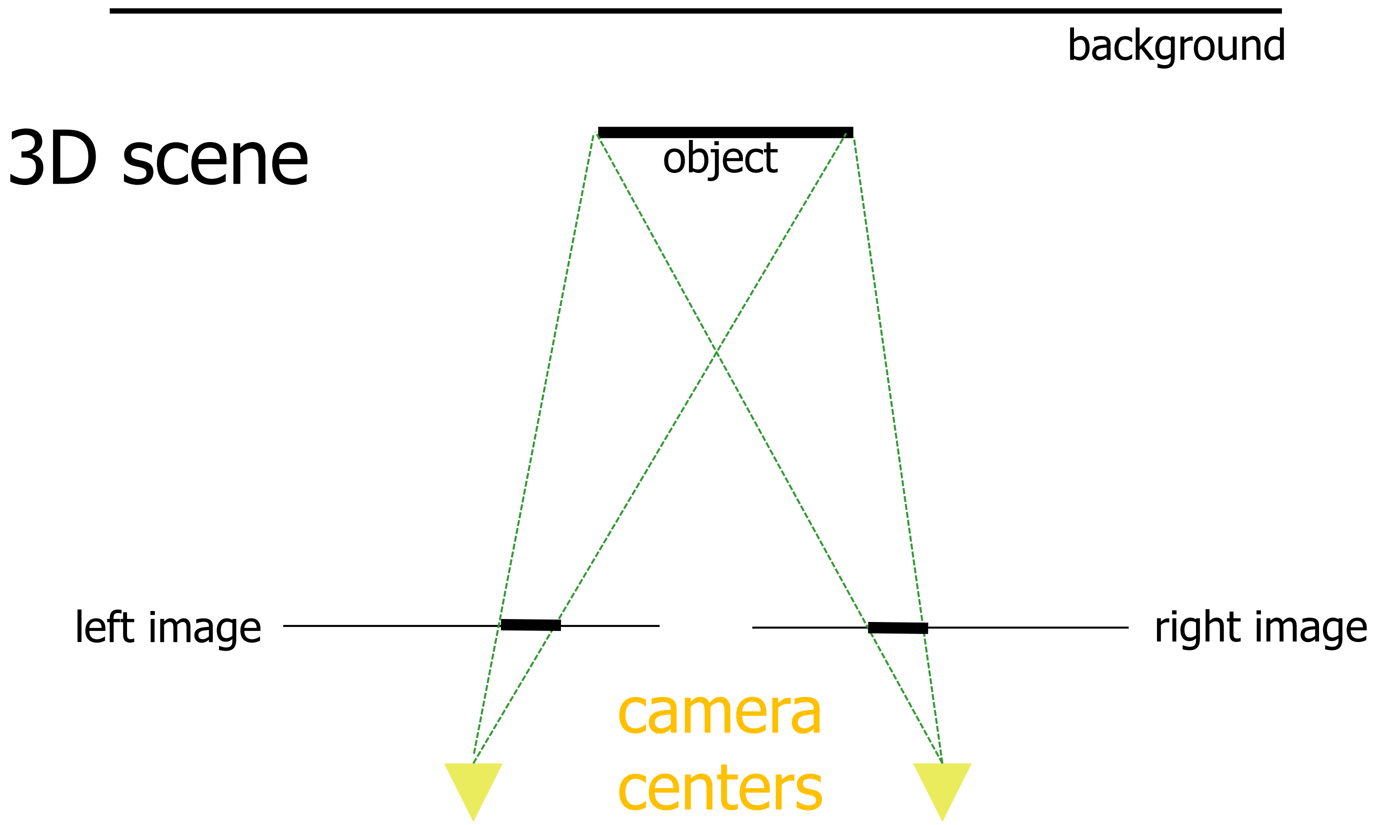


Edge weights:

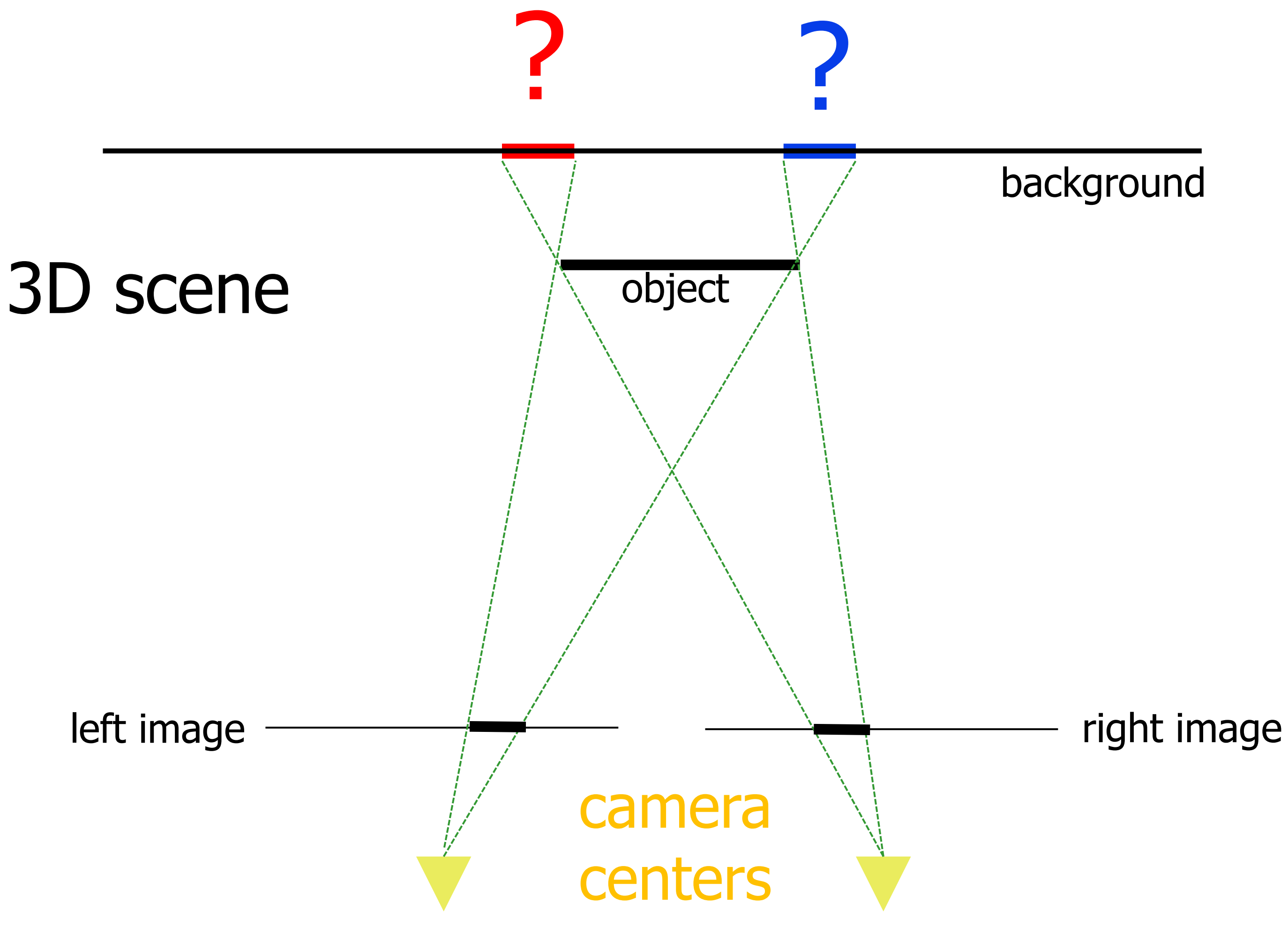


What is "occlusion" in general ?

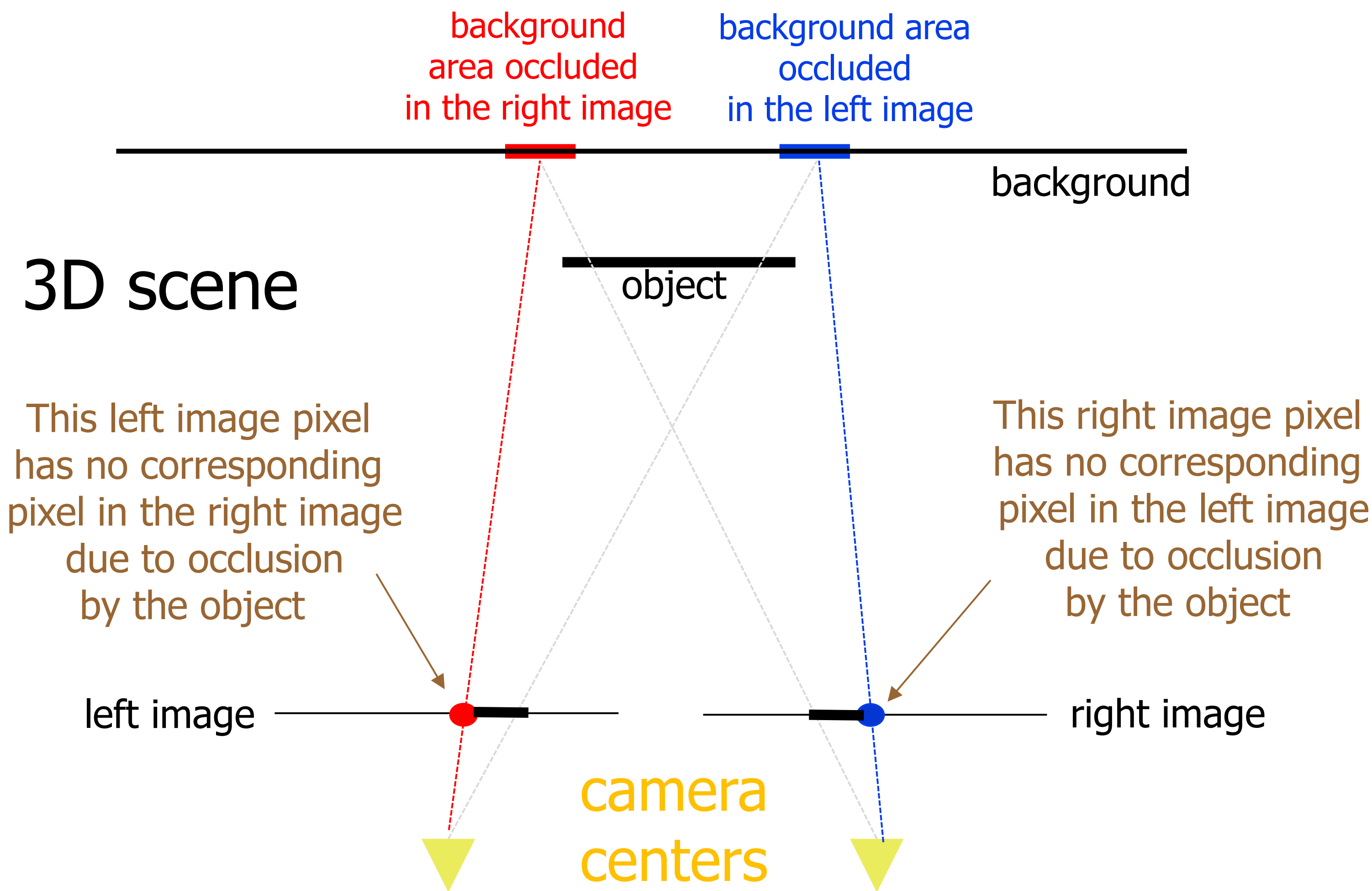
Occlusion in stereo



Occlusion in stereo

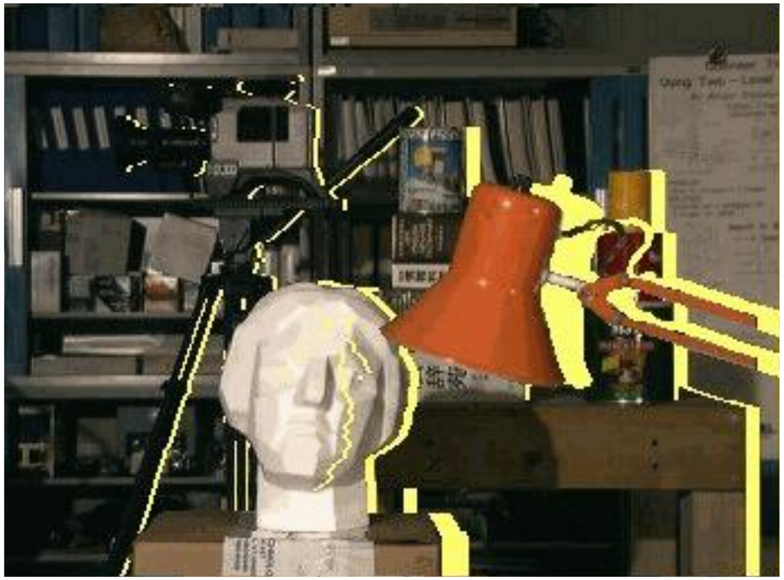


Occlusion in stereo



Note: **occlusions occur at depth discontinuities/jumps**

Stereo

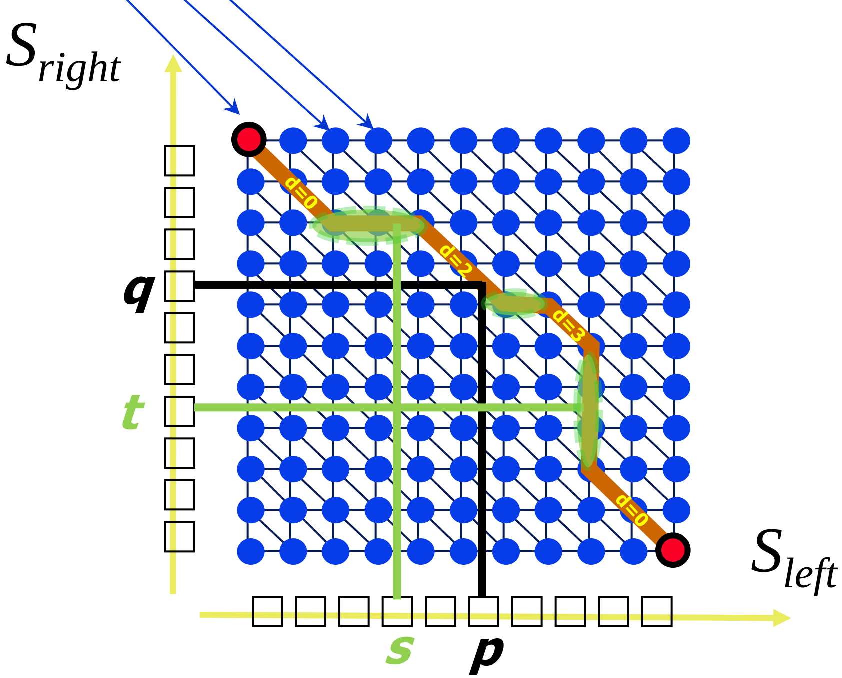


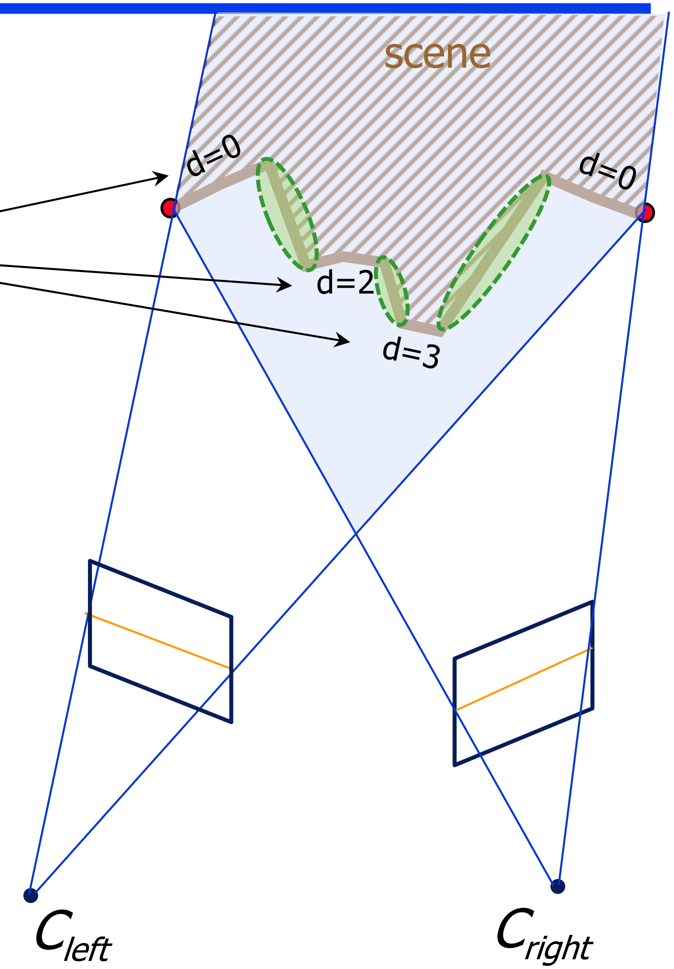
yellow marks occluded points in different viewpoints
(points not visible from the central/base viewpoint).

Note: **occlusions occur at depth discontinuities/jumps**

Occlusions vs disparity/depth jumps

NOTE: diagonal lines on this graph represent *disparity levels* (shifts between corresponding pixels) that can be seen as *depth layers*

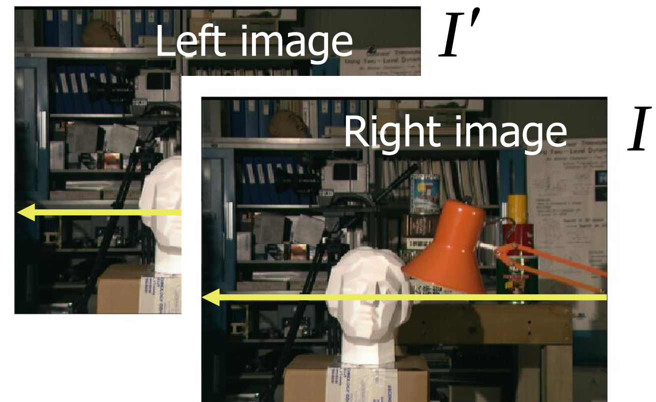


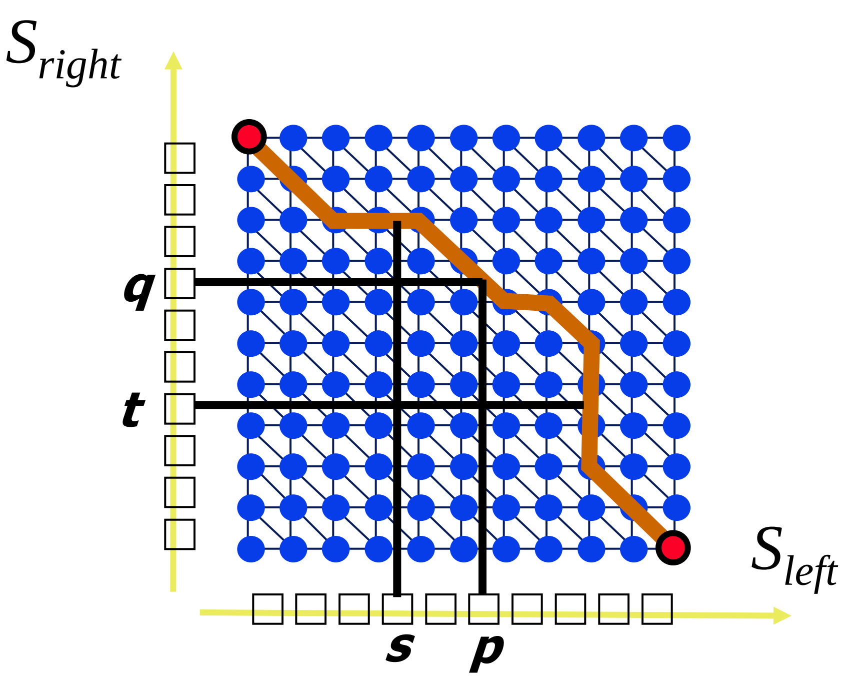


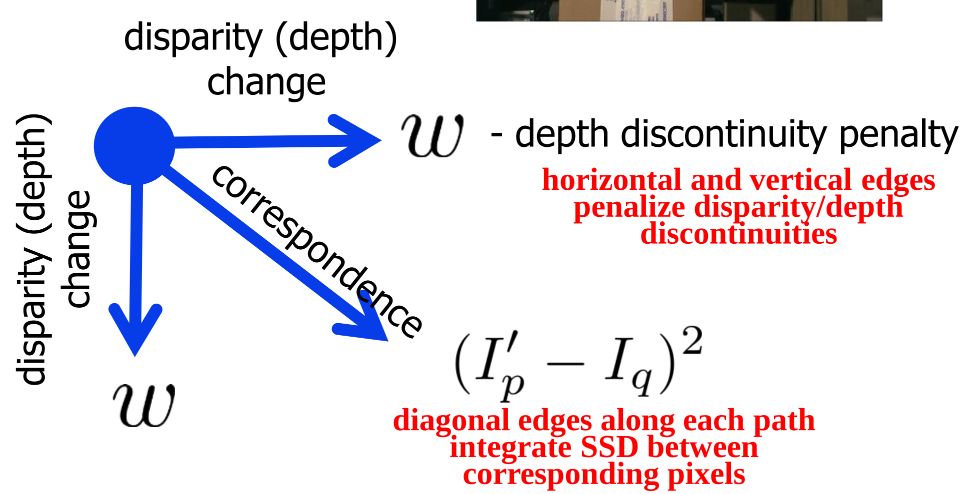
horizontal and vertical edges on this graph describe occlusions, as well as disparity jumps or depth discontinuities

Use Dijkstra to find *the shortest path* corresponding to certain edge costs

e.g. Ohta&Kanade'85, Cox at.al.'96



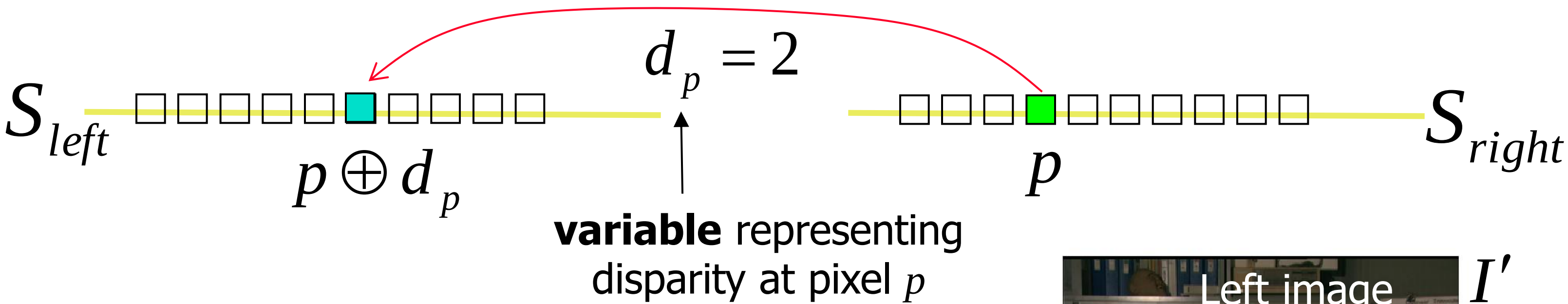


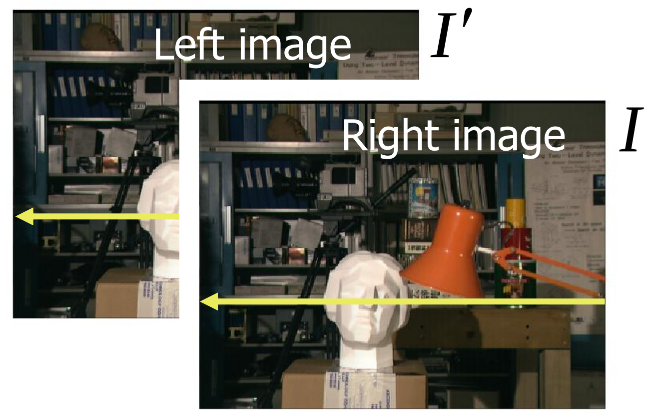


Each path implies certain depth/disparity configuration. Dijkstra can find the best one.

But, the actual implementation in OK'85 and C'96 uses *Viterbi* algorithm (DP) explicitly assigning "optimal" disparity labels d_p to all pixels p as follows…

More common representation of disparity map in stereo algorithms

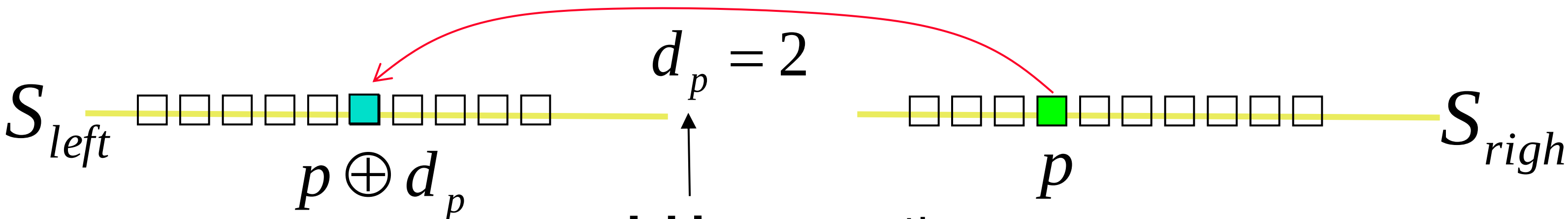




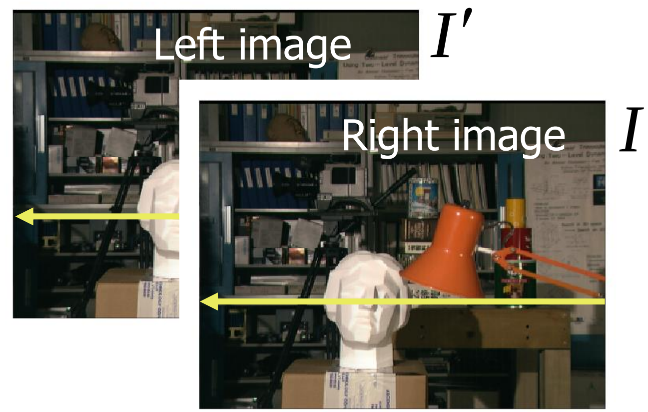
$$\mathbf{d} = \{d_p \mid p \in G\}$$

configuration (map) of disparities for pixels p on image grid

More common representation of disparity map in stereo algorithms



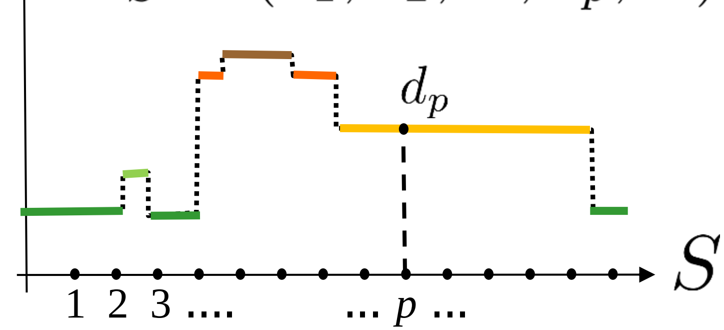
variable representing disparity at pixel p



$$\mathbf{d}_S = \{d_p \,|\, p \in S\}$$

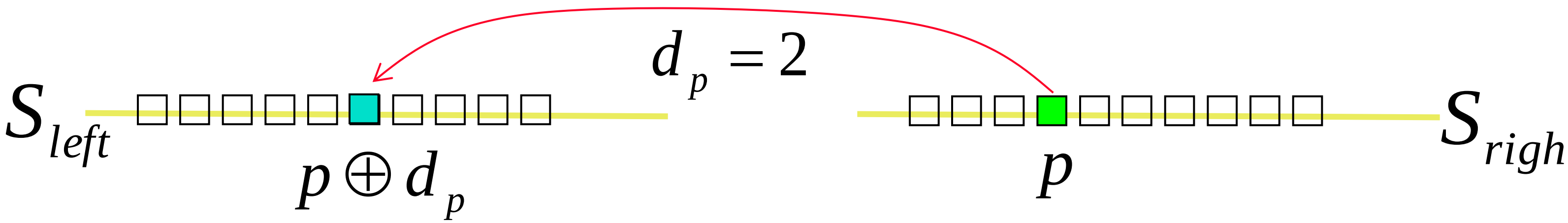
configuration (map) of disparities for pixels p on a scan line

$$\mathbf{d}_S = (d_1, d_2, ..., d_p, ...)$$

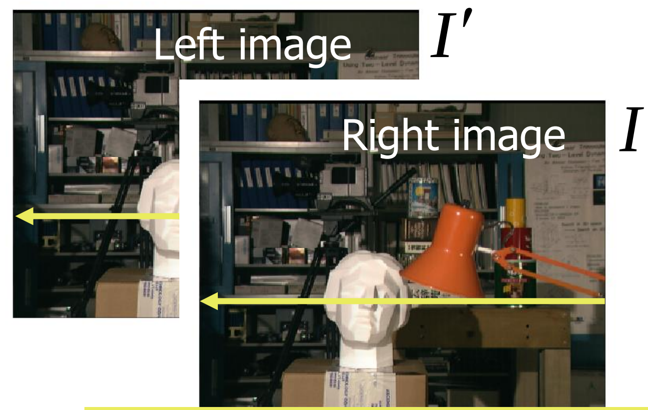


NOTE: *Loss*, ***Energy***, *Cost*, and *Objective* function mean the same thing and are used indiscriminately in different contexts

DP for scan-line stereo



Viterbi algorithm can find an optimal *disparity configuration* $\mathbf{d}_S = (d_1, d_2, ..., d_p, ...)$ for pixels p on a given scan-line S_{right} minimizing the following **loss function**



$$E(\mathbf{d}_S) = \sum_{p \in S} D_p(d_p) + \sum_{p \in S} V(d_p, d_{p+1}) = \sum_{\{p,q\} \in N} E(d_p, d_q)$$

$D_p(d_p) = |I_p - I'_{p \oplus d_p}|$ — photo consistency

$V(d_p, d_{p+1}) = w\,|d_p - d_{p+1}|$ — spatial coherence

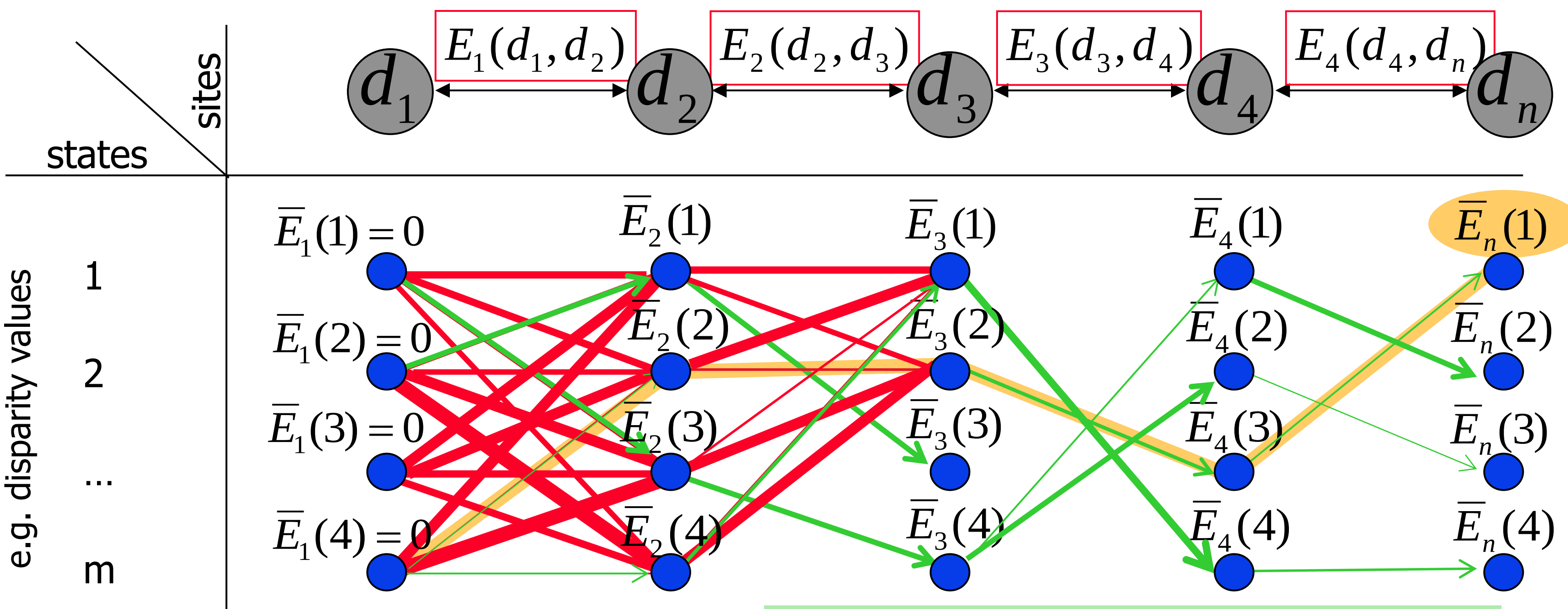
Such pairwise loss can be optimized in $O(nm^2)$ on non-loopy graphs (e.g., chains)

Dynamic Programming (DP)
Viterbi Algorithm

$d_i \in \{0, 1, ..., m-1\}$

Consider **pair-wise interactions** between sites (pixels) on a **chain** (scan-line)

$$E(d_1, ..., d_n) \;=\; E_1(d_1, d_2) + E_2(d_2, d_3) + \cdots + E_{n-1}(d_{n-1}, d_n)$$



$\bar{E}_p(k)$ - internal "loss counter" at "site" p and "state" k

$$\bar{E}_{p+1}(k) = \min_i \left(\bar{E}_p(i) + E_p(i,k)\right)$$

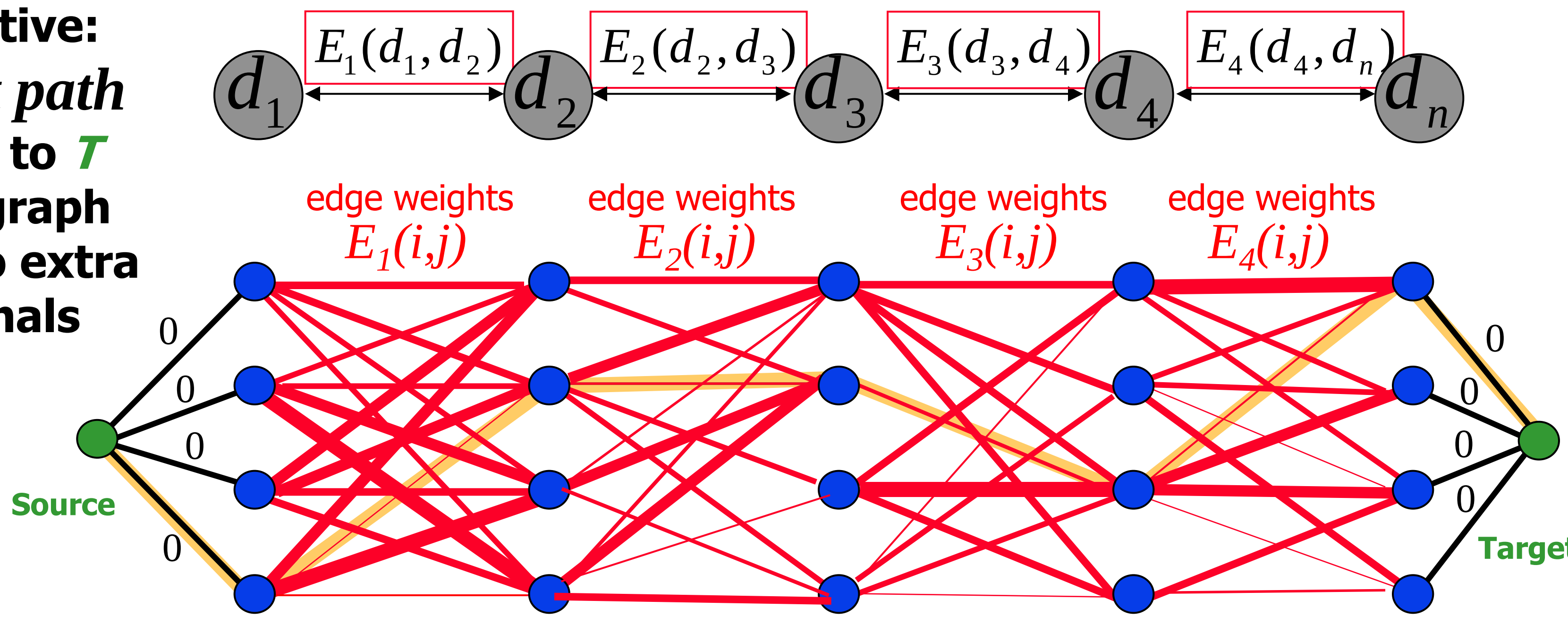
Complexity: $O(nm^2)$, worst case = best case

Dynamic Programming (DP)
Shortest paths Algorithm

Consider **pair-wise interactions** between sites (pixels) on a **chain** (scan-line)

$$E_1(d_1,d_2)+E_2(d_2,d_3)+\ldots+E_{n-1}(d_{n-1},d_n)$$

Alternative:
shortest path
from *S* to *T*
on the graph
with two extra
terminals



Complexity: $O(nm^2+nm\log(nm))$ - worst case

But, the best case could be better than Viterbi. Why?

Estimating (optimizing) disparities: over **points** vs. **scan-lines** vs. **grid**

Consider energy (loss) function over disparities

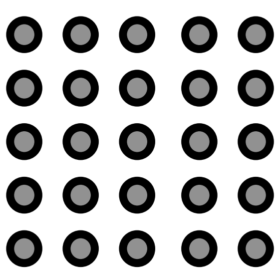
$\mathbf{d} = \{d_p \mid p \in G\}$ for pixels p on grid G

$$E(\mathbf{d}) = \sum_{p \in G} \underbrace{D_p(d_p)}_{= \, |I_p - I'_{p \oplus d_p}|} + \sum_{\{p,q\} \in N} \underbrace{V(d_p, d_q)}_{= \, w\,|d_p - d_q|}$$

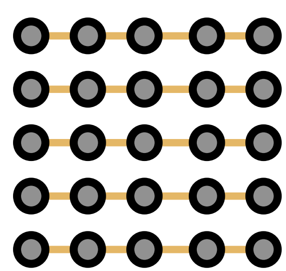
$|I_p - I'_{p \oplus d_p}|$ photo consistency

$w\,|d_p - d_q|$ spatial coherence

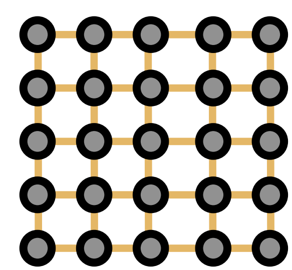
Consider three different neighborhood systems N:



$N = \varnothing$



$N = \{\{p, p\pm 1\} : p \in G\}$



$N = \{\{p,q\} \subset G : |pq| \leq 1\}$

Estimating (optimizing) disparities: over **points** vs. **scan-lines** vs. **grid**

Consider energy (loss) function over disparities

$\mathbf{d} = \{d_p \mid p \in G\}$ for pixels p on grid G

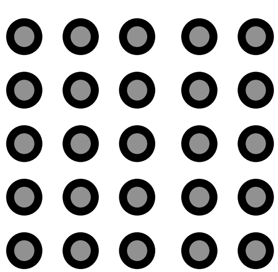
$$E(\mathbf{d}) = \sum_{p \in G} D_p(d_p) + \sum_{\{p,q\} \in N} V(d_p, d_q)$$

$D_p(d_p) = |I_p - I'_{p \oplus d_p}|$ photo consistency

$V(d_p, d_q) = w\,|d_p - d_q|$ spatial coherence

smoothness term disappears

CASE 1



$N = \varnothing$

Q: how to optimize $E(\mathbf{d})$ in this case?

$$\forall p \in G \quad \hat{d}_p = \arg\min_{d} D_p(d) \qquad O(nm)$$

Q: How does this relate to window-based stereo?

Estimating (optimizing) disparities: over **points** vs. **scan-lines** vs. **grid**

Consider energy (loss) function over disparities $\mathbf{d} = \{d_p \mid p \in G\}$ for pixels p on grid G

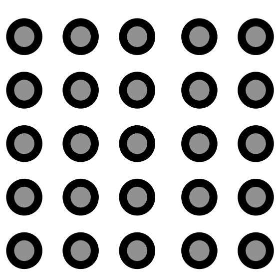
$$E(\mathbf{d}) = \sum_{p \in G} D_p(d_p) + \sum_{\{p,q\} \in N} V(d_p, d_q)$$

$$D_p(d_p) = |I_p - I'_{p \oplus d_p}| \quad \text{photo consistency}$$

$$V(d_p, d_q) = w\,|d_p - d_q| \quad \text{spatial coherence}$$

smoothness term disappears

CASE 1



$N = \varnothing$

Nodes/pixels do not interact (are independent).
Optimization of the sum of **unary terms**,
e.g. $\sum_{p \in G} D_p(d_p)$, is trivial: *O(nm)*

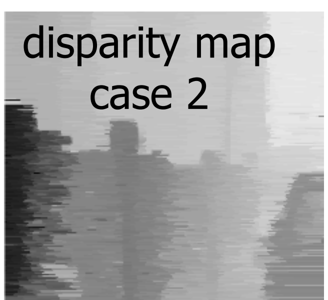
Estimating (optimizing) disparities: over **points** vs. **scan-lines** vs. **grid**

Consider energy (loss) function over disparities $\mathbf{d} = \{d_p \mid p \in G\}$ for pixels p on grid G

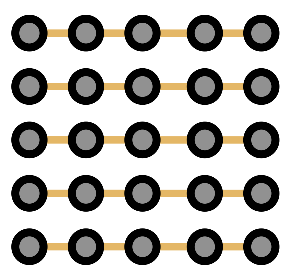
$$E(\mathbf{d}) = \sum_{p \in G} D_p(d_p) + \sum_{\{p,q\} \in N} V(d_p, d_q)$$

$D_p(d_p) = |I_p - I'_{p \oplus d_p}|$ photo consistency

$V(d_p, d_q) = w\,|d_p - d_q|$ spatial coherence



CASE 2



$N = \{\{p, p\pm 1\} : p \in G\}$

Pairwise coherence is enforced, but only between pixels on the same scan line.

Q: how do we optimize such $E(\mathbf{d})$?

$O(nm^2)$

Estimating (optimizing) disparities: over **points** vs. **scan-lines** vs. **grid**

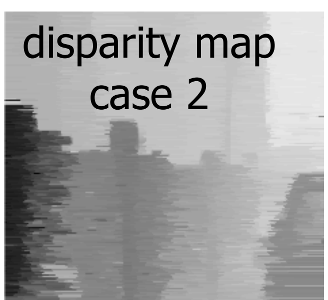
Consider energy (loss) function over disparities

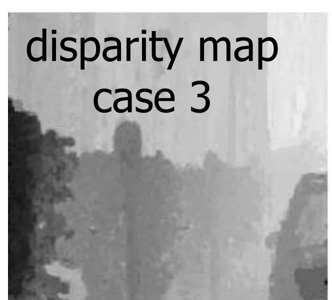
$$\mathbf{d} = \{d_p \mid p \in G\}$$ for pixels p on grid G

$$E(\mathbf{d}) = \sum_{p \in G} D_p(d_p) + \sum_{\{p,q\} \in N} V(d_p, d_q)$$

$D_p(d_p) = |I_p - I'_{p \oplus d_p}|$ — photo consistency

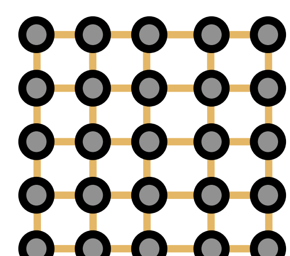
$V(d_p, d_q) = w\,|d_p - d_q|$ — spatial coherence





Pairwise smoothness of the disparity map is enforced both horizontally and vertically.

CASE 3



$$N = \{\{p,q\} \subset G : |pq| \le 1\}$$

NOTE: *depth map* regularity/smoothness should be isotropic since 3D scene surface is independent of scan-lines (epiplar lines) orientation.

Estimating (optimizing) disparities: over **points** vs. **scan-lines** vs. **grid**

Consider energy (loss) function over disparities

$\mathbf{d} = \{d_p \mid p \in G\}$ for pixels p on grid G

$$E(\mathbf{d}) = \sum_{p \in G} D_p(d_p) + \sum_{\{p,q\} \in N} V(d_p, d_q)$$

$D_p(d_p) = |I_p - I'_{p \oplus d_p}|$ — photo consistency

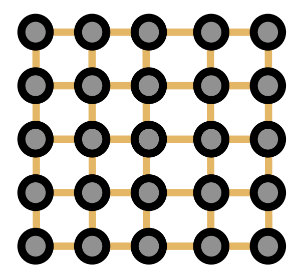
$V(d_p, d_q) = w\,|d_p - d_q|$ — spatial coherence

How to optimize "pairwise" loss on loopy graphs?

NOTE 1: **Viterbi does not apply**, but its extensions (e.g. *message passing*) provide approximate solutions on loopy graphs.

NOTE 2: "*Gradient descent*" can find only local minima for a continuous relaxation of $E(\mathbf{d})$ combining non-convex photo-consistency (1st term) and convex *total variation* of $\mathbf{d}$ (2nd term).

CASE 3



$N = \{\{p,q\} \subset G : |pq| \leq 1\}$

Estimating (optimizing) disparities: over **points** vs. **scan-lines** vs. **grid**

Consider energy (loss) function over disparities $\mathbf{d} = \{d_p \mid p \in G\}$ for pixels p on grid G

$$E(\mathbf{d}) = \sum_{p \in G} \underbrace{D_p(d_p)}_{= \, |I_p - I'_{p \oplus d_p}|} + \sum_{\{p,q\} \in N} \underbrace{V(d_p, d_q)}_{= \, w\, |d_p - d_q|}$$

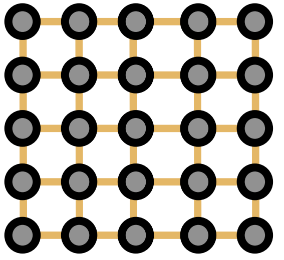
photo consistency

spatial coherence

Can **globally minimize** such pairwise losses **over any neighborhood** ***N*****,** e.g. *graph cut* algorithms

(**convex potentials** *V* **only** [Ishikawa et al 1998])

CASE 3



$N = \{\{p,q\} \subset G : |pq| \leq 1\}$

Useful extension:
local affinities w_{pq} and "edge alignment"

Consider energy (loss) function over disparities

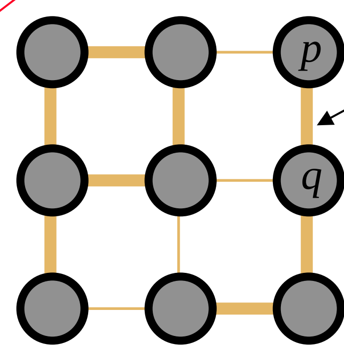
$\mathbf{d} = \{d_p \mid p \in G\}$ for pixels p on grid G

$$E(\mathbf{d}) = \sum_{p \in G} \underbrace{D_p(d_p)}_{= \; |I_p - I'_{p \oplus d_p}|} + \sum_{\{p,q\} \in N} \underbrace{V(d_p, d_q)}_{= \; w_{pq} \cdot |d_p - d_q|}$$

$|I_p - I'_{p \oplus d_p}|$ photo consistency

$w_{pq} \cdot |d_p - d_q|$ spatial coherence

In general, one can use pairwise affinities w_{pq} specific to each pair of neighboring pixels p and q



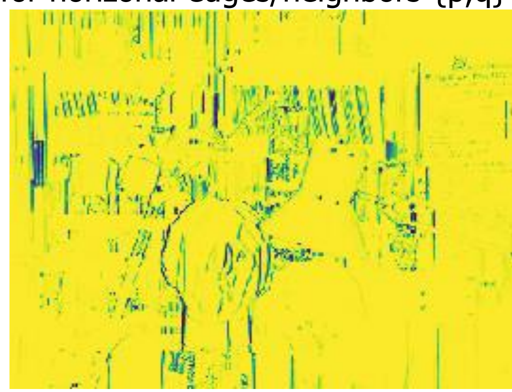
w_{pq} - weights of neighborhood edges (e.g. may be assigned according to local intensity contrast in the **reference image**)

Useful extension:
local affinities w_{pq} and "edge alignment"

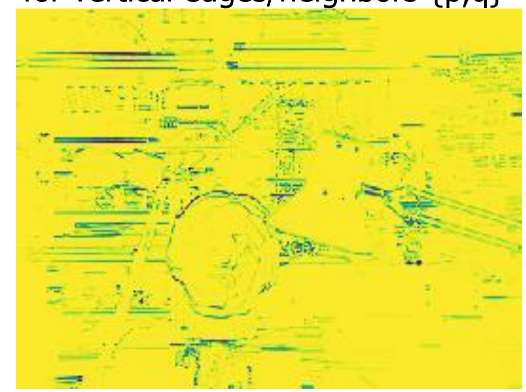
visualization of weights w_{pq}

reference (e.g. right) image I

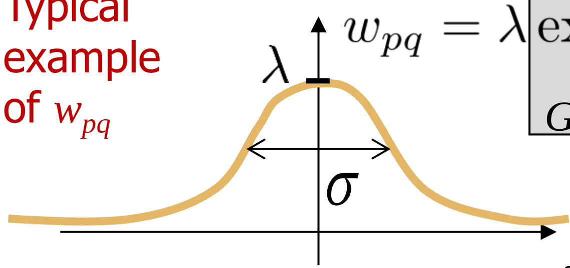
for horizonal edges/neighbors {p,q}



for vertical edges/neighbors {p,q}



Typical example of w_{pq}

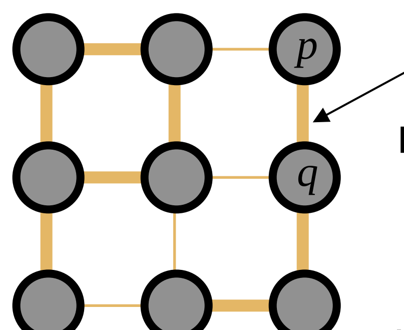


$$w_{pq} = \lambda \exp\left\{-\frac{\|I_p - I_q\|^2}{2\sigma^2}\right\}$$

Gaussian (affinity) *kernel*

Similar "edge aligning" affinity kernels are common in low-level segmentation (see topic 9). "Deep features" f_p can replace I_p (topic 12).

$\Delta I_{pq} = I_p - I_q$
differences in the **reference image**



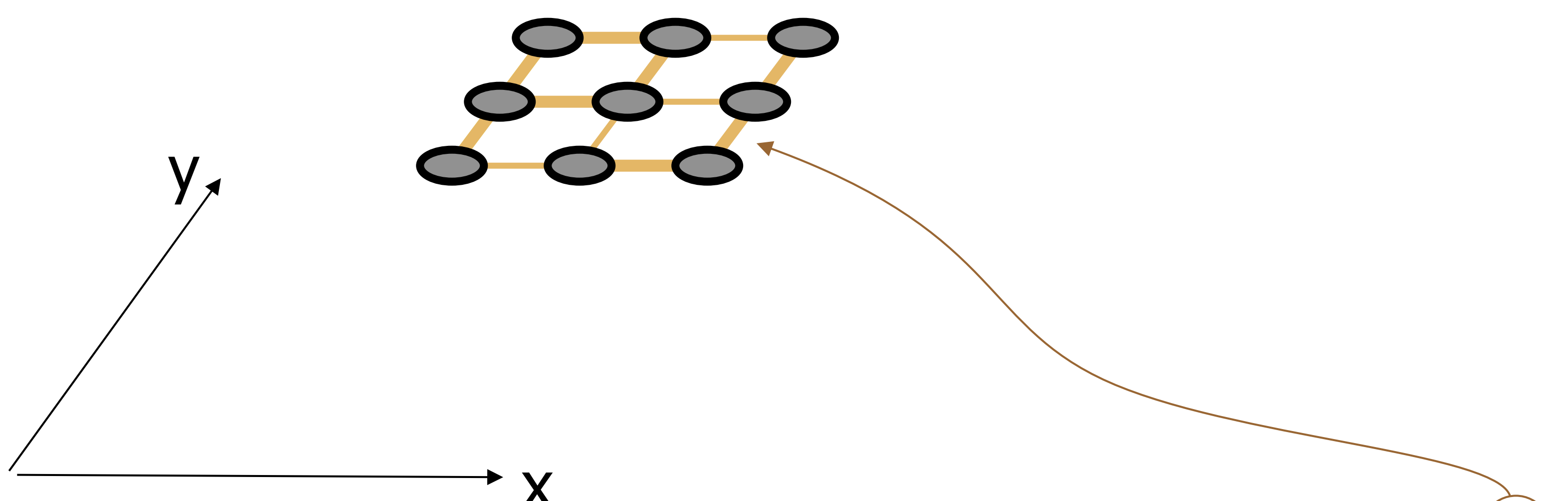
w_{pq} - weights of neighborhood edges (e.g. may be assigned according to local intensity contrast in the **reference image**)

Motivation: such *static cues* (in ref. image) help to **align depth boundaries to high contrast edges** since the loss function gets lower when disparity changes $|d_p-d_q|$ happen near edges where $|\Delta I_{pq}| > \sigma$.

NOTE: **parameter σ** is important, it works as (soft) **edge detection threshold**.

Multi-scan-line stereo with *s*-*t* graph cuts [Roy&Cox'98, Ishikawa'98]

optional slides (grad students)



$$E(\mathbf{d}) = \sum_{p \in G} D_p(d_p) + \sum_{\{pq\} \in N} w_{pq} |d_p - d_q|$$

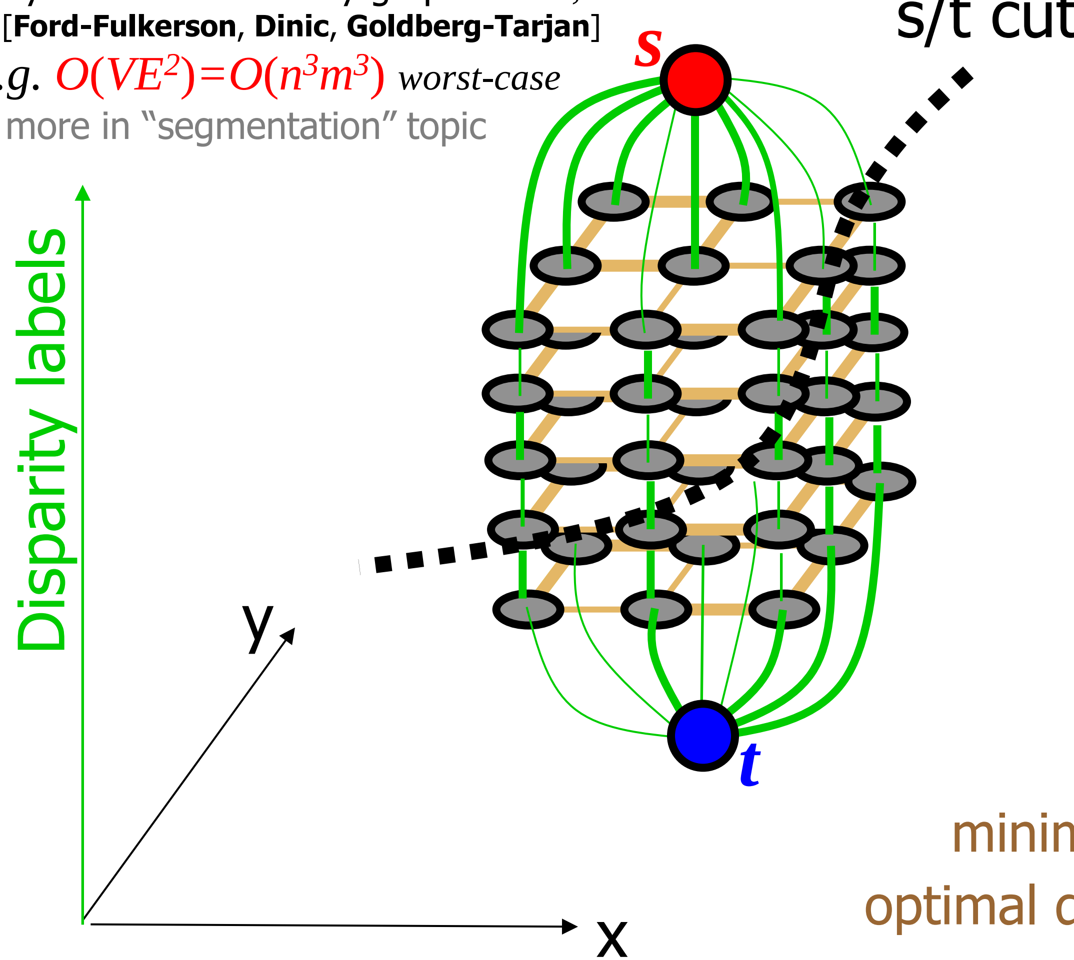
Multi-scan-line stereo with *s-t* graph cuts [Roy&Cox'98, Ishikawa'98]

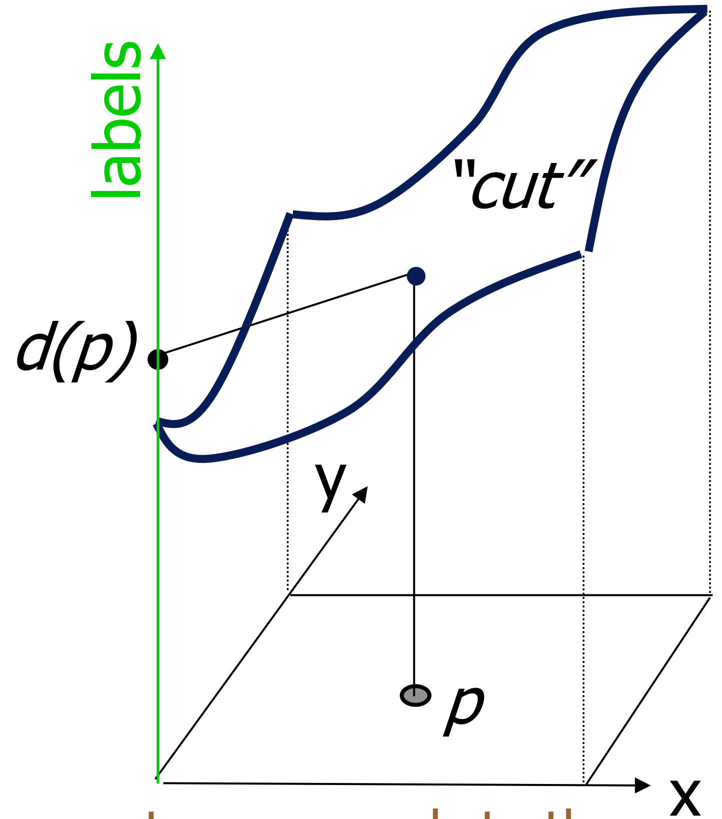
optional slides (grad students)

Minimum s/t cuts can be found in low-order polynomial time on any graph *G=<V,E>* [**Ford-Fulkerson**, **Dinic**, **Goldberg-Tarjan**]

e.g. $O(VE^2)=O(n^3m^3)$ *worst-case*

more in "segmentation" topic





minimum cut corresponds to the optimal disparity map $\mathbf{d} = \{d_p\}$ for loss

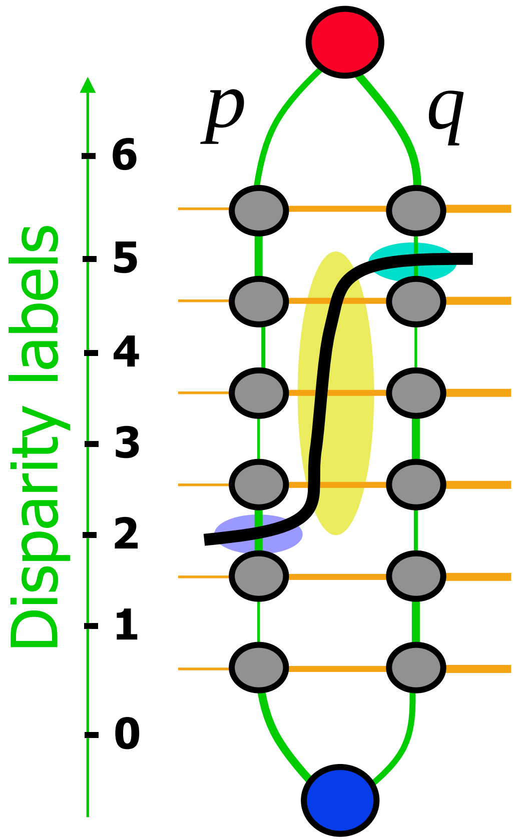
$$E(\mathbf{d}) = \sum_{p \in G} D_p(d_p) + \sum_{\{pq\} \in N} w_{pq}|d_p - d_q|$$

Ishikawa&Geiger 98

What loss function do we minimize this way?

optional slides (grad students)

Concentrate on one pair of neighboring pixels $\{p,q\} \in N$



cost of vertical edges

$$E(d_p, d_q) = D_p(2) + D_q(5) + \ldots$$

$$+ \quad w_{pq} \cdot |3| \quad + \ldots$$

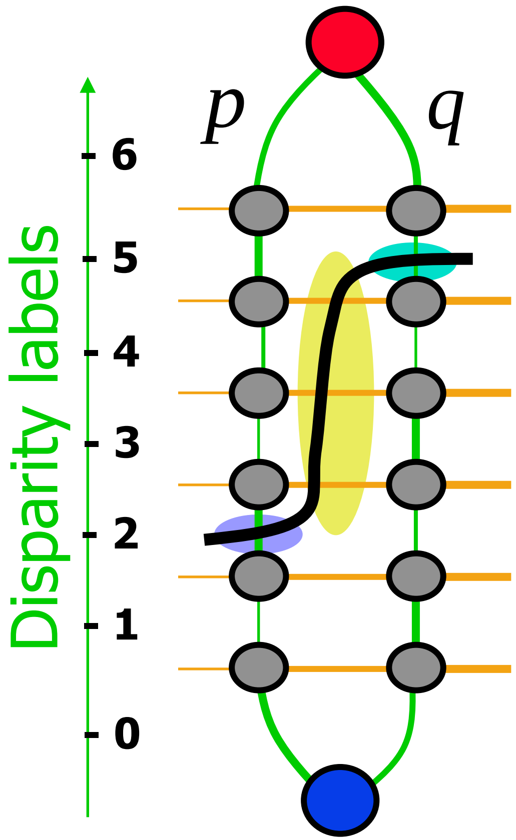
cost of horizontal edges

assuming cut has no folds (optional slides later shows how to make sure) optional slides (grad students)

Ishikawa&Geiger 98

What loss function do we minimize this way?

Concentrate on one pair of neighboring pixels $\{p,q\} \in N$



cost of vertical edges

$$E(d_p, d_q) = D_p(d_p) + D_q(d_q) + \ldots$$

$$+ \; w_{pq} \cdot |d_p - d_q| + \ldots$$

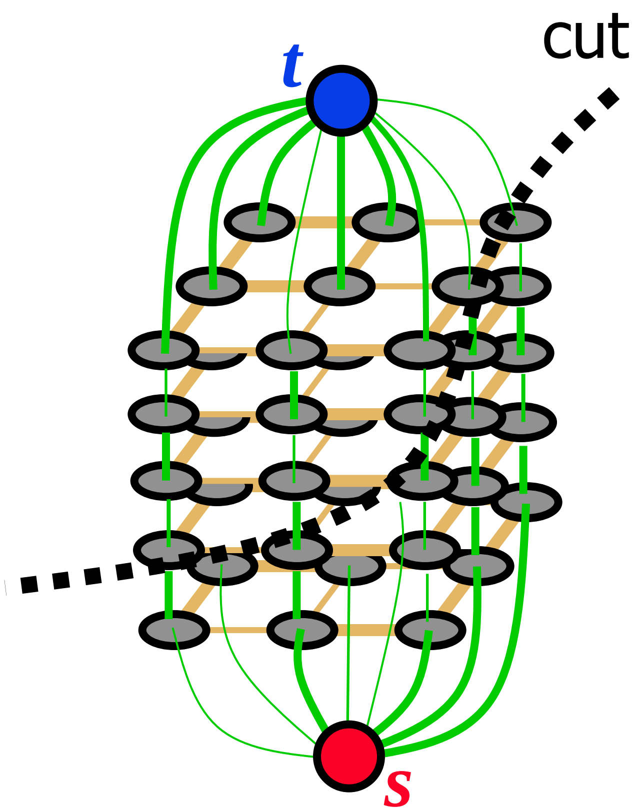
cost of horizontal edges

Ishikawa&Geiger 98

optional slides (grad students)

What loss function do we minimize this way?

The combined loss over the entire grid G is



(**photo consistency**, e.g. SSD)
cost of vertical edges

$$E(\mathbf{d}) = \sum_{p\in G} D_p(d_p)$$

$$+ \sum_{\{p,q\}\in N} w_{pq}\cdot|d_p - d_q|$$

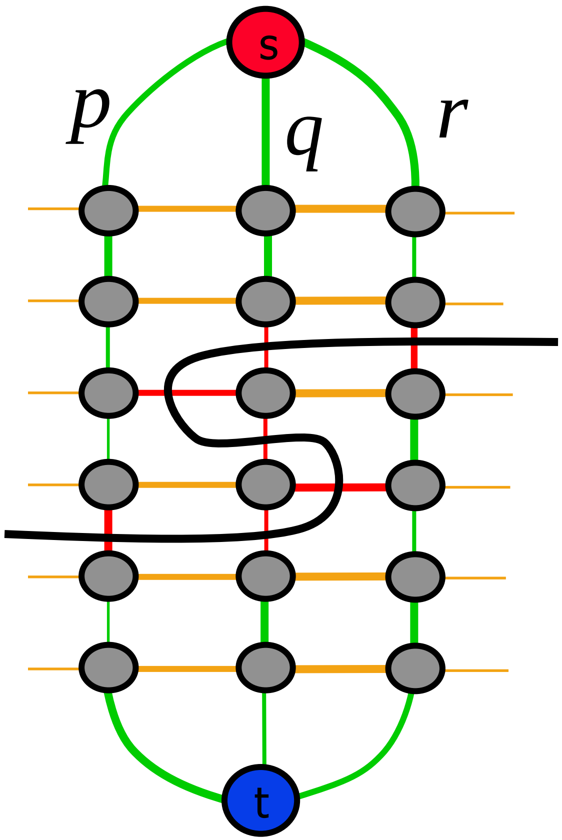
cost of horizontal edges
(**spatial consistency**)

optional slides
(grad students)

UNIVERSITY OF WATERLOO

How to avoid folding?

consider three pixels $\{p, q, r\}$

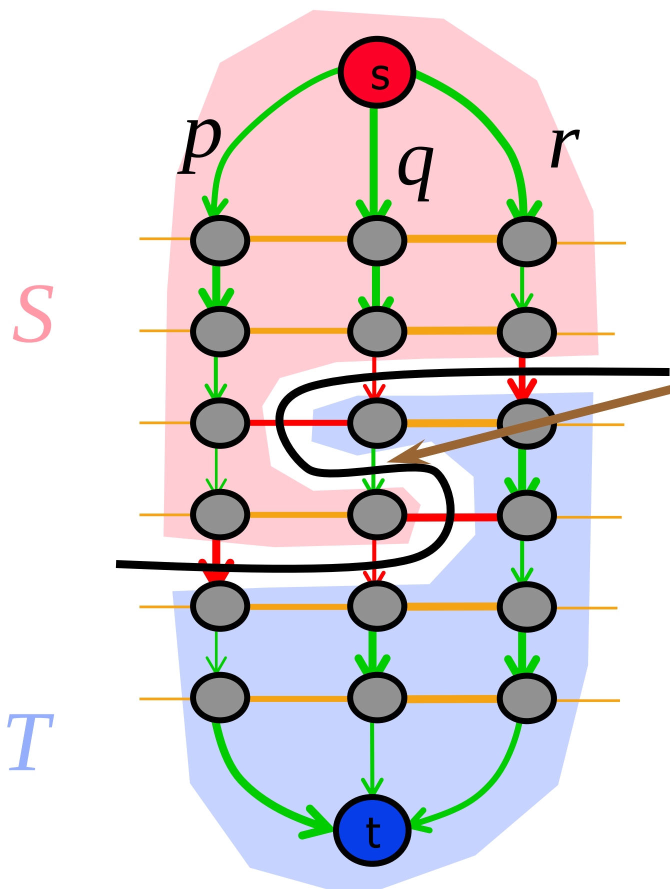


"severed" edges are shown in **red**

optional slides (grad students)

How to avoid folding?

consider three pixels $\{p,q,r\}$



introduce <u>directed</u> *t-links*

NOTE: this directed *t-link* is not "severed"

WHY?

Formally, s/t cut is a <u>partitioning of graph nodes</u>

$C=\{S,T\}$ and its cost is $\|C\| = \sum_{\substack{(pq)\in N \\ p\in S \\ q\in T}} c_{pq}$

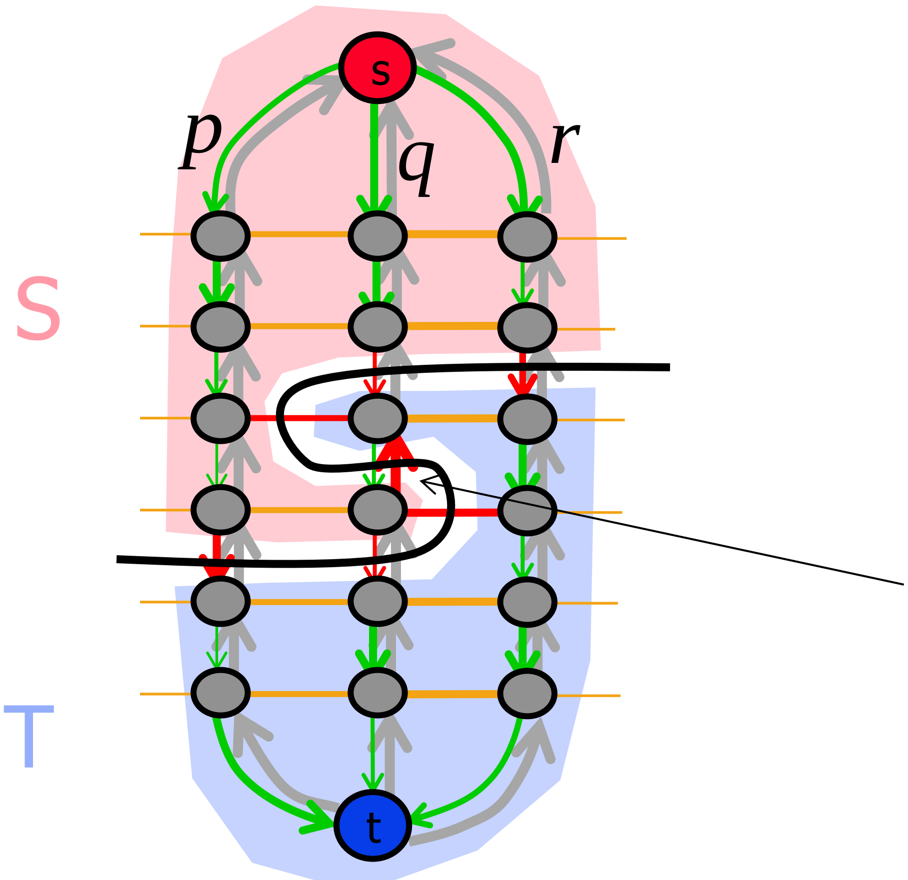
only edges from S to T matter

optional material

optional slides (grad students)

How to avoid folding?

consider three pixels $\{p, q, r\}$



Solution prohibiting **folds**:

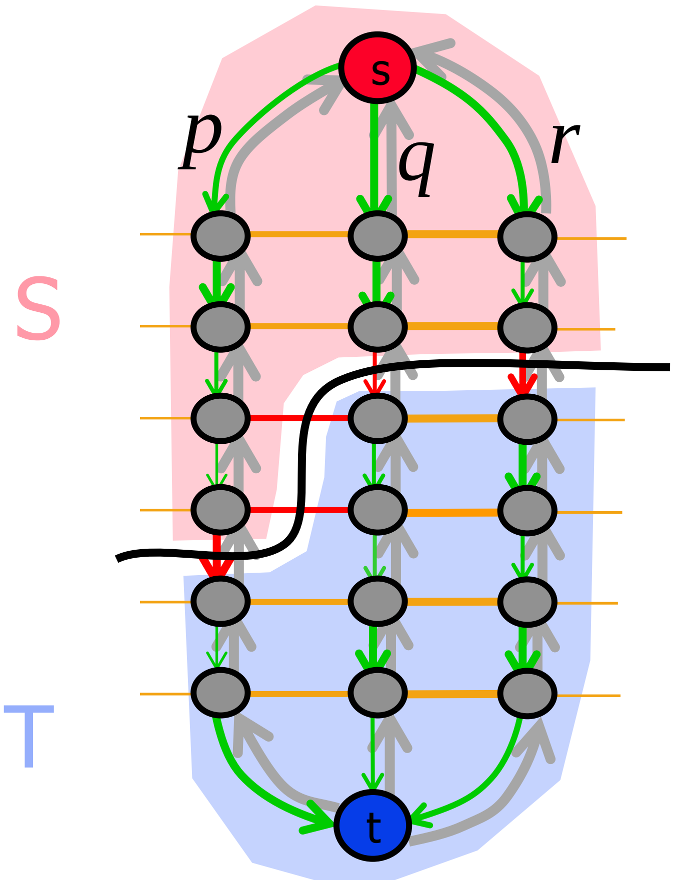
add infinity cost t-links in the "up" direction ↑

NOTE: **folding cuts** $C = \{S, T\}$ sever at least one of such t-links making such cuts **infeasible** optional material

optional slides (grad students)

How to avoid folding?

consider three pixels $\{p, q, r\}$

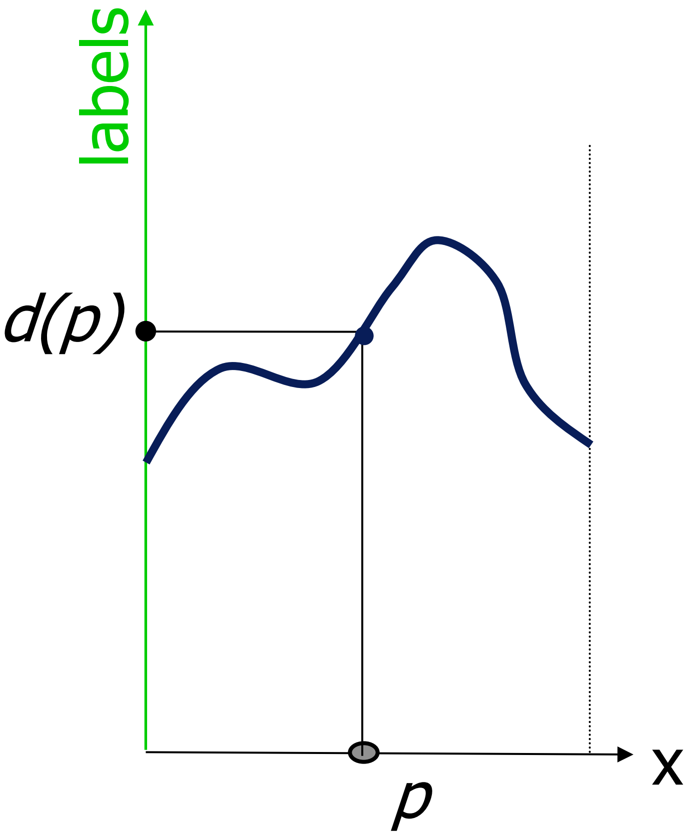


Solution prohibiting **folds**:

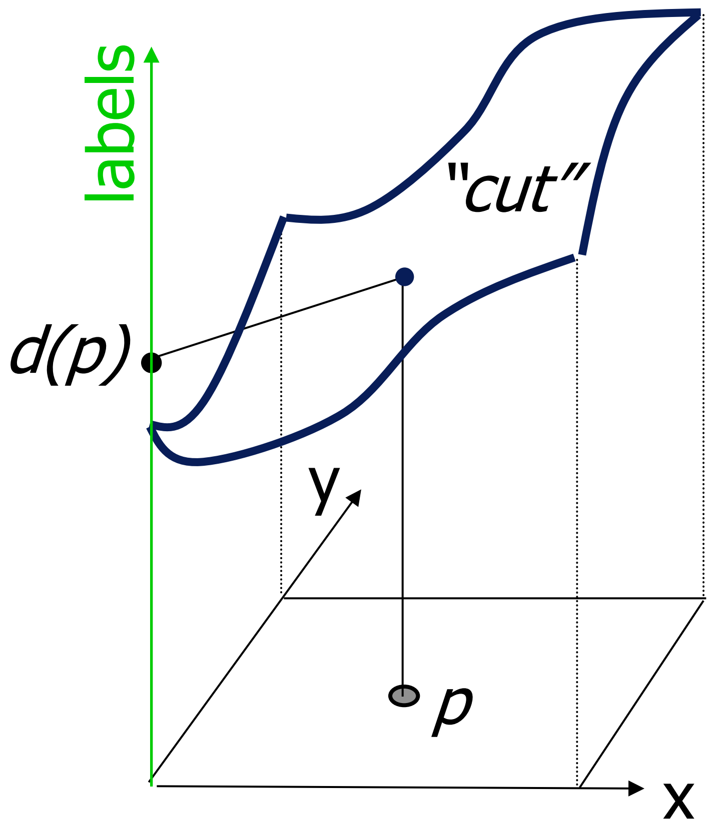
add <u>infinity</u> cost t-links in the "up" direction ↑

NOTE: **non-folding cuts** $C = \{S, T\}$ do not sever such t-links

Scan-line stereo vs. Multi-scan-line stereo (on whole grid)

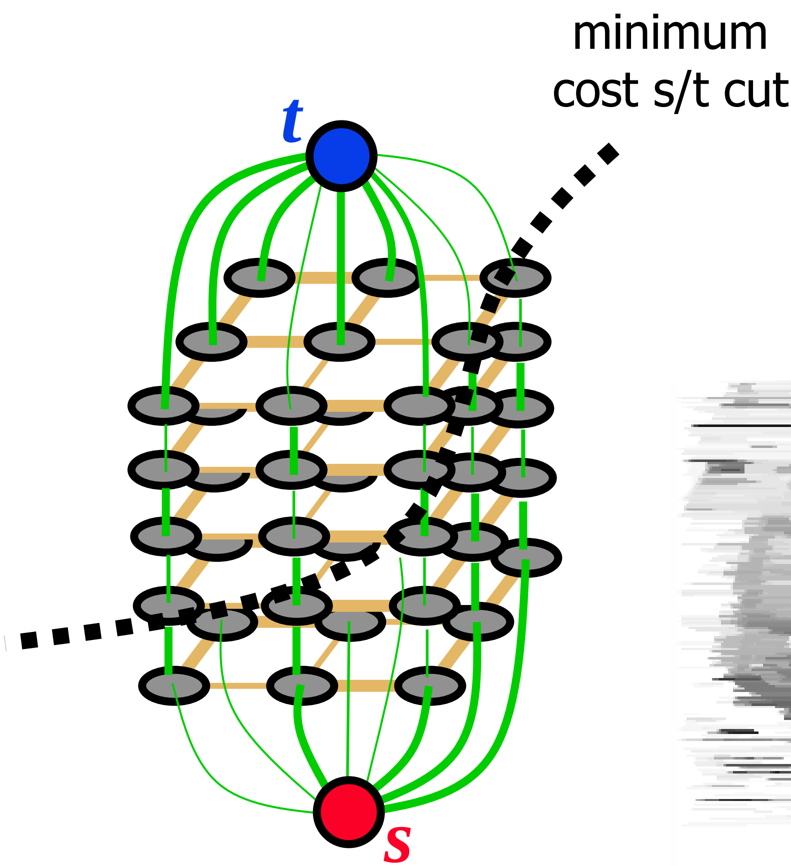


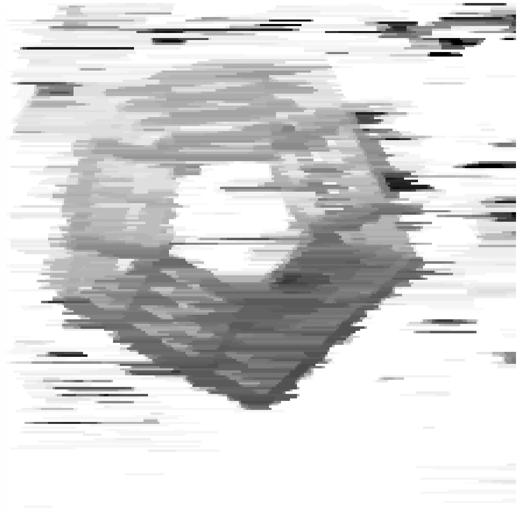
Dynamic Programming
(single scan line optimization)



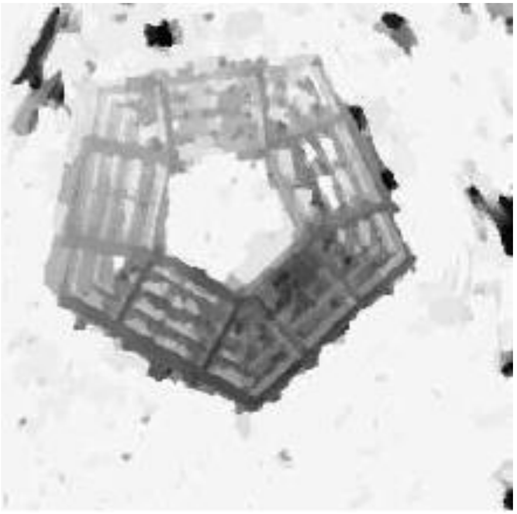
s-t Graph Cuts
(grid optimization)

Some results from Roy&Cox



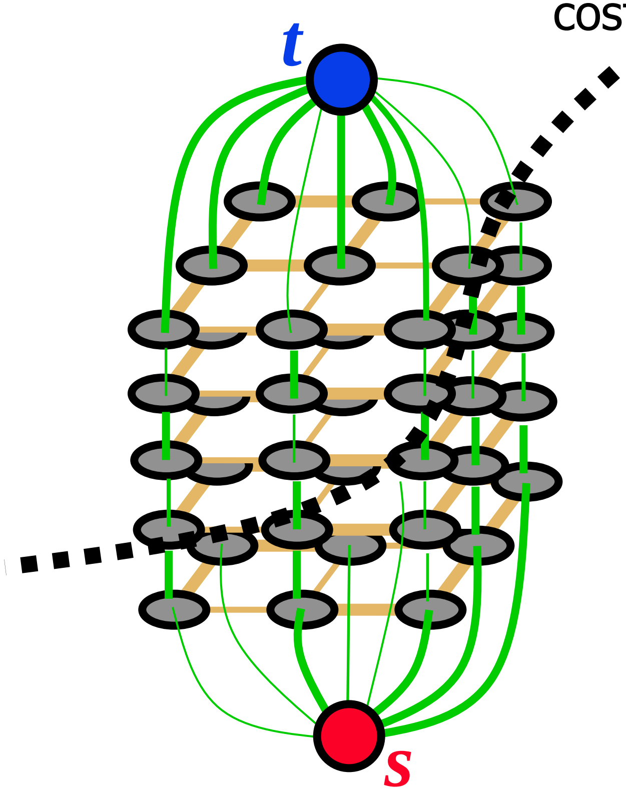


single scan-line stereo
(DP)



multi scan line stereo
(graph cuts)

Some results from Roy&Cox



single scan-line stereo
(DP)

multi scan line stereo
(graph cuts)

Simple Examples:
Stereo with only 2 depth layers

binary stereo

essentially,
depth-based binary
segmentation

[Kolmogorov et al. CVPR 2005, IJCV 2008]

Simple Examples: Stereo with only 2 depth layers

background substitution

essentially,
depth-based binary
segmentation

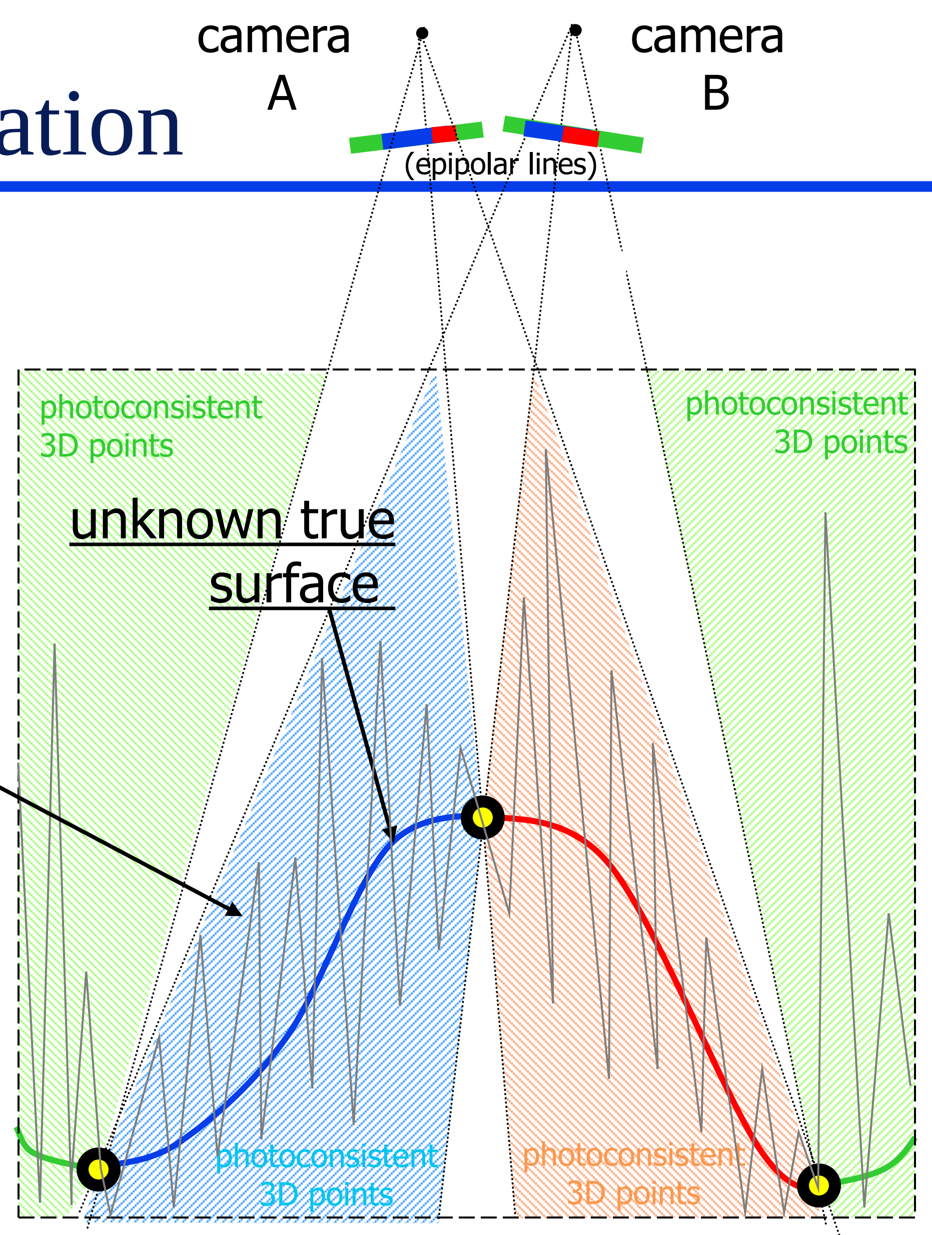
[Kolmogorov et al. CVPR 2005, IJCV 2008]

Features and Spatial Regularization

photo-consistency term

$$E(\mathbf{d}) \;=\; \sum_{p\in G} |I_p - I'_{p+d_p}|$$

photoconsistent depth map



3D volume where surface is being reconstructed (epipolar plane)

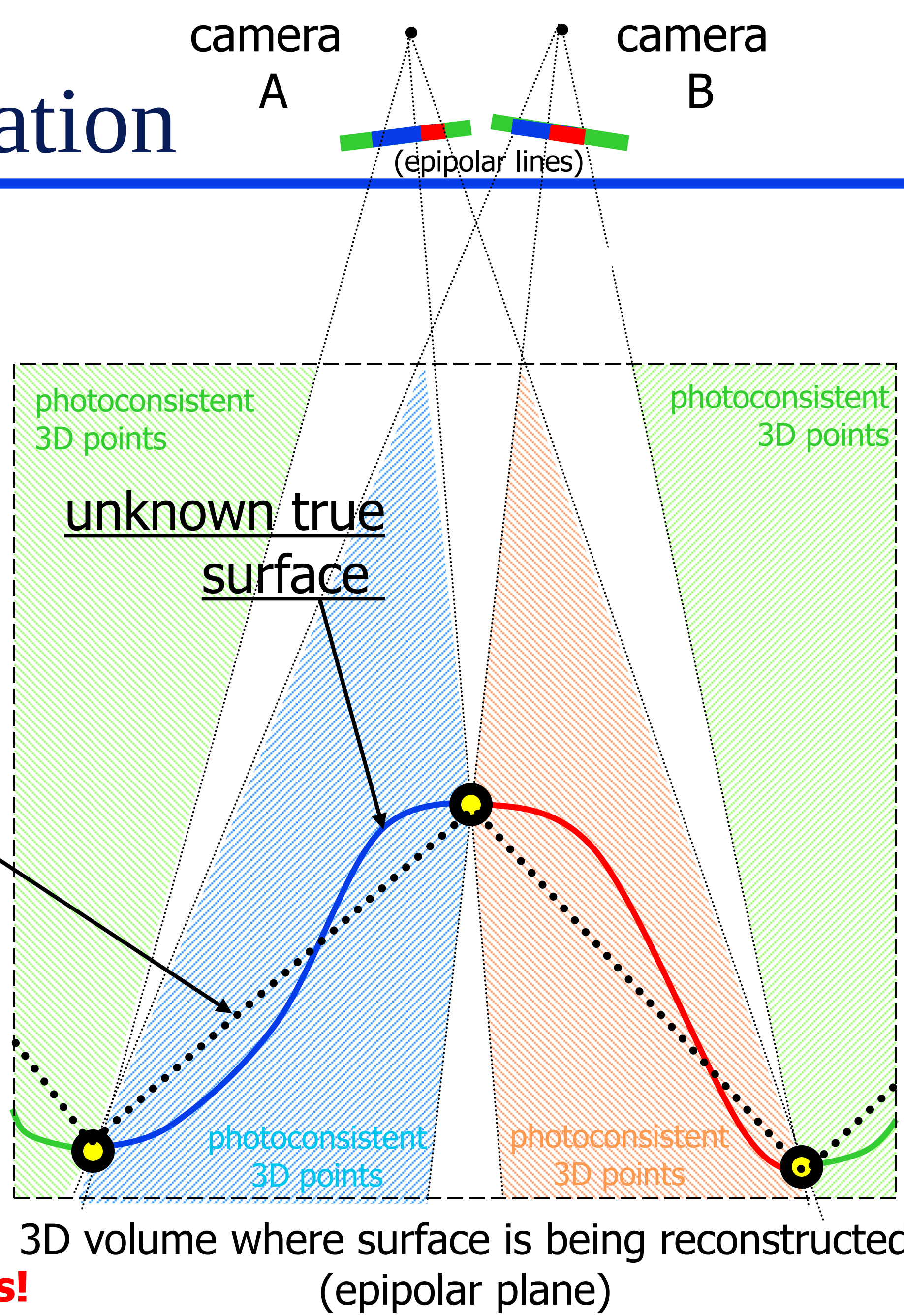
Features and Spatial Regularization

UNIVERSITY OF WATERLOO

photo-consistency term

$$E(\mathbf{d}) \;=\; \sum_{p\in G} |I_p - I'_{p+d_p}| \;+\; \sum_{pq\in N} w\,|d_p - d_q|$$

regularization term



regularized depth map

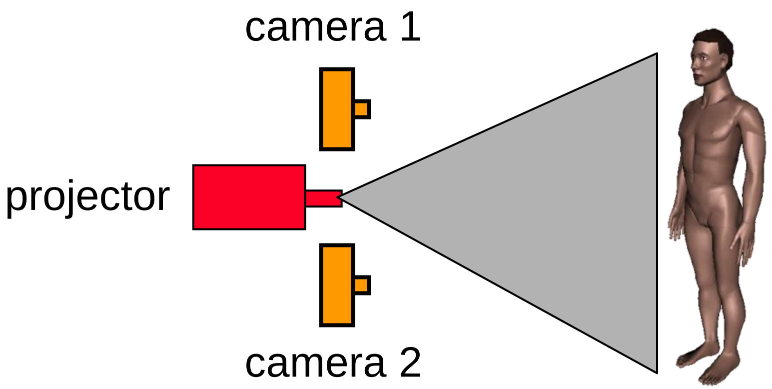
- regularization helps to find **smooth** depth map consistent with points uniquely matched by photoconsistency

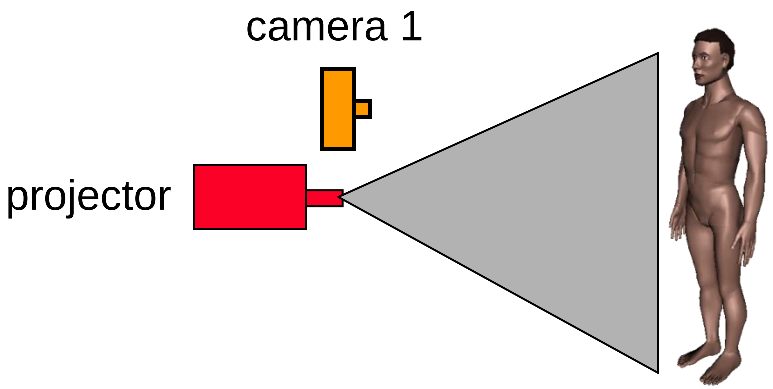
- regularization propagates information from textured regions (features) to ambiguous textureless regions

More features/texture always helps!

Active Stereo (with structured light)

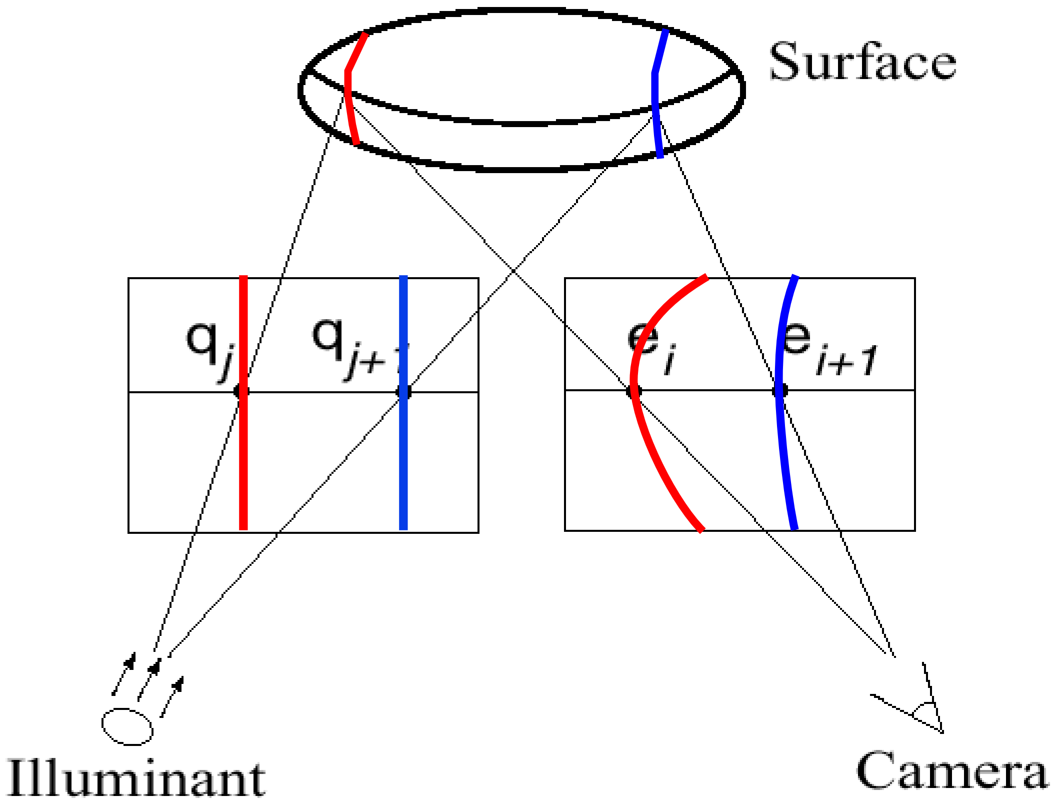
Li Zhang's one-shot stereo



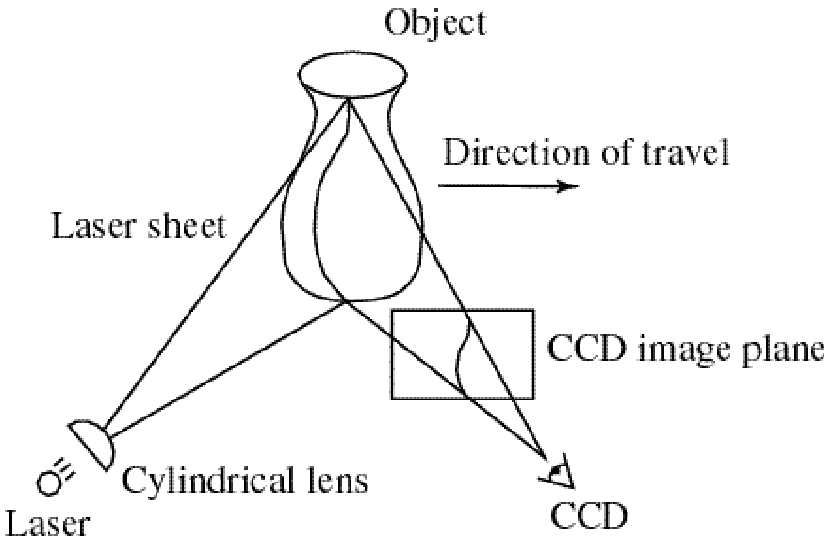


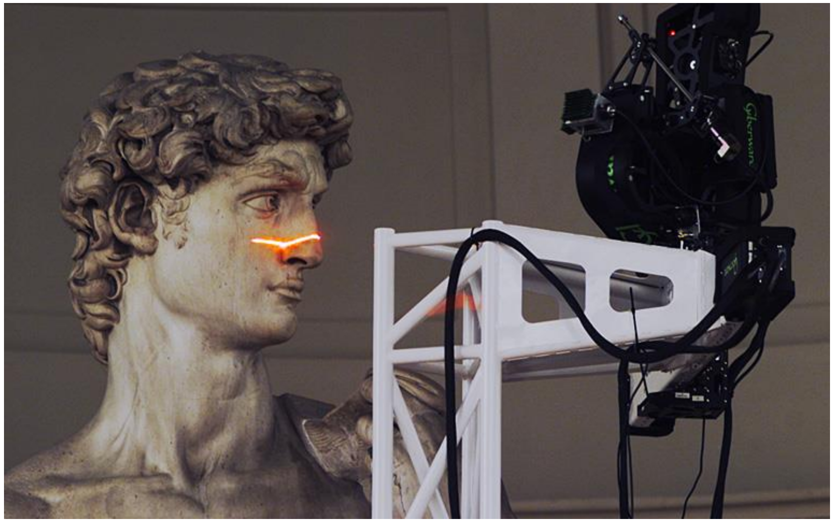
- Project "structured" light patterns onto the object
  - simplifies the correspondence problem

Active Stereo (with structured light)



Laser scanning

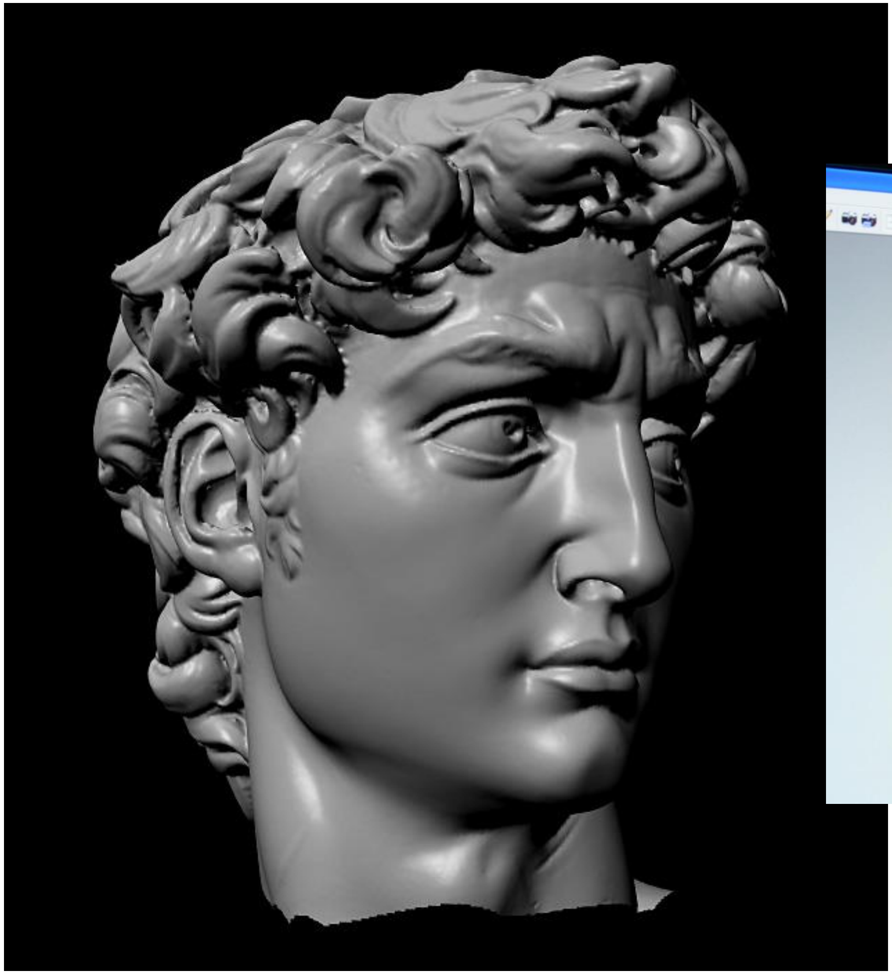




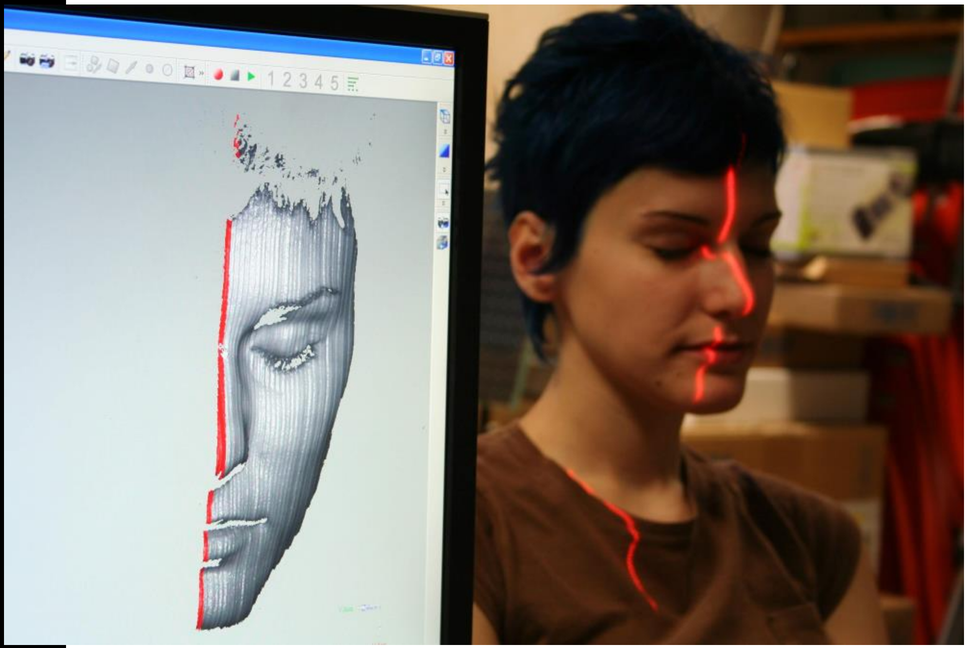
Digital Michelangelo Project [Levoy et al.]
http://graphics.stanford.edu/projects/mich/

- Optical triangulation
  - Project a single stripe of laser light
  - Scan it across the surface of the object
  - This is a very precise version of structured light scanning

Laser scanning

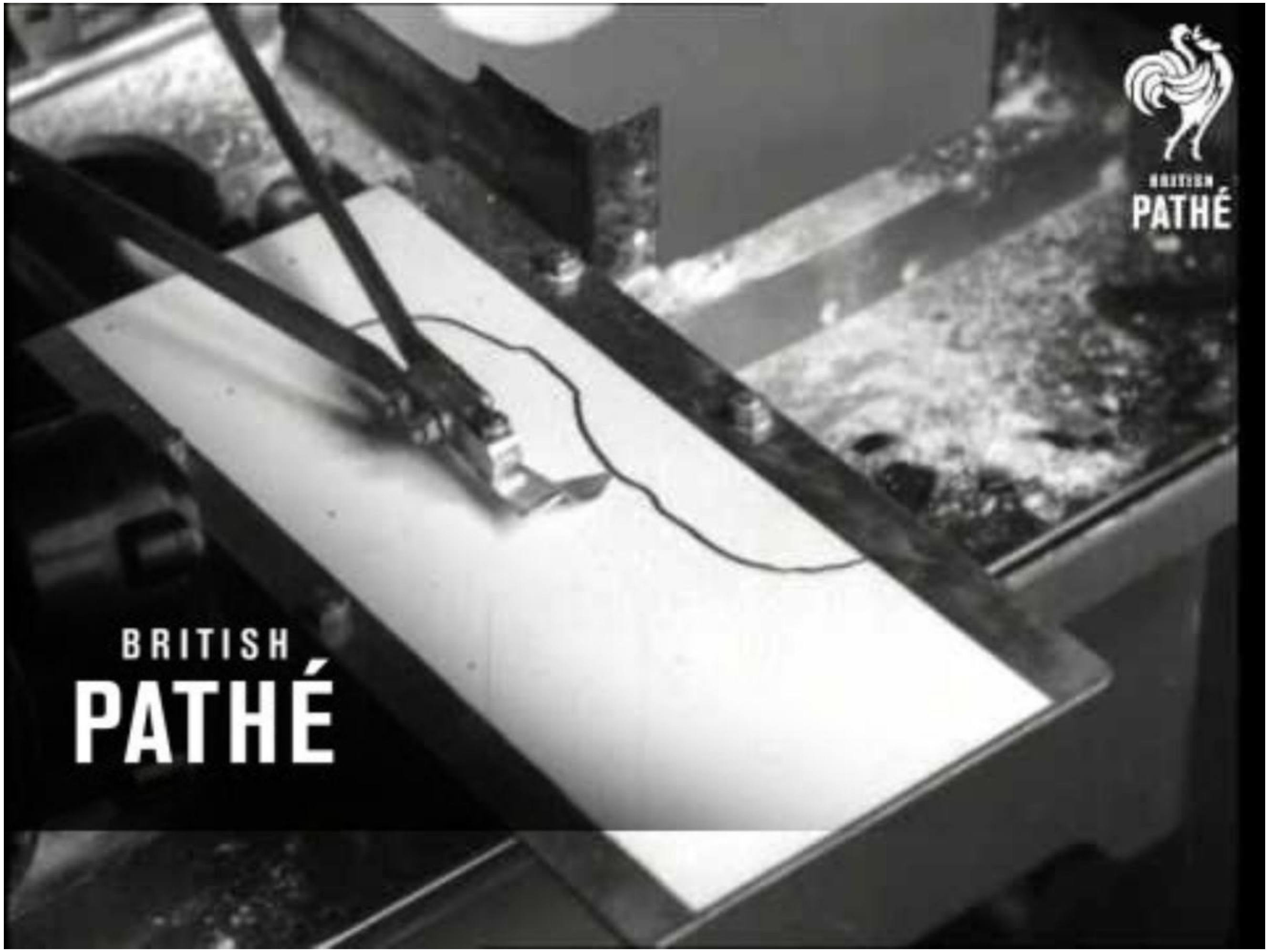


Digital Michelangelo Project [Levoy et al.]
http://graphics.stanford.edu/projects/mich/



basic laser scanner

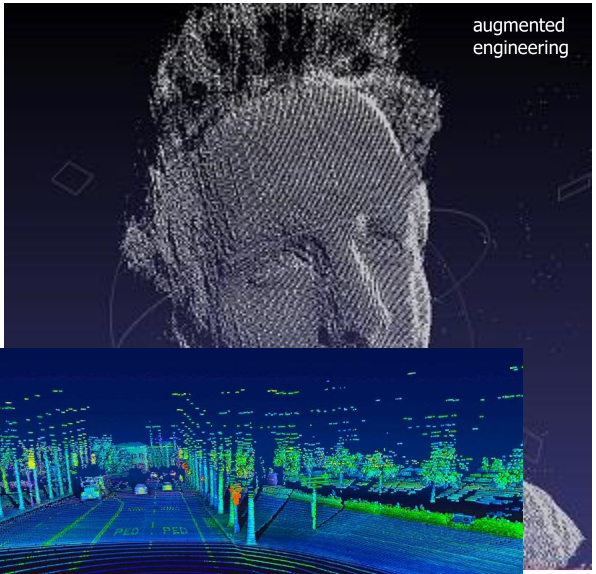
Photo Sculpture (1939)



https://youtu.be/jS_rcwG9mxU?si=JcrzZs2VSZb6yvuS

3D scanning

- Stereo
  - uses photo sensors
  - requires ambient light, does not work in dark environment
  - ambiguous in textureless regions, noisy on specular surfaces



- Active Stereo
  - uses photo sensors and active light sources (e.g. laser)
  - problems with specular or non-reflective surfaces
  - problems with bright ambient light

- Lidar
  - uses *time-of-flight* sensors and active light (laser)
  - good range, no baseline required
  - problems with specular or non-reflective surfaces
  - problems with bright ambient light
  - relatively sparse output (cloud of points)

To produce **dense scene reconstruction**, all methods should address noise and ambiguities by fitting various dense surface models, typically using **surface regularization techniques**

Disparity map $d(p)$ is an example of (regularised) surface model, more in Topic 9B

further considerations:
Robust error/penalty functions

$$E(\mathbf{d}) = \sum_{p \in G} D_p(d_p) + \sum_{\{p,q\} \in N} V(d_p, d_q)$$

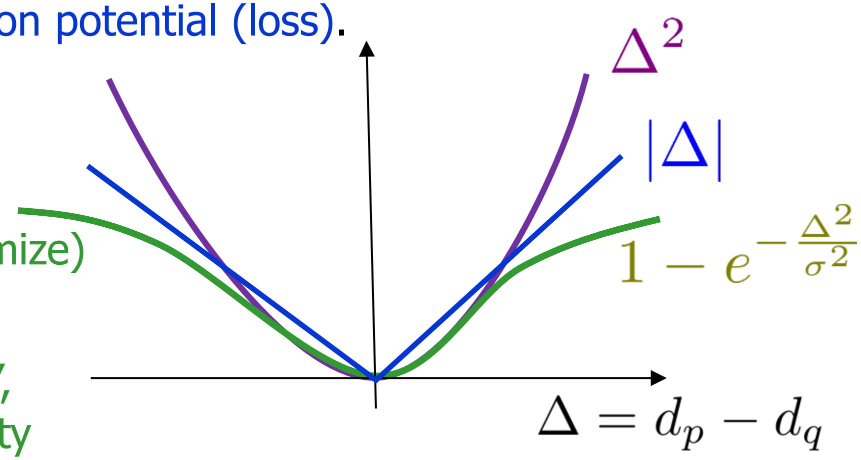
photo-consistency $D_p(d_p) = |I_p - I'_{p \oplus d_p}|$

$V(d_p, d_q) = w_{pq} \cdot |d_p - d_q|$ spatial coherence

The last term is an example of **convex** regularization potential (loss).

- easier to optimize, but
- tend to over-smooth

practically preferred
robust regularization
(non convex – harder to optimize)



Note: once deviation/error Δ is "large enough",
there is no reason to keep increasing the penalty

further considerations:

Robust error/penalty functions

$$E(\mathbf{d}) = \sum_{p\in G} \underbrace{D_p(d_p)}_{=\,|I_p - I'_{p\oplus d_p}|} + \sum_{\{p,q\}\in N} \underbrace{V(d_p, d_q)}_{=\,w_{pq}\cdot|d_p - d_q|}$$

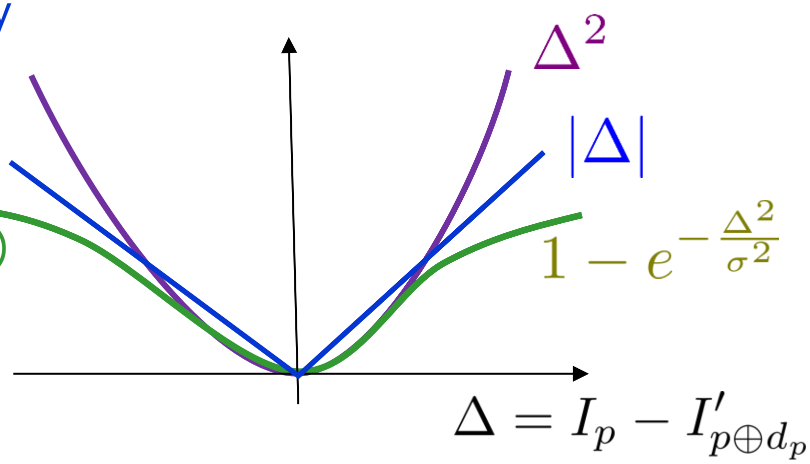
photo-consistency $|I_p - I'_{p\oplus d_p}|$

$w_{pq}\cdot|d_p - d_q|$ spatial coherence

Similarly, robust losses are needed for photo-consistency to handle occlusions & "specularities"

practically preferred
robust regularization
(non convex – harder to optimize)

Note: once deviation/error Δ is "large enough", there is no reason to keep increasing the penalty



further considerations:
Higher-order regularization

$$E(\mathbf{d}) = \sum_{p \in G} D_p(d_p) + \sum_{\{p,q\} \in N} \cancel{V(d_p, d_q)}$$

photo-consistency $D_p(d_p) = |I_p - I'_{p \oplus d_p}|$

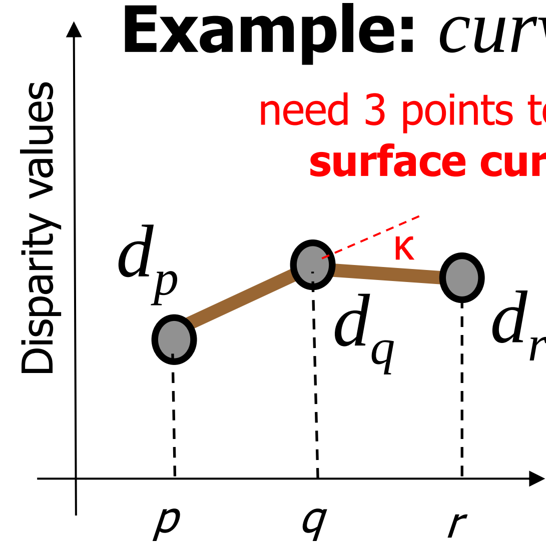
$$V(d_p, d_q, d_r) \quad p, q, r \in N$$

higher-order "coherence"

Many state-of-the-art methods use <u>higher-order regularizers</u>

Example: *curvature*

need 3 points to estimate **surface curvature**



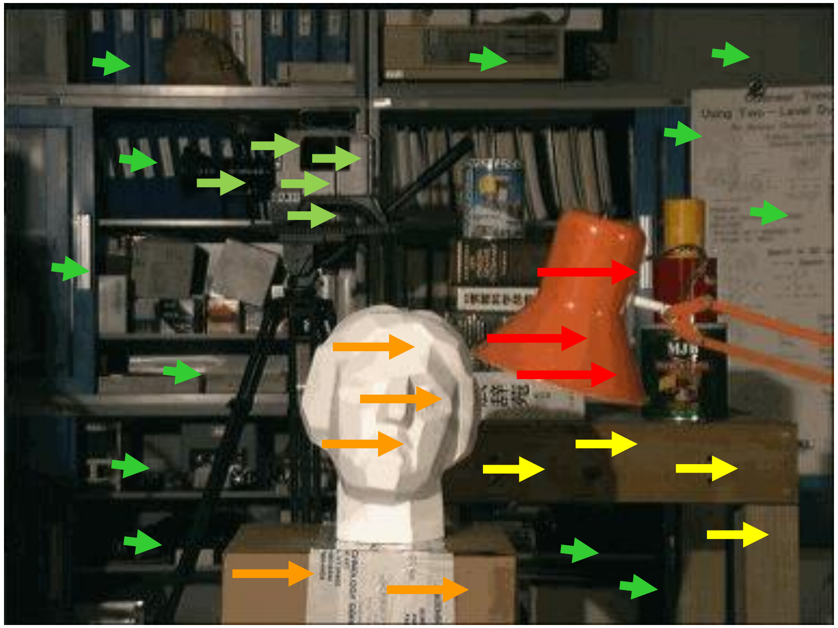
Q: why penalizing depth curvature instead of depth change?

From 1D correspondence (stereo) to 2D correspondence problems (motion)

1D shifts along **epipolar lines**.

Assumption for stereo:

only camera moves,
3D scene is stationary



vector field (motion) with a priori known direction

From 1D correspondence (stereo) to 2D correspondence problems (motion)

1D shifts along **epipolar lines**.

Assumption for stereo:

only camera moves,
<u>3D scene is stationary</u>

vector field (motion) with a priori *known direction*

⇨ We estimate only *magnitude* represented by a **scalar field** (disparity map)

From 1D correspondence (stereo) to 2D correspondence problems (motion)

In general, correspondences between two images
may not be described by global models (like *homography*) or
by shifts along known **epipolar lines**.

if 3D scene
is NOT stationary
motion is
vector field
with **arbitrary directions**
(no epipolar line constraints)

Society Of Robots

From 1D correspondence (stereo) to 2D correspondence problems (motion)

In general, correspondences between two images may not be described by global models (like *homography*) or by shifts along known **epipolar lines**.

For (non-rigid) motion the correspondences between two video frames are described by a general ***optical flow***

if 3D scene is NOT stationary motion is **vector field** with **arbitrary directions** (no epipolar line constraints)

Society Of Robots

From 1D correspondence (stereo) to 2D correspondence problems (motion)

$$E(\mathbf{v}) = \sum_{p \in G} \underbrace{D_p(v_p)}_{\text{color-consistency}} + \sum_{\{p,q\} \in N} \underbrace{V(v_p, v_q)}_{\text{regularity}}$$

$$D_p(v_p) = (I_p^t - I_{p+v_p}^{t+1})^2 \qquad V(v_p, v_q) = w \cdot \|v_p - v_q\|^2$$

Horn-Schunck 1981
optical flow regularization
- 2nd order optimization (pseudo Newton)
- Rox/Cox/Ishikawa's method only works for scalar-valued variables

optical flow
$\mathbf{v} = \{v_p\}$

if 3D scene is NOT stationary motion is **vector field** with **arbitrary directions**
(no epipolar line constraints)

SOCIETY OF ROBOTS

more difficult problem

need 2D shift vectors v_p

(no epipolar line constraint)

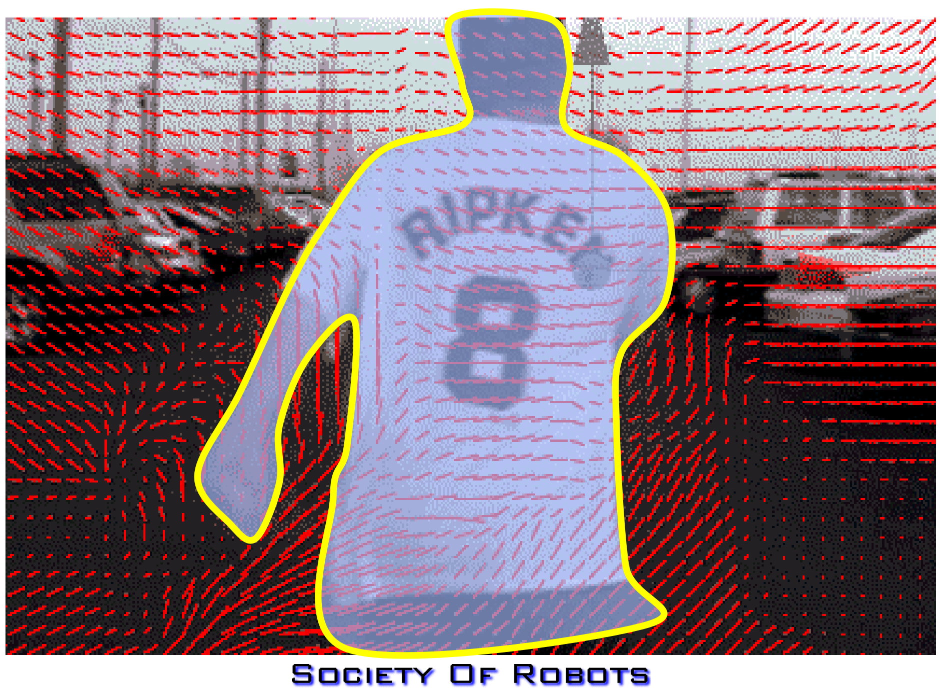
over-smoothed vector field
(robust regularization losses can preserve sharp changes in motion between objects)

From 1D correspondence (stereo) to 2D correspondence problems (motion)

State-of-the-art methods **segment** independently moving objects

We will discuss segmentation problem next

if 3D scene is <u>NOT stationary</u> motion is **vector field** with **arbitrary directions**
(no epipolar line constraints)



optical flow

$$\mathbf{v} = \{v_p\}$$

more difficult problem

need 2D shift vectors v_p

(no epipolar line constraint)

over-smoothed vector field
(robust regularization losses can preserve sharp changes in motion between objects)