

CS325 Artificial Intelligence

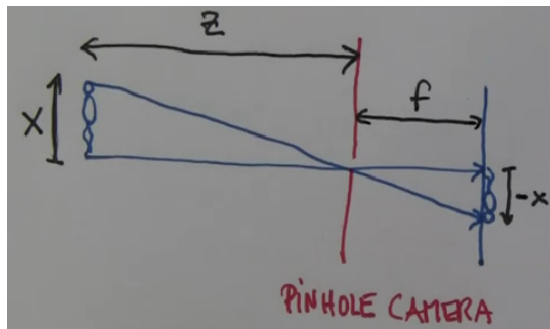
Computer Vision II – 3D Vision (Ch. 24)

Dr. Cengiz Günay, Emory Univ.



Spring 2013

Limits of 2D Projection



- 3D world is projected onto 2D image
- What happens to depth information?

Getting the Depth from Perspective Projection



Getting the Depth from Perspective Projection



Getting the Depth from Perspective Projection



Getting the Depth from Perspective Projection



- Giant panda, or just close?

Getting the Depth from Perspective Projection



- Giant panda, or just close?
- Can only tell if we know exactly the size.

Alternative?

Alternative?

Use your two eyes: Stereo Vision



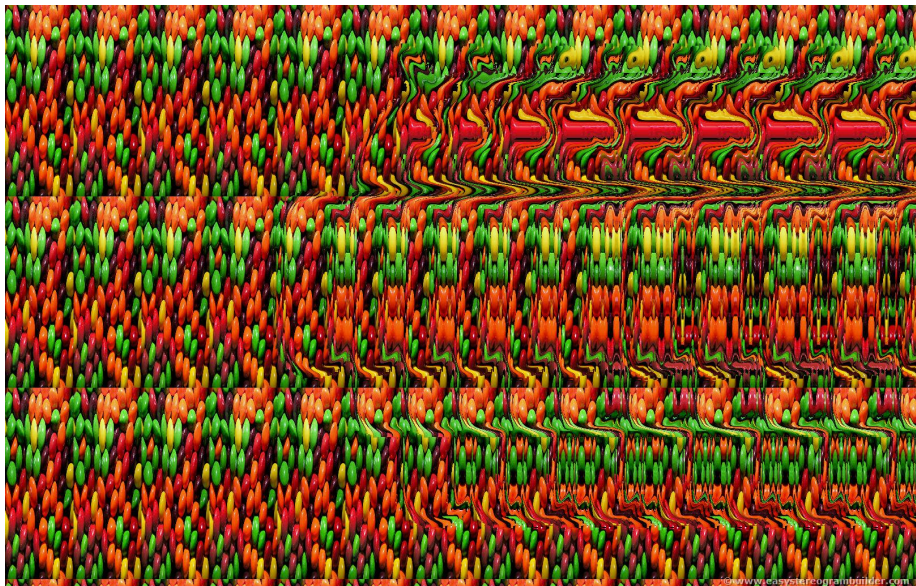
Exit survey: Computer Vision I – Object Recognition

- List some problematic states of objects for which an object recognition algorithm must be invariant for.
- What kind of a filter mask would you convolve with an image to detect *diagonal* lines?

Entry survey: Computer Vision II – 3D Vision (0.25 points)

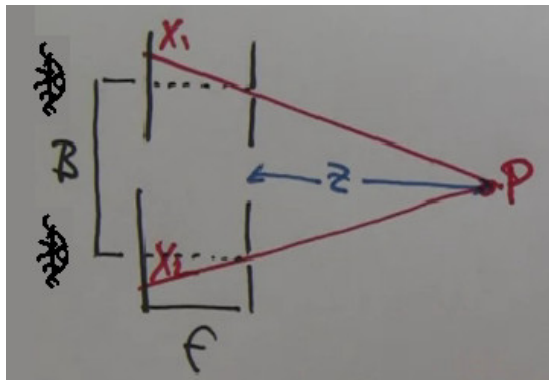
- What tasks would you find difficult if you had only one eye open?
- How do you think stereograms are made?





©www.easystereogrambuilder.com

Stereo Vision



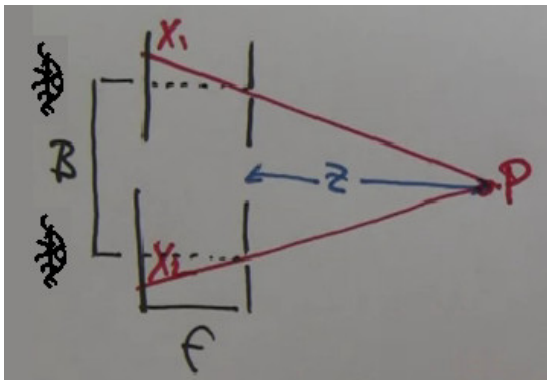
P : Target object.

Z : Distance to object.

B : Baseline; separation between eyes.

x_1, x_2 : Disparity or parallax; different offsets at each eye.

Stereo Vision



P : Target object.

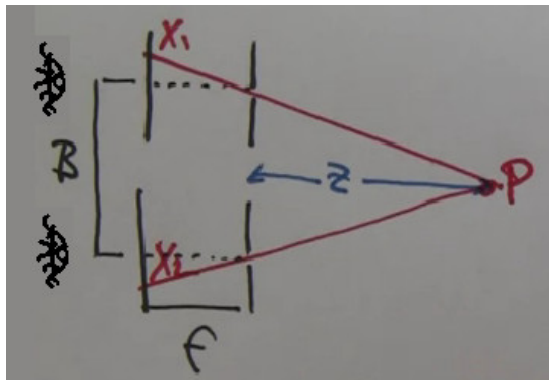
Z : Distance to object.

B : Baseline; separation between eyes.

x_1, x_2 : Disparity or parallax; different offsets at each eye.

Can we always find depth of P ?

Stereo Vision



P : Target object.

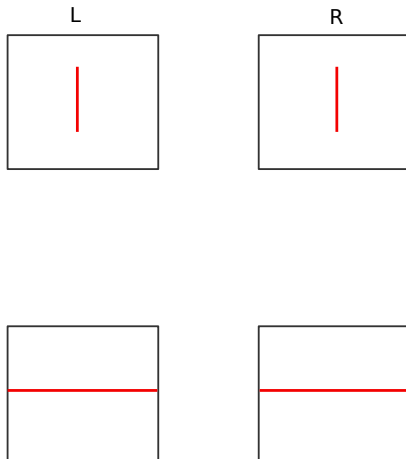
Z : Distance to object.

B : Baseline; separation between eyes.

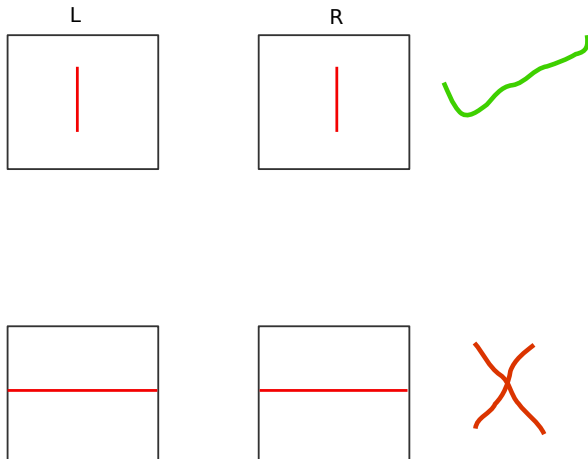
x_1, x_2 : Disparity or parallax; different offsets at each eye.

Can we always find depth of P ? No, only sometimes.

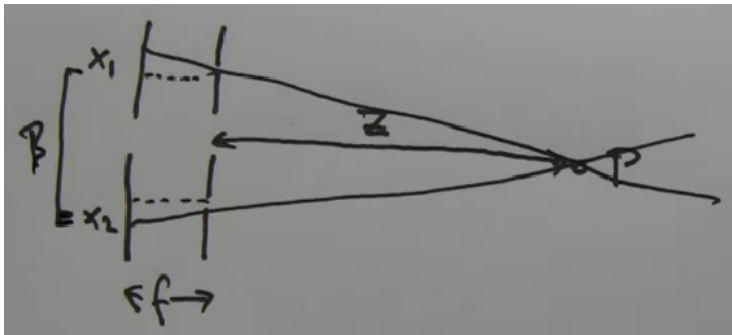
Stereo Vision: Which One is Easier?



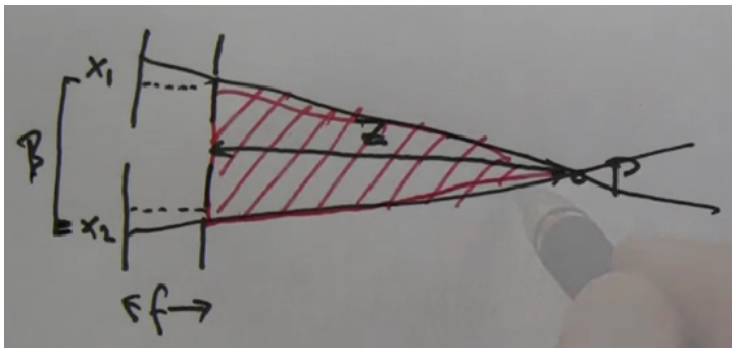
Stereo Vision: Which One is Easier?



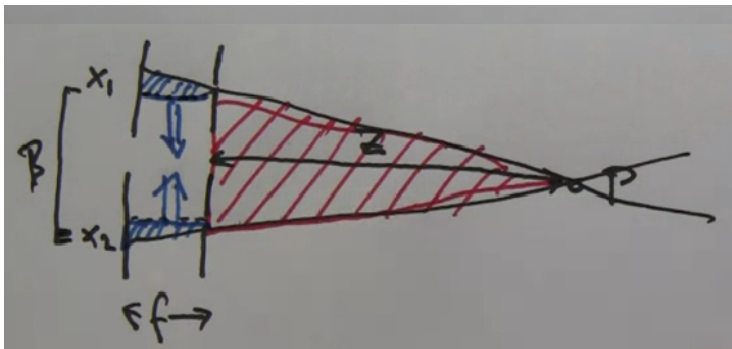
Stereo Vision: How to Find Depth?



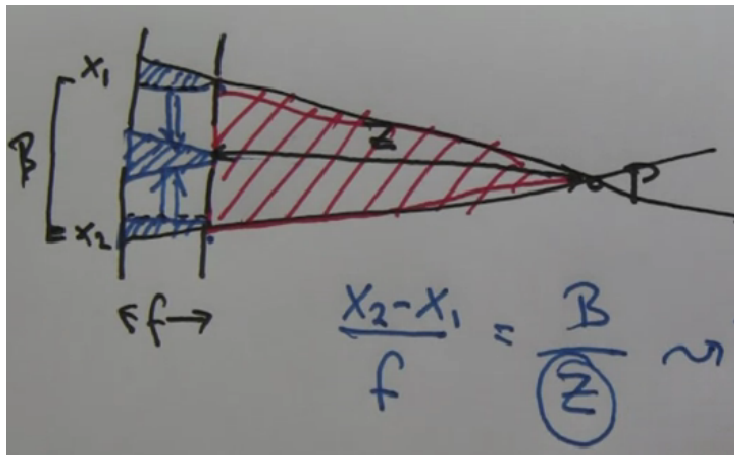
Stereo Vision: How to Find Depth?



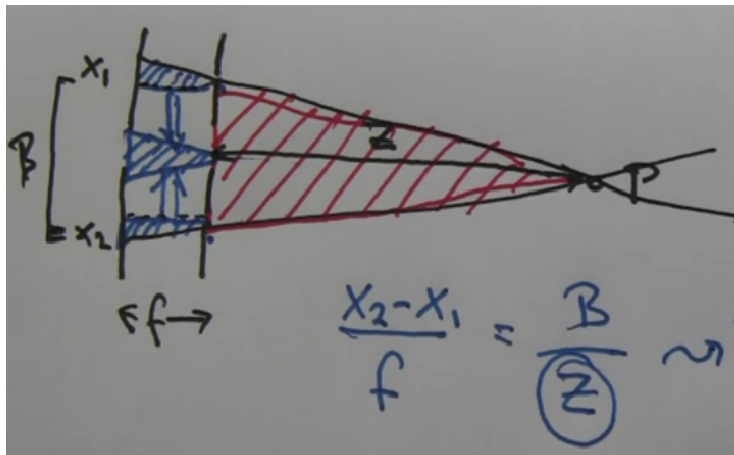
Stereo Vision: How to Find Depth?



Stereo Vision: How to Find Depth?

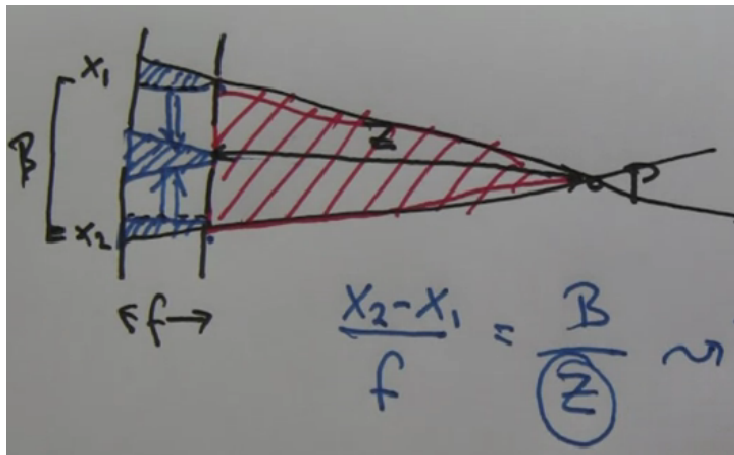


Stereo Vision: How to Find Depth?



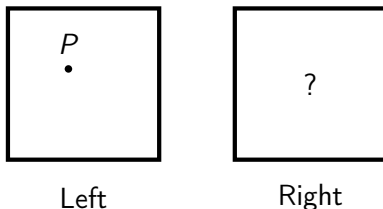
What's different here? What don't we need to find depth?

Stereo Vision: How to Find Depth?



What's different here? What don't we need to find depth? **Original size.**

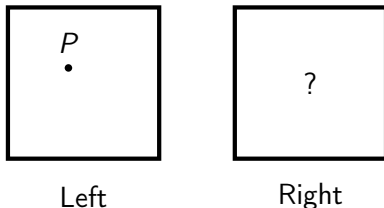
Finding Correspondence Between Left and Right Images



Where do we search on the right image?

- 1 2D: everywhere
- 2 1D: on a line
- 3 0D: we know the point

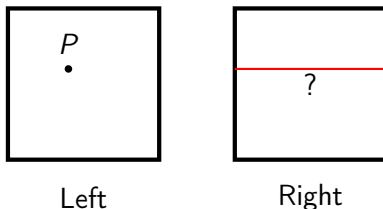
Finding Correspondence Between Left and Right Images



Where do we search on the right image?

- 1 2D: everywhere
- 2 **1D: on a line**
- 3 0D: we know the point

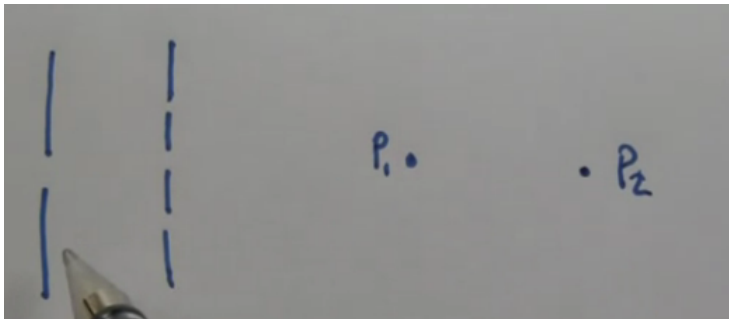
Finding Correspondence Between Left and Right Images



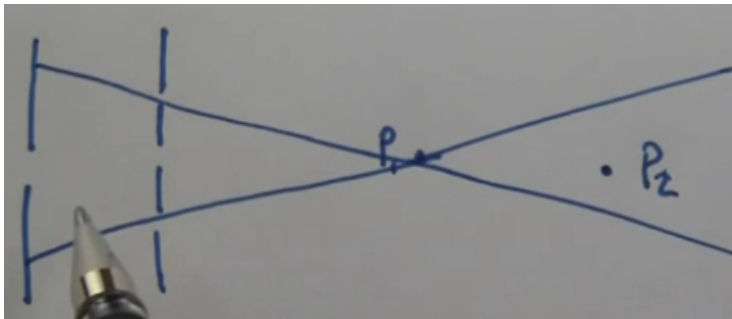
Where do we search on the right image?

- 1 2D: everywhere
- 2 **1D: on a line**
- 3 0D: we know the point

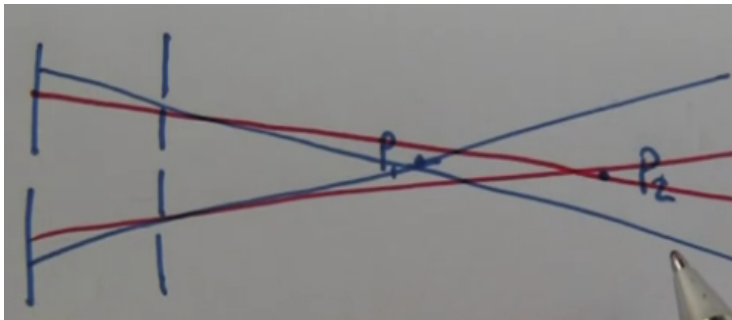
Correspondence Problems



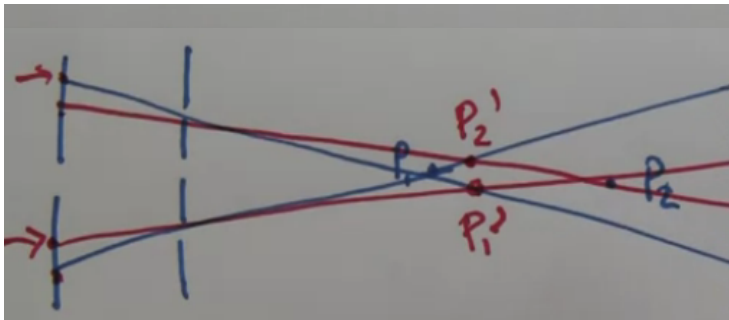
Correspondence Problems



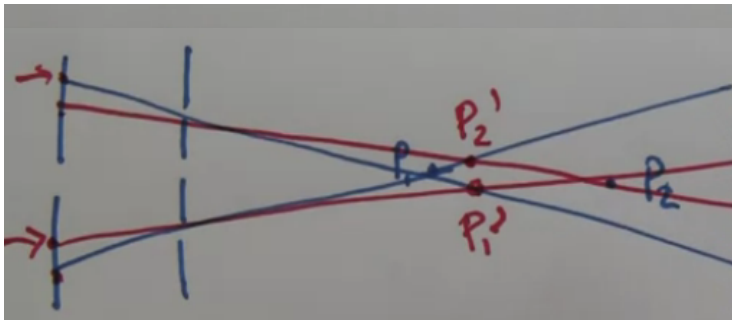
Correspondence Problems



Correspondence Problems



Correspondence Problems



You get **Phantom Points** if you get the correspondence wrong.

A Real Correspondence Example

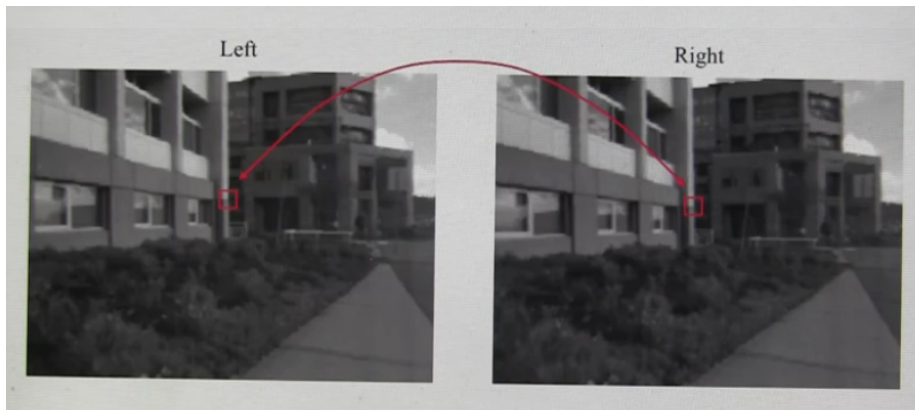
Left



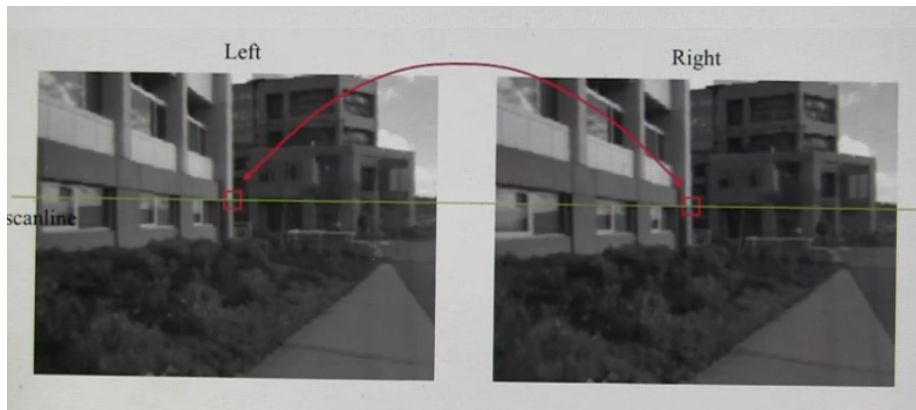
Right



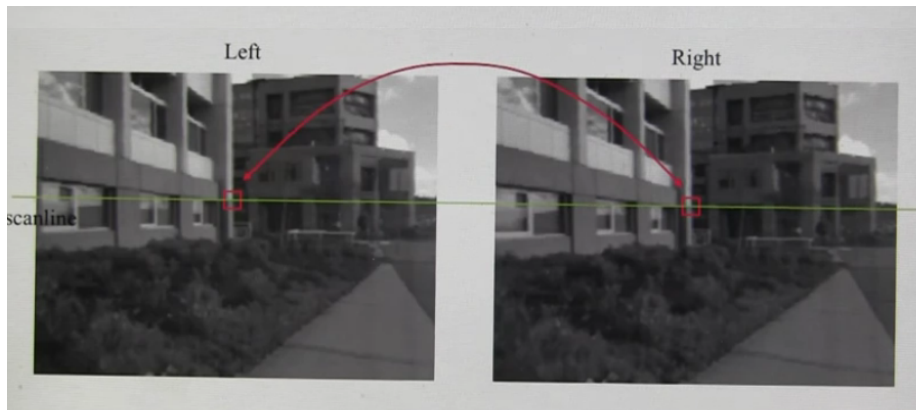
A Real Correspondence Example



A Real Correspondence Example



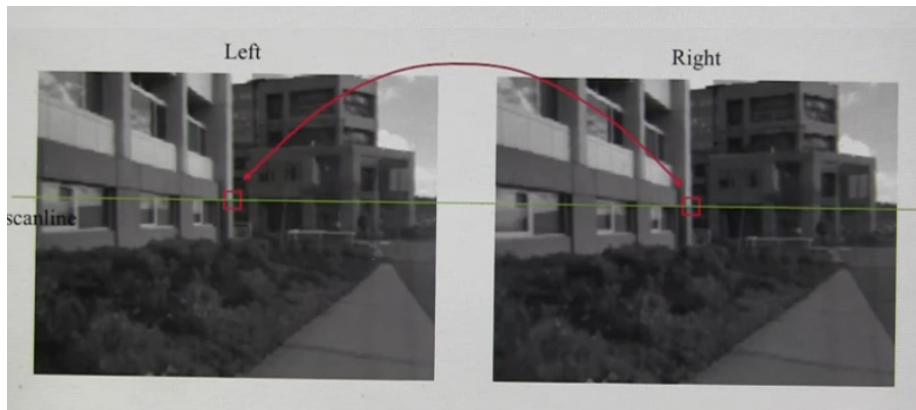
A Real Correspondence Example



Can we find correspondence with any of:

- 1 Texture match?
- 2 Feature match?

A Real Correspondence Example



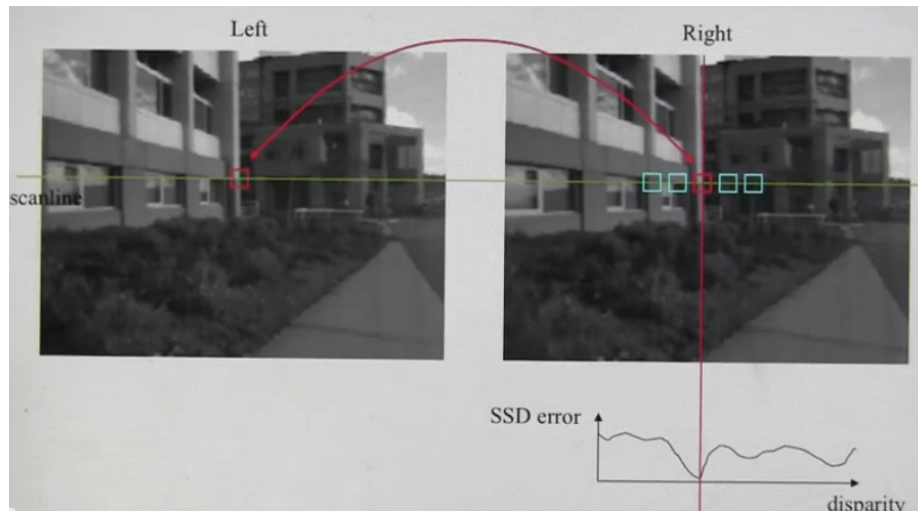
Can we find correspondence with any of:

- 1 Texture match?
- 2 Feature match?

Both, actually.

Texture Match with SSD

SSD is not **solid state drive**, but it is **sum of squared distance**

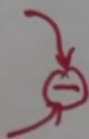


Sum of Squared Distance (SSD)

SSD minimization

L → normalize

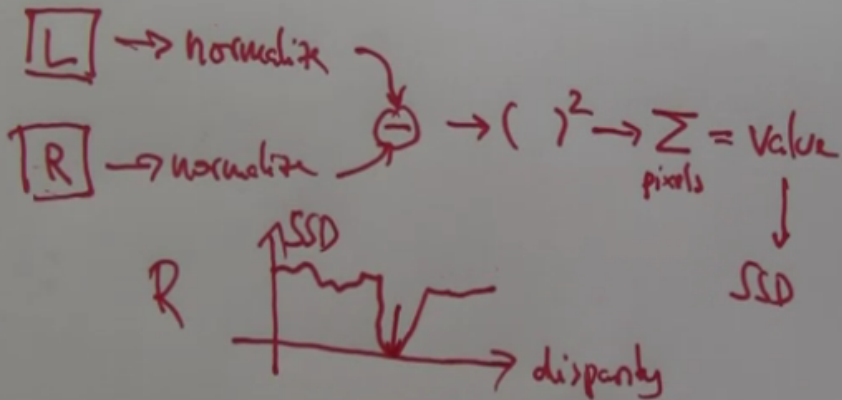
R → normalize



$()^2 \rightarrow \sum = \text{Value}$
pixels

Sum of Squared Distance (SSD)

SSD minimization



The Result: Disparity Maps

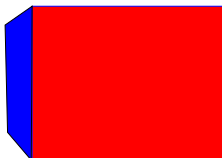
Left



Disparity Map



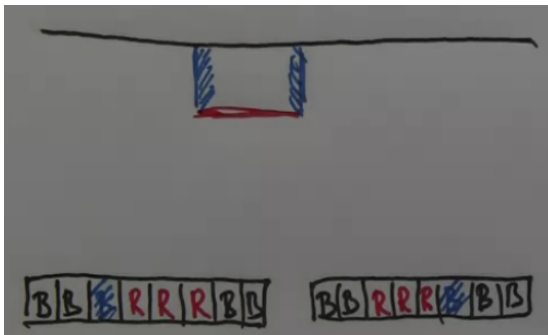
How About Occlusions?



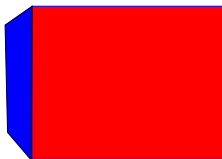
Left



Right



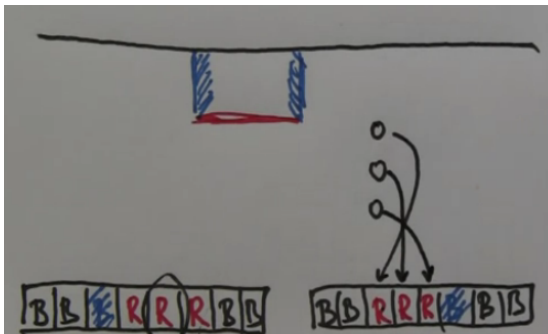
How About Occlusions?



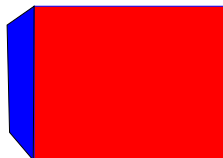
Left



Right



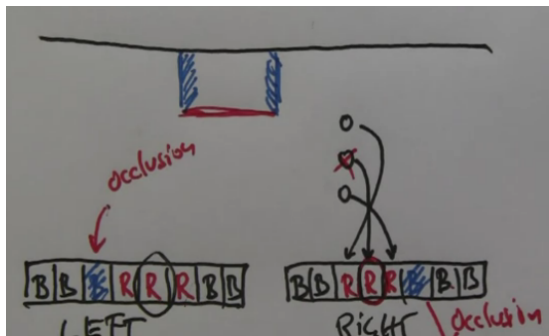
How About Occlusions?



Left



Right



Using Cost to Optimize Correspondence

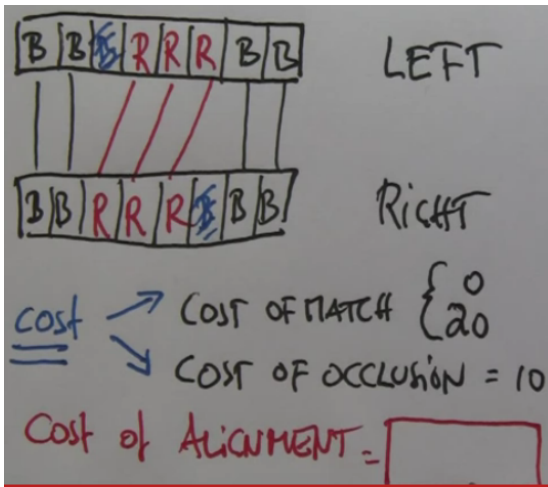
LEFT: B B ~~B~~ R R R B B

RIGHT: B B R R R ~~B~~ B B

cost \rightarrow COST OF MATCH $\begin{cases} 0 \\ 20 \end{cases}$

\searrow COST OF OCCLUSION = 10

Using Cost to Optimize Correspondence



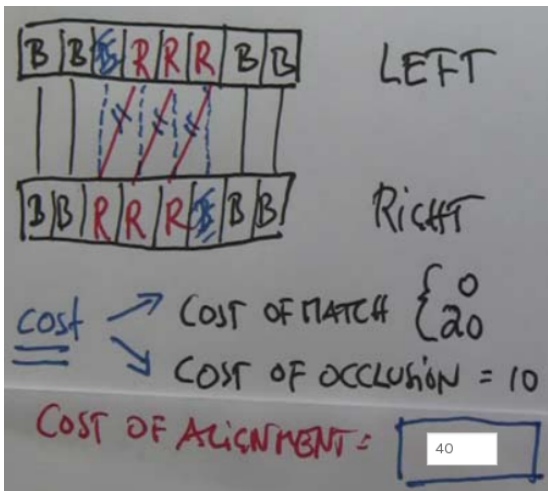
Using Cost to Optimize Correspondence

LEFT
RIGHT

cost \rightarrow COST OF MATCH $\begin{cases} 0 \\ 20 \end{cases}$
 \searrow COST OF OCCLUSION = 10

Cost of ALIGNMENT = 20

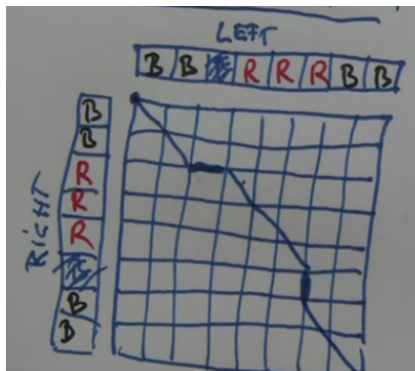
Using Cost to Optimize Correspondence



So How to Compute Best Alignment?

Use **dynamic programming**:

- calculate **correspondence matrix** with $O(n^2)$:



So How to Compute Best Alignment?

Use **dynamic programming**:

- calculate **correspondence matrix** with $O(n^2)$:



Does this look familiar?

So How to Compute Best Alignment?

Use **dynamic programming**:

- calculate **correspondence matrix** with $O(n^2)$:



Does this look familiar?
Can we use MDP?

So How to Compute Best Alignment?

Use **dynamic programming**:

- calculate **correspondence matrix** with $O(n^2)$:



Does this look familiar?

Can we use MDP?

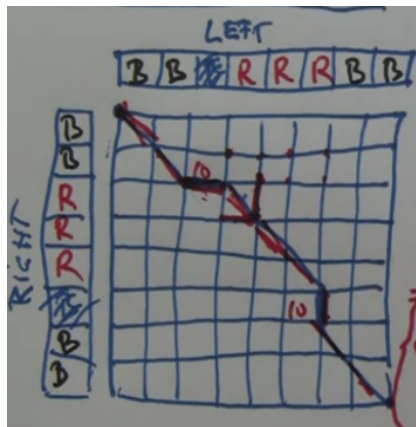
$V(i, j) =$

$$\max \begin{cases} \text{match}(i, j) + V(i-1, j-1) \\ \text{occl}(i, j) + V(i-1, j) \\ \text{occl}(i, j) + V(i, j-1) \end{cases}$$

So How to Compute Best Alignment?

Use **dynamic programming**:

- calculate **correspondence matrix** with $O(n^2)$:



Does this look familiar?

Can we use MDP?

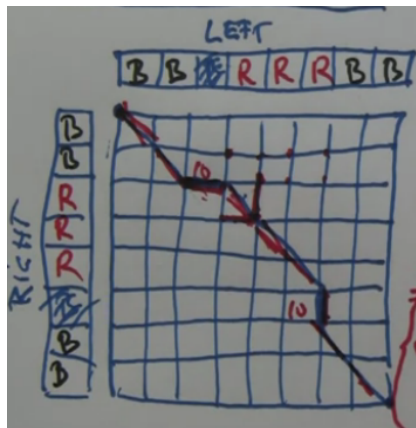
$V(i, j) =$

$$\max \begin{cases} \text{match}(i, j) + V(i - 1, j - 1) \\ \text{occl}(i, j) + V(i - 1, j) \\ \text{occl}(i, j) + V(i, j - 1) \end{cases}$$

So How to Compute Best Alignment?

Use **dynamic programming**:

- calculate **correspondence matrix** with $O(n^2)$:



Does this look familiar?

Can we use MDP?

$V(i, j) =$

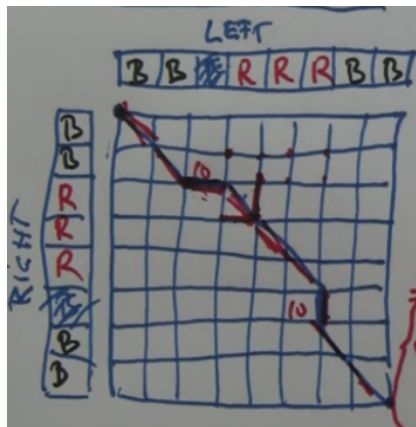
$$\max \begin{cases} \text{match}(i, j) + V(i - 1, j - 1) \\ \text{occl}(i, j) + V(i - 1, j) \\ \text{occl}(i, j) + V(i, j - 1) \end{cases}$$

State-of-the-art in computer vision!

So How to Compute Best Alignment?

Use **dynamic programming**:

- calculate **correspondence matrix** with $O(n^2)$:



Does this look familiar?

Can we use MDP?

$V(i, j) =$

$$\max \begin{cases} \text{match}(i, j) + V(i - 1, j - 1) \\ \text{occl}(i, j) + V(i - 1, j) \\ \text{occl}(i, j) + V(i, j - 1) \end{cases}$$

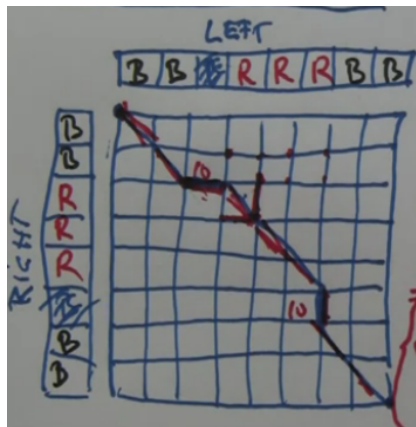
State-of-the-art in computer vision!

How would the brain do it?

So How to Compute Best Alignment?

Use **dynamic programming**:

- calculate **correspondence matrix** with $O(n^2)$:



Does this look familiar?

Can we use MDP?

$$V(i, j) = \max \begin{cases} \text{match}(i, j) + V(i - 1, j - 1) \\ \text{occl}(i, j) + V(i - 1, j) \\ \text{occl}(i, j) + V(i, j - 1) \end{cases}$$

State-of-the-art in computer vision!

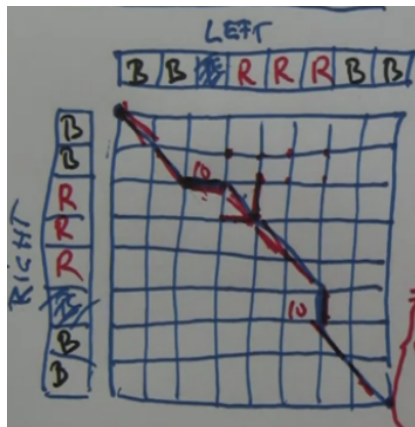
How would the brain do it?

- In parallel, each node in matrix a separate neuron
- How fast?

So How to Compute Best Alignment?

Use **dynamic programming**:

- calculate **correspondence matrix** with $O(n^2)$:



Does this look familiar?

Can we use MDP?

$V(i, j) =$

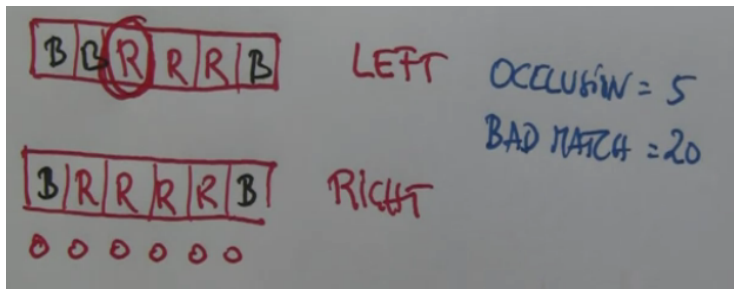
$$\max \begin{cases} \text{match}(i, j) + V(i - 1, j - 1) \\ \text{occl}(i, j) + V(i - 1, j) \\ \text{occl}(i, j) + V(i, j - 1) \end{cases}$$

State-of-the-art in computer vision!

How would the brain do it?

- In parallel, each node in matrix a separate neuron
- How fast? $O(1)$

Alignment Questions



Alignment Questions

Diagram illustrating sequence alignment between two strings: "B B R R R B" (top) and "B R R R R B" (bottom). The 'R' in the top string is circled. The diagram is labeled "LEFT" and "RIGHT". To the right, it specifies "OCCLUSION = 5" and "BAD MATCH = 20". Below the strings are seven small circles representing a sequence of elements.

Alignment Questions

B|R(B|B|B|B L

B|B|B|B|R|B R

○ ○ ○ ○ ● ●

OCCLUSION = 60
BAD MATCH = 20

Alignment Questions

B R B B B L

B B B B R B R

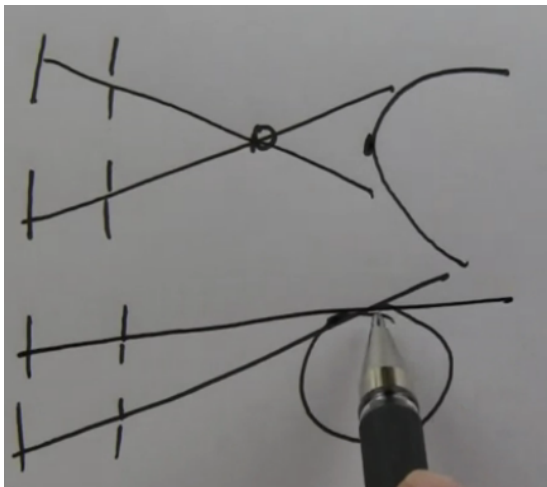
OCCLUSION = 60

BAD MATCH = 20

● ● ● ● ● ● ●

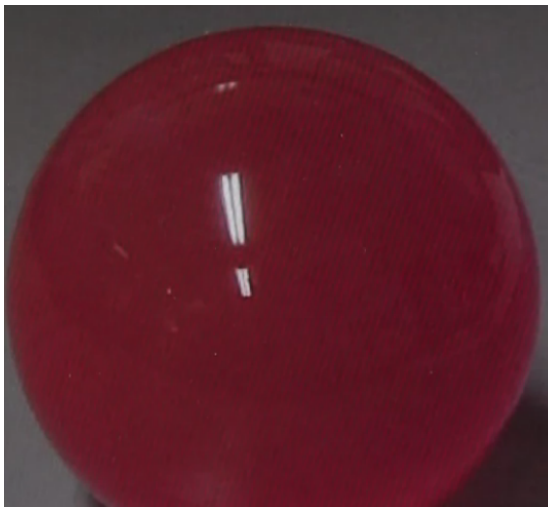
Problem Cases for Alignment

Problems with: Foreground-background separation and circular edges.



Problem Cases for Alignment

Problems with: Reflection.



New Technologies

