Cyclope **vs.** **Odysseus**

Stereo

Kari Pulli

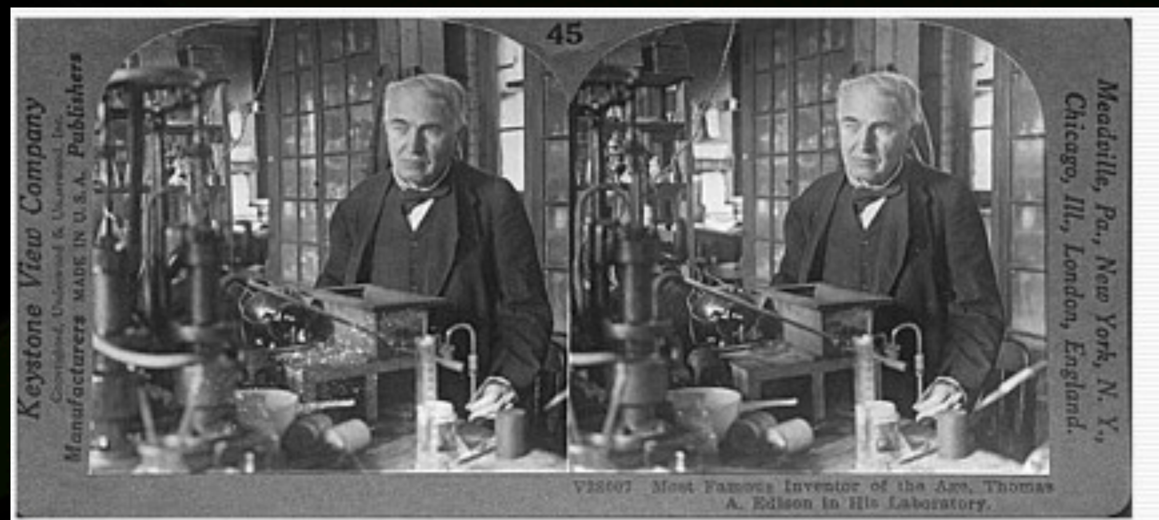
Senior Director

NVIDIA Research

NVIDIA

Binocular stereo

- Given a calibrated binocular stereo pair
  - fuse it to produce a depth image
  - humans can do it

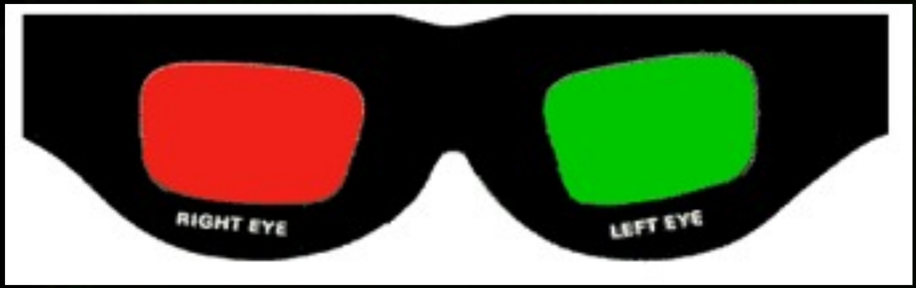


Stereograms: Invented by Sir Charles Wheatstone, 1838

Anaglyphs

- Two images of complementary color are overlaid
- Glasses required
  - red filter cancels out red image component
  - green filter cancels out green component
- Each eye gets one image
  - 3D impression
- Current 3D movie theaters use this principle
  - except they use polarization filters instead of color filters

(Anaglyph Image)



(Red/Green Glasses)

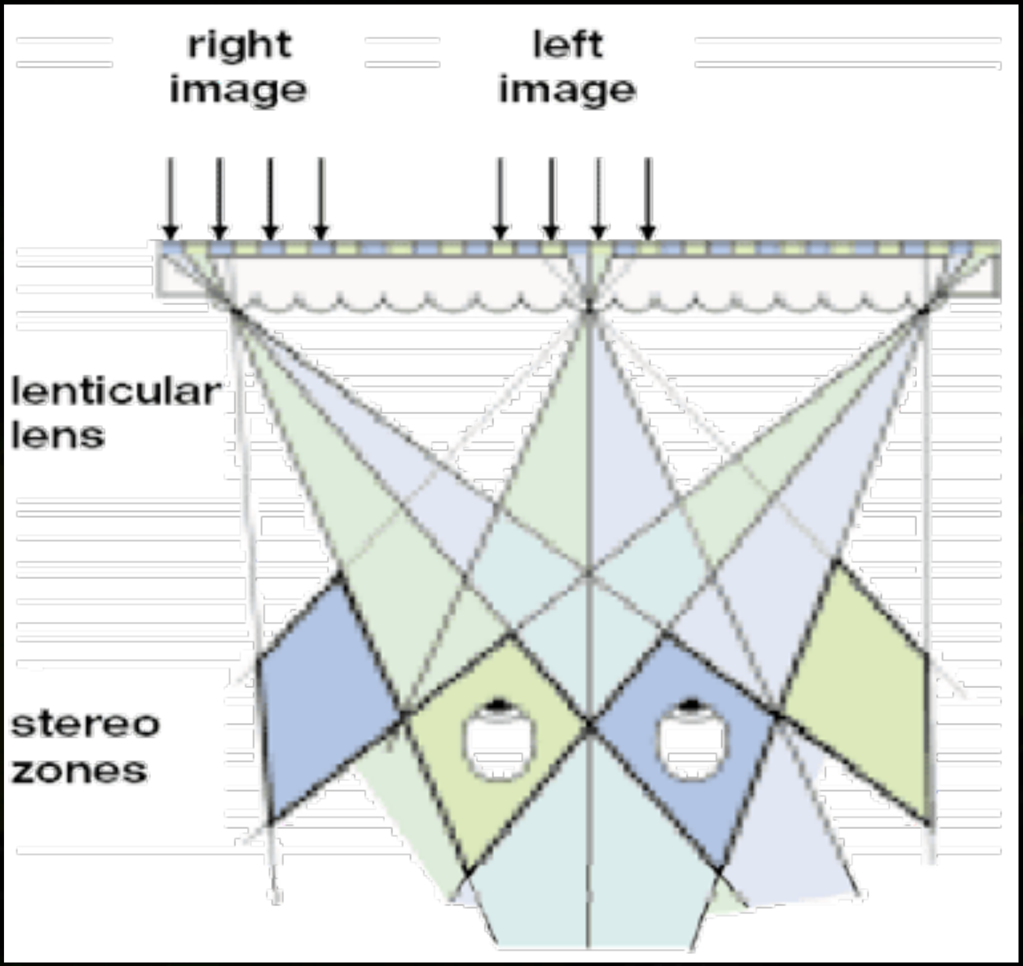
Shutter Glasses

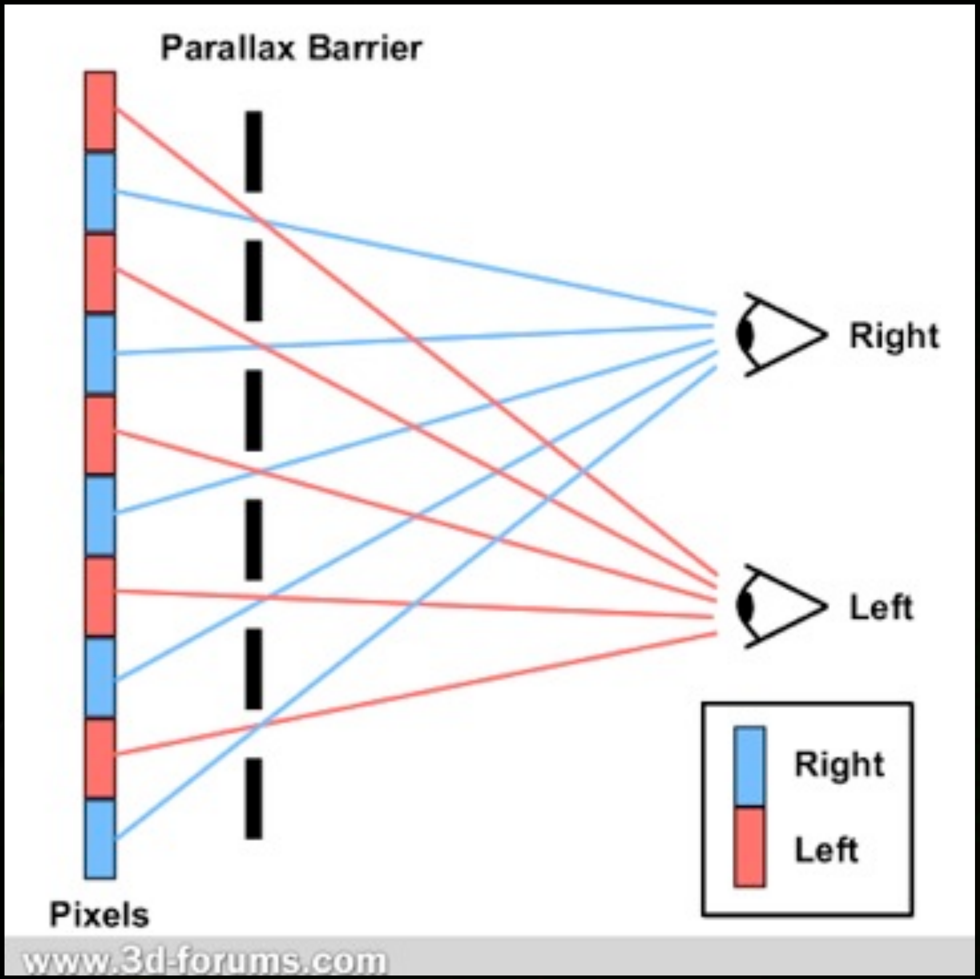
- Display flickers between left and right image
  - i.e. each even frame shows left image, each odd frame shows right image
- When left frame is shown, shutter glasses close right eye and vice versa
- Requires new displays of high frame rate (120Hz)

(Shutter Glasses and 120 Hz Display)

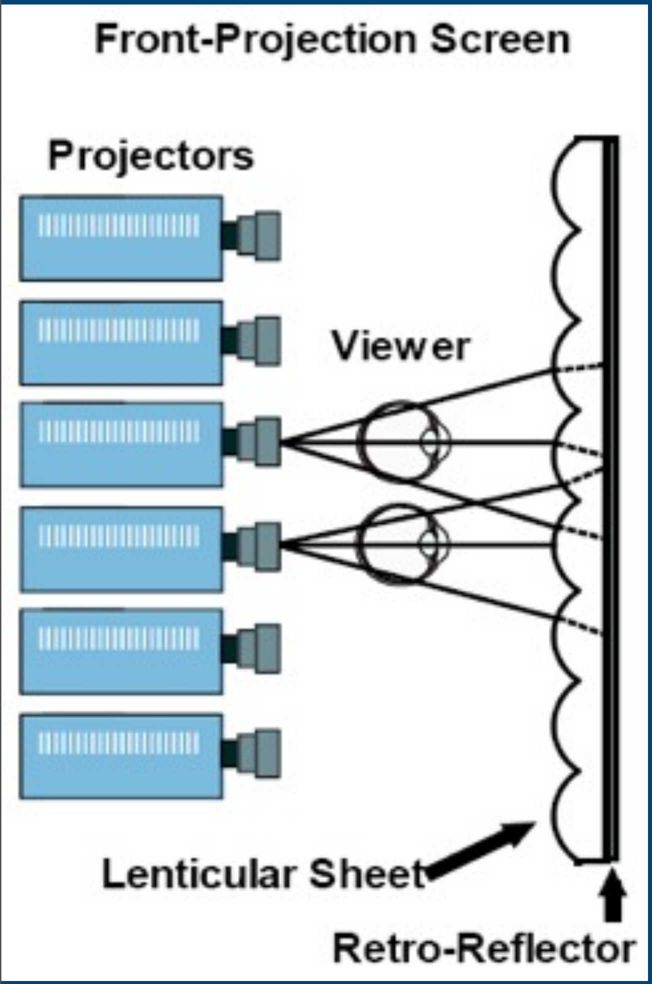
(NVIDIA Artwork)

Autostereoscopic Displays





Camera, Display Array: MERL 3D-TV System

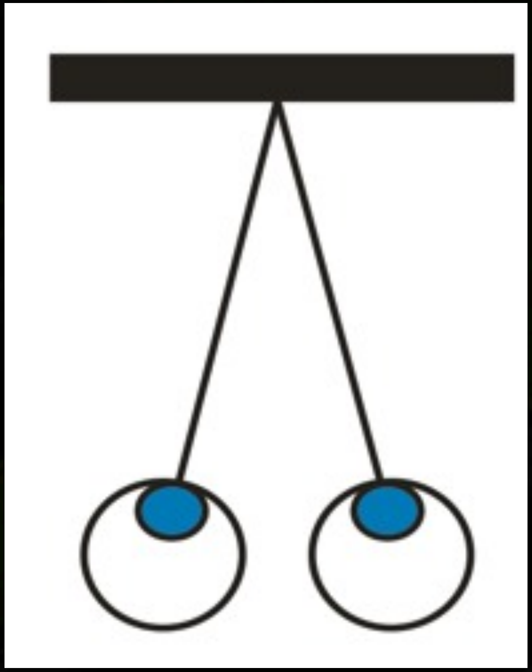


- 16 camera Horizontal Array
- Auto-Stereoscopic Display
- Efficient coding, transmission

Anthony Vetro, Wojciech Matusik, Hanspeter Pfister, and Jun Xin

Free Viewing (No glasses required, but some practice)

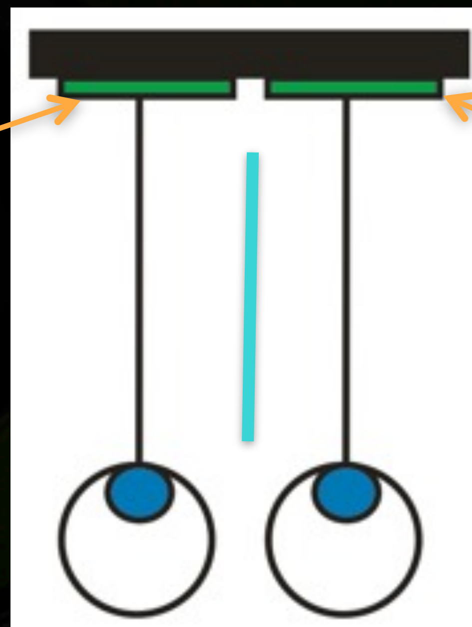
- The way how you usually look at the display (no 3D):



Free Viewing (No glasses required, but some practice)

- Parallel Viewing:

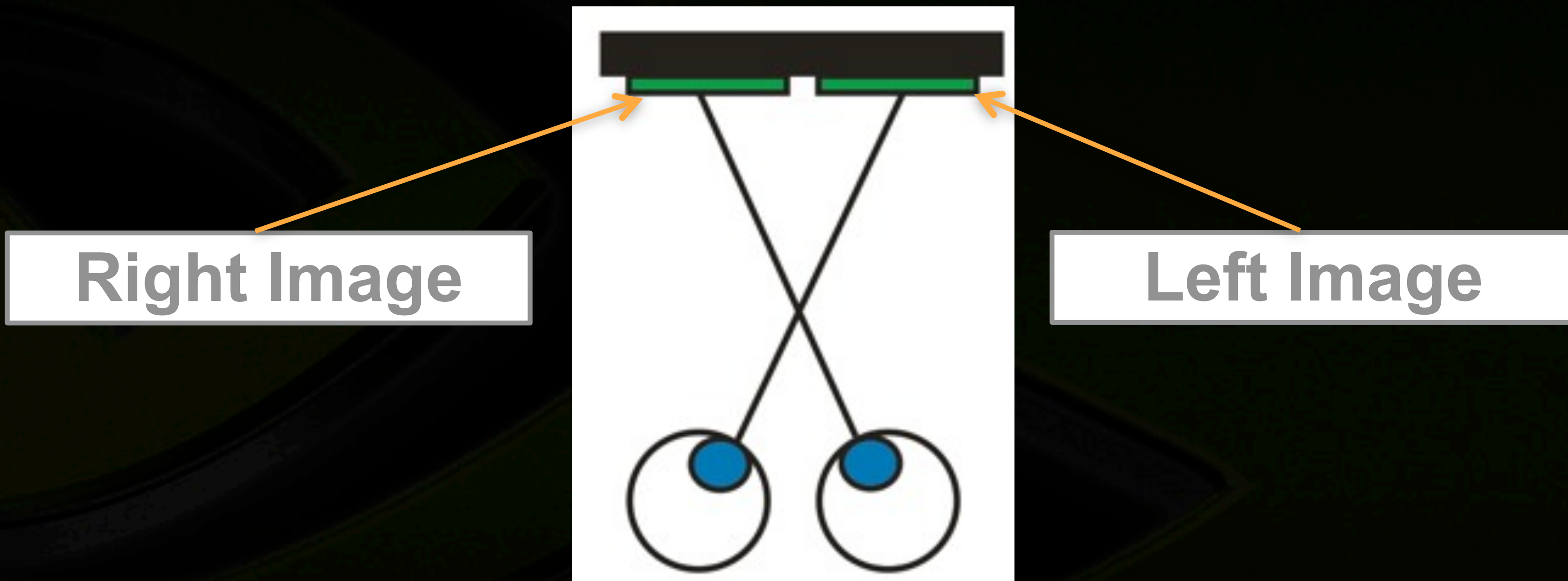
Left Image



Right Image

Free Viewing (No glasses required, but some practice)

- Cross Eye Viewing:



- Most likely the simpler method

Learning Cross Eye Viewing

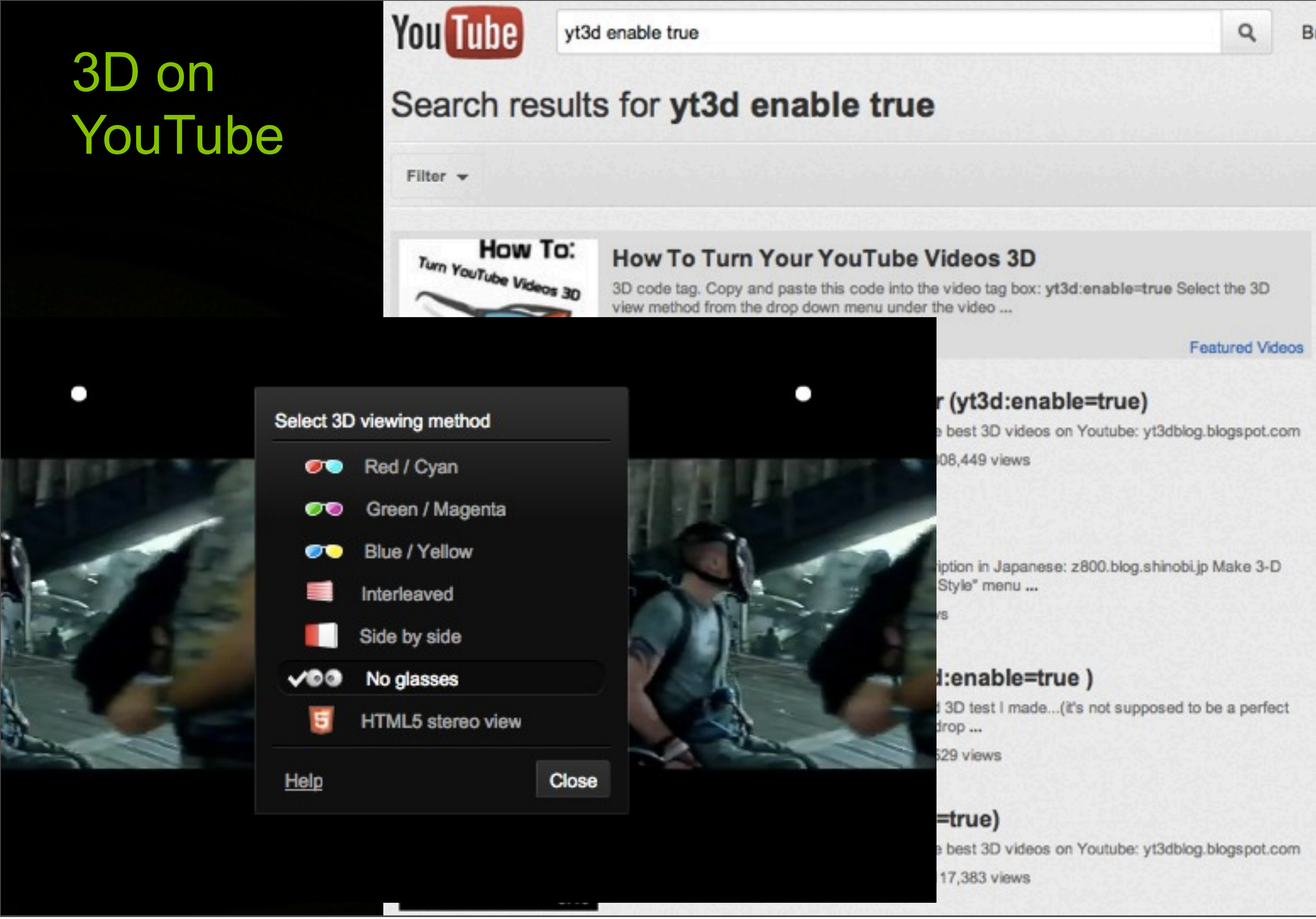
Right 2D Image

Left 2D Image

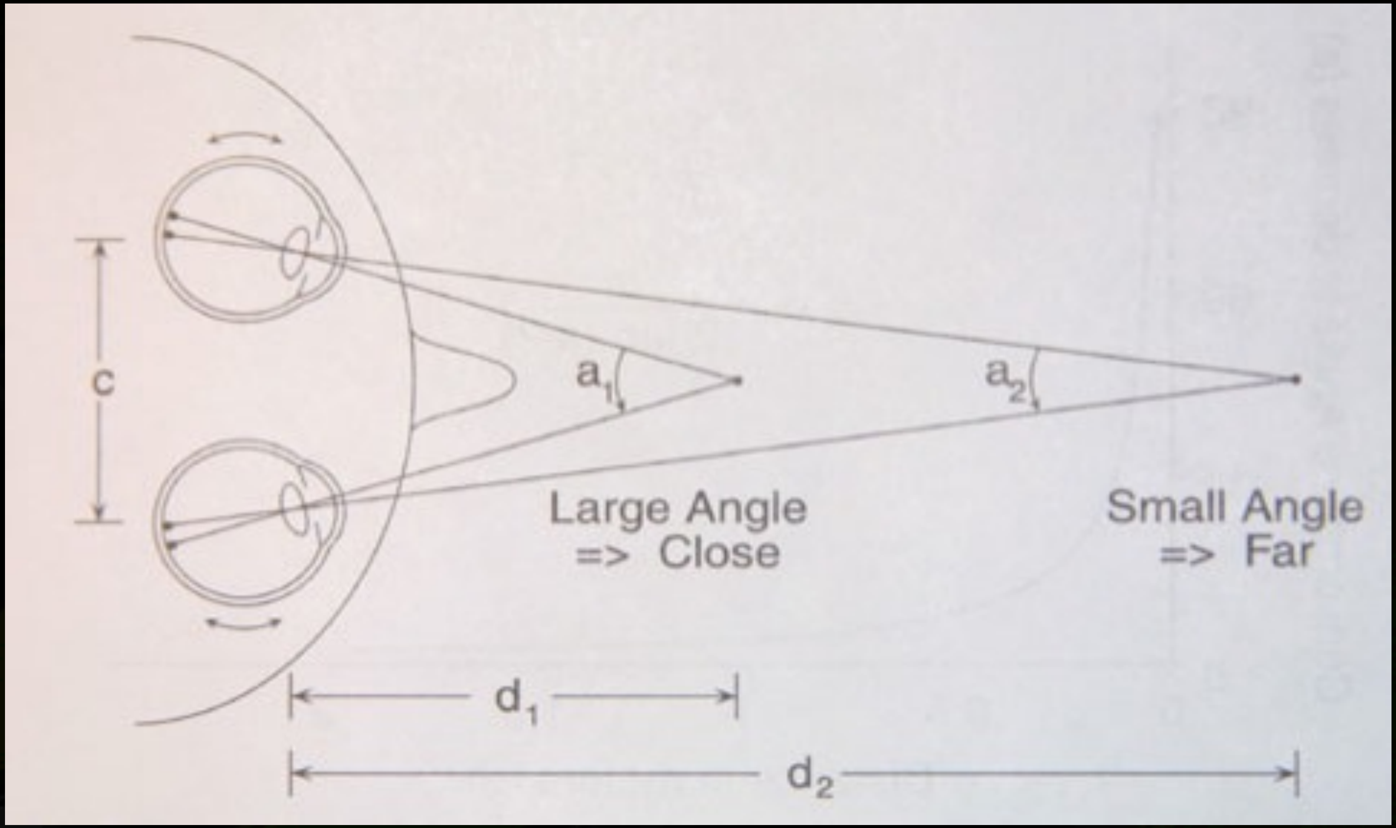
- Hold a finger in the middle of your eyes
- Look at the finger and slowly change its distance
- If you found the right distance
  - you see a third image in between left and right images
  - look at the third image (instead of the finger), it is in 3D

Monocular stereo

3D on YouTube

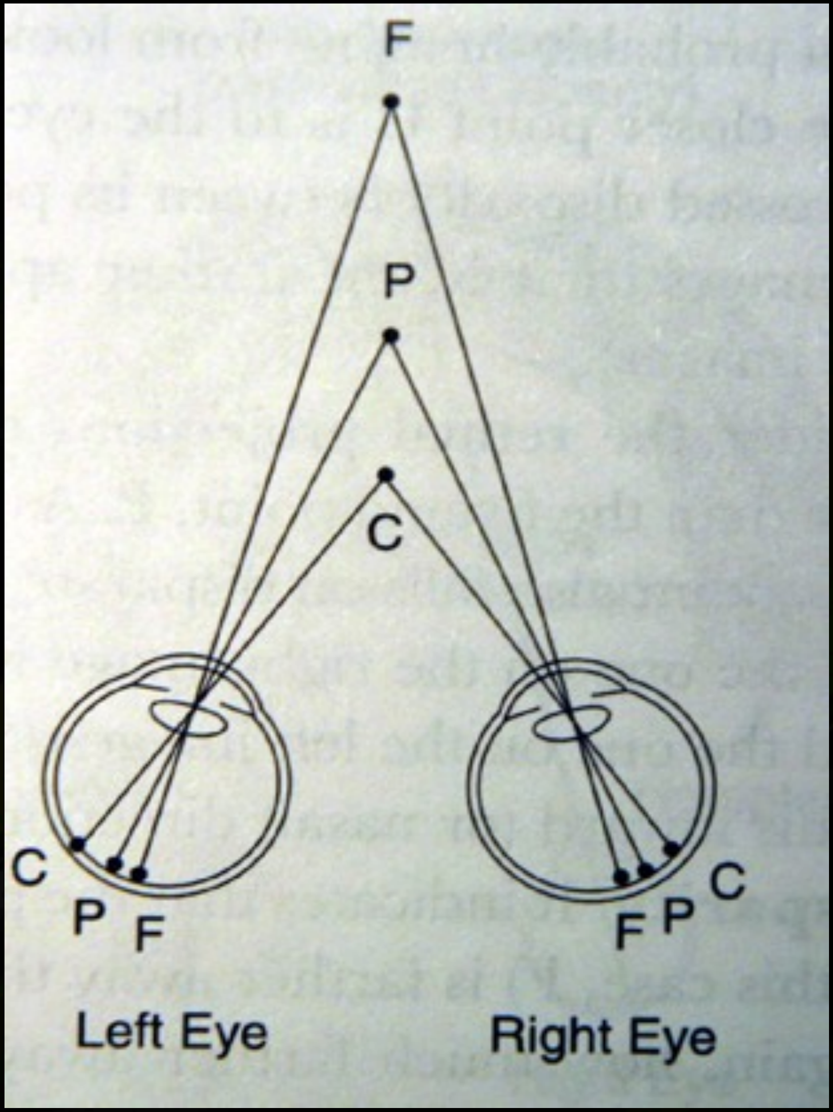


Depth from Convergence



Human performance: up to 2-3 meters

Depth from binocular disparity



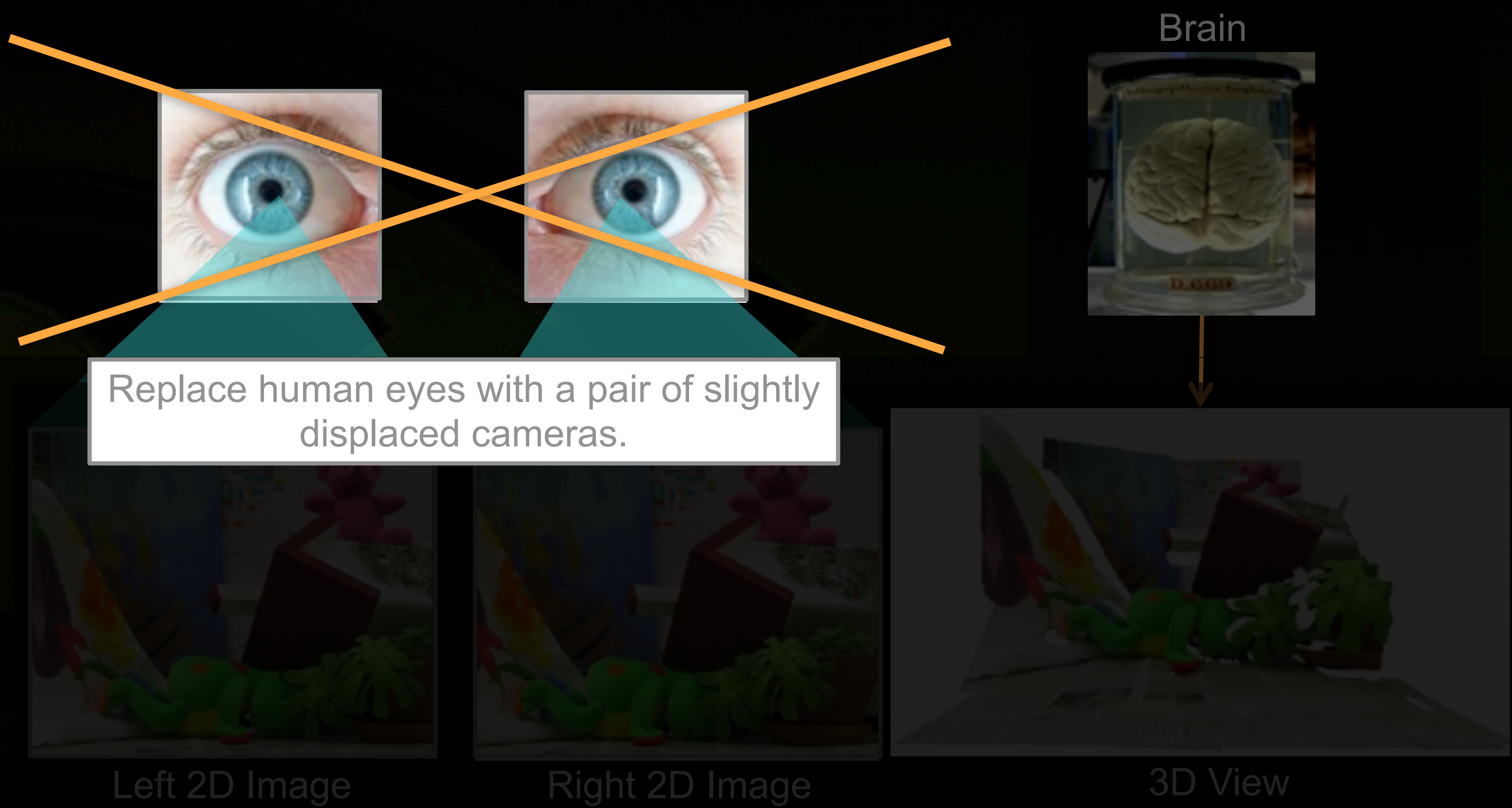
Sign and magnitude of disparity

P: converging point

C: object nearer projects to the outside of the P, disparity = +

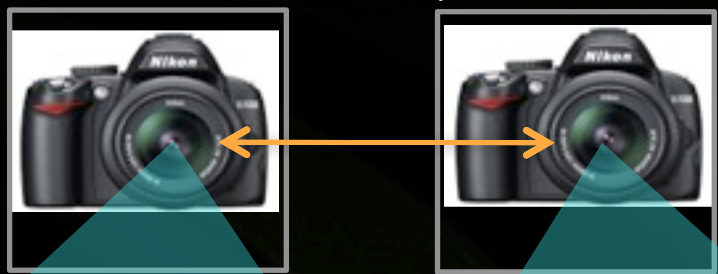
F: object farther projects to the inside of the P, disparity = –

Computational Stereo

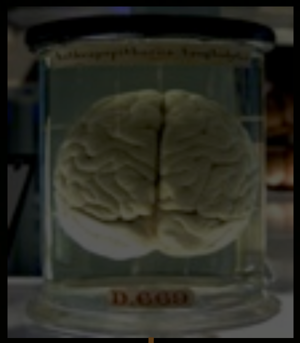


Computational Stereo

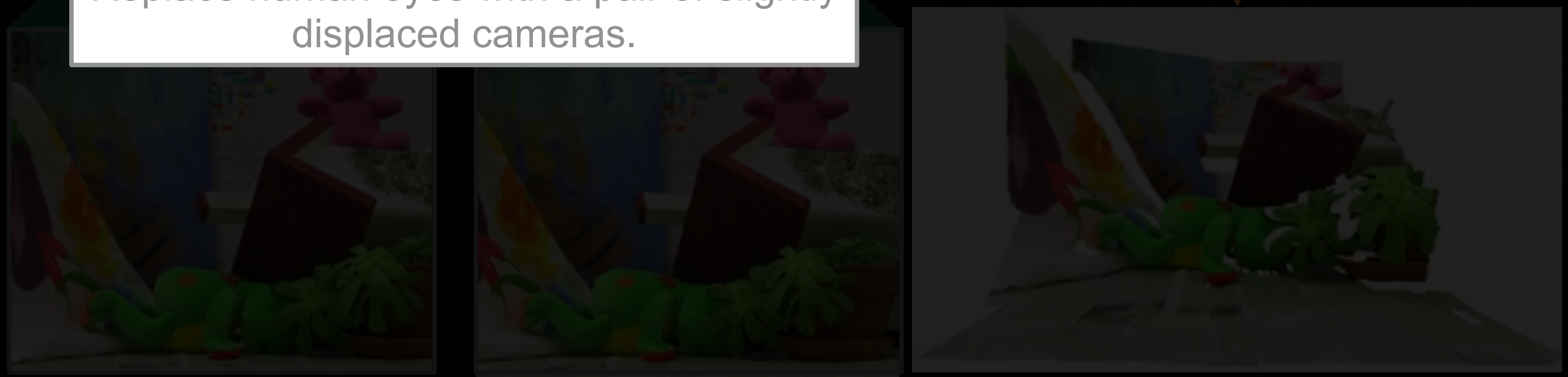
Displacement (Stereo Baseline)



Brain



Replace human eyes with a pair of slightly displaced cameras.

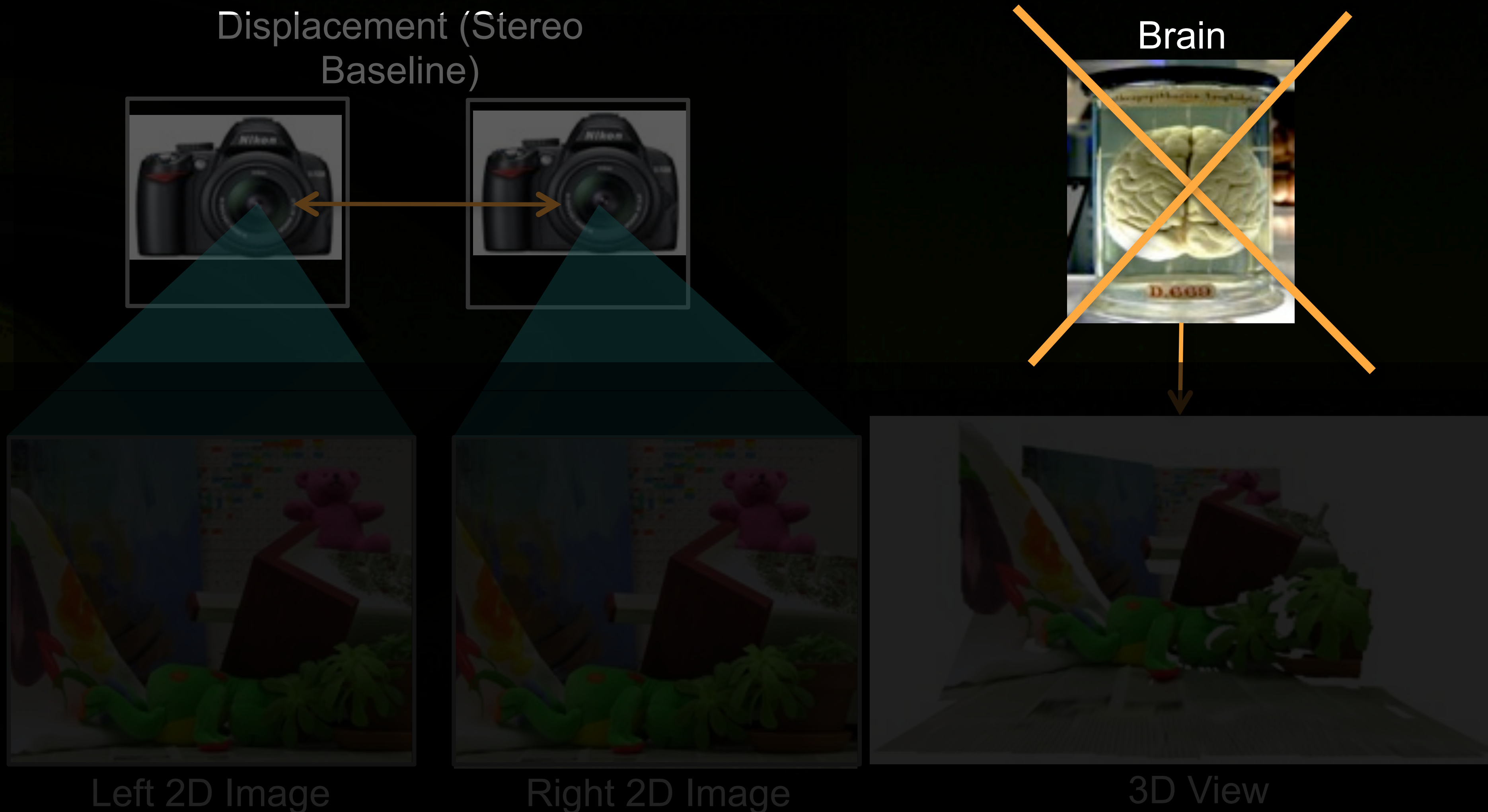


Left 2D Image

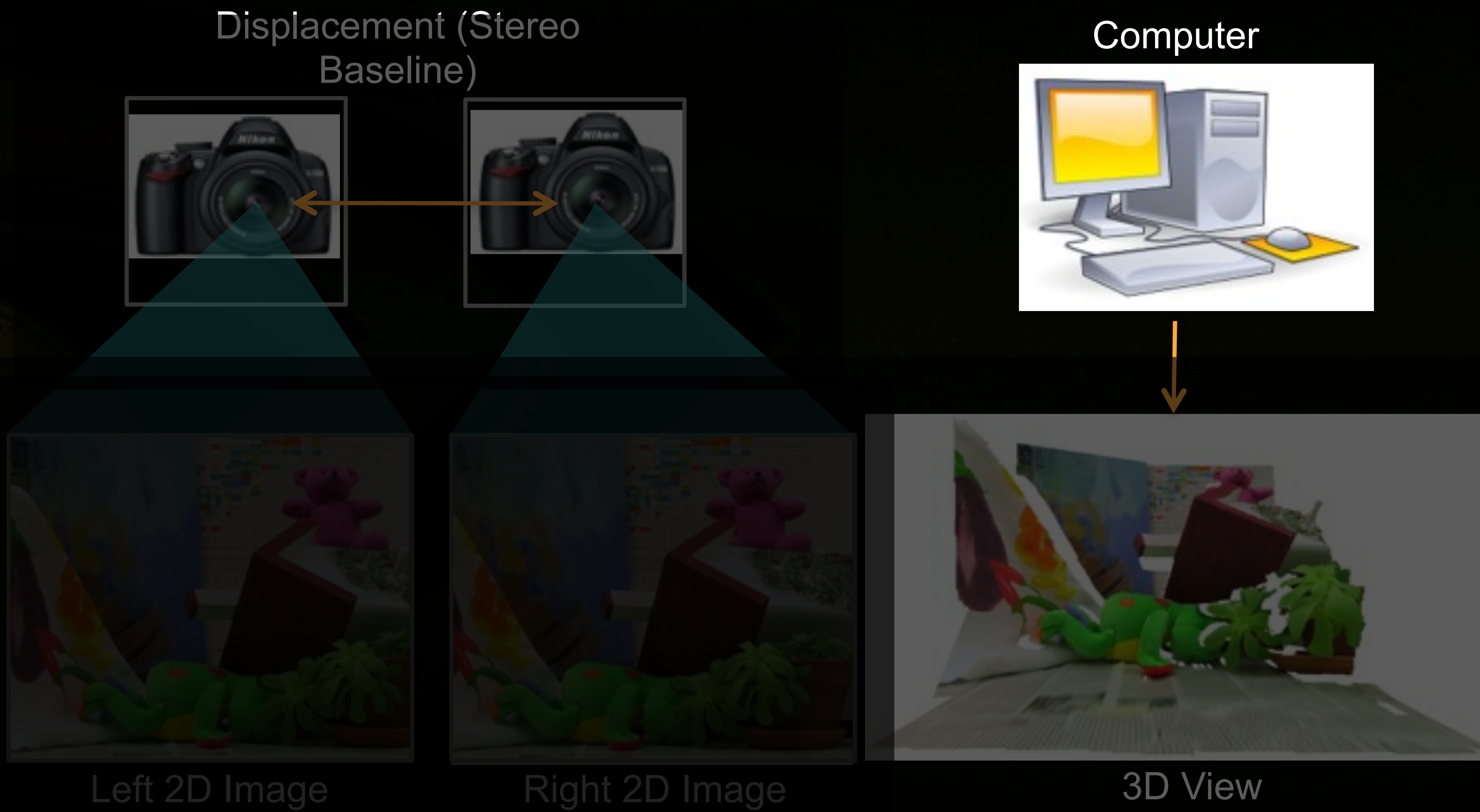
Right 2D Image

3D View

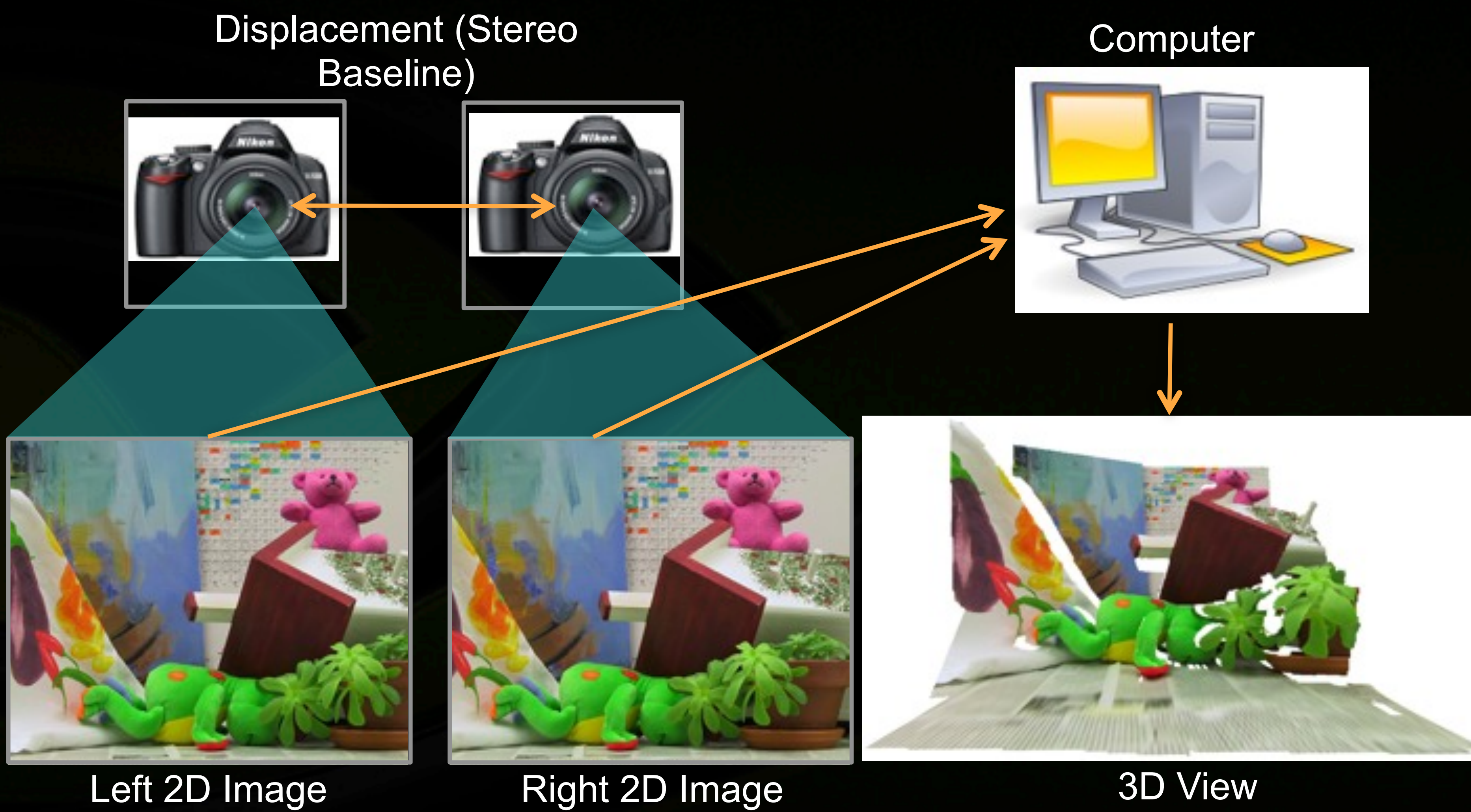
Computational Stereo



Computational Stereo



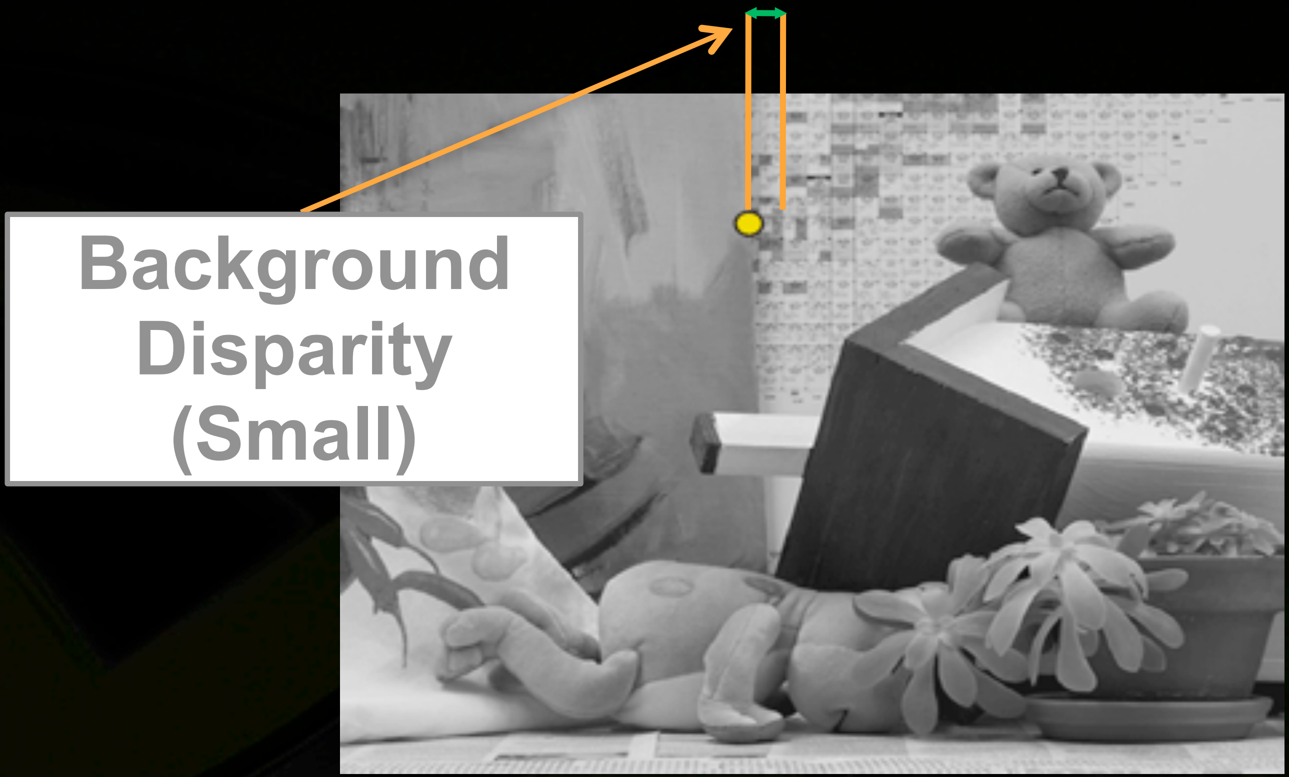
Computational Stereo



What is Disparity?

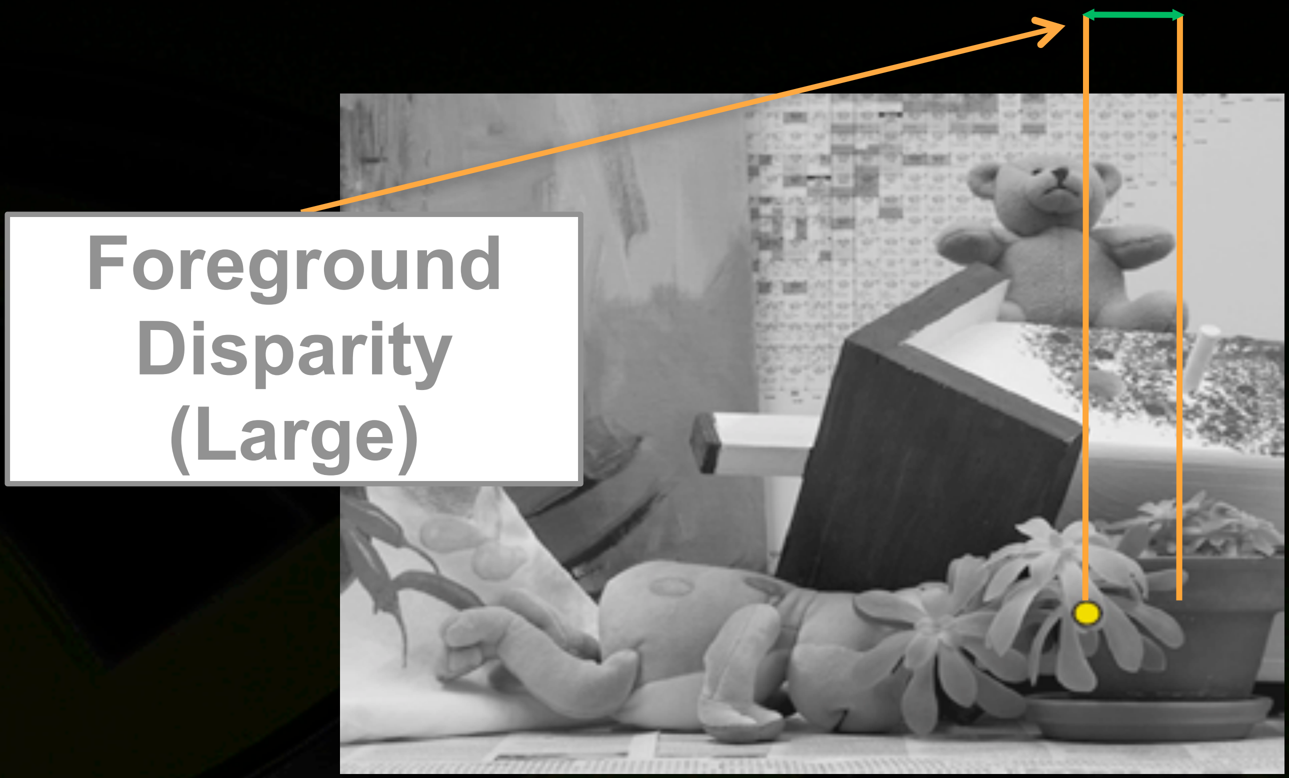
- The amount to which a single pixel is displaced in the two images is called disparity
- A pixel's disparity is inversely proportional to its depth in the scene

What is Disparity?



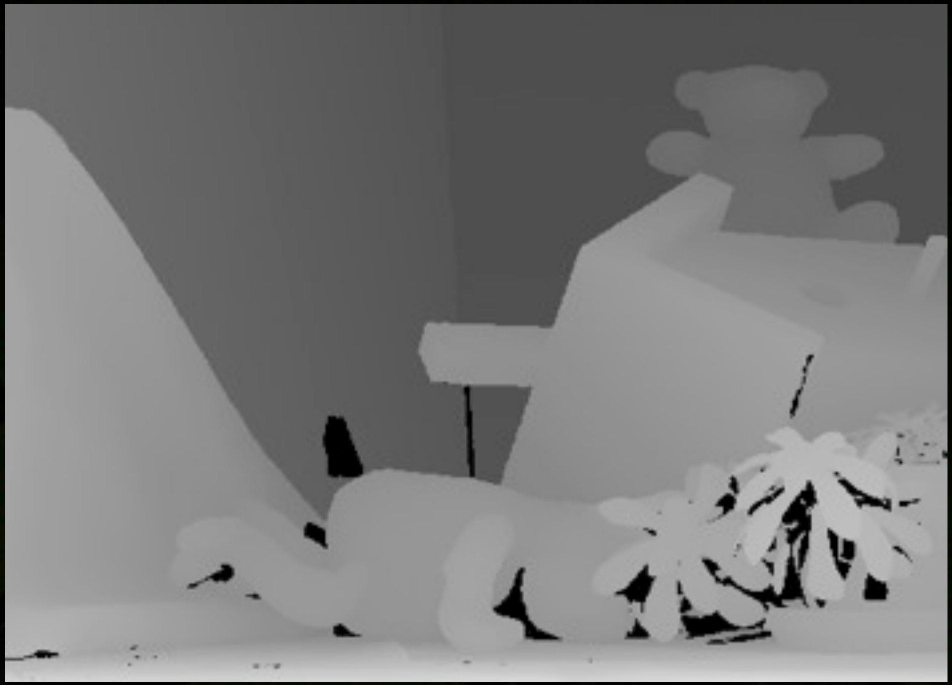
- The amount to which a single pixel is displaced in the two images is called disparity
- A pixel's disparity is inversely proportional to its depth in the scene

What is Disparity?



- The amount to which a single pixel is displaced in the two images is called disparity
- A pixel's disparity is inversely proportional to its depth in the scene

Disparity Encoding

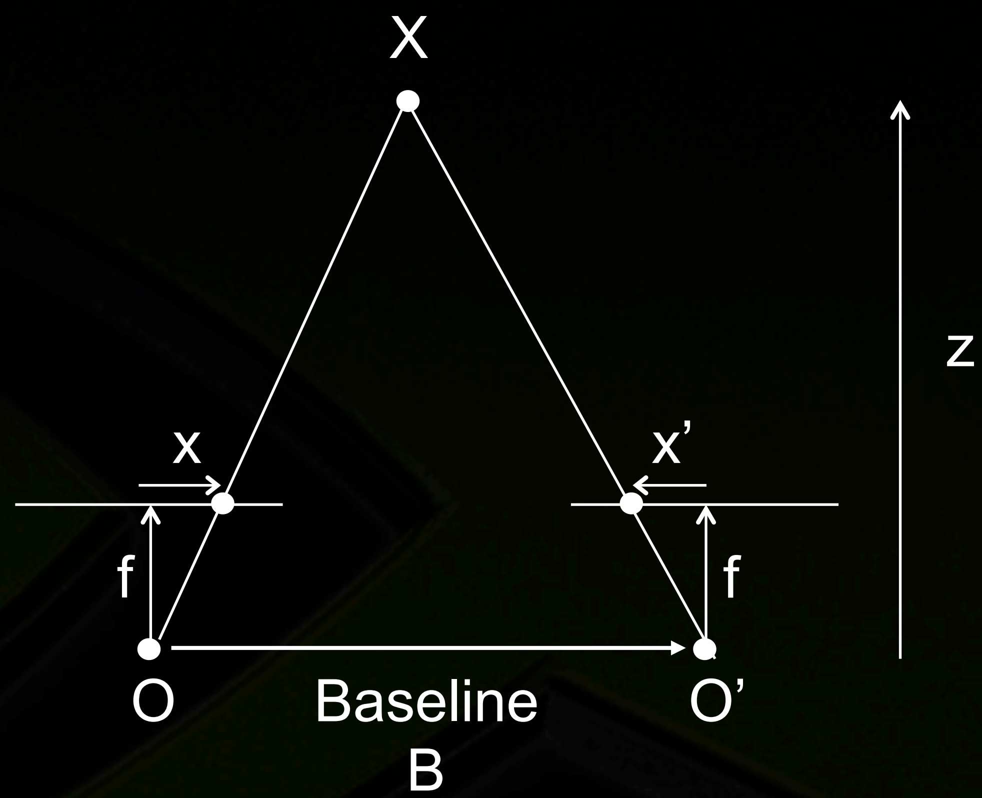


- The disparity of each pixel is encoded by a gray value
- Light gray values represent high disparities
  - and dark gray values small disparities
- The resulting image is called disparity map

Disparity and Depth

- The disparity map contains sufficient information for generating a 3D model

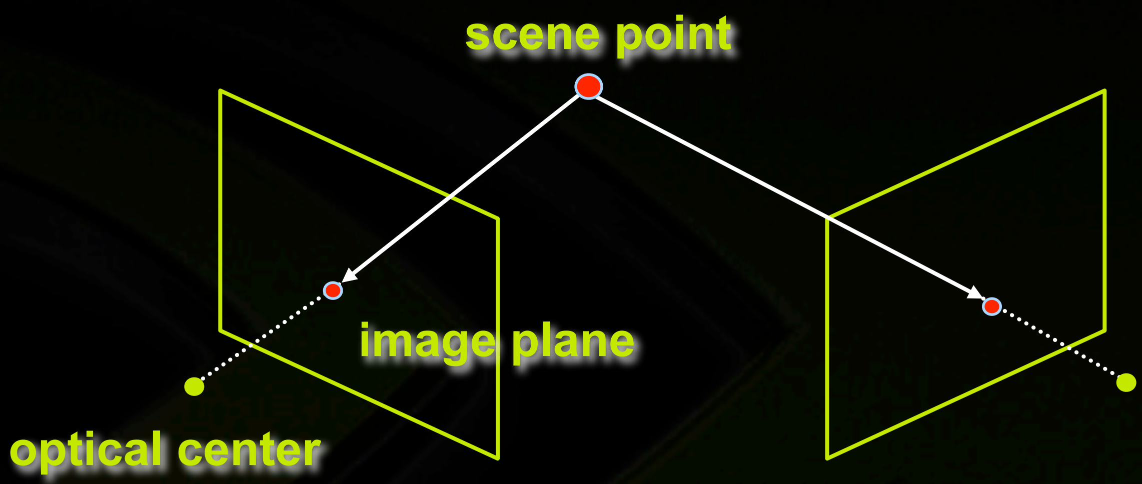
Depth from disparity



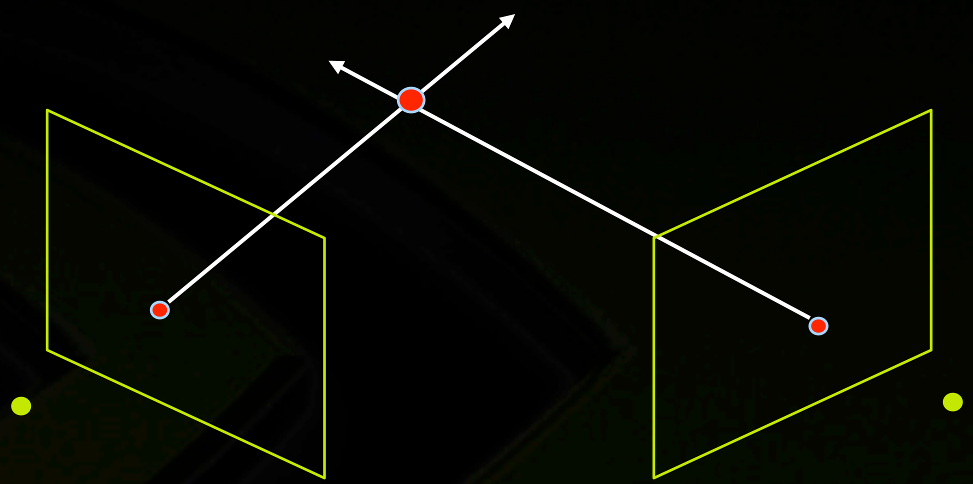
$$disparity = x - x' = \frac{B \times f}{z}$$

Disparity is inversely proportional to depth!

Stereo



Stereo

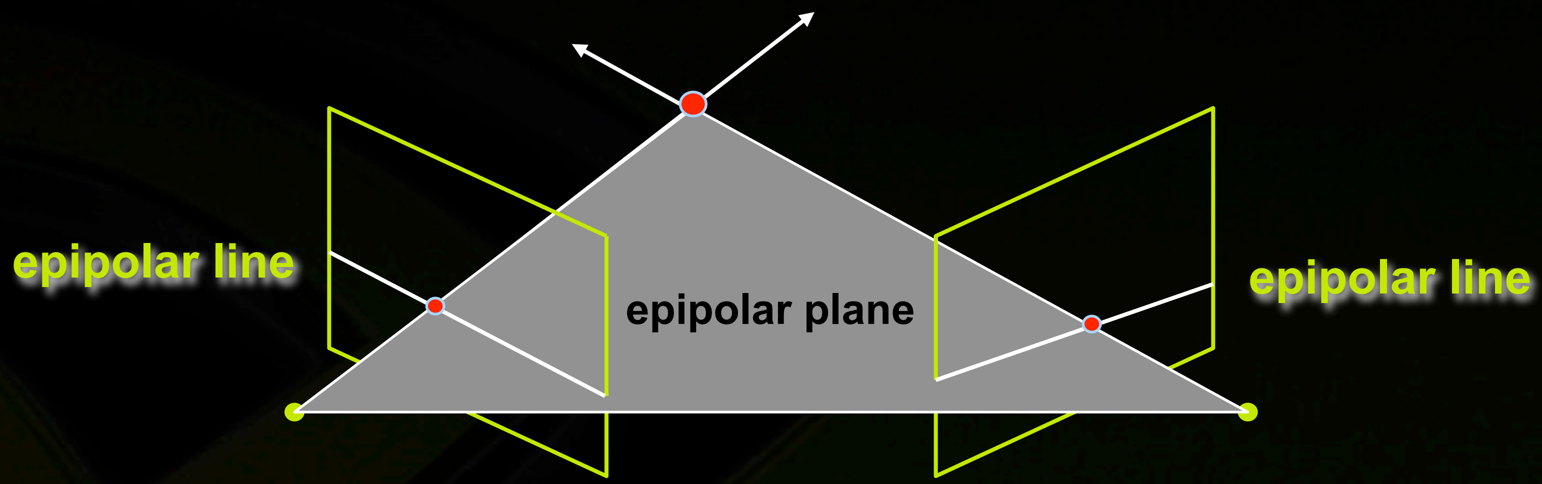


Basic Principle: Triangulation

- Gives reconstruction as intersection of two rays
- Requires
  - camera pose (calibration)
  - ***point correspondence***

Stereo correspondence

- Determine Pixel Correspondence
  - Pairs of points that correspond to same scene point

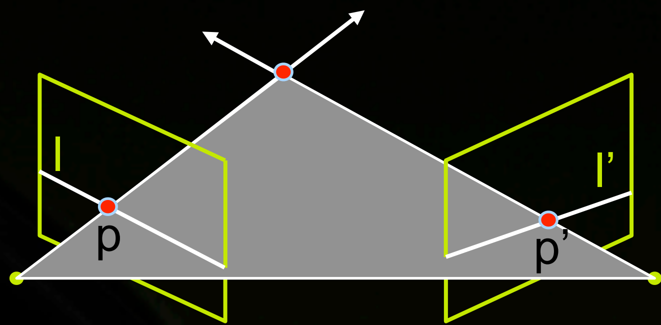


Epipolar Constraint

- Reduces correspondence problem to 1D search along *conjugate epipolar lines*
- Java demo: http://www.ai.sri.com/~luong/research/Meta3DViewer/EpipolarGeo.html

Fundamental matrix

- Let *p* be a point in left image, *p'* in right image



- Epipolar relation
  - *p* maps to epipolar line *l'*
  - *p'* maps to epipolar line *l*
- Epipolar mapping described by a 3x3 matrix *F*

$$
\begin{aligned}
l' &= Fp \\
l &= p'F
\end{aligned}
$$

- It follows that

$$p'Fp =$$

Fundamental matrix

- This matrix F is called
  - the "Essential Matrix"
    - when image intrinsic parameters are known
  - the "Fundamental Matrix"
    - essential matrix in pixel coordinates
- Can solve for F from point correspondences
  - Each (p, p') pair gives one linear equation in entries of F

$$p' F p = 0$$

  - 8 points give enough to solve for F (8-point algorithm)
  - see Marc Pollefey's notes for a nice tutorial

http://cs.unc.edu/~marc/tutorial/node53.html cs.unc.edu/~marc/tutorial/node54.html

NVIDIA

Eight-point algorithm

The two view structure is equivalent to the fundamental matrix. Since the fundamental matrix $\mathbf{F}$ is a 3×3 matrix determined up to an arbitrary scale factor, 8 equations are required to obtain a unique solution. The simplest way to compute the fundamental matrix consists of using Equation (3.26). This equation can be rewritten under the following form:

$$\begin{bmatrix} xx' & yx' & x' & xy' & yy' & y' & x & y & 1 \end{bmatrix} \mathbf{f} = 0 \qquad \text{(D6)}$$

with $\mathtt{m} = [x\, y\, 1]^\top, \mathtt{m}' = [x' y' 1]^\top$ and $\mathbf{f} = [F_{11} F_{12} F_{13} F_{21} F_{22} F_{23} F_{31} F_{32} F_{33}]^\top$ a vector containing the elements of the fundamental matrix $\mathbf{F}$. By stacking eight of these equations in a matrix $\mathbf{A}$ the following equation is obtained:

$$\mathbf{A}\mathbf{f} = 0 \qquad \text{(D7)}$$

This system of equation is easily solved by Singular Value Decomposition (SVD) [43]. Applying SVD to $\mathbf{A}$ yields the decomposition $\mathbf{USV}^\top$ with $\mathbf{U}$ and $\mathbf{V}$ orthonormal matrices and $\mathbf{S}$ a diagonal matrix containing the singular values. These singular values σ_i are positive and in decreasing order. Therefore in our case σ_9 is guaranteed to be identically zero (8 equations for 9 unknowns) and thus the last column of $\mathbf{V}$ is the correct solution (at least as long as the eight equations are linearly independent, which is equivalent to all other singular values being non-zero).

Add singularity constraint

Algorithm

Input: n correspondences with $n \geq 8$

1. Construct homogeneous system $Ax = 0$ where A is an nx9 matrix. Suppose $A = UDV^T$ is its SVD.

(i.e., corresponding to the smallest singular value)

2. The entries of F are proportional to the components of the last column of V.

Enforcing the constraint $rank(F) = 2$**:** (singularity constraint)

3. compute the SVD of F

$$F = U_F D_F V_F^T$$

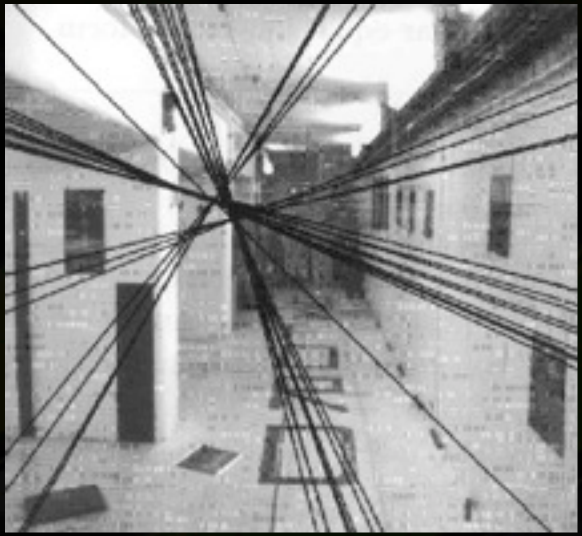
4. Set the smallest singular value equal to 0; Let D'_F be the corrected matrix.

5. The corrected estimate of F, F', is given by

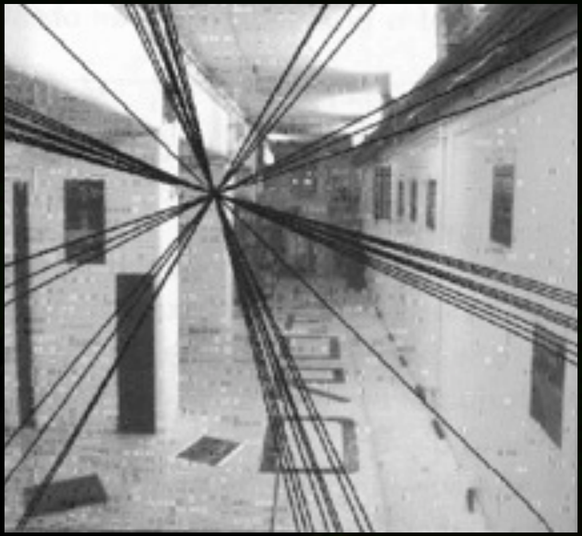
$$F' = U_F D'_F V_F^T$$

Epipolar lines must intersect at epipole!

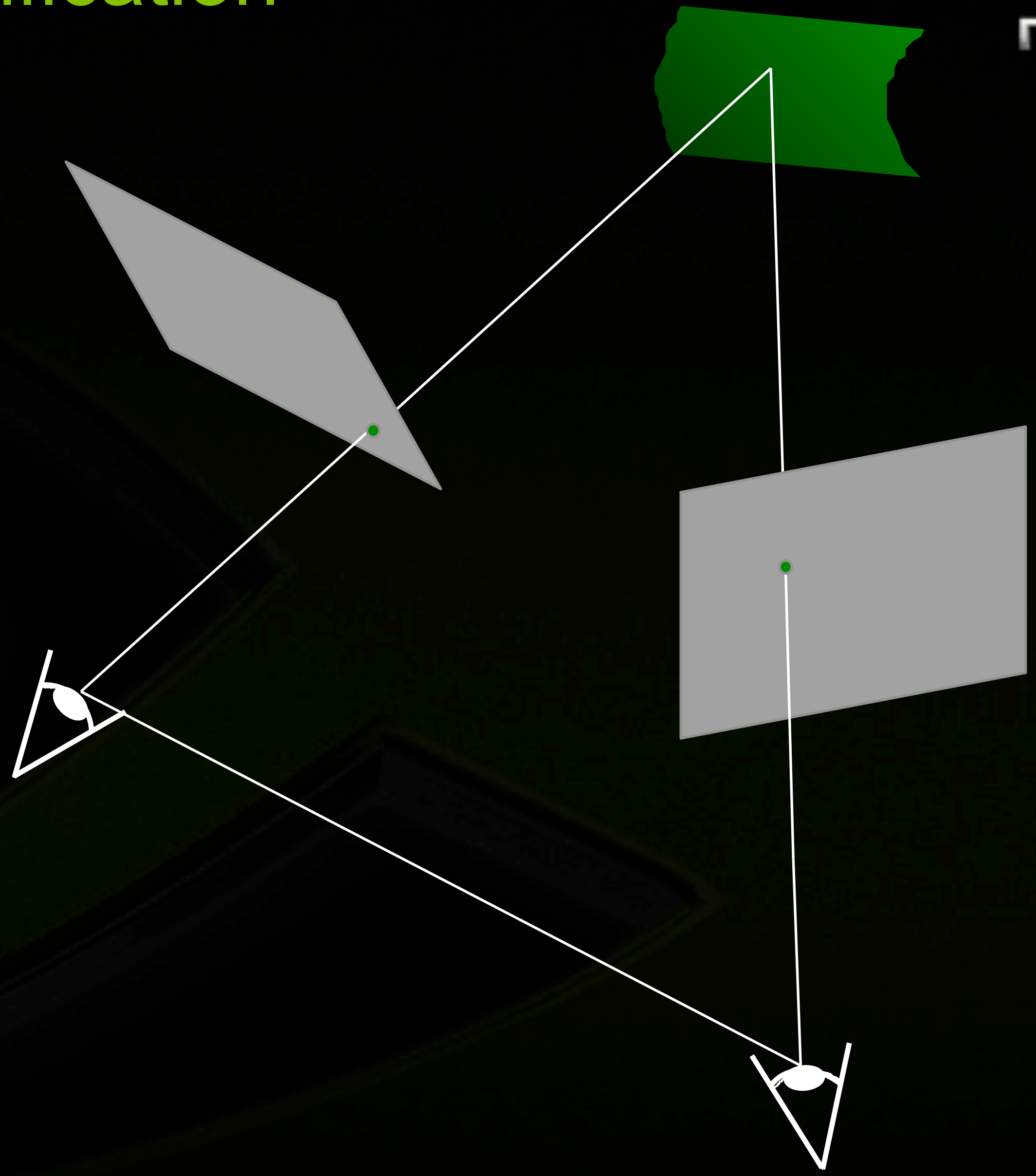
Uncorrected F



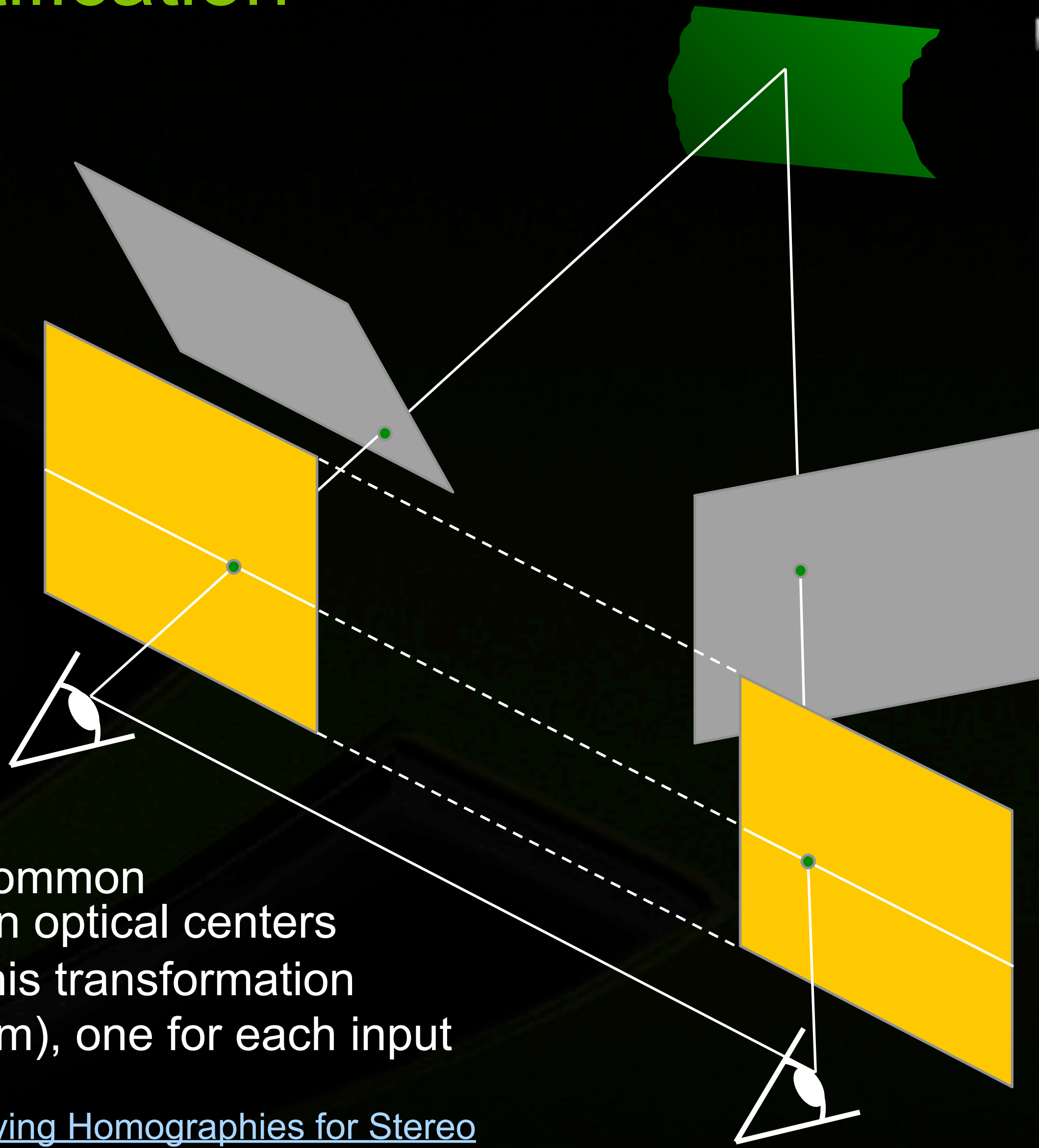
Corrected F



Stereo image rectification



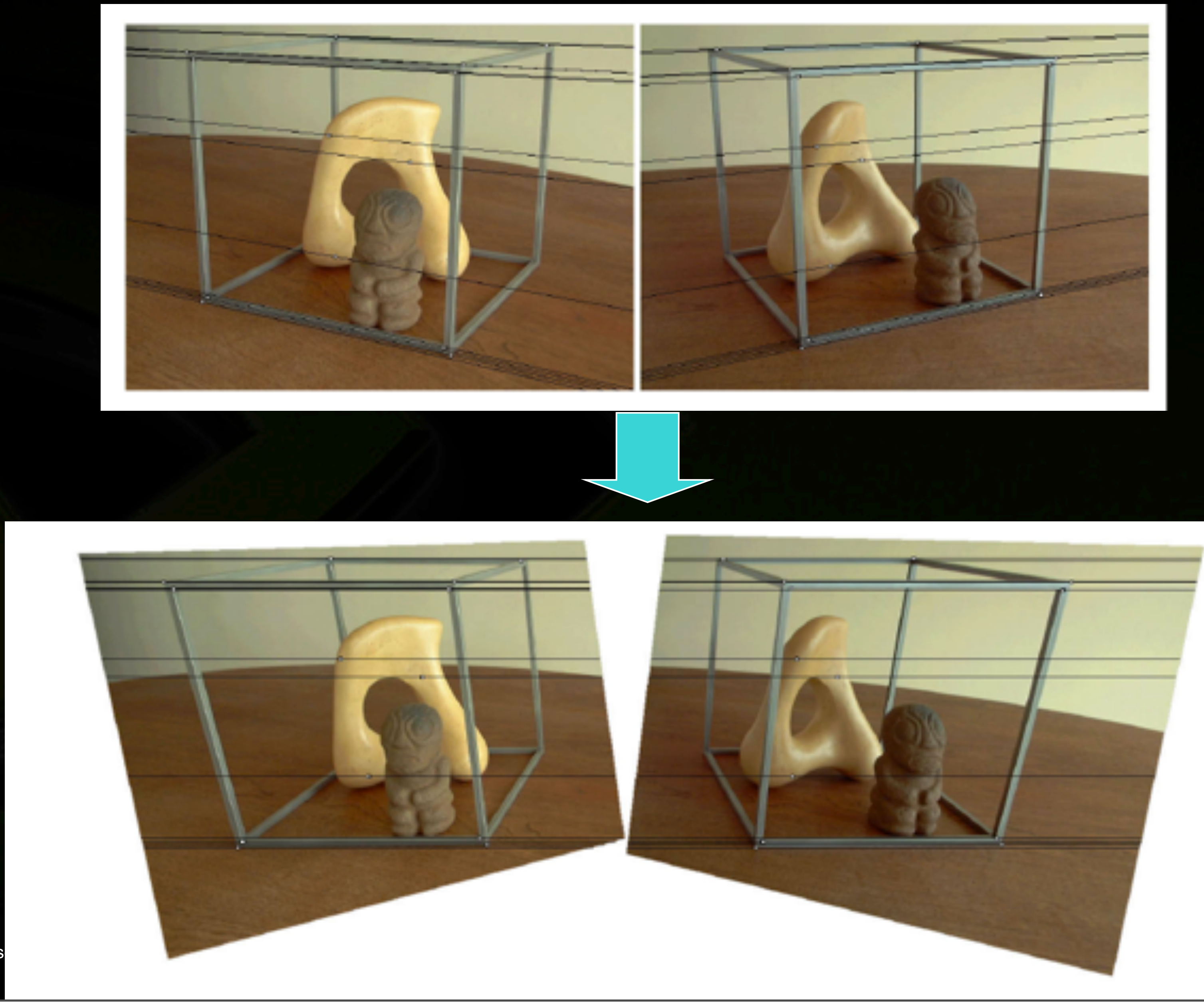
Stereo image rectification



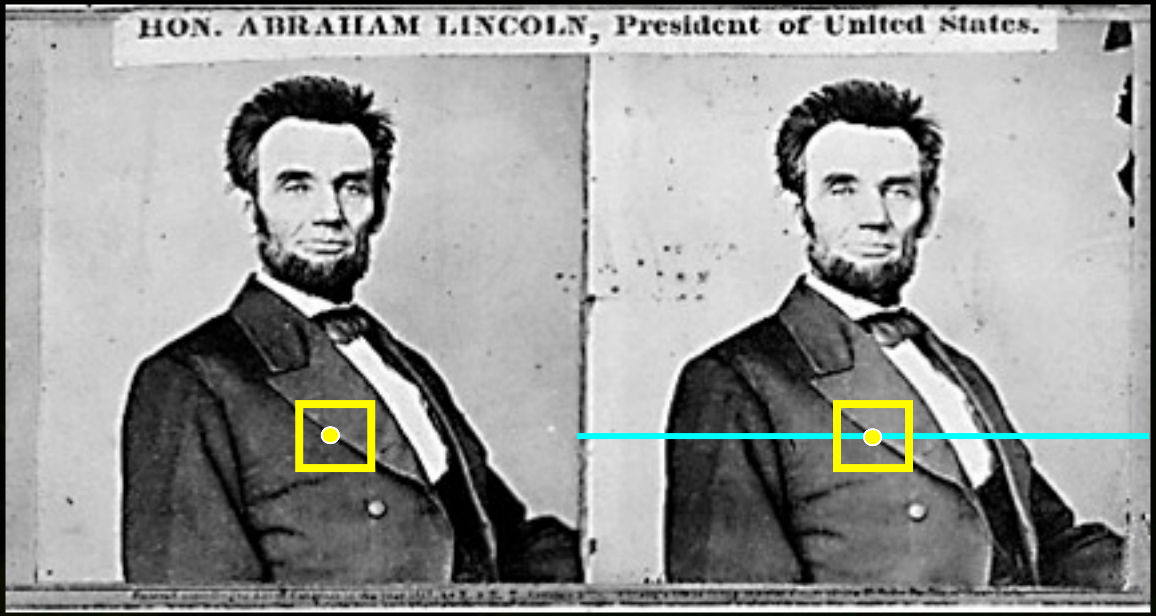
- reproject image planes onto a common plane parallel to the line between optical centers
- pixel motion is horizontal after this transformation
- two homographies (3x3 transform), one for each input image reprojection

➢ C. Loop and Z. Zhang. Computing Rectifying Homographies for Stereo Vision
IEEE Conf. Computer Vision and Pattern Recognition, 1999

Stereo Rectification

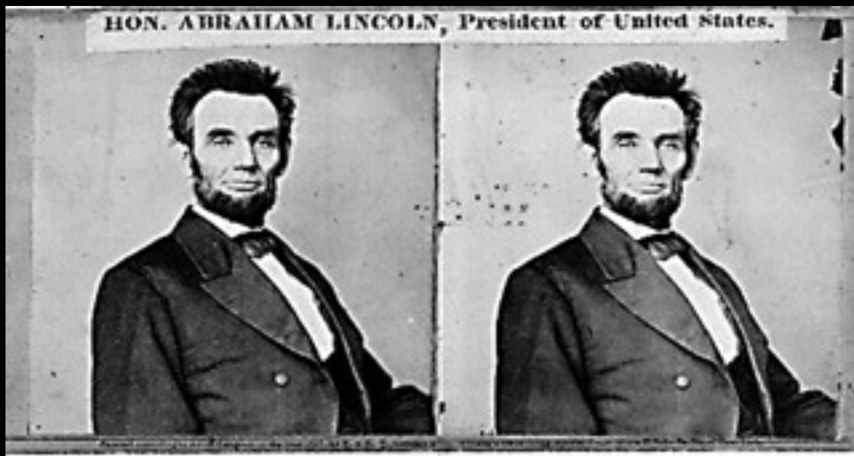


Correspondences from similarities



- Corresponding regions in two images should be similar in appearance
- …and non-corresponding regions should be different
- When will the similarity constraint fail?

Limitations of similarity constraint

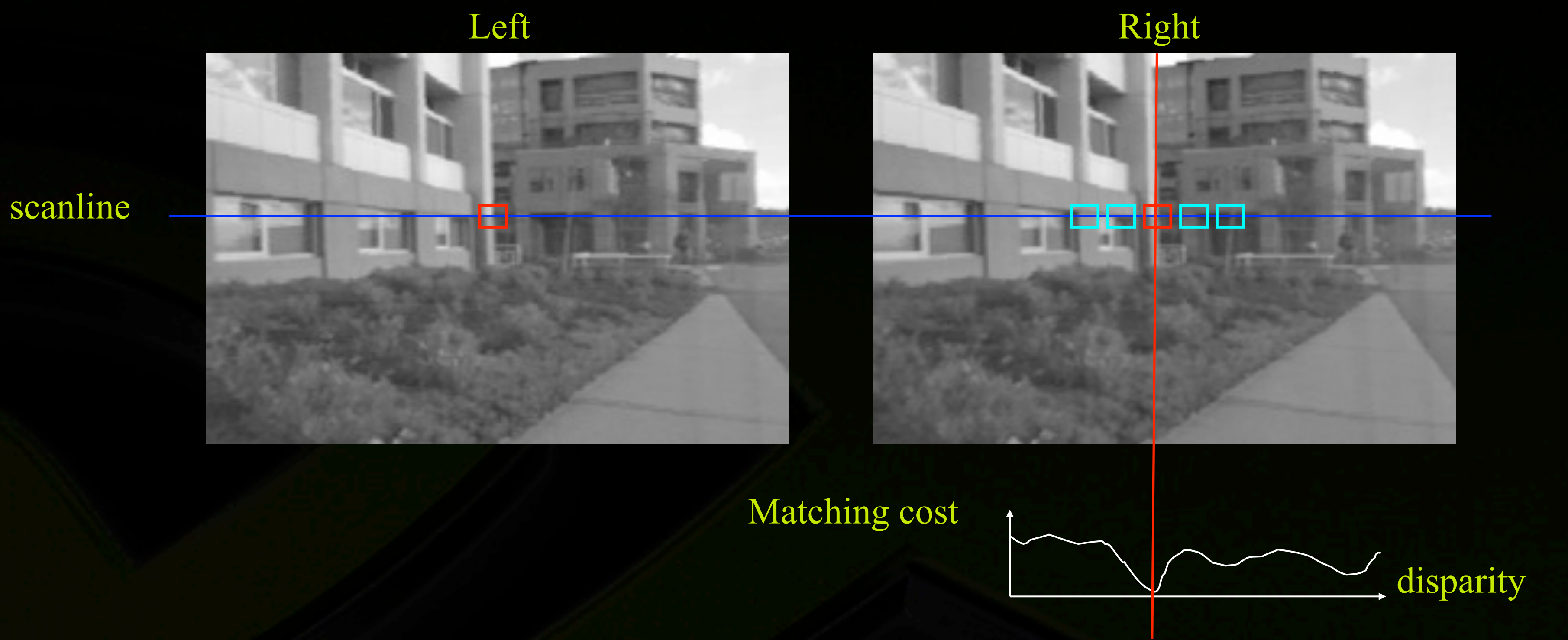


Textureless surfaces

Occlusions, repetition

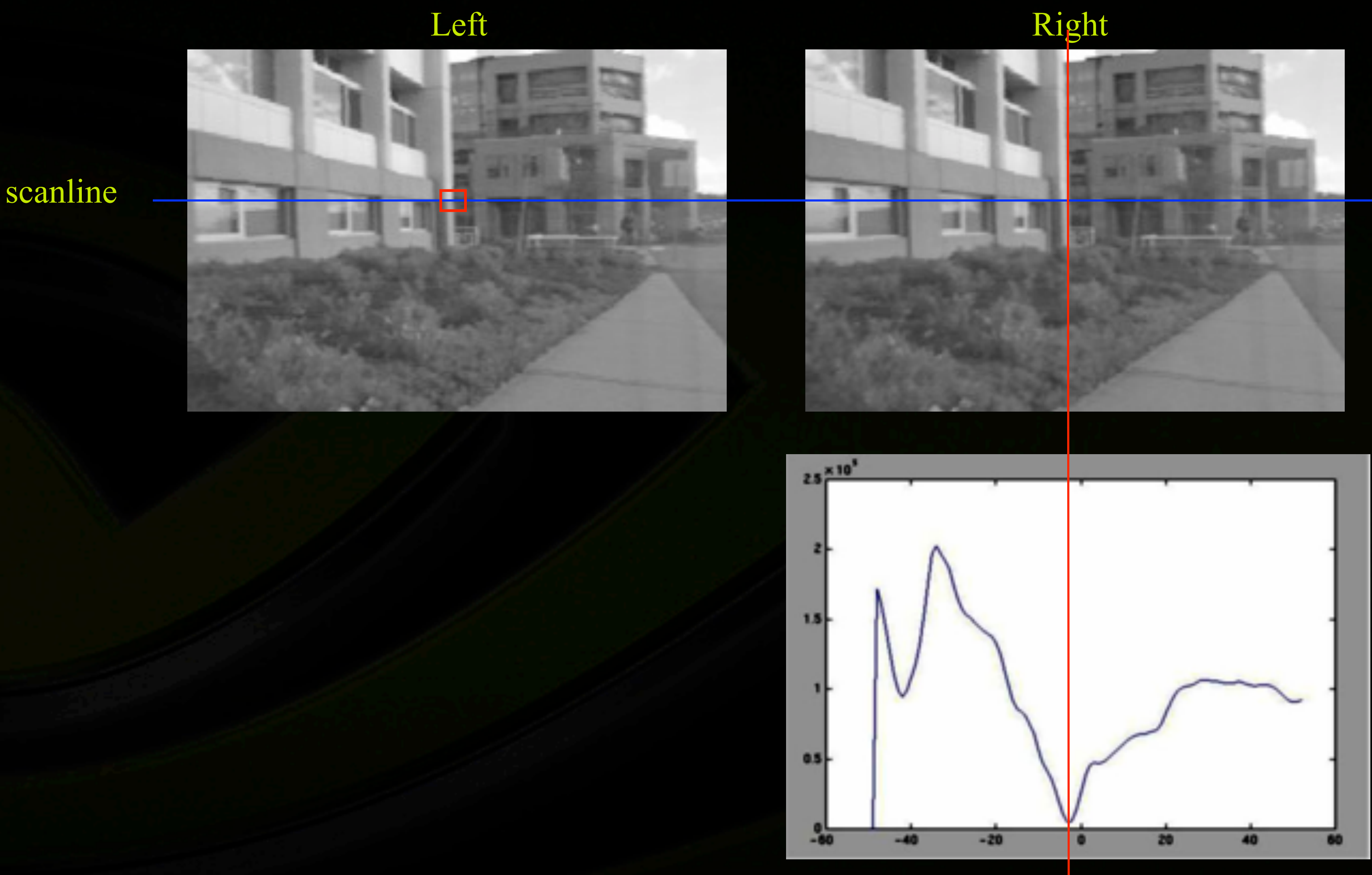
Non-Lambertian surfaces, specularities

Correspondences from similarities



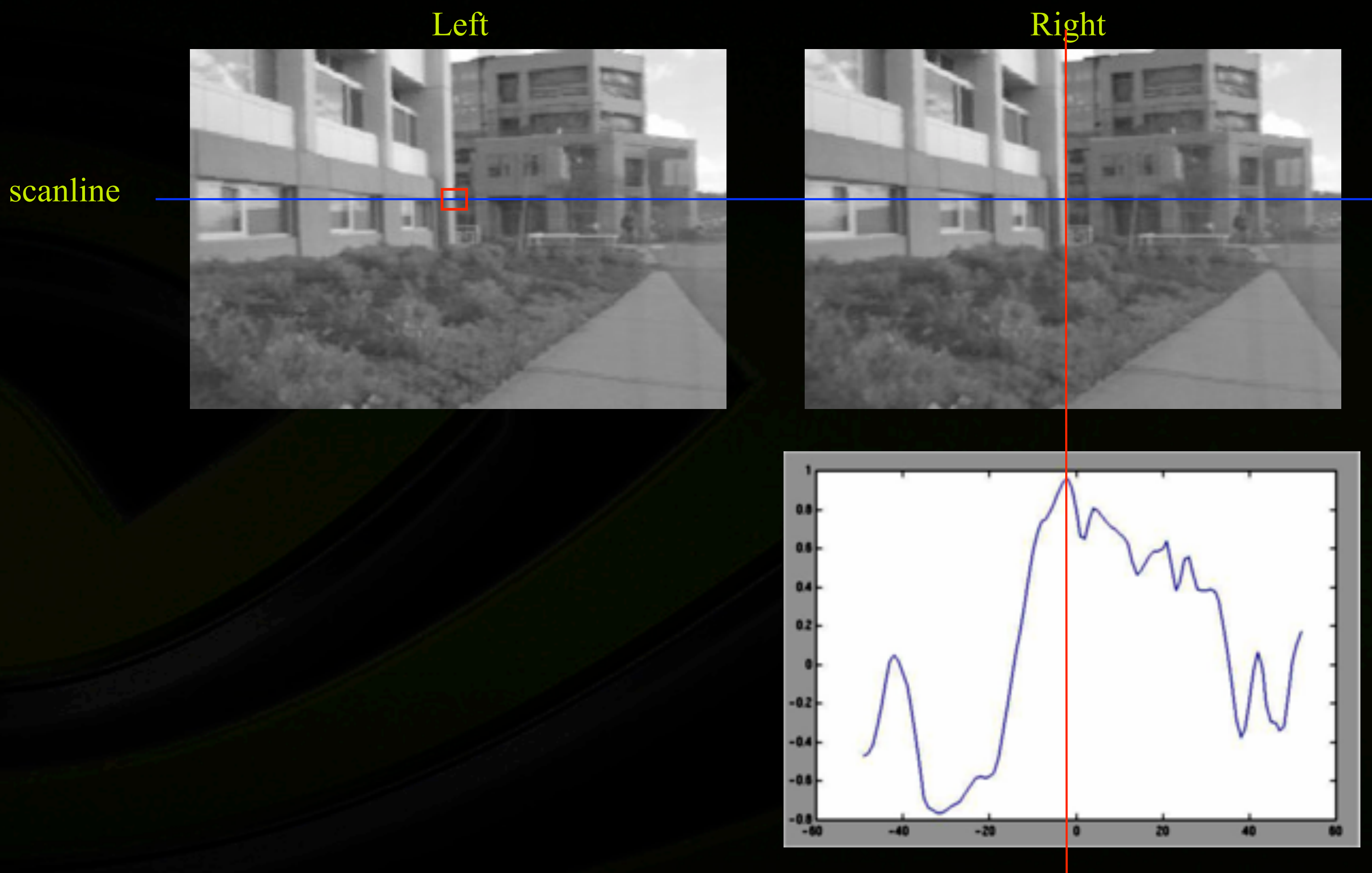
- Slide a window along the right scanline and compare contents of that window with the reference window
- Matching cost: SSD or NCC

Correspondences from similarities



SSD = sum of squared differences

Correspondences from similarities



NCC = Normalized cross-correlation

Image Normalization

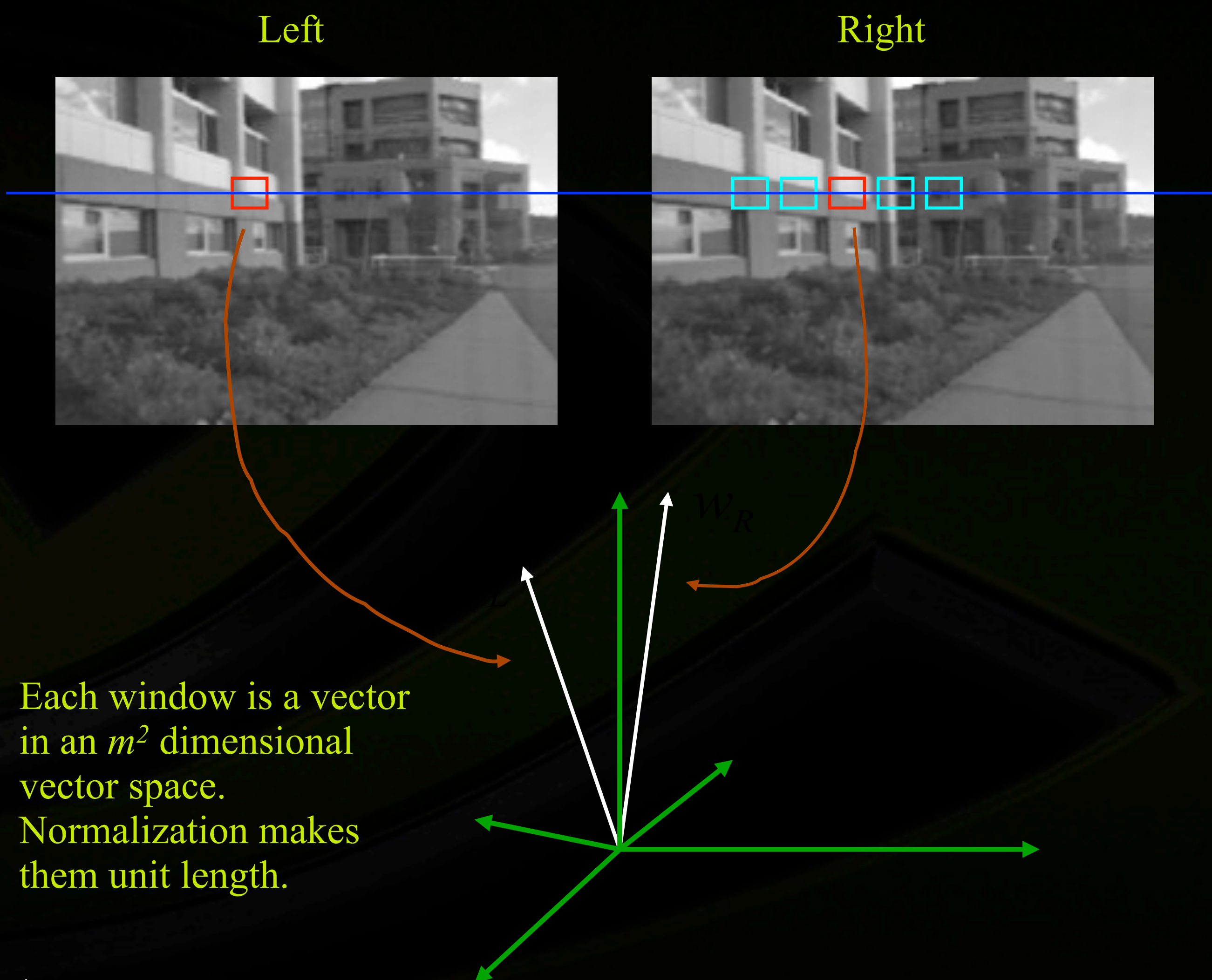
- Even when the cameras are identical models, there can be differences in gain and sensitivity
- The cameras do not see exactly the same surfaces, so their overall light levels can differ
- For these reasons and more, it is a good idea to normalize the pixels in each window:

$$\bar{I} = \frac{1}{|W_m(x,y)|} \sum_{(u,v)\in W_m(x,y)} I(u,v) \qquad \text{Average pixel}$$

$$\|I\|_{W_m(x,y)} = \sqrt{\sum_{(u,v)\in W_m(x,y)} [I(u,v)]^2} \qquad \text{Window magnitude}$$

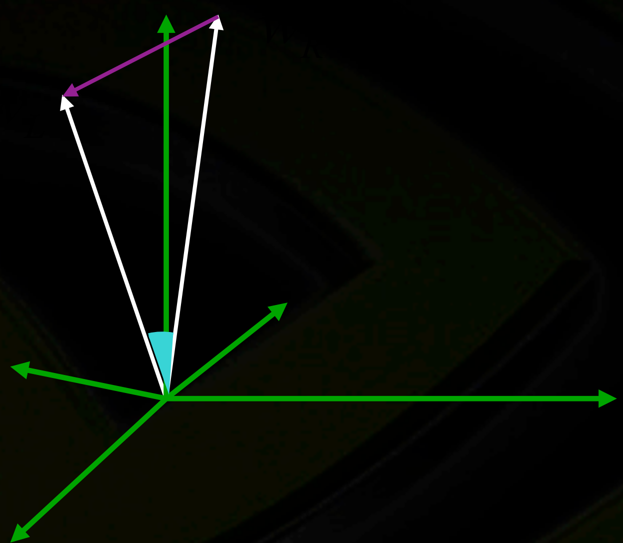
$$\hat{I}(x,y) = \frac{I(x,y) - \bar{I}}{\|I - \bar{I}\|_{W_m(x,y)}} \qquad \text{Normalized pixel}$$

Images as Vectors



Each window is a vector in an m^2 dimensional vector space. Normalization makes them unit length.

Image Metrics



(Normalized) Sum of Squared Differences

$$C_{\mathrm{SSD}}(d) = \sum_{(u,v)\in W_m(x,y)} [\hat{I}_L(u,v) - \hat{I}_R(u-d,v)]^2$$

$$= \left\| w_L - w_R(d) \right\|^2$$

Normalized Cross-Correlation

$$C_{\mathrm{NC}}(d) = \sum_{(u,v)\in W_m(x,y)} \hat{I}_L(u,v)\hat{I}_R(u-d,v)$$

$$= w_L \times w_R(d) = \cos\theta$$

Census

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

125	126	125
127	128	130
129	132	135

→

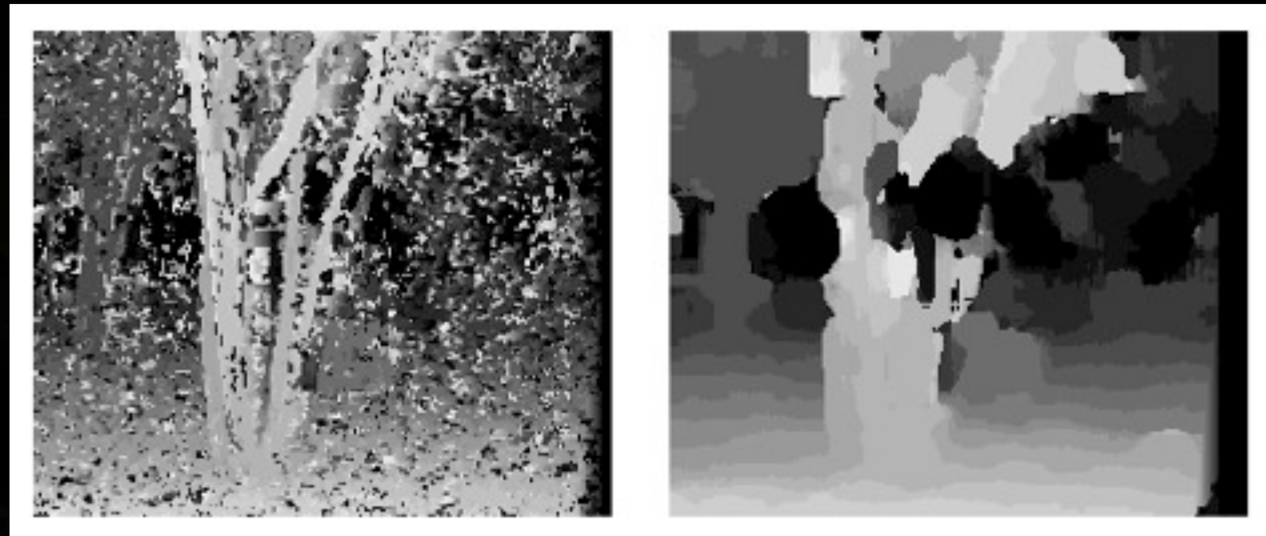
0	0	0
0		1
1	1	1

→ [00001111]

only compare bit signature

(Real-time chip from TYZX based on Census)

Effect of window size

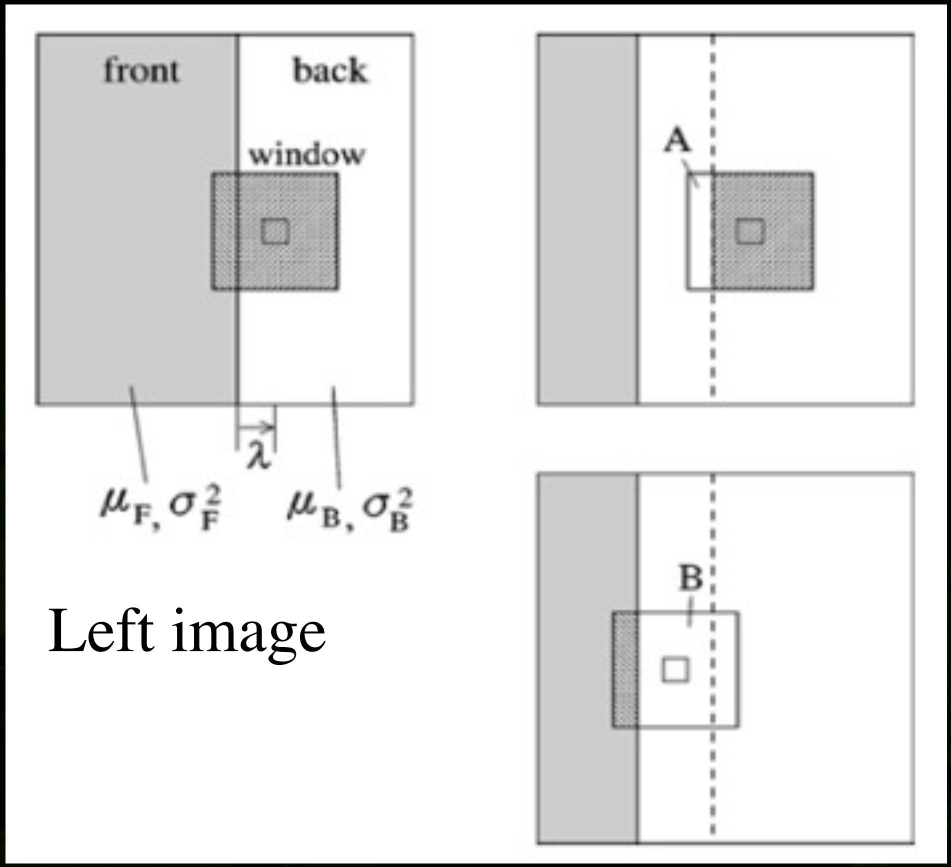


W = 3 W = 20

- Smaller window
  - \+ More detail
  - More noise

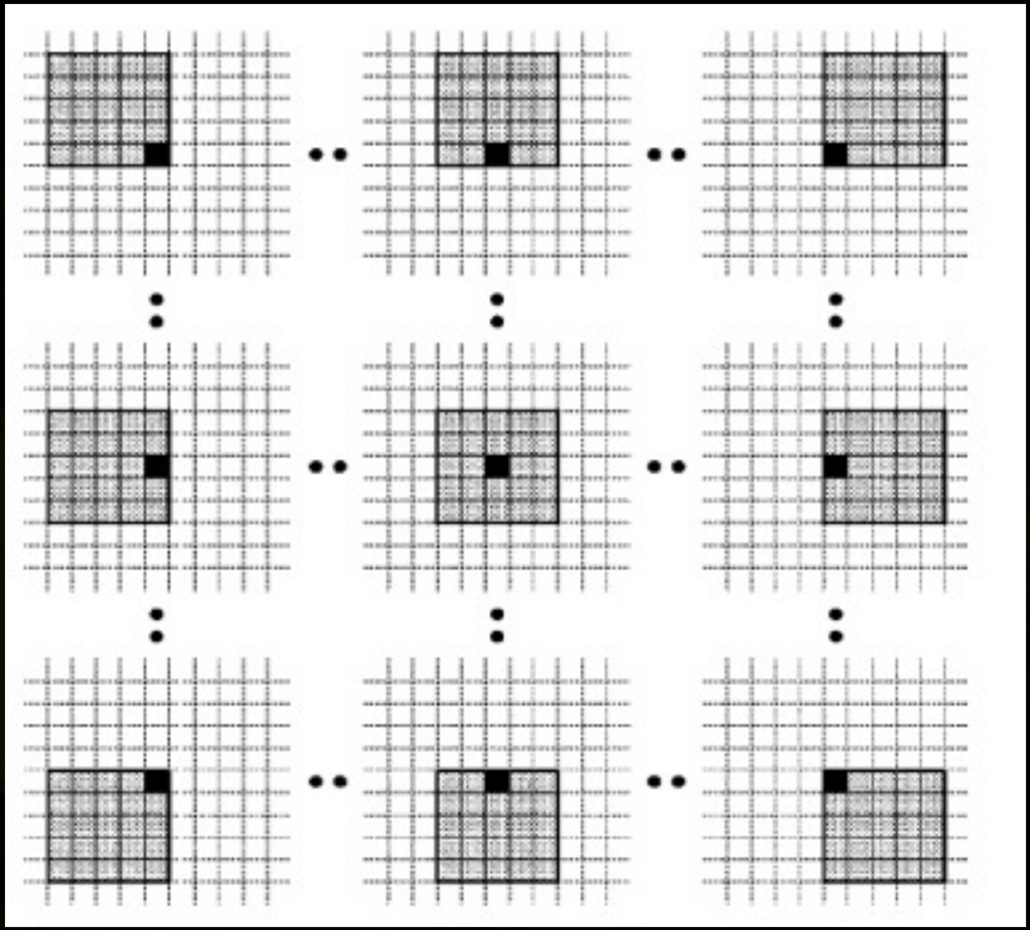
- Larger window
  - \+ Smoother disparity maps
  - Less detail

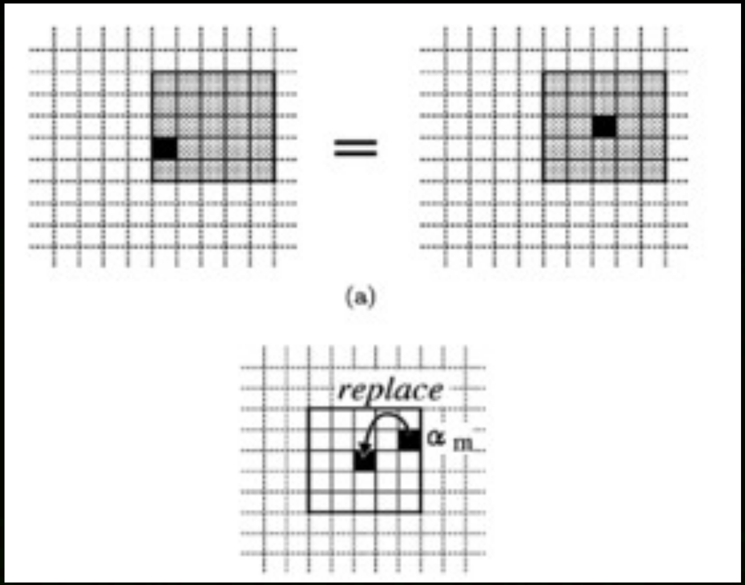
Occlusion edges are difficult



Right image

Offset windows





- Equivalent to using min of nearby cost
- Loss of depth accuracy

Discontinuity detection

- Use offset windows only where appropriate

Locally adaptive support

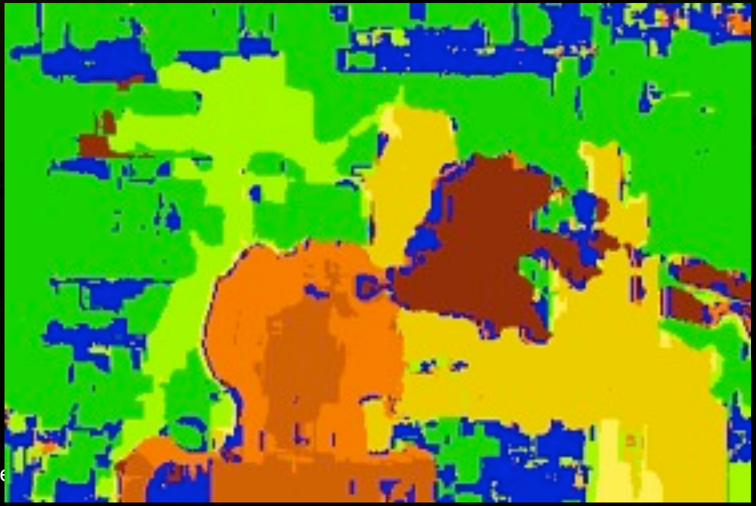
- Apply weights to contributions of neighboring pixels according to similarity and proximity [Yoon CVPR05]

(a) left support window (b) right support window (c) color difference between (a) and (b)

Results with window search

Data

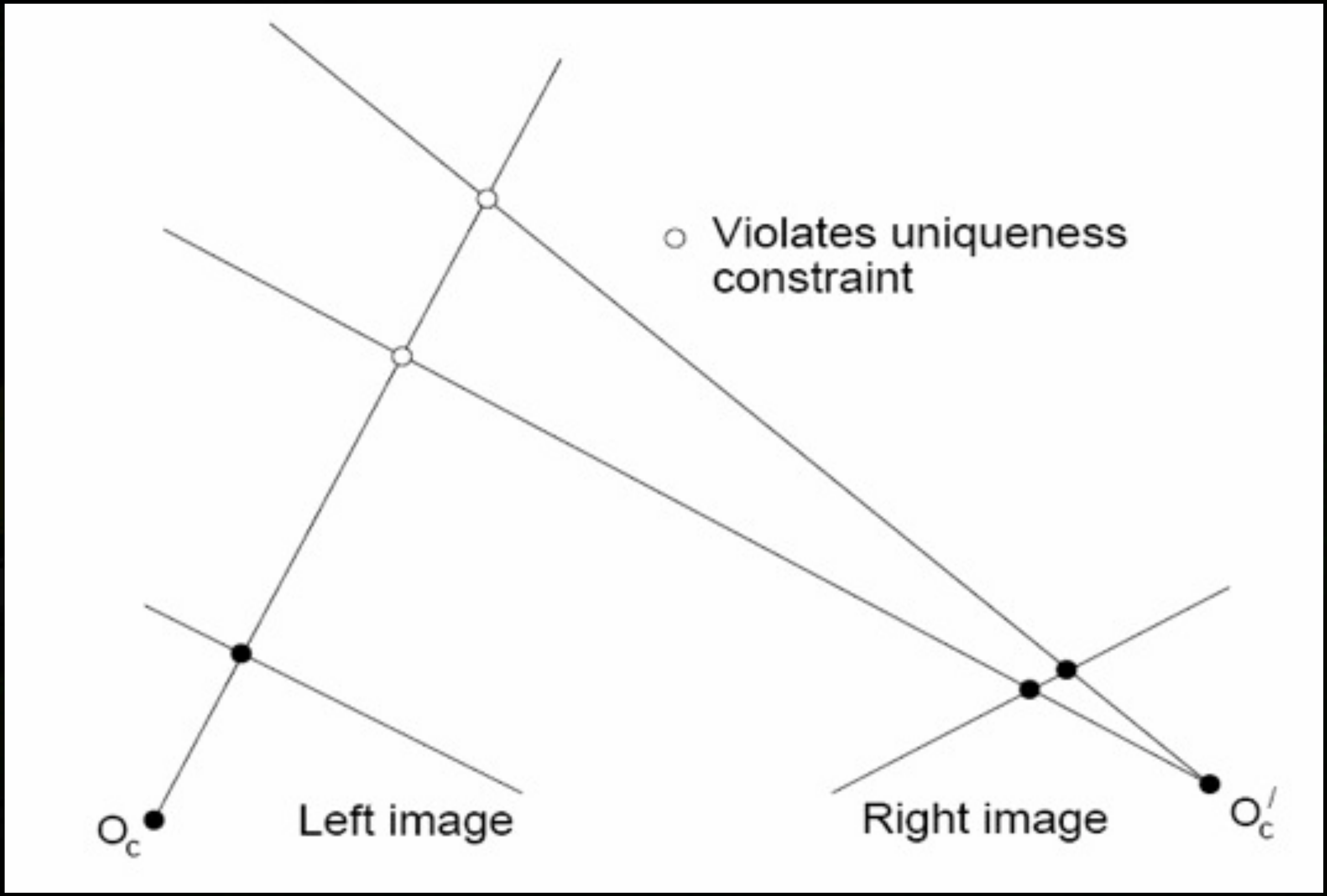
Window-based matching



Ground truth

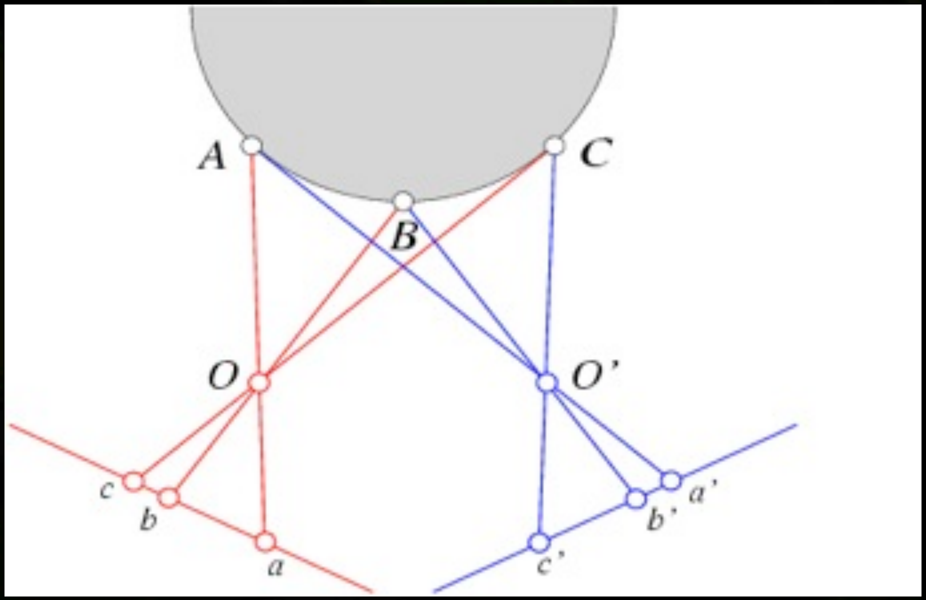
Non-local constraints

- Uniqueness
  - For any point in one image, there should be at most one matching point in the other image



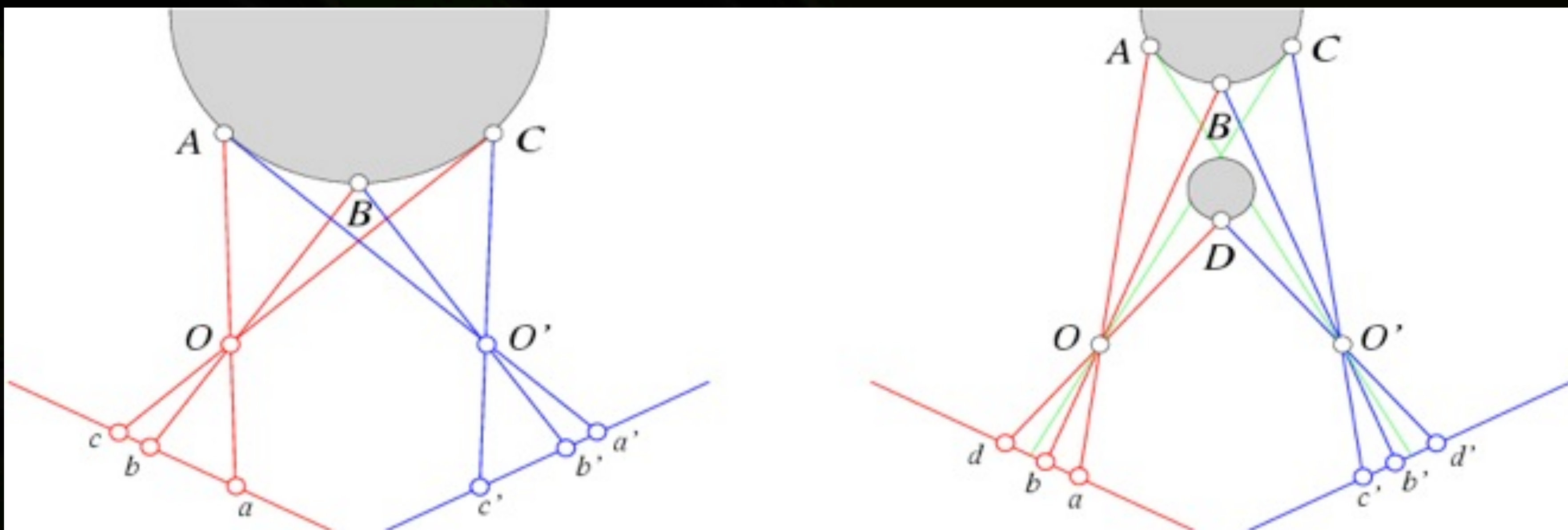
Non-local constraints

- Uniqueness
  - For any point in one image, there should be at most one matching point in the other image
- Ordering
  - Corresponding points should be in the same order in both views



Non-local constraints

- Uniqueness
  - For any point in one image, there should be at most one matching point in the other image
- Ordering
  - Corresponding points should be in the same order in both views



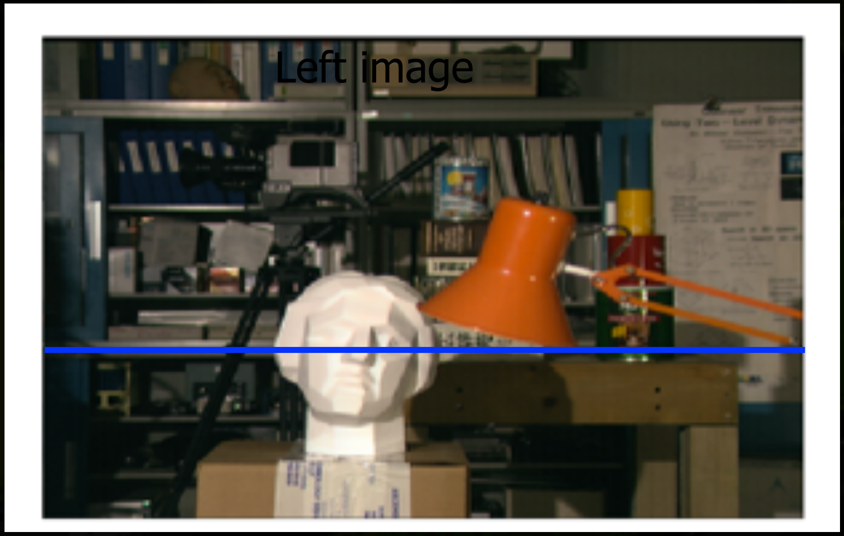
Ordering constraint does not hold

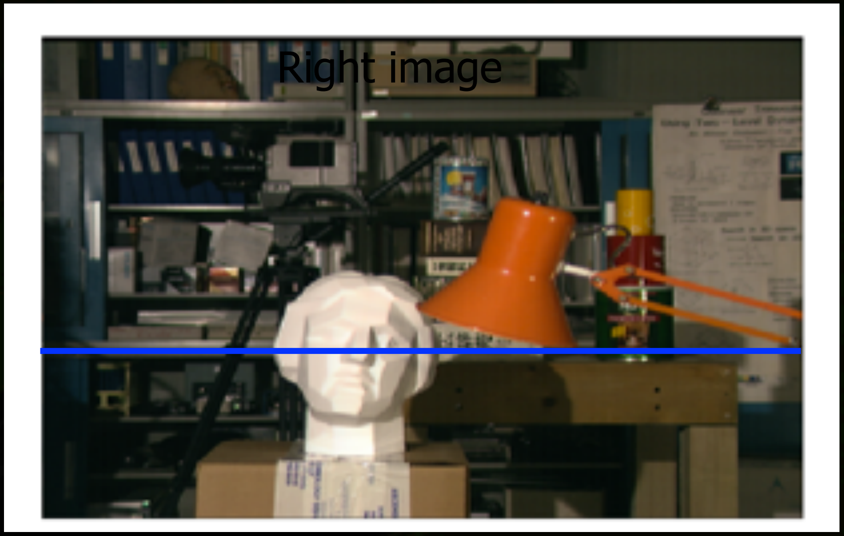
Non-local constraints

- Uniqueness
  - For any point in one image, there should be at most one matching point in the other image
- Ordering
  - Corresponding points should be in the same order in both views
- Smoothness
  - We expect disparity values to change slowly (for the most part)

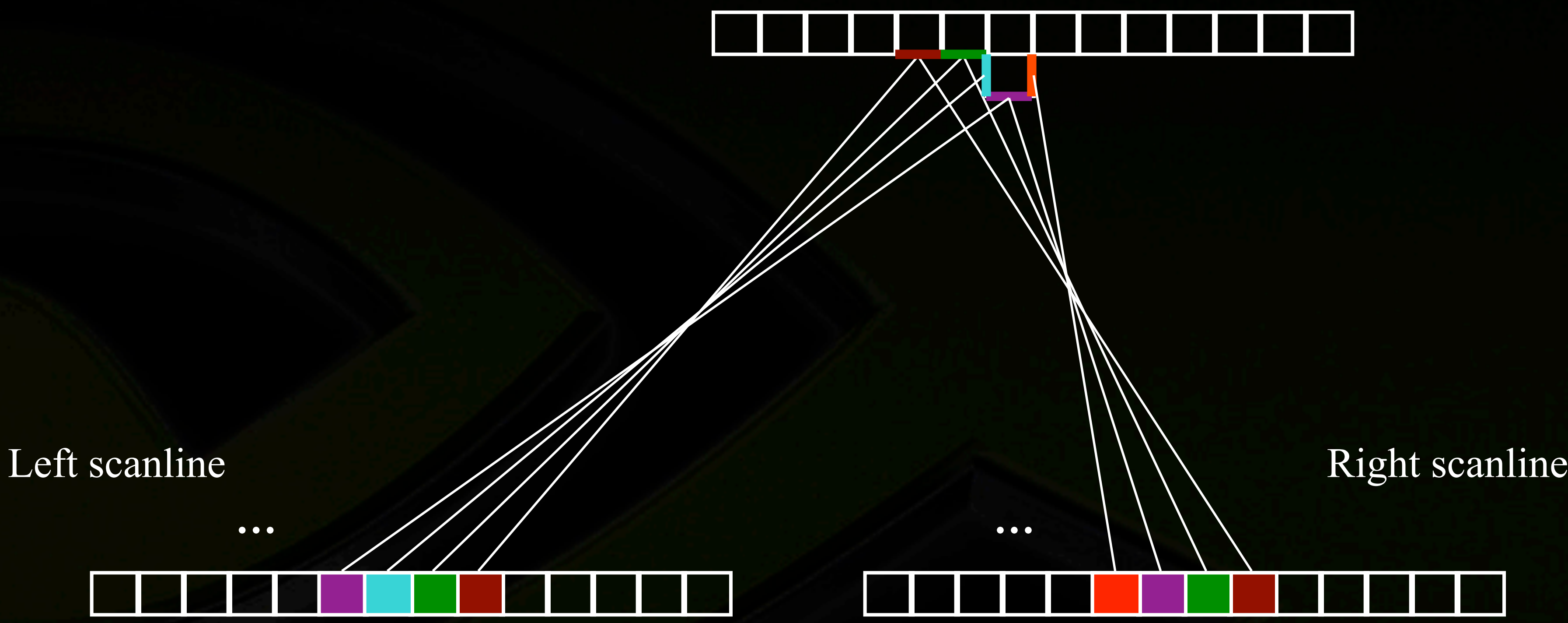
Scanline stereo

- Try to coherently match pixels on the entire scanline
- Different scanlines are still optimized independently

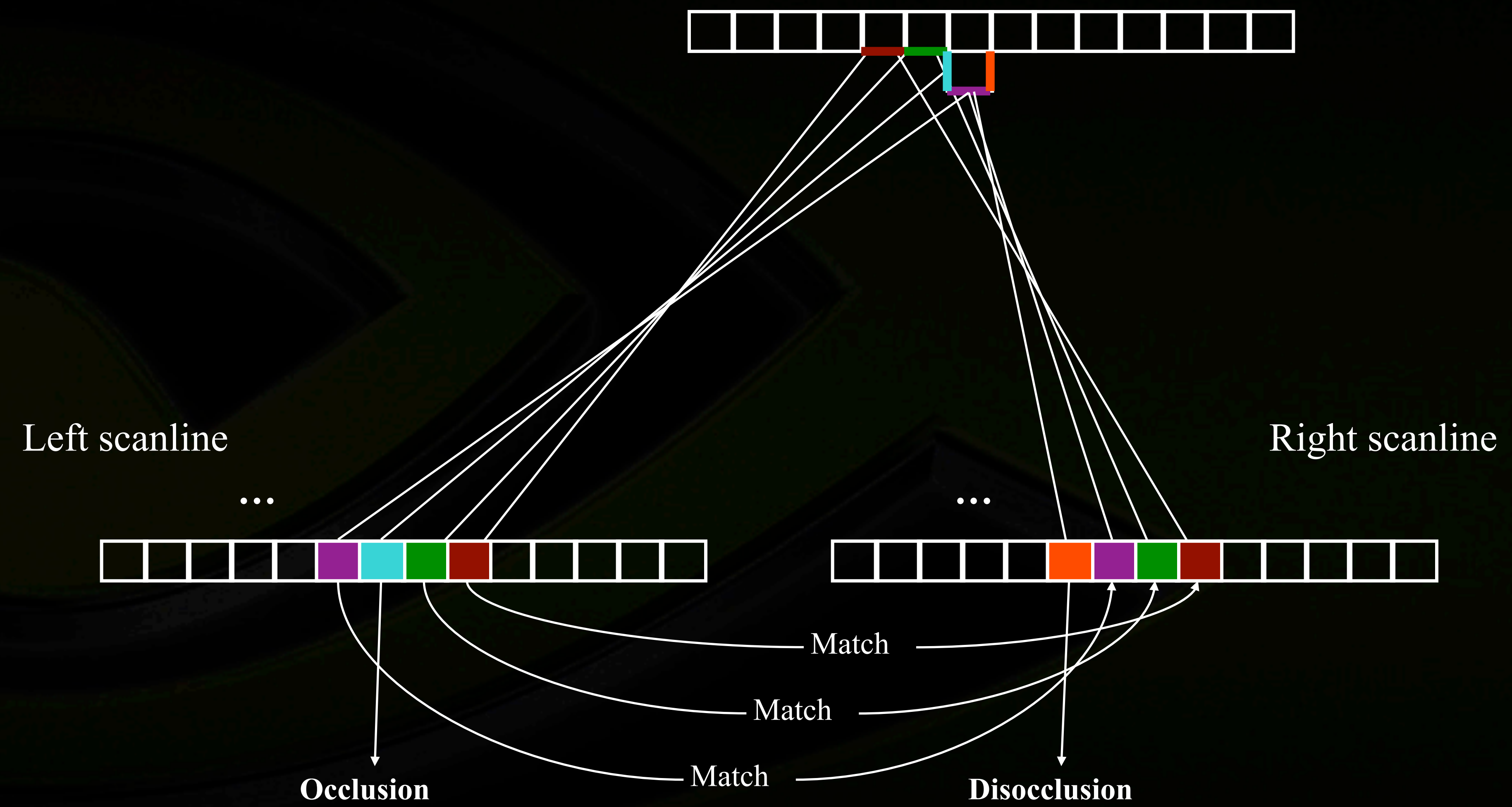




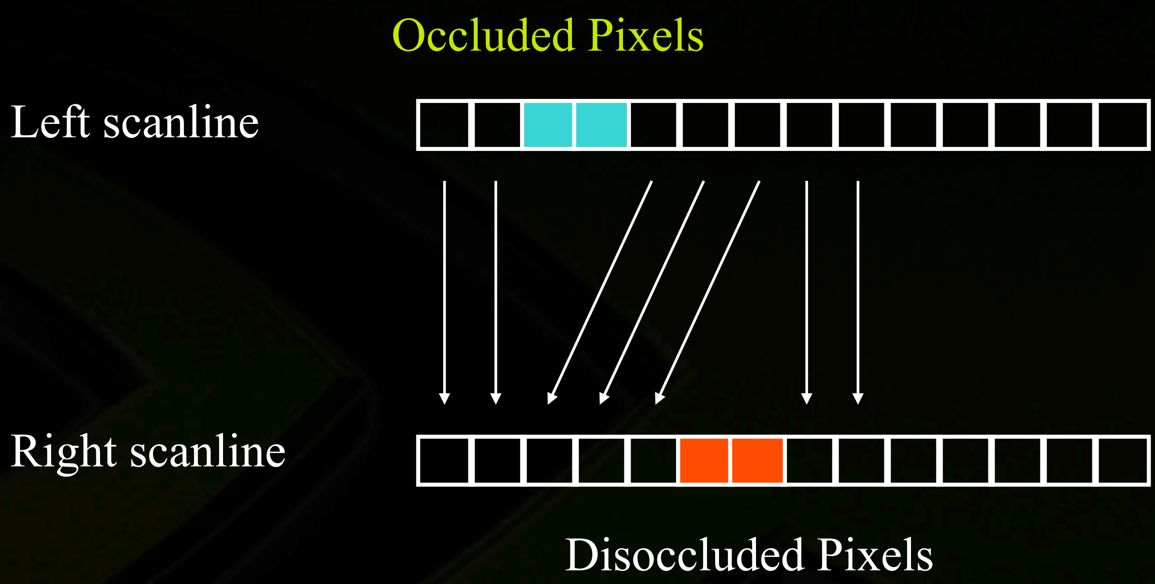
Stereo Correspondences



Stereo Correspondences



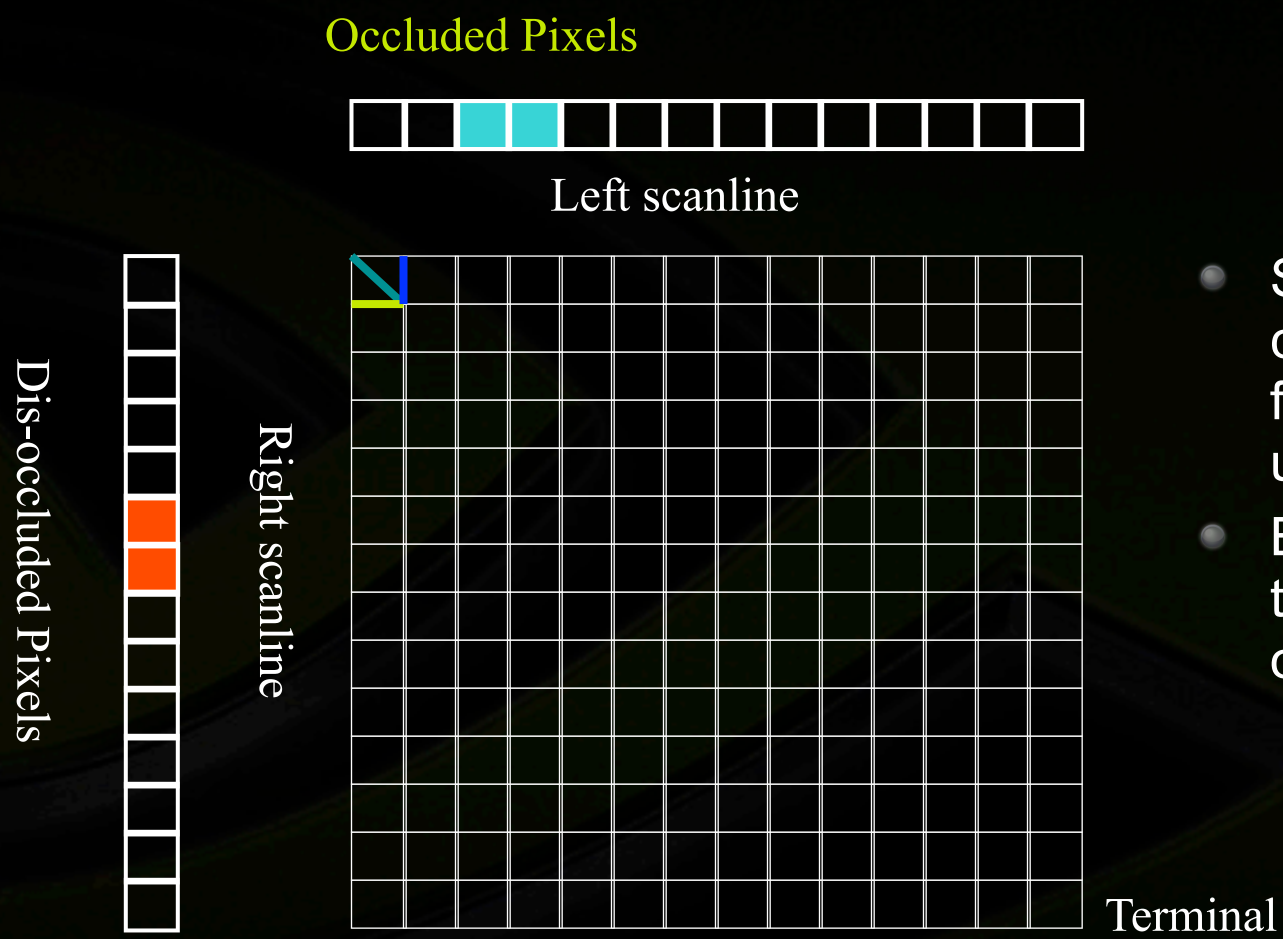
Search Over Correspondences



Three cases:

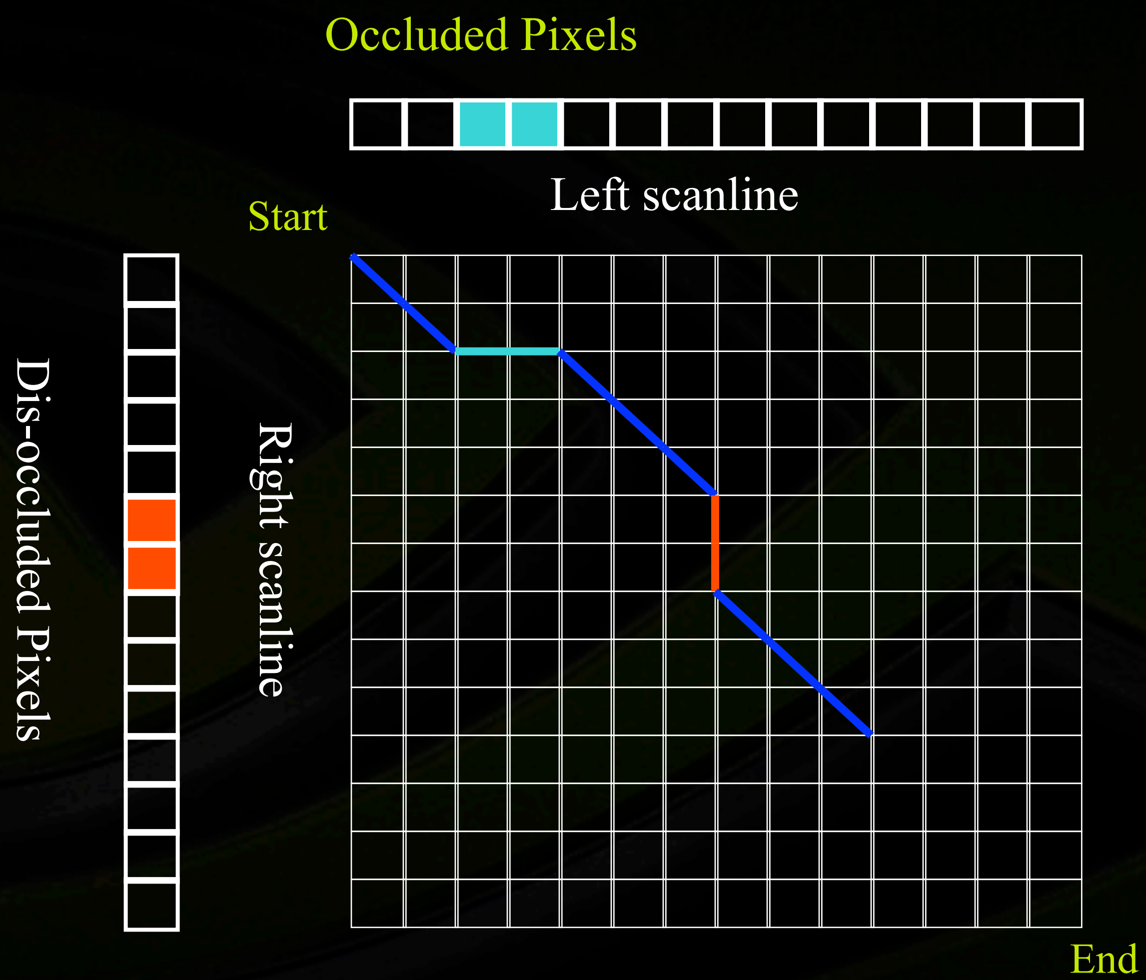
- Sequential – cost of match
- Occluded – cost of no match
- Disoccluded – cost of no match

Stereo Matching with Dynamic Programming



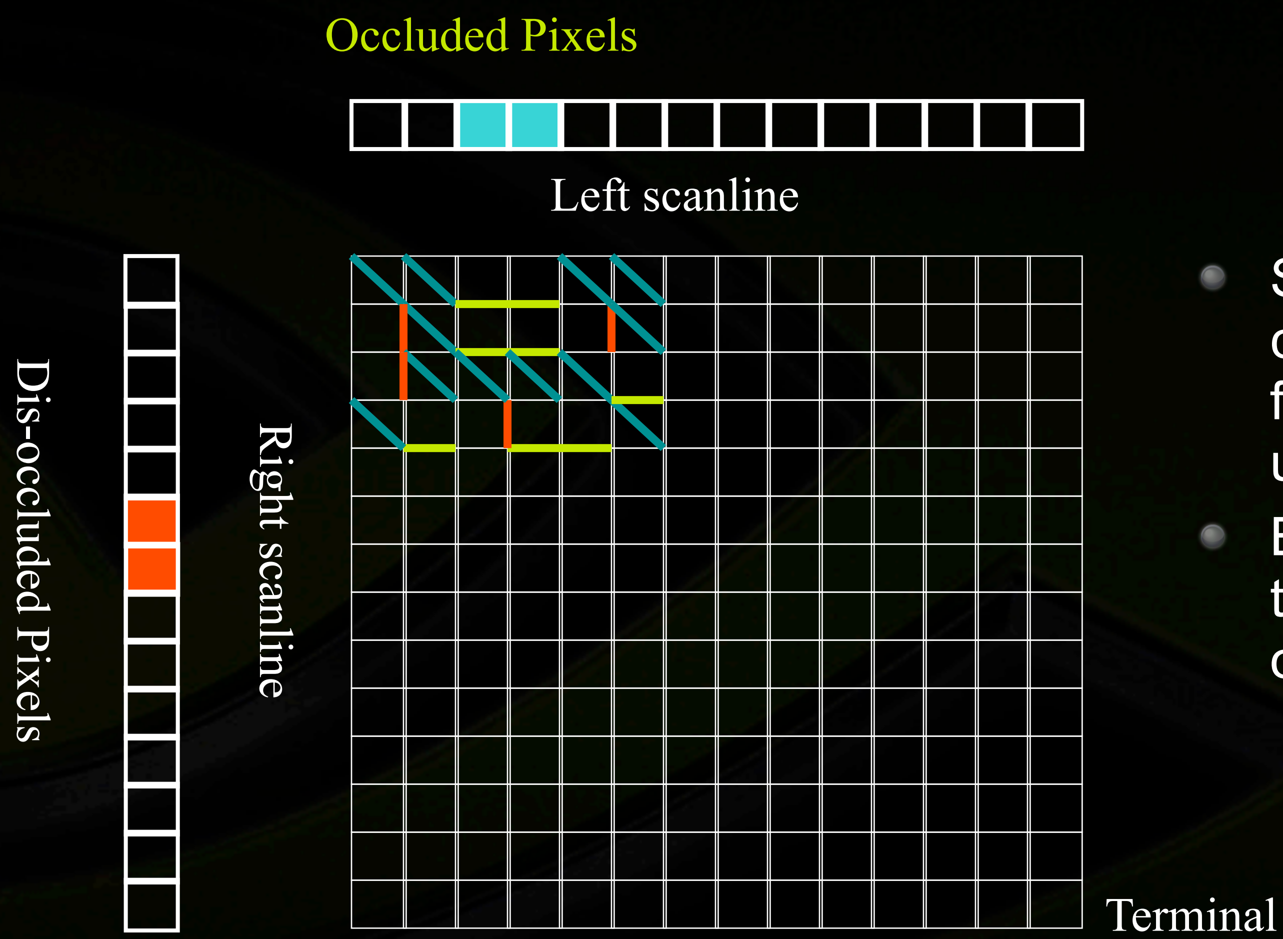
- Scan across grid computing optimal cost for each node given its upper-left neighbors
- Backtrack from the terminal to get the optimal path

Stereo Matching with Dynamic Programming



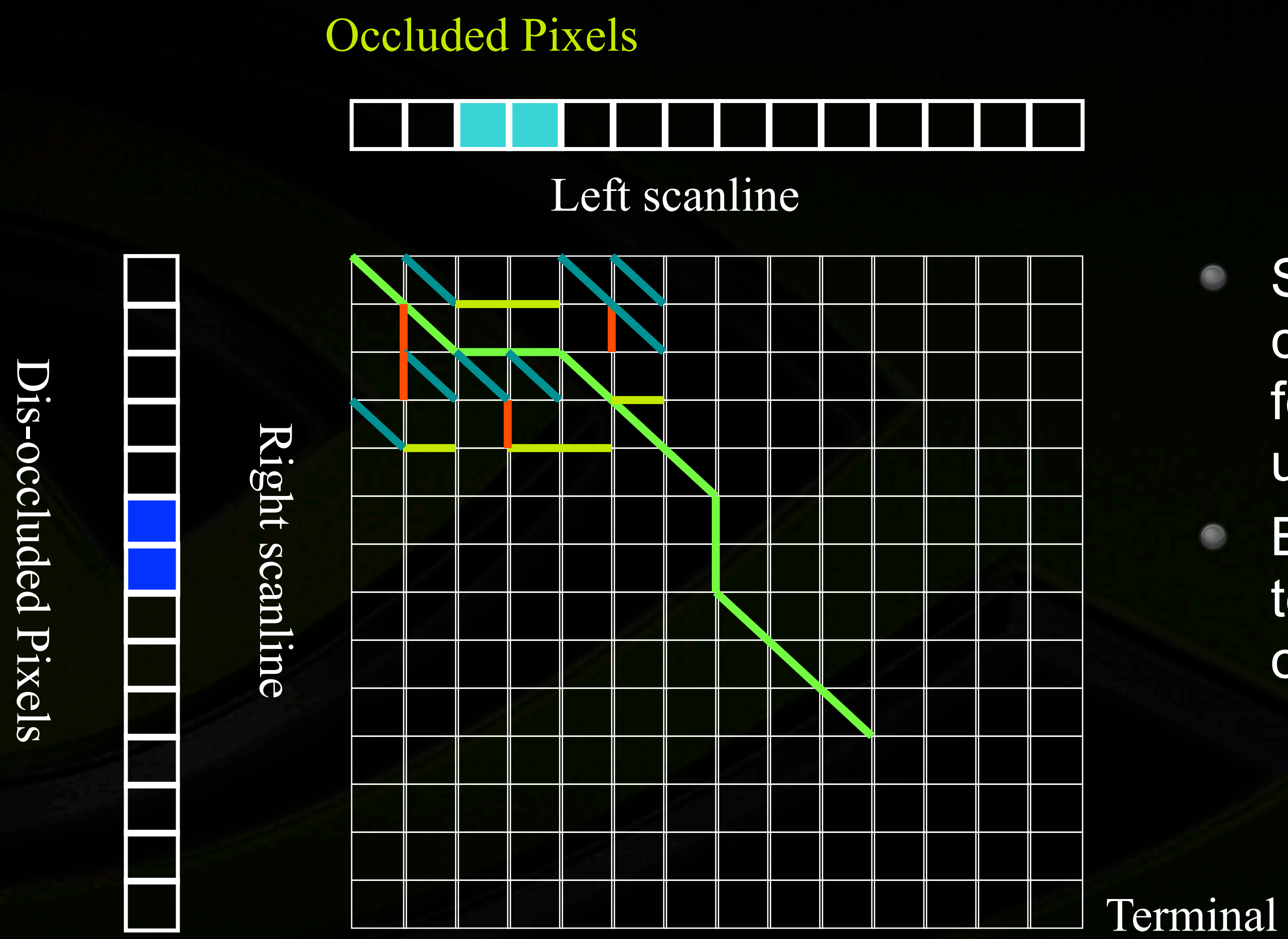
- Dynamic programming yields the optimal path through grid
- This is the best set of matches that satisfy the ordering constraint

Stereo Matching with Dynamic Programming



- Scan across grid computing optimal cost for each node given its upper-left neighbors
- Backtrack from the terminal to get the optimal path

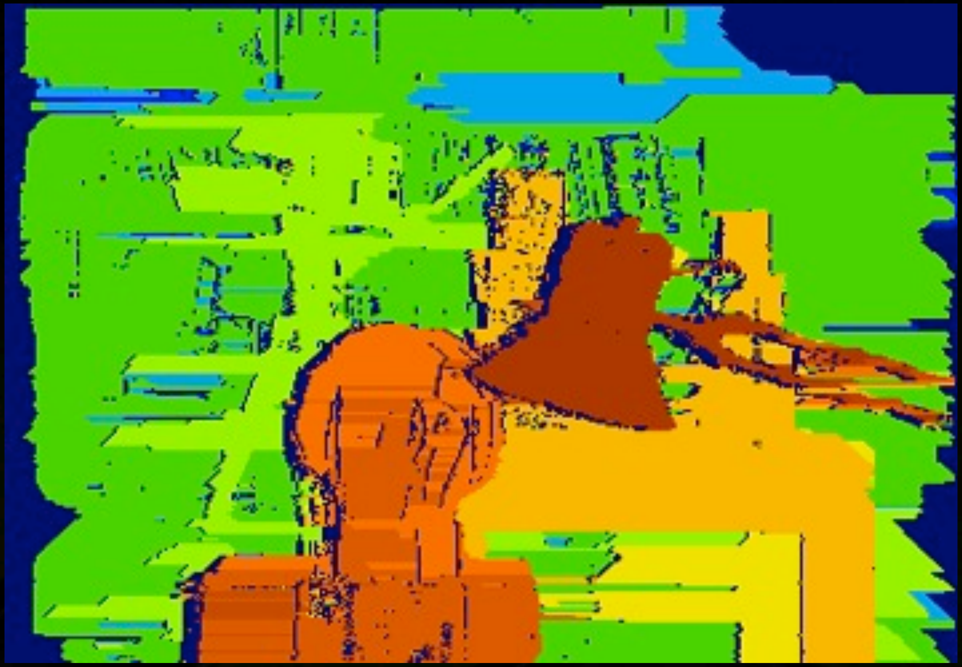
Stereo Matching with Dynamic Programming



- Scan across grid computing optimal cost for each node given its upper-left neighbors
- Backtrack from the terminal to get the optimal path

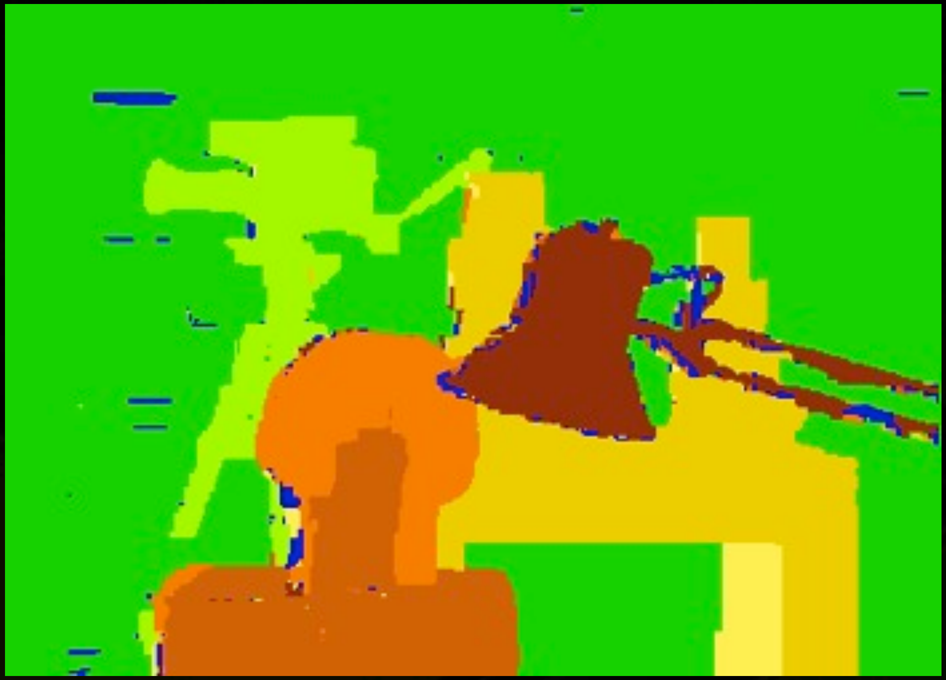
Coherent stereo on 2D grid

- Scanline stereo generates streaking artifacts



- Can’t use dynamic programming to find spatially coherent disparities/ correspondences on a 2D grid

Better methods exist...



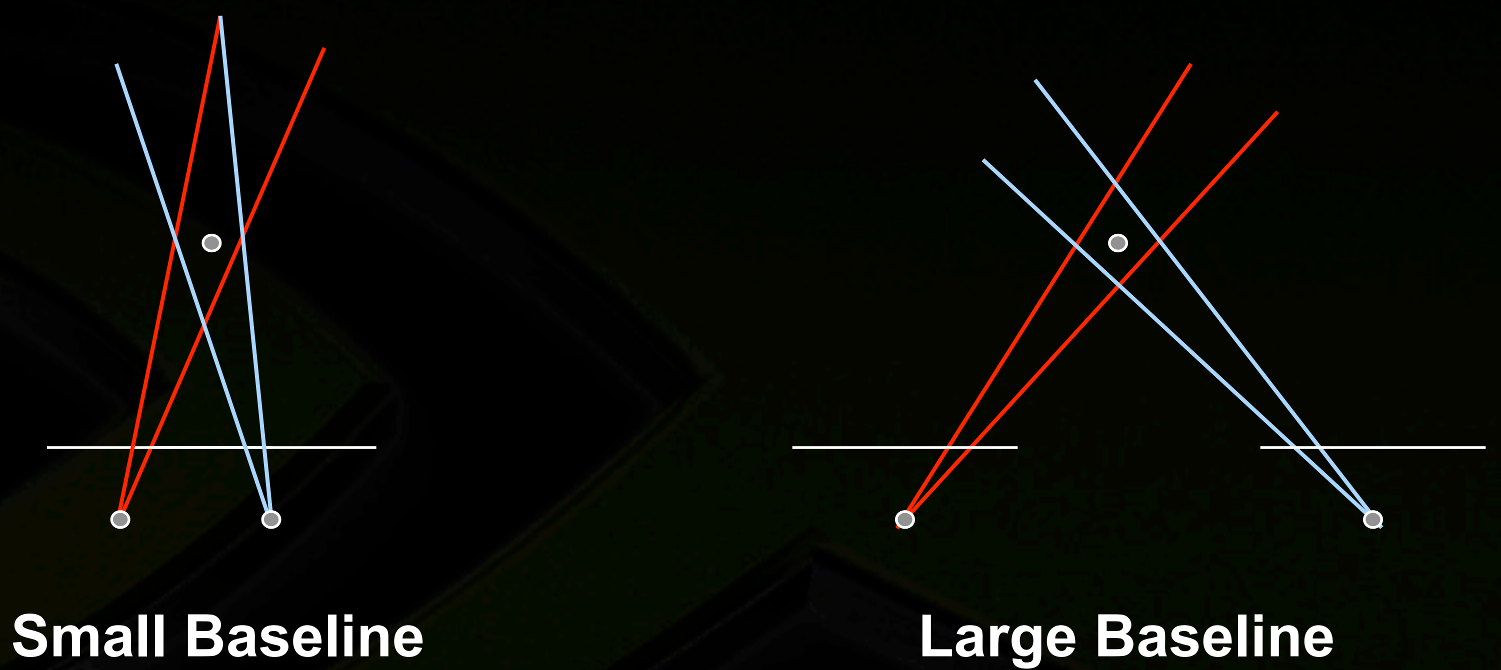
Graph cuts

Ground truth

Y. Boykov, O. Veksler, and R. Zabih, Fast Approximate Energy Minimization via Graph Cuts, PAMI 2001

For the latest and greatest: http://www.middlebury.edu/stereo/

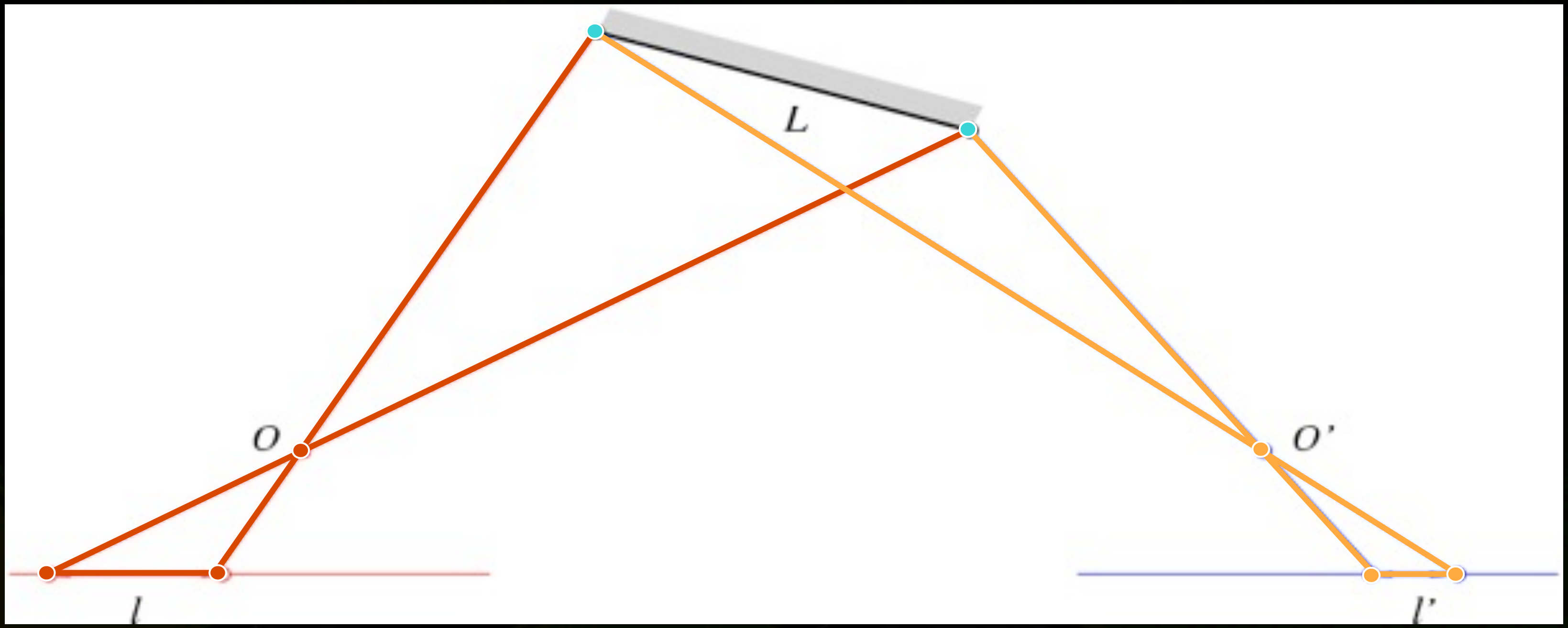
Small baseline → large depth error



NVIDIA Research

Wednesday, March 14, 12

Large baseline → Foreshortening (and occlusions)



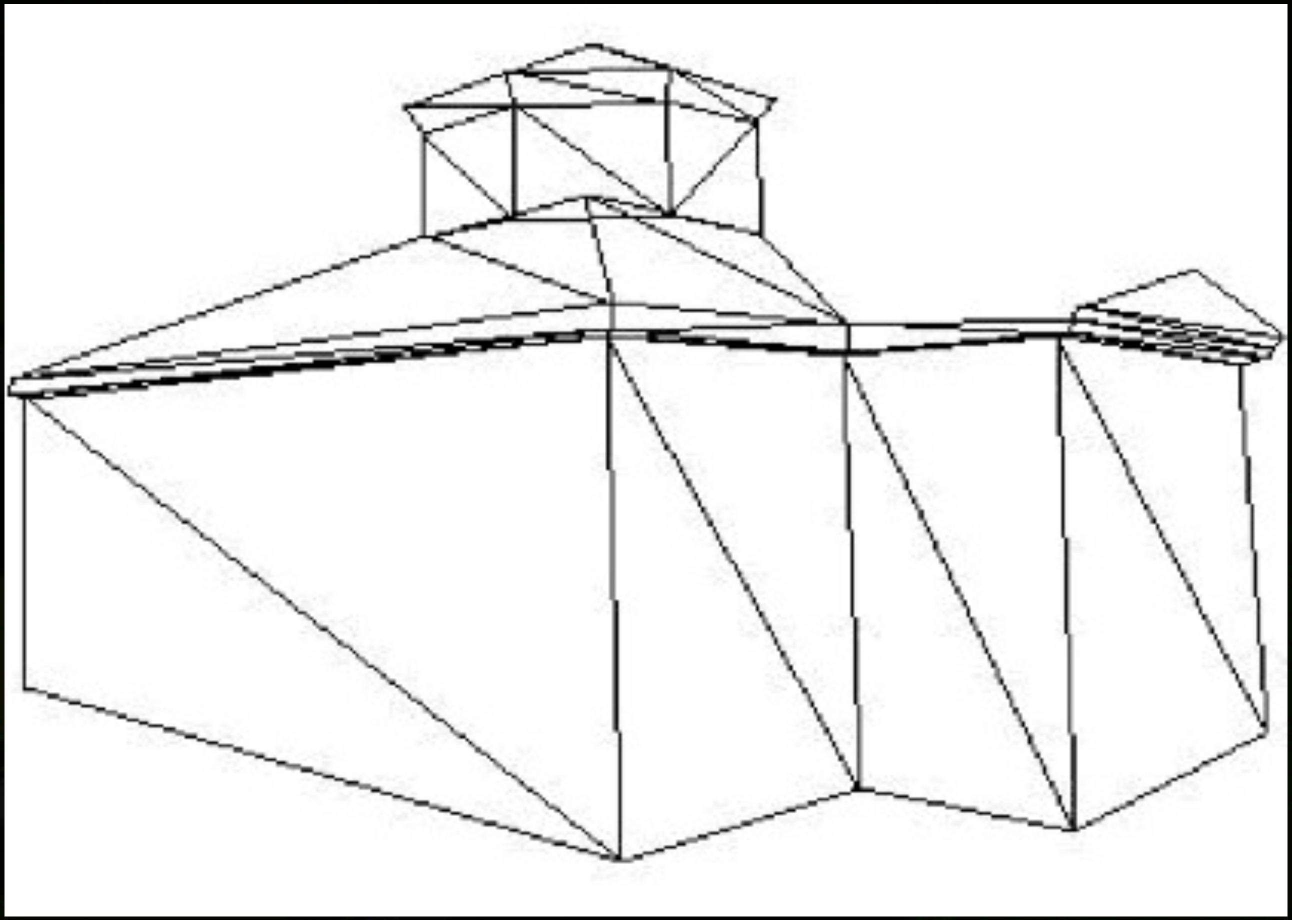
- Matching with fixed-size windows will fail!
- Possible solution: adaptively vary window size
- Another solution: *model-based stereo*

Model-based stereo: Manual modeling

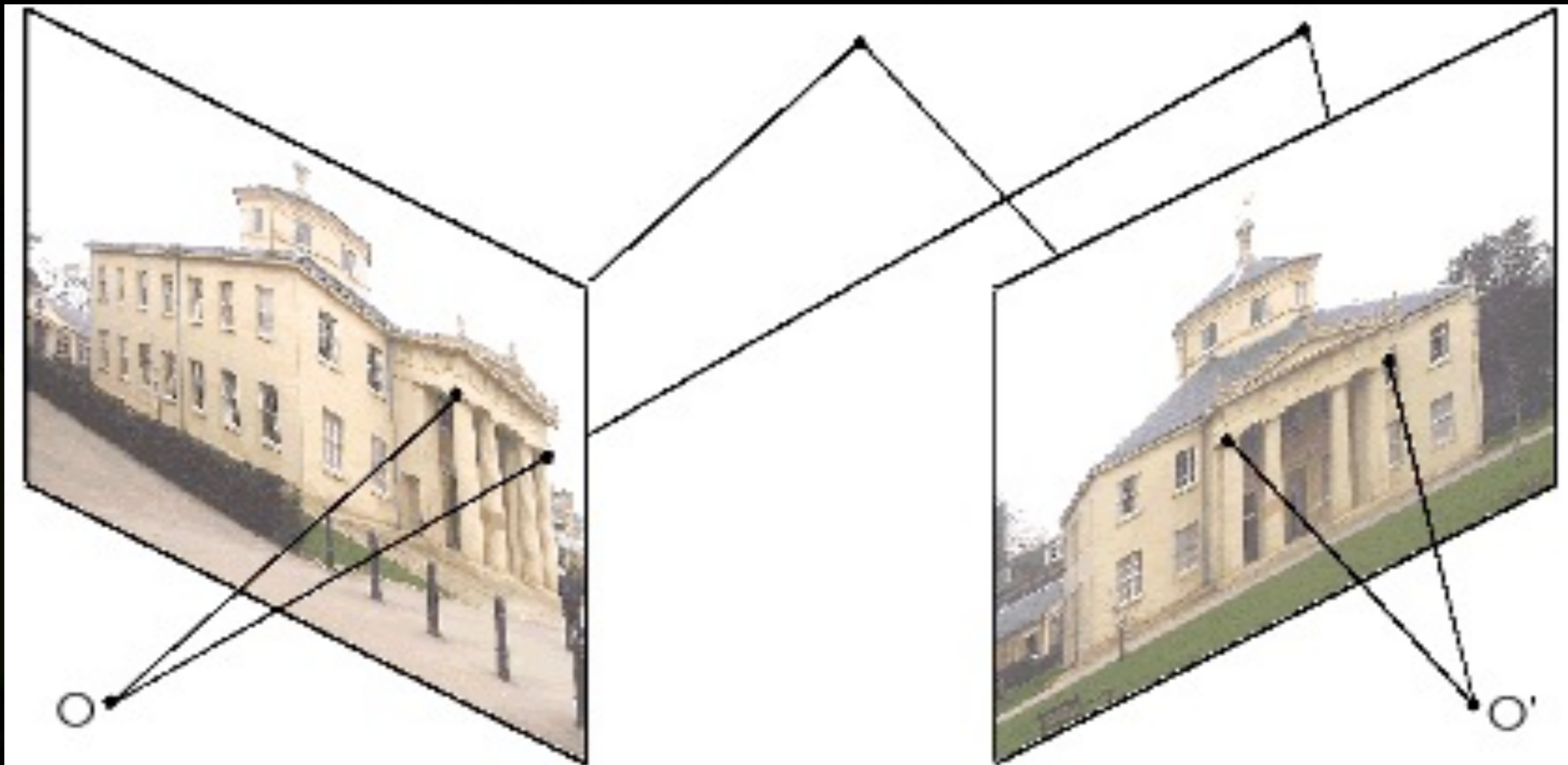
Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and Render
Architecture from Photographs. SIGGRAPH 1996.

Manually select corresponding features

… and define a polygonal model



Bundle adjustment



Add textures

More details from model-based stereo

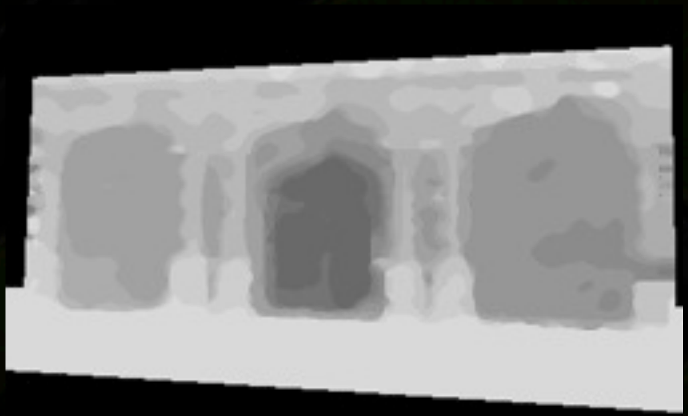
key image

offset image

Model-based stereo

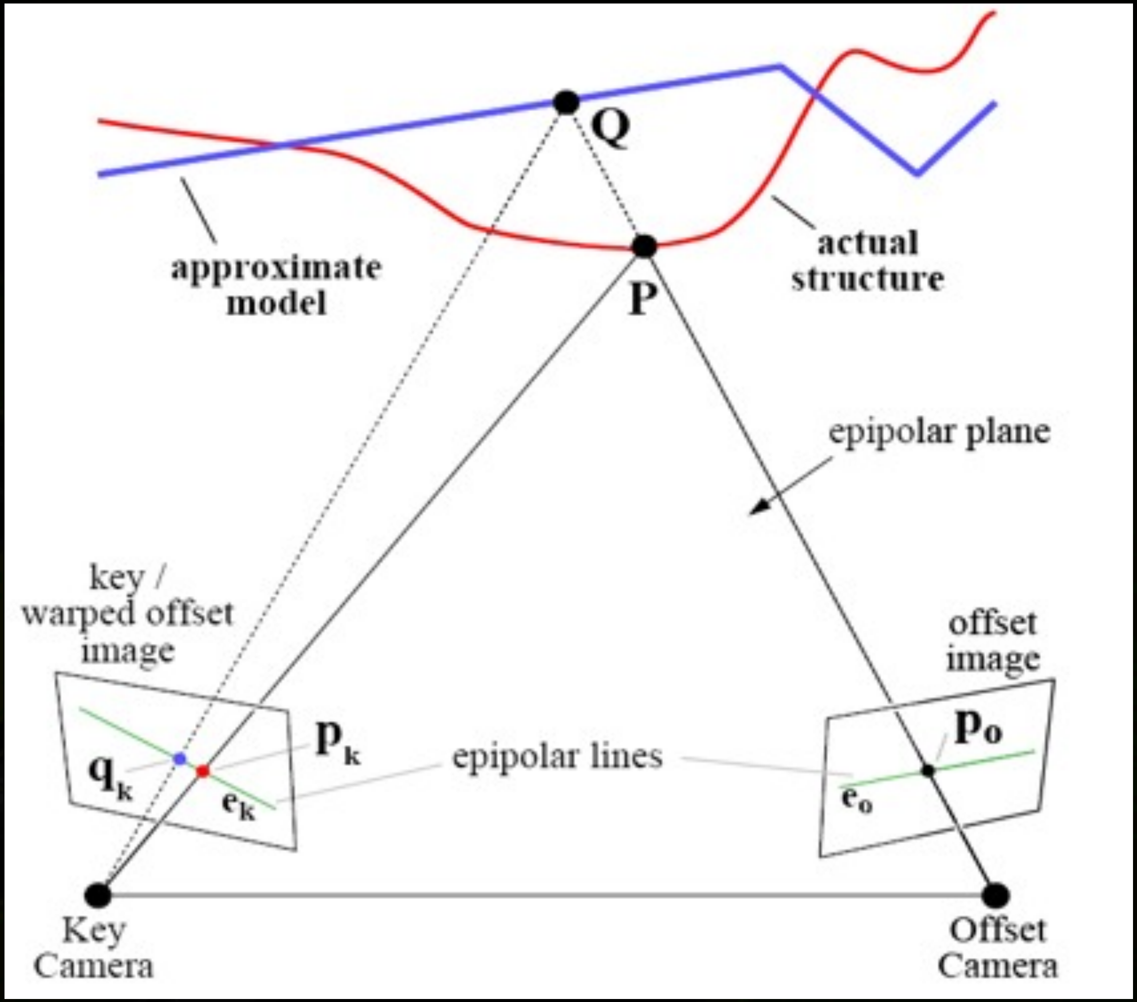
key image

warped offset image

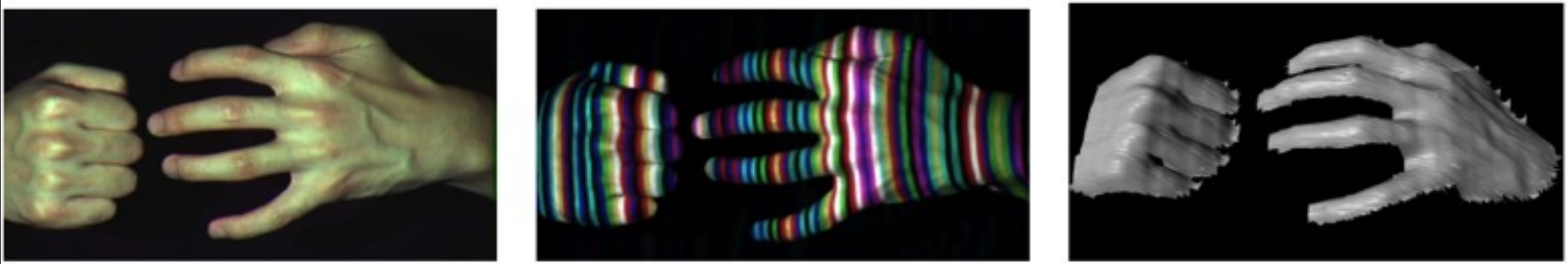


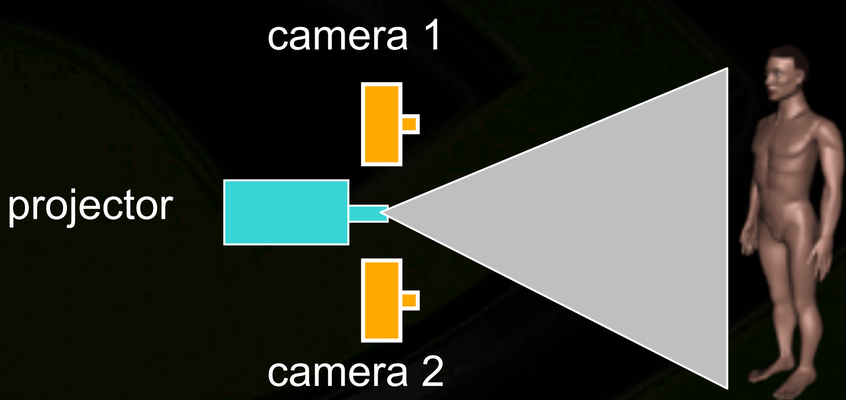
displacement map

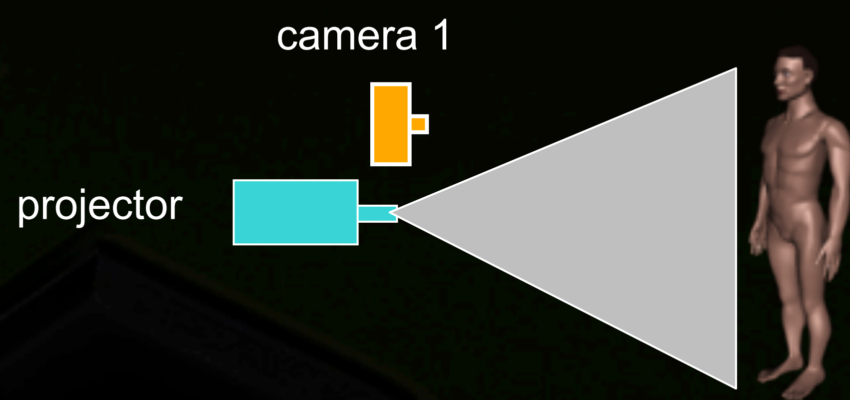
Model-based stereo



Active stereo with structured light







- Project “structured” light patterns onto the object
  - simplifies the correspondence problem

Scanner setup

Dense range (and color) from stereo + active light

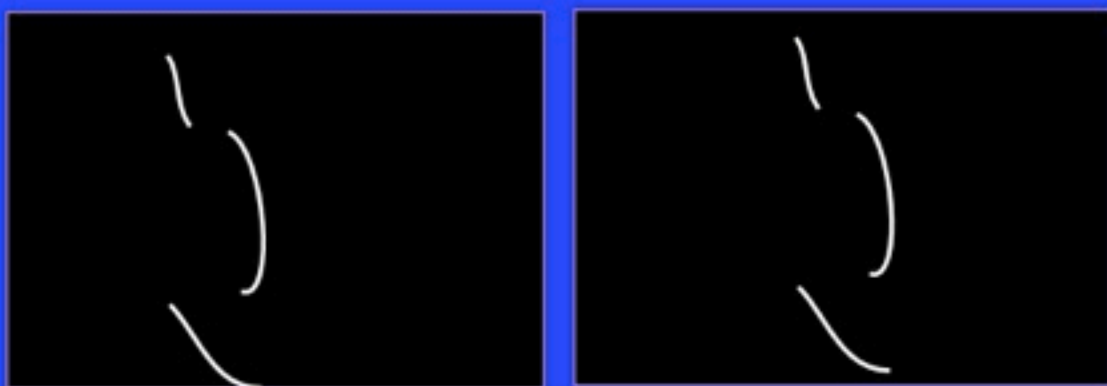
Wednesday, March 14, 12

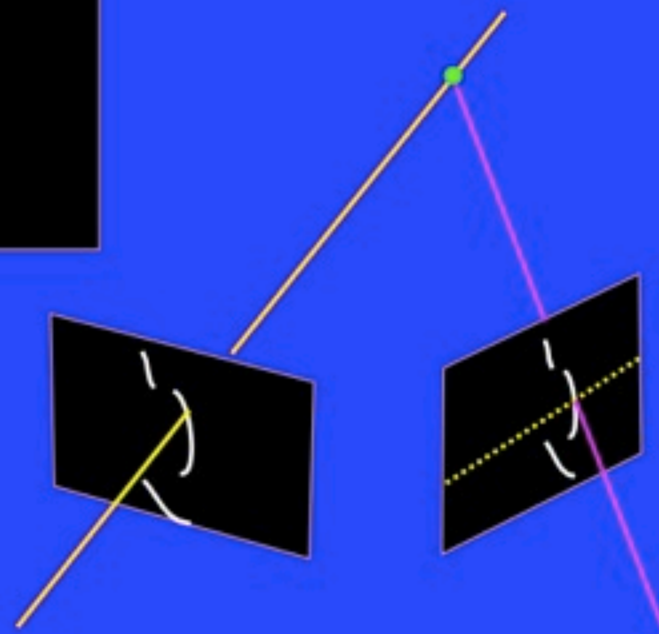
Calibrate cameras

Calibrate cameras from known 3D-2D point correspondences [Tsai '87]

Scanning

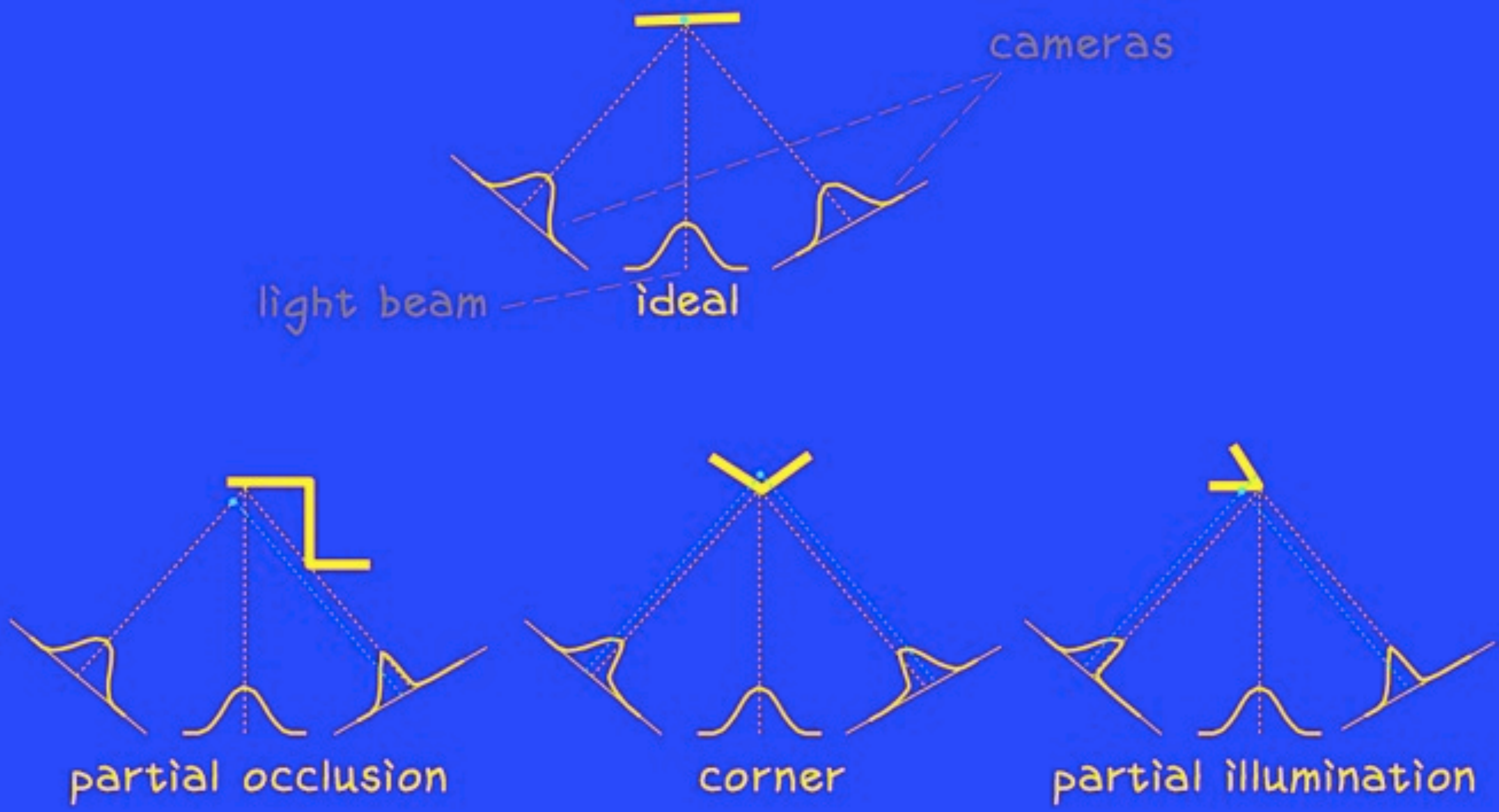
- Sweep a vertical light stripe in small steps
- Detect the stripe from camera images





- Project a line from a pixel to the right image
- Triangulate

Beam location errors

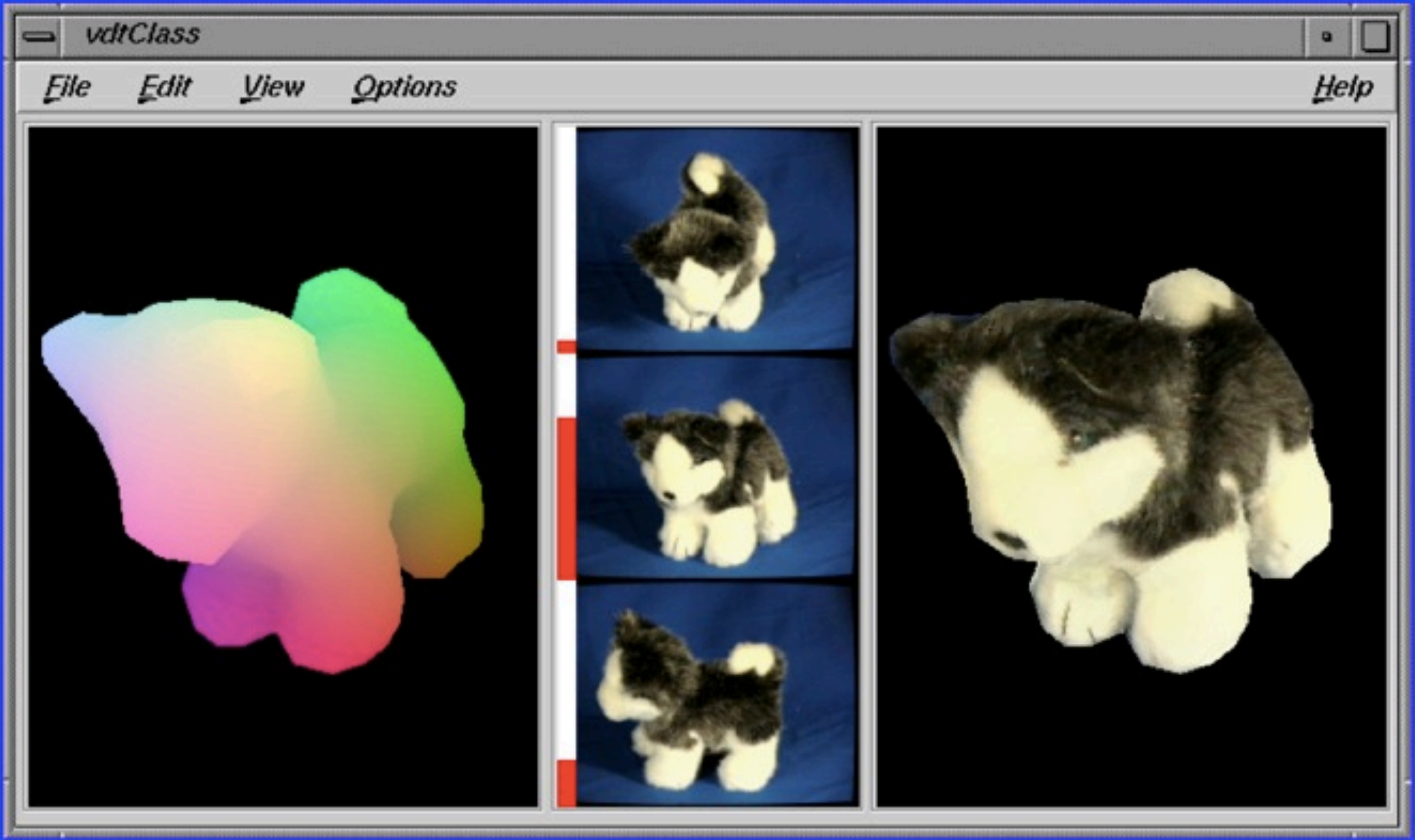


Spacetime analysis

- Don't look at a single image where the beam is centered
- Look at a sequence of images when the beam was centered at each pixel

- Invert the function to locate beam in each image [Curless and Levoy '95]

Our viewer



Video View Interpolation

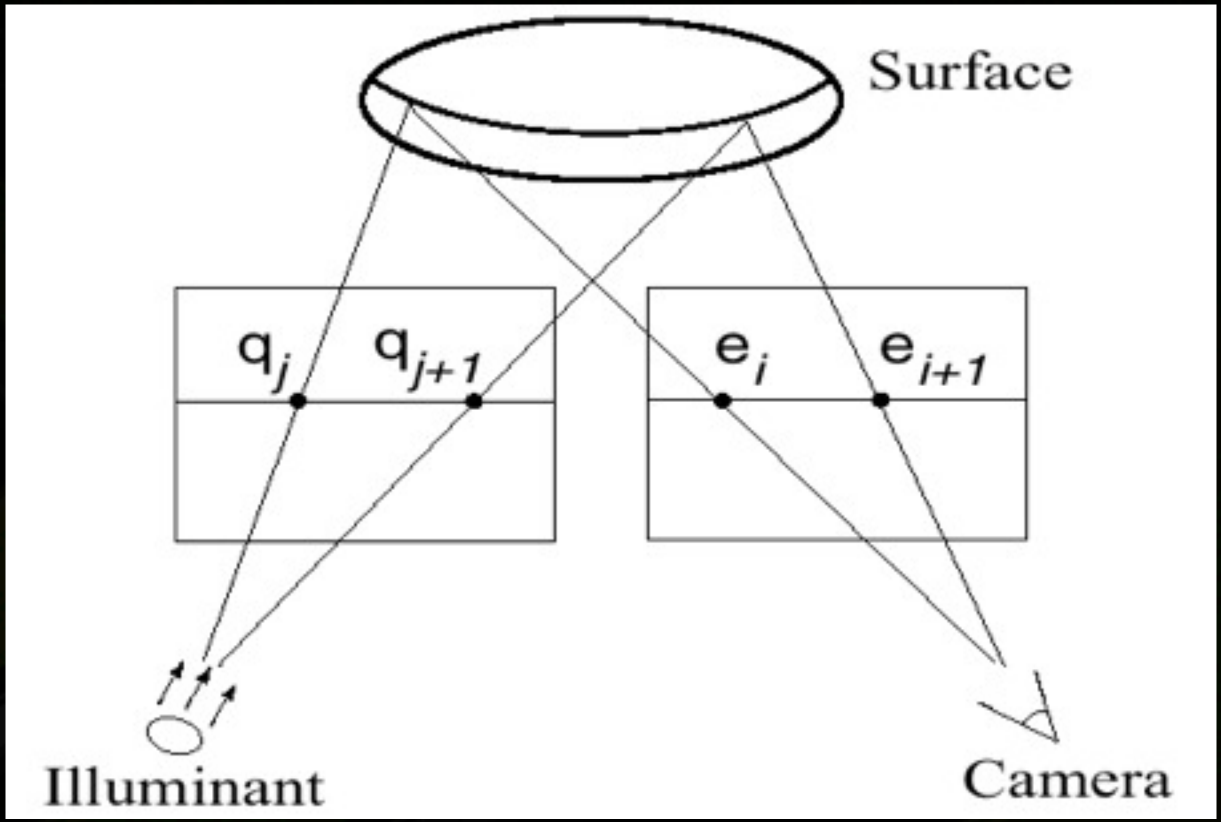
- http://research.microsoft.com/users/larryz/videoviewinterpolation.htm

Left View (recorded) | Virtual Interpolated View (not recorded) | Right View (recorded)

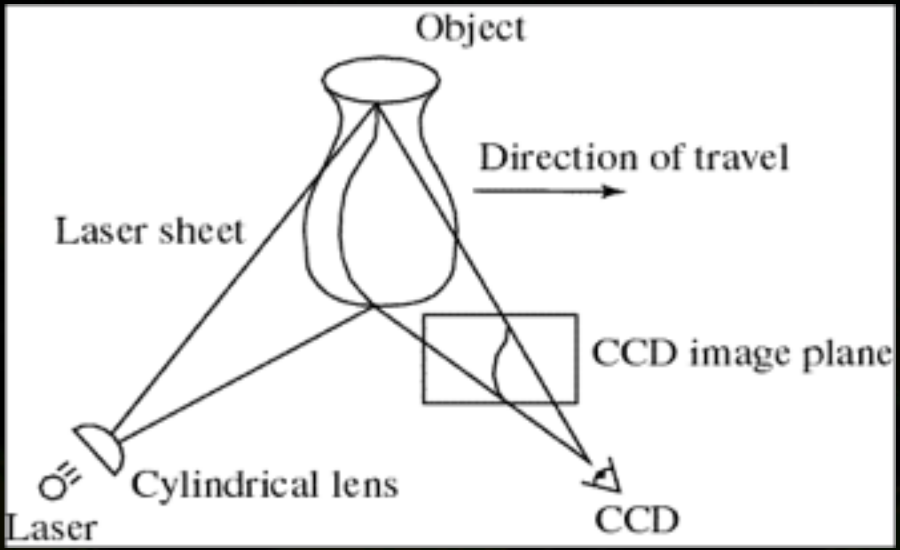
- Given a 3D model of the scene, one can use a virtual camera to record new views from arbitrary viewpoints
- For example: Freeze frame effect from the movie Matrix
- [L. Zitnick, S. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation", SIGGRAPH, 2004]

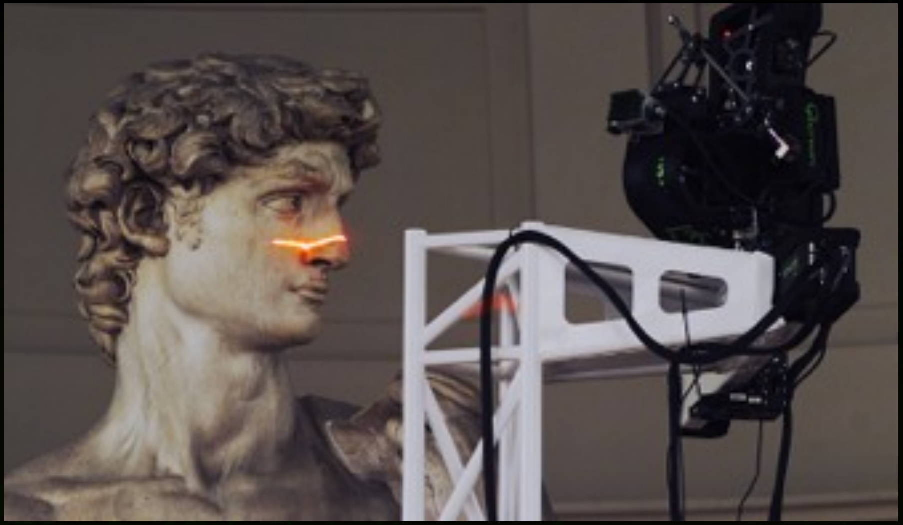

Created with Flip4Mac WMV Demo
www.Flip4Mac.com

Active stereo with structured light



Laser scanning

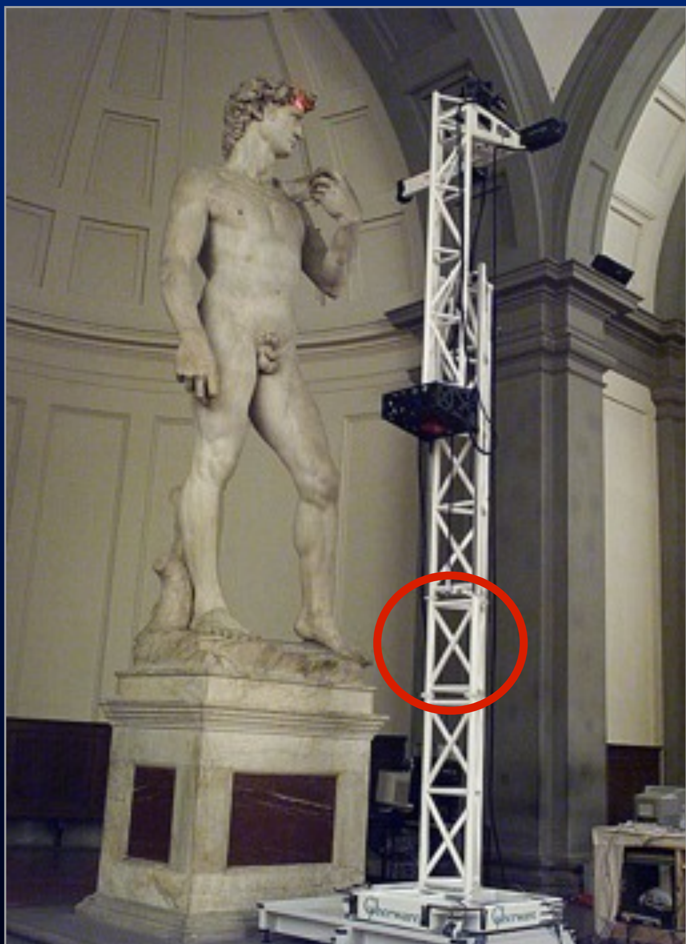


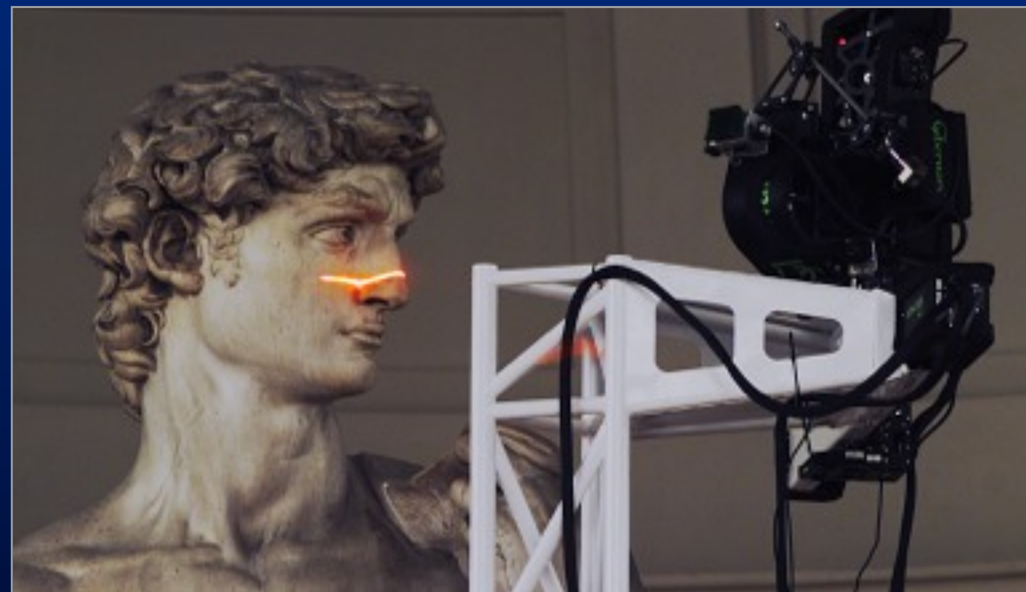


Digital Michelangelo Project
http://graphics.stanford.edu/projects/mich/

- 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

Scanning the David



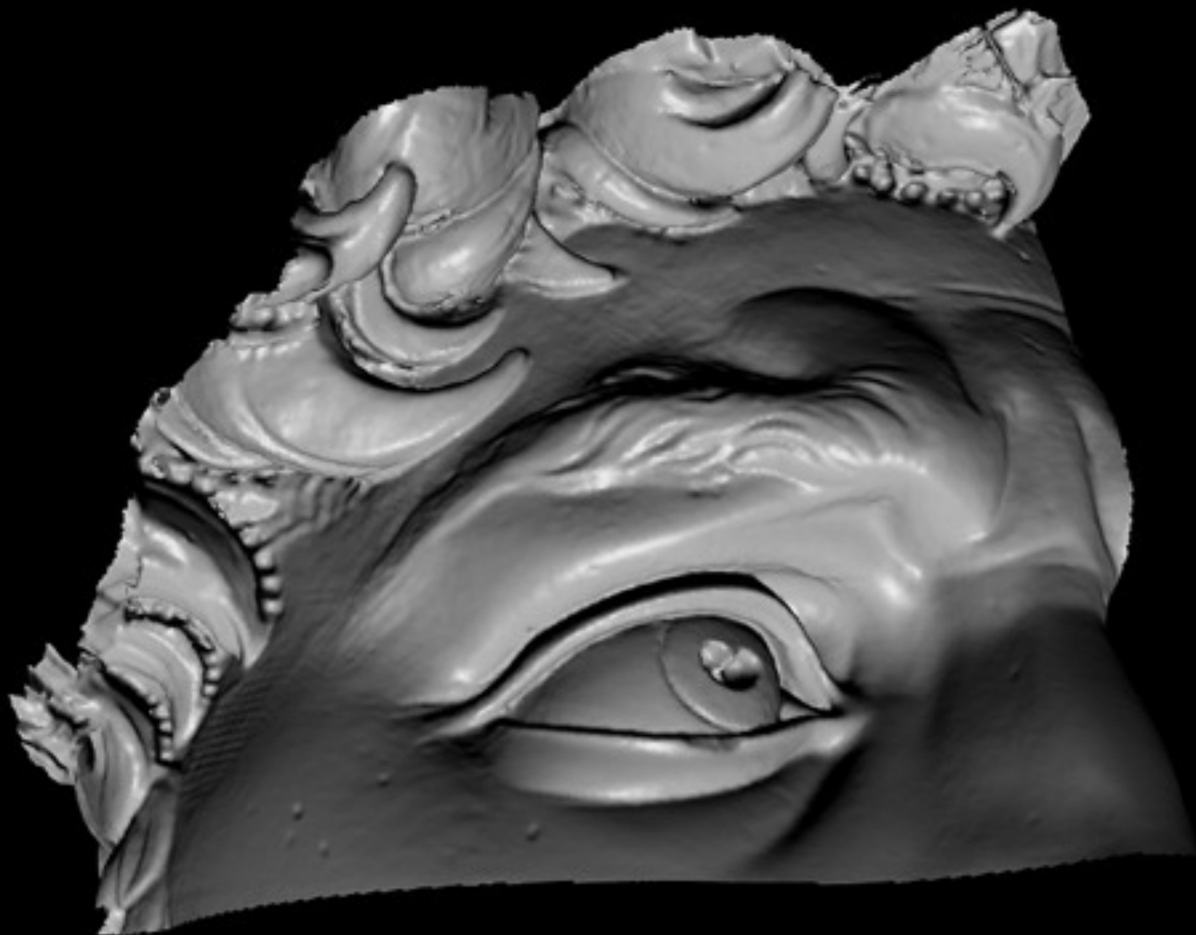


height of gantry: 7.5 meters
weight of gantry: 800 kilograms

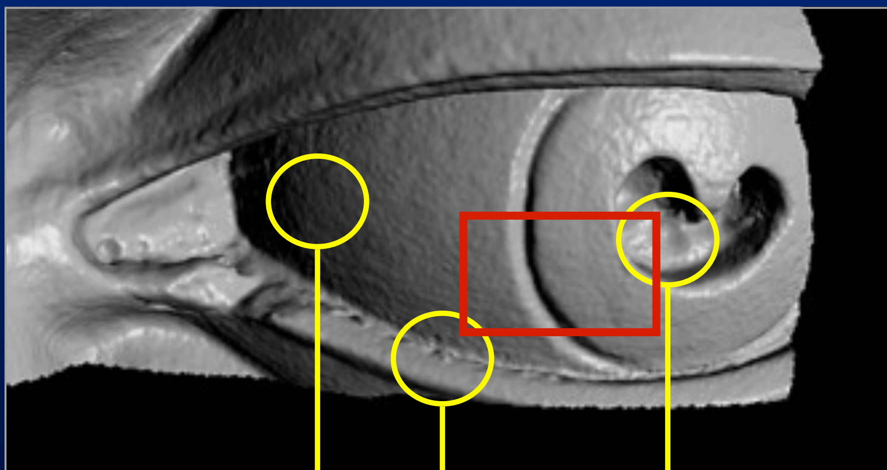
Statistics about the scan

- 480 individual scans
- 2 billion polygons
- 7,000 color images
- 32 gigabytes
- 30 nights of scanning
- 1,080 man-hours
- 22 people

NVIDIA Research



David’s left eye



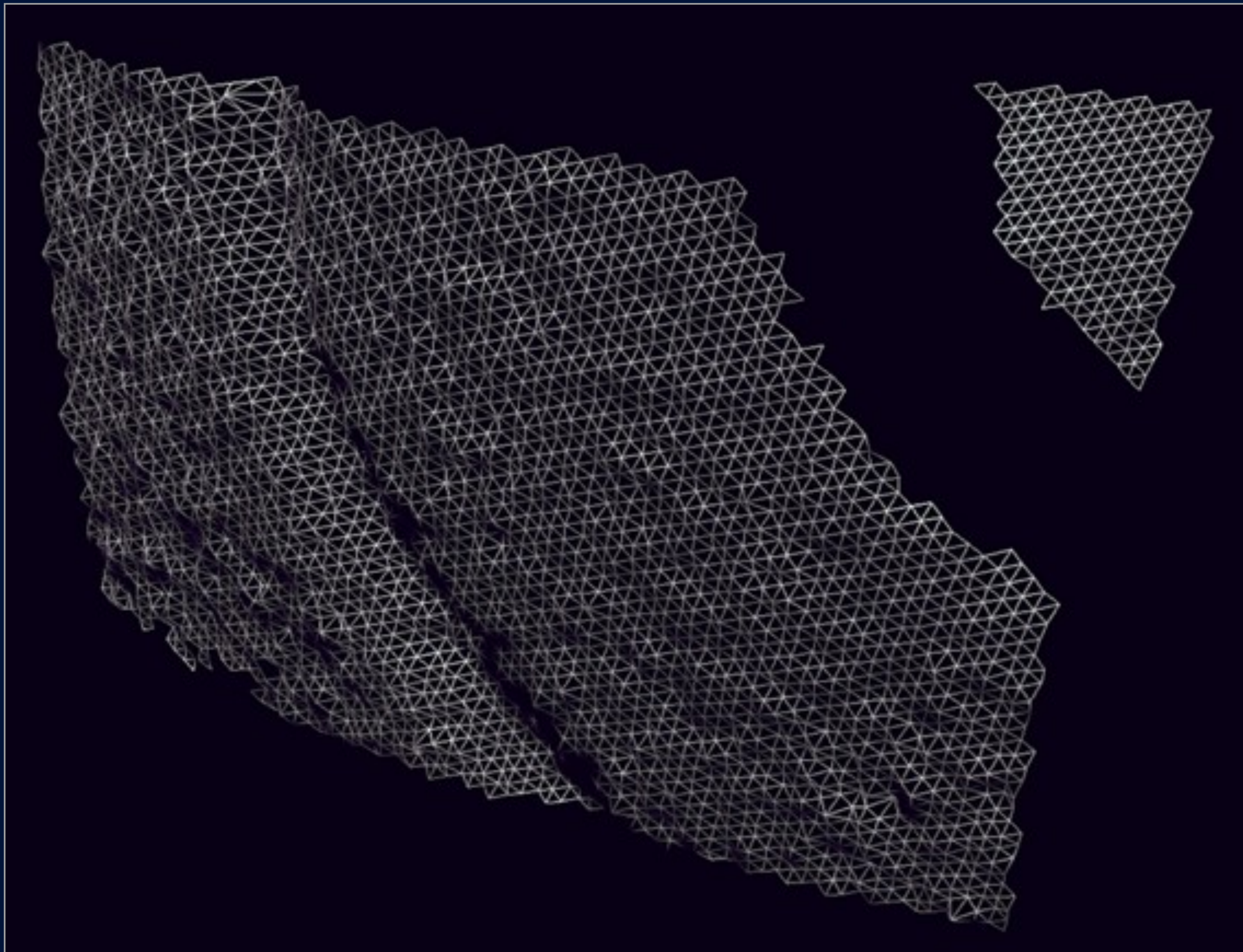
0.25 mm model

photograph

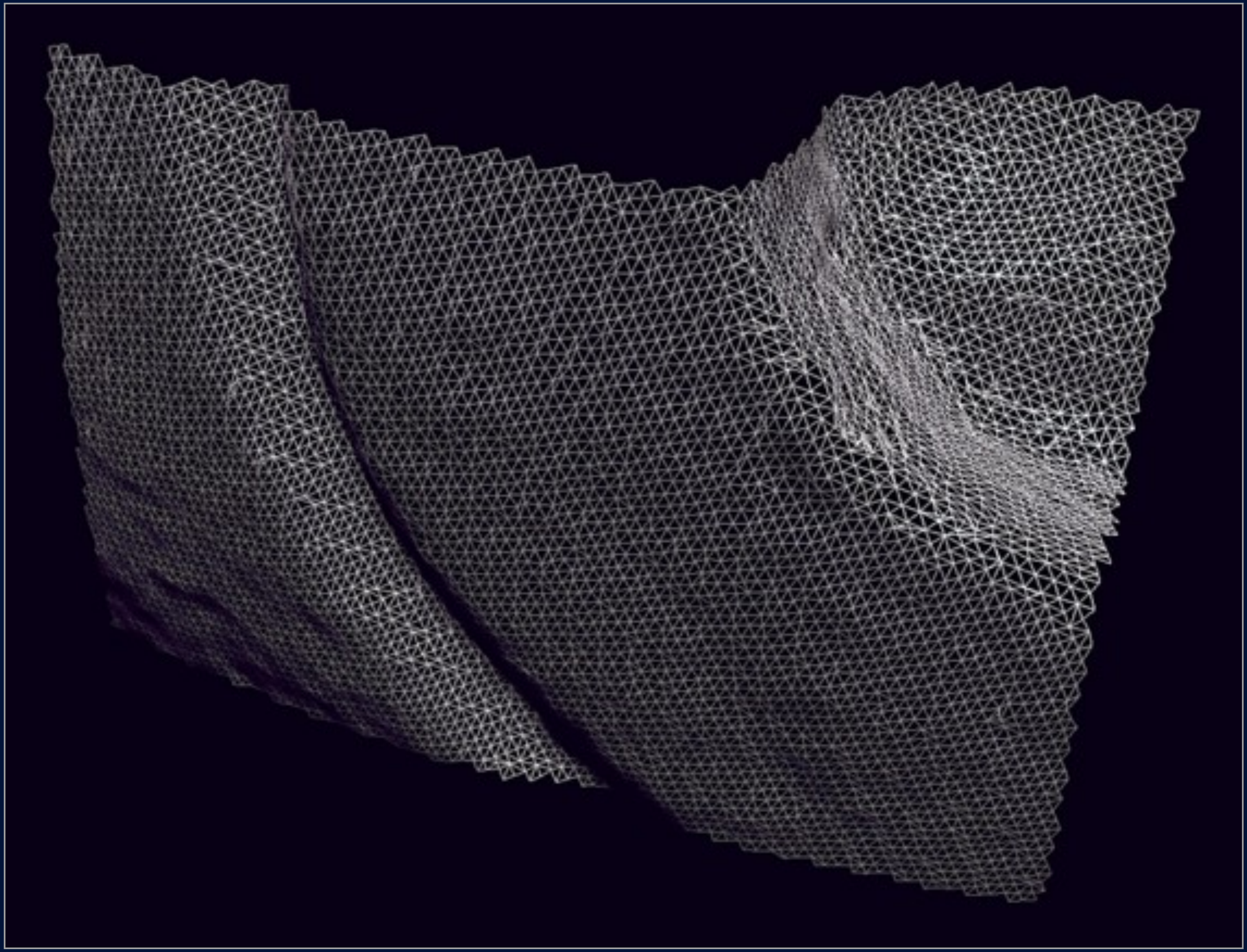
holes from Michelangelo’s drill

artifacts from space carving

noise from laser scatter



Single scan of David’s cornea



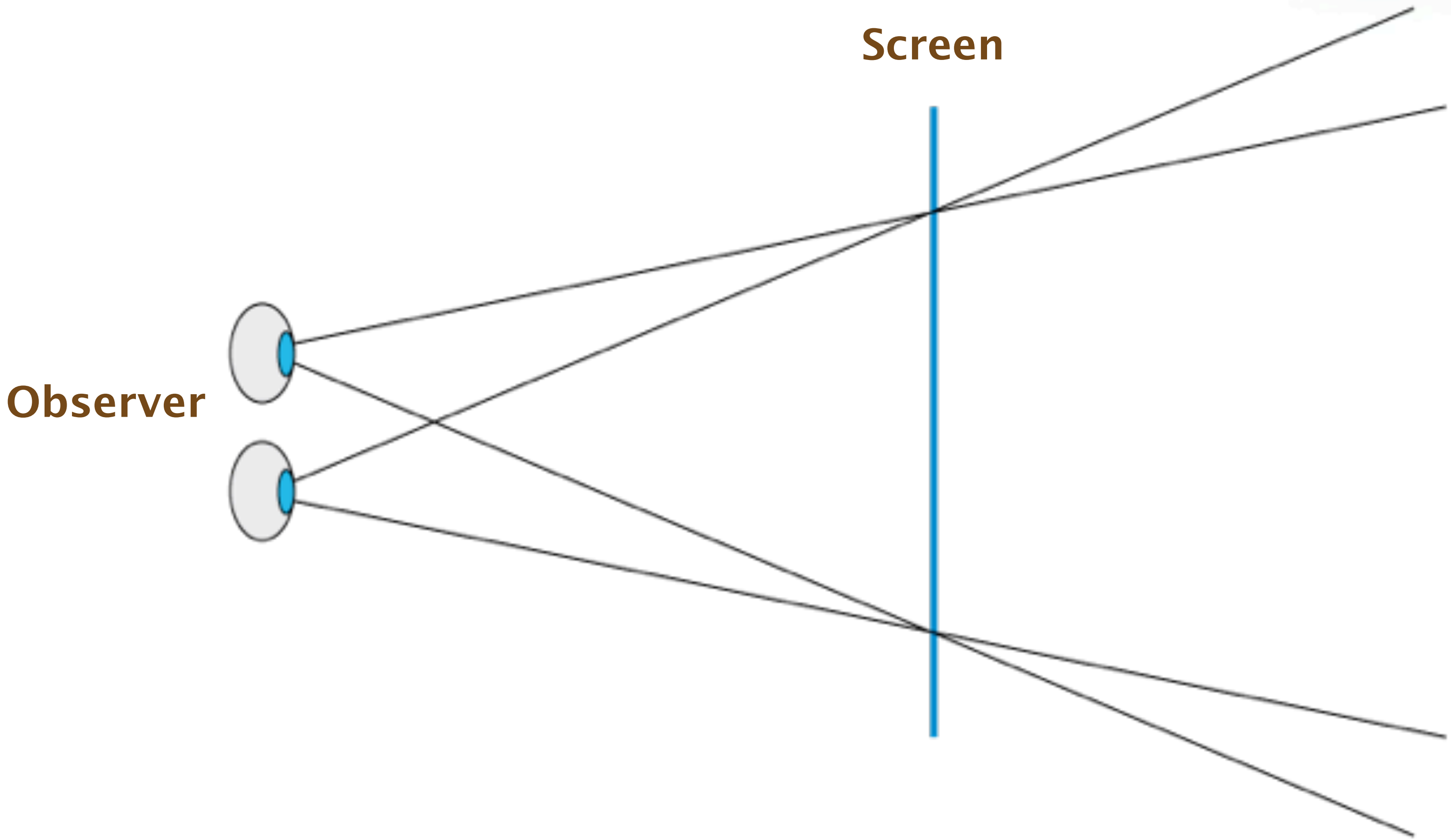
Mesh constructed from several scans

Computational Stereo Camera System with Programmable Control Loop

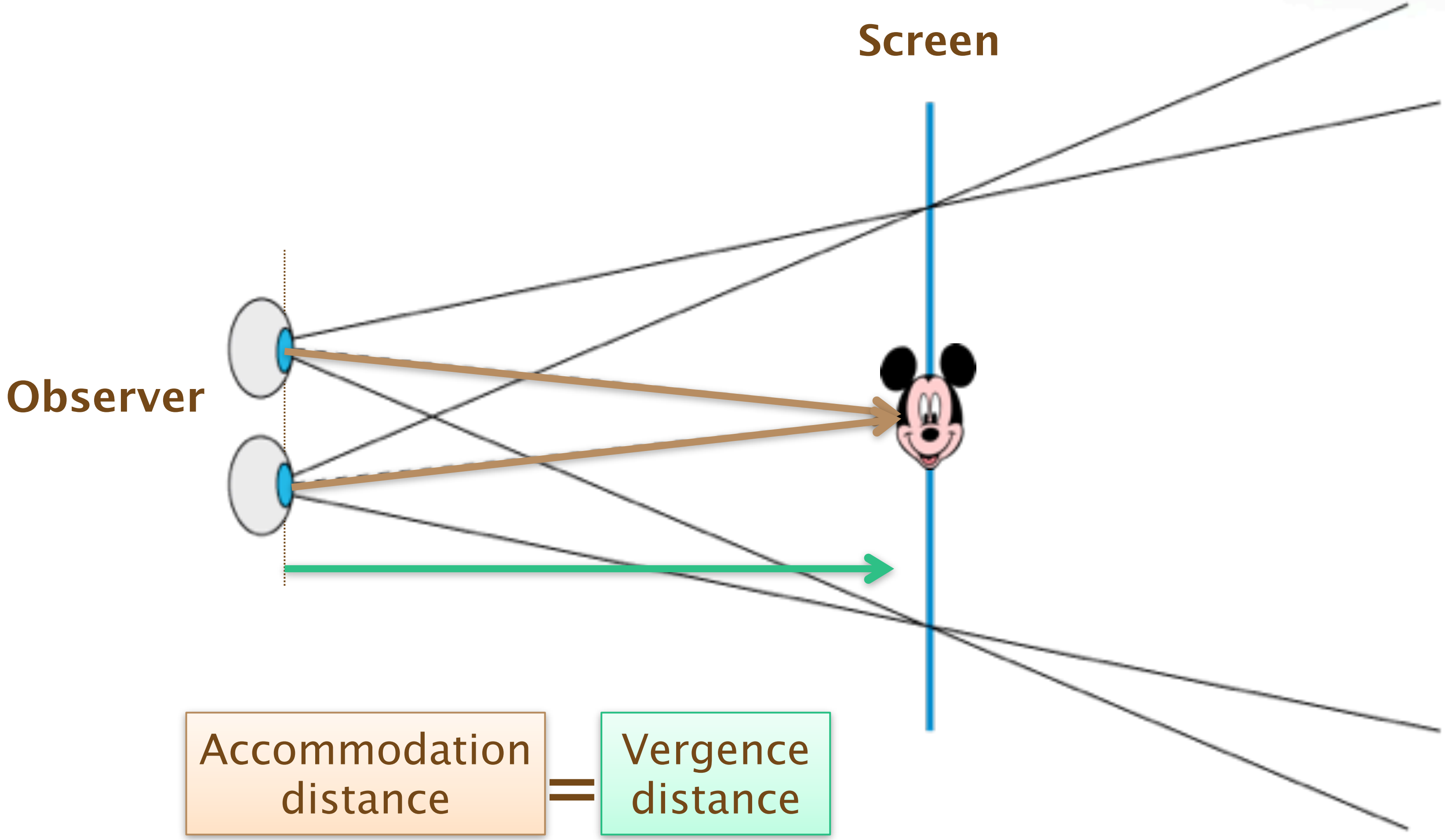
Simon Heinzle[1] Pierre Greisen[1;2] David Gallup[3]
Christine Chen[1] Daniel Saner[2] Aljoscha Smolic[1]
Andreas Burg[2] Wojciech Matusik[1] Markus Gross[1;2]

1 Disney Research, Zurich
2 ETH Eidgenössische Technische Hochschule Zürich Swiss Federal Institute of Technology Zurich
3 The University of North Carolina at Chapel Hill

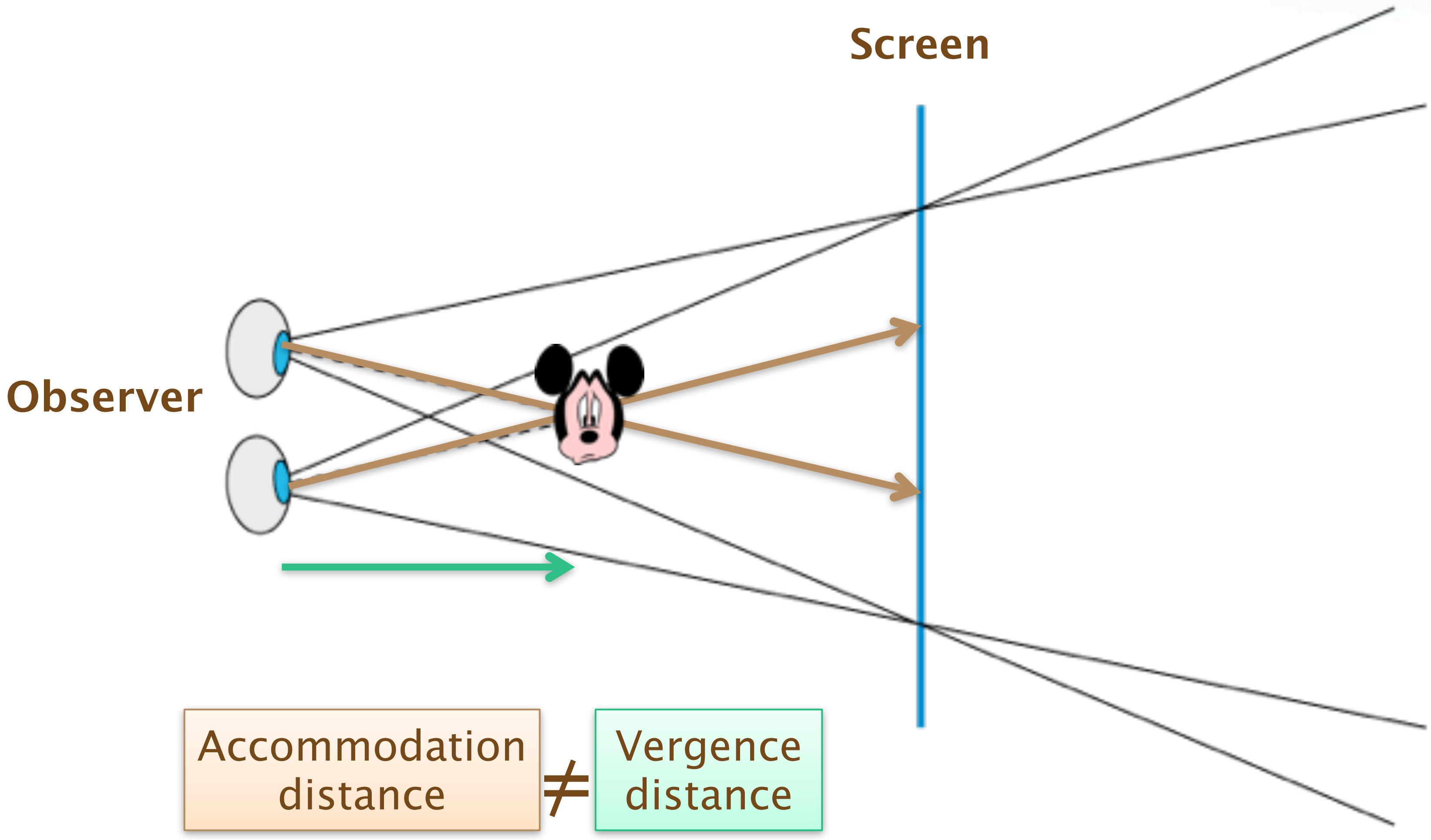
Comfort zone



Comfort zone

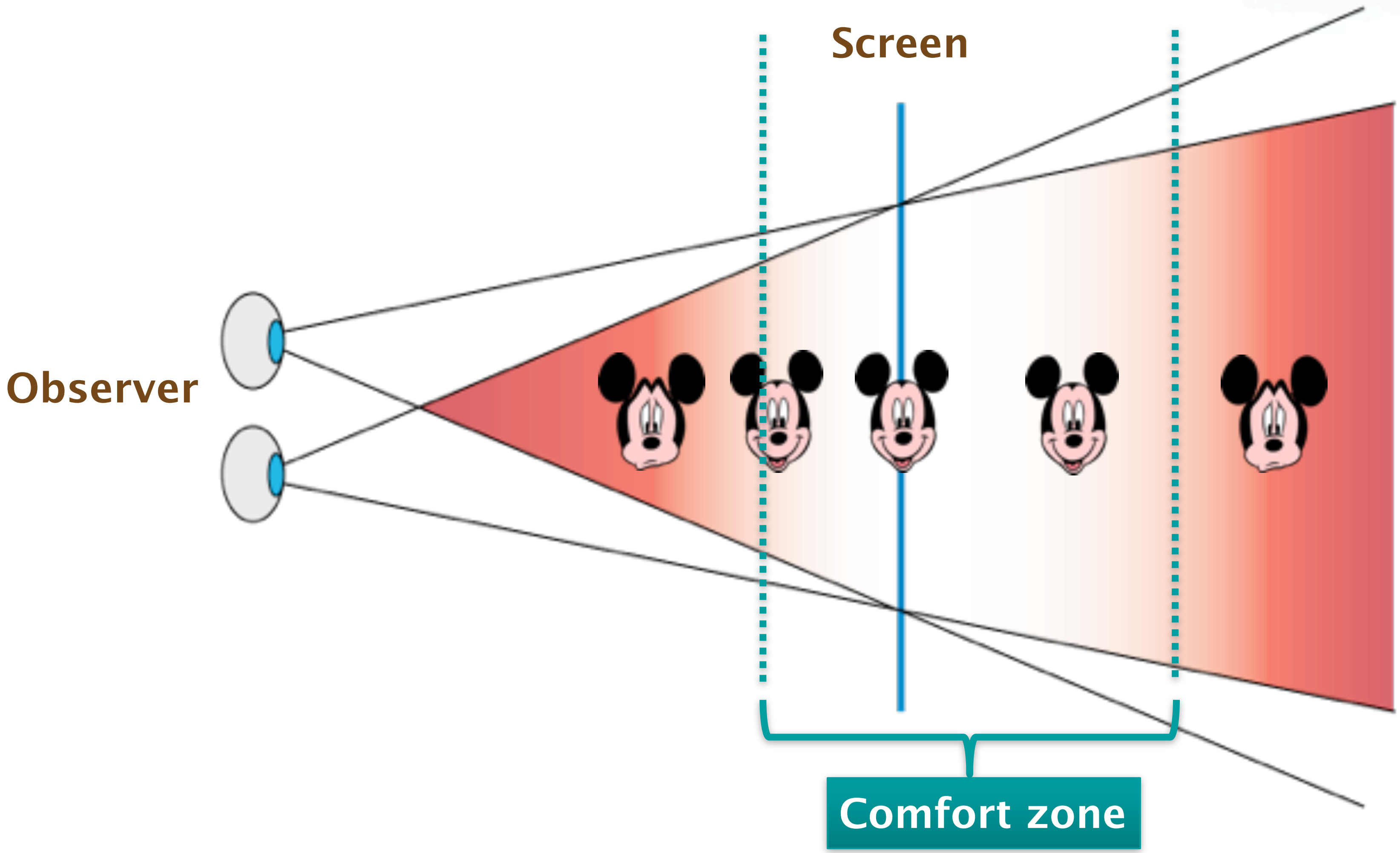


Comfort zone



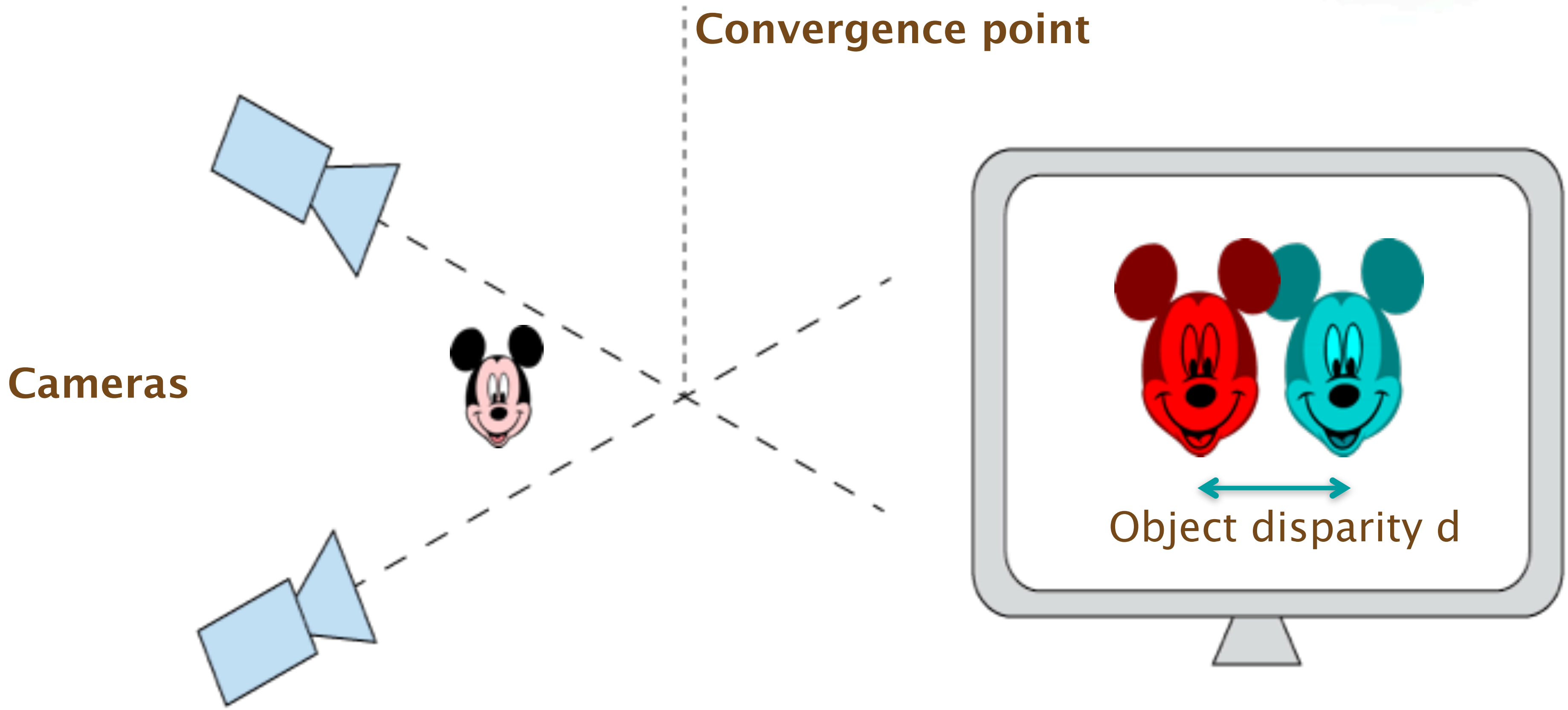
Vergence-accomodation conflict

Comfort zone



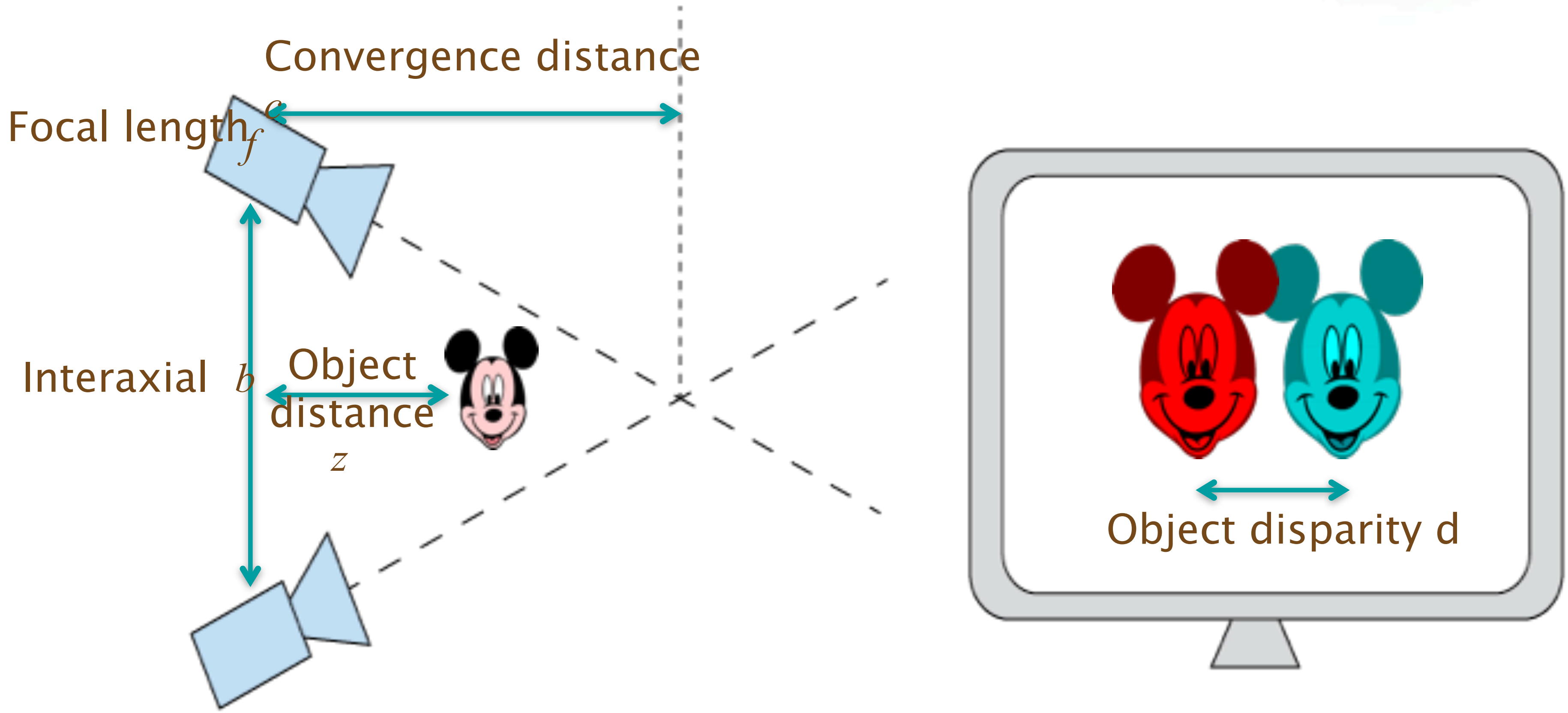
Vergence-accomodation conflict defines maximum 'pop-out' effect

Screen space disparity



- Disparity induces 3D illusion
- Directly relates to comfort zone
- Central input for our controller

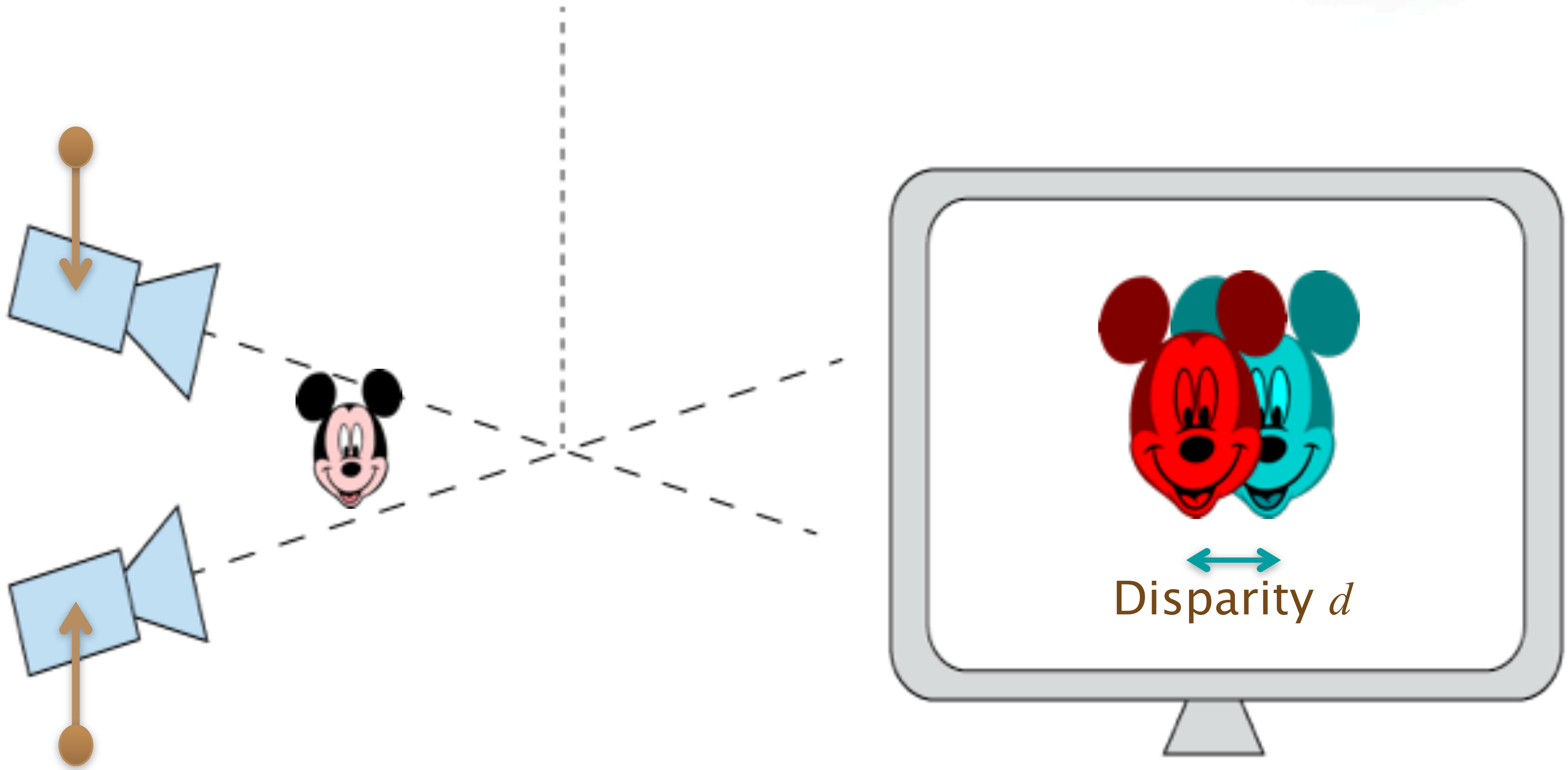
Disparity calculation



- Disparity induces 3D illusion
- Directly relates to comfort zone
- Central input for our controller

$$d \approx -f \times b \times \left(\frac{1}{z} - \frac{1}{c}\right)$$

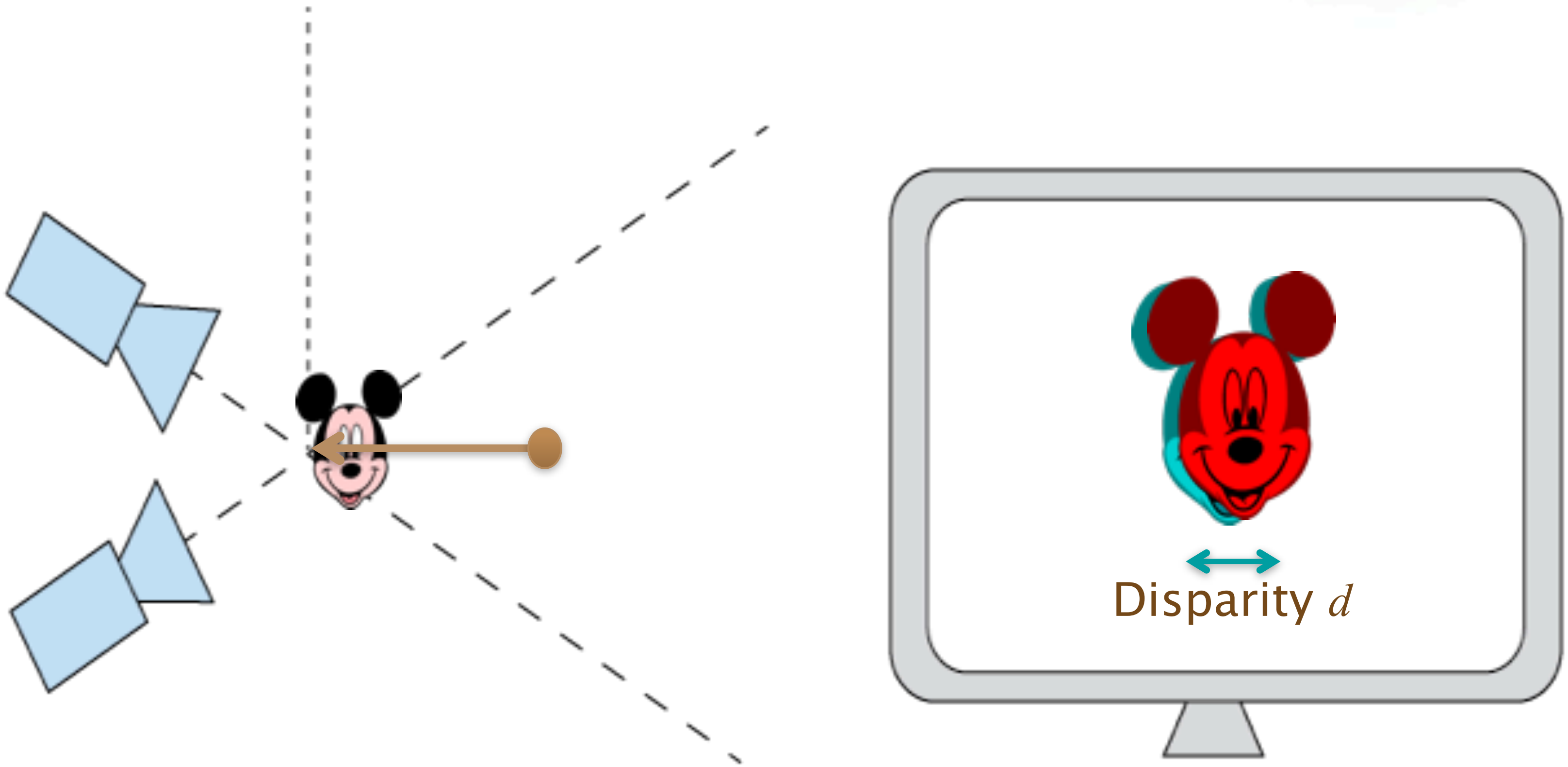
Baseline change



$$d \approx -f \times b \times \left(\frac{1}{z} - \frac{1}{c} \right)$$

Baseline change 'stretches' disparities

Convergence change



$$d \approx -f \times b \times \left(\frac{1}{z} - \frac{1}{c} \right)$$

Convergence ‘shifts’ disparities:

Disparity range limits

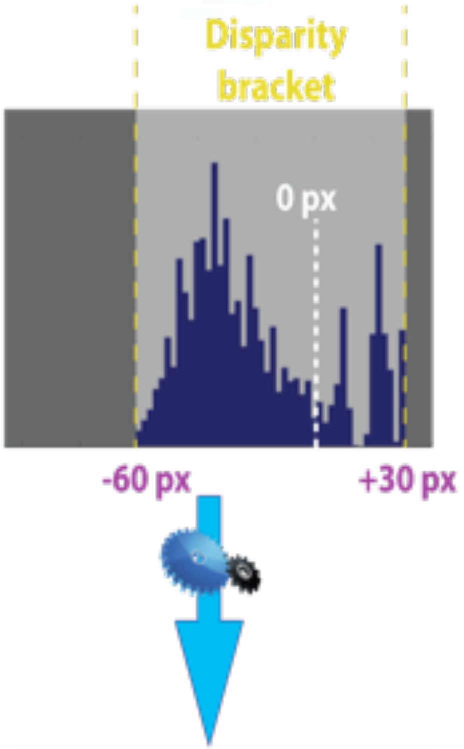
Disparity range (degree)	Sensation
0～0.5	Comfortable
0.5～1	Small fatigue
1～2	Fatigue
2～3	Pain
3～	No stereoscopic fusion

Disparity-based control

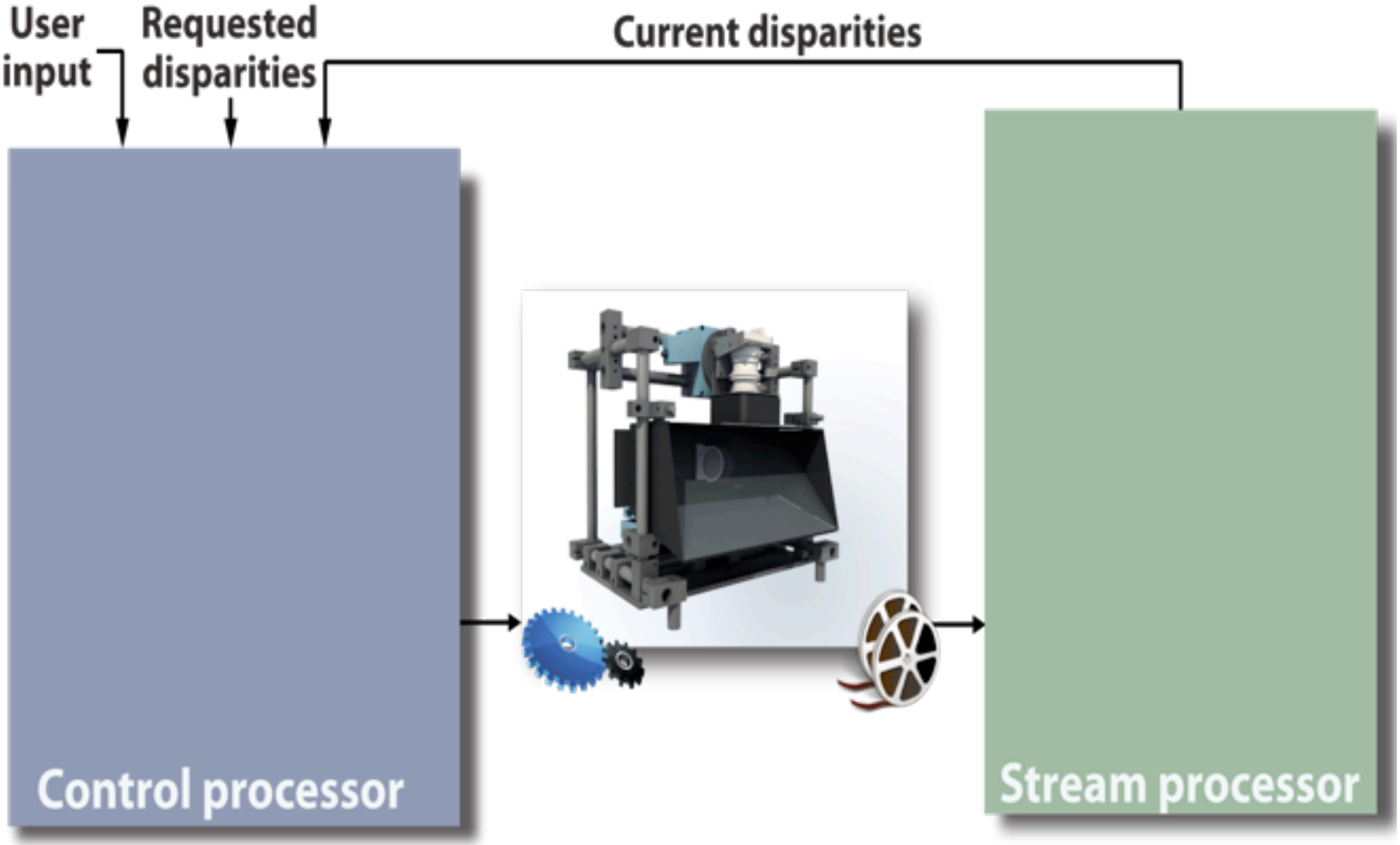
- Comfort zone disparity bracket

$$d_{min} < d < d_{max}$$

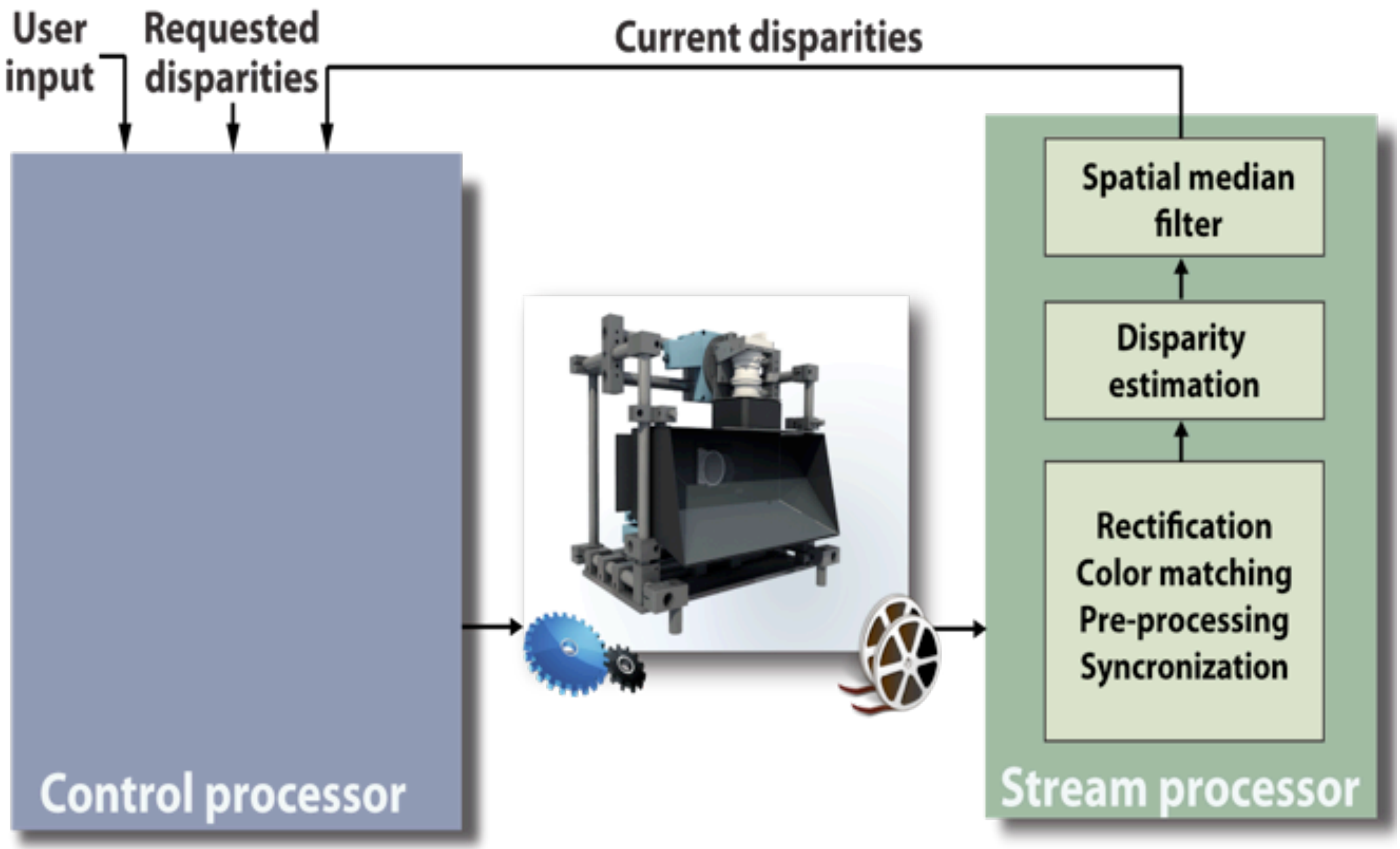
- Idea:
  - Measure disparities
  - Adjust camera parameters
    - vergence
    - baseline



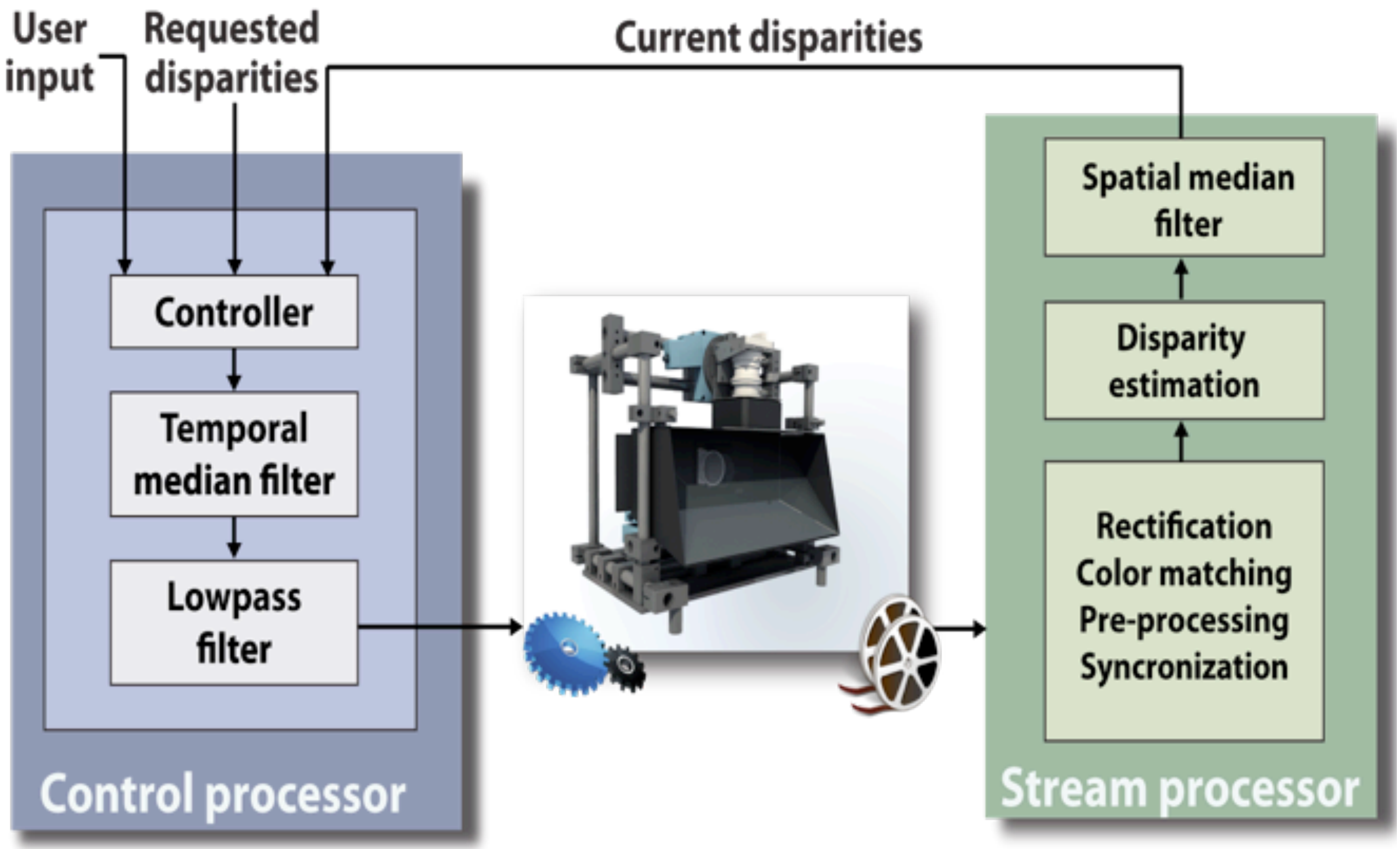
Disparity-based control



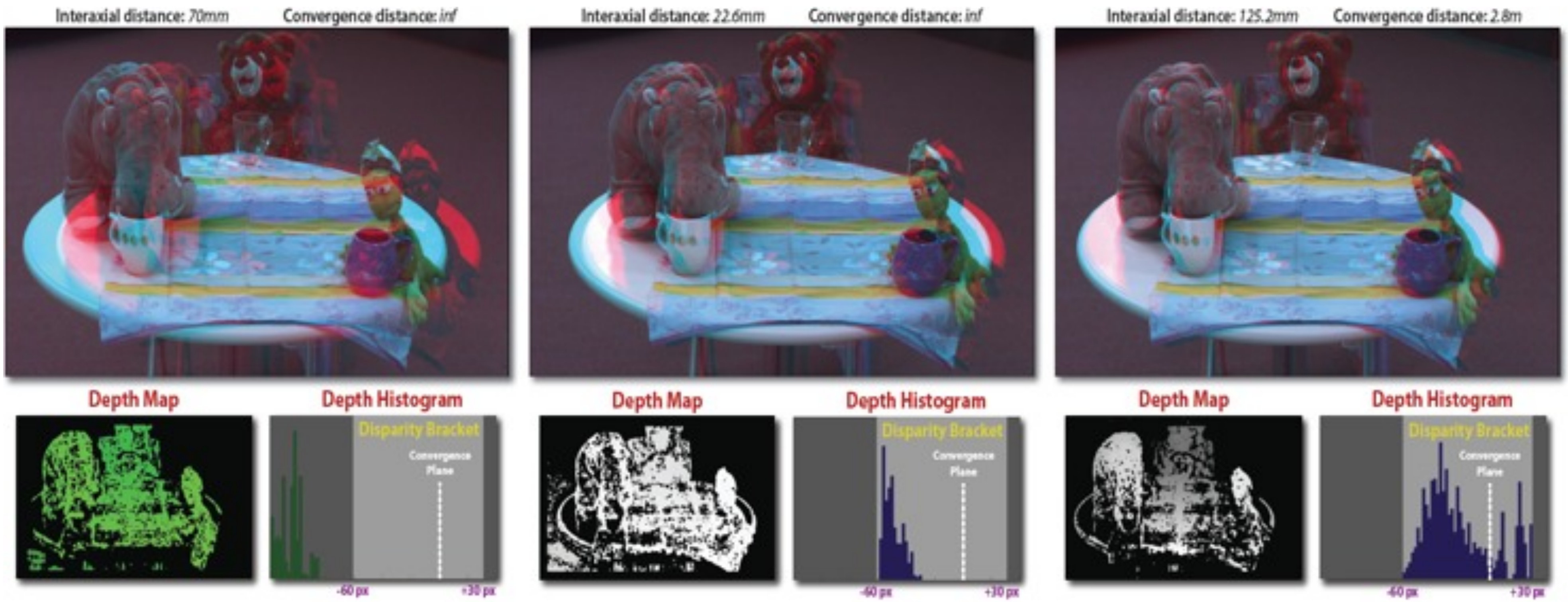
Disparity-based control



Disparity-based control



Automatic control example



Start condition
Disparities violate comfort zone

Automatic interaxial only
Disparities compressed, negative disparities only

Automatic interaxial and convergence
Disparities distributed

Tracking example

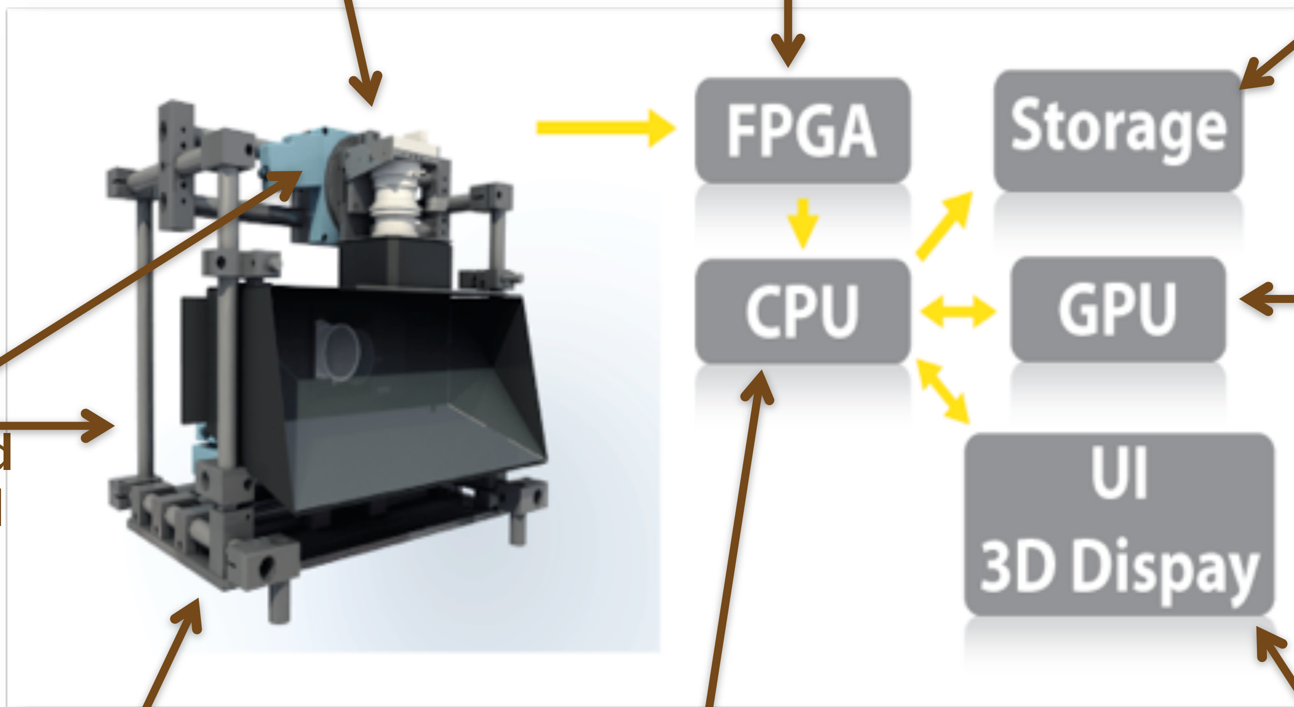
Touch focus and convergence

Hardware architecture

DSLR-sized camera sensor
with motorized optics

FPGA video preprocessing
(Stratix III 340L)

RAID array
for data storage
(uncompressed)



GPU for
disparity
estimation
(NVIDIA GTX
480)

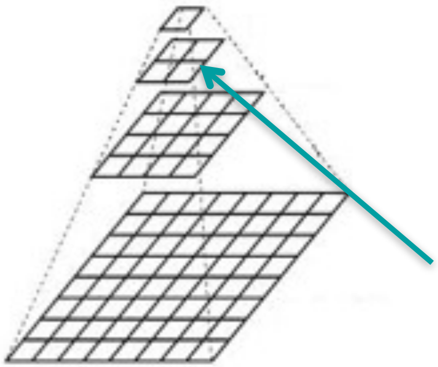
Linear and
rotational
stages

Beam-splitter rig

Camera control algorithms
and tracking
(Intel i7, 6 cores)

Multi-touch UI
3D display

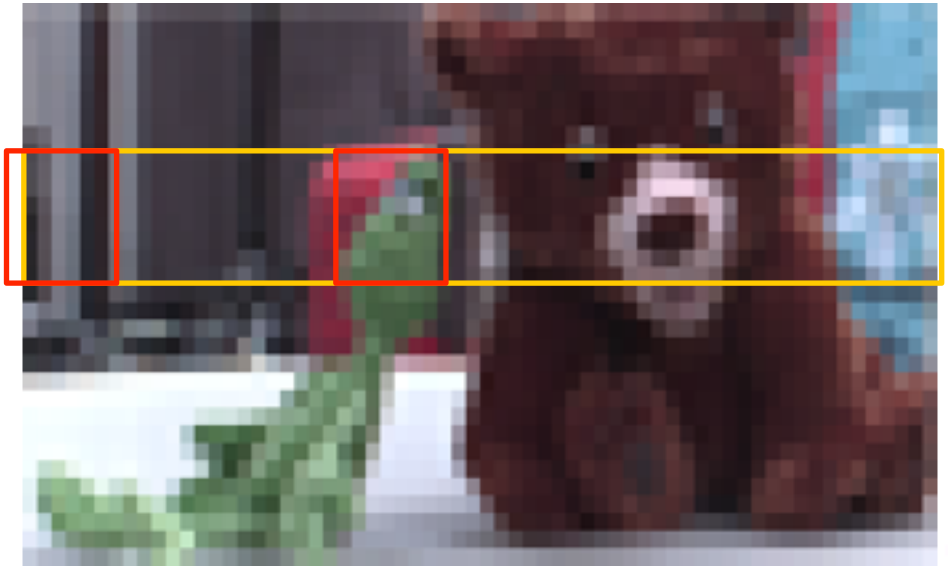
Hierarchical disparity



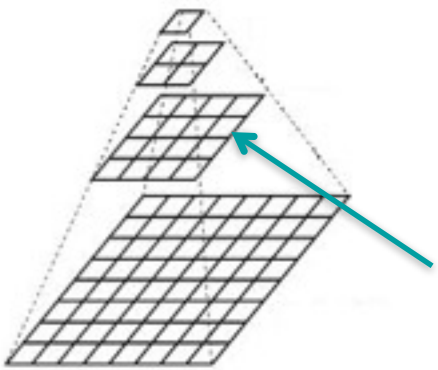
Left, level 2:

Perform initial matching

Right, level 2:



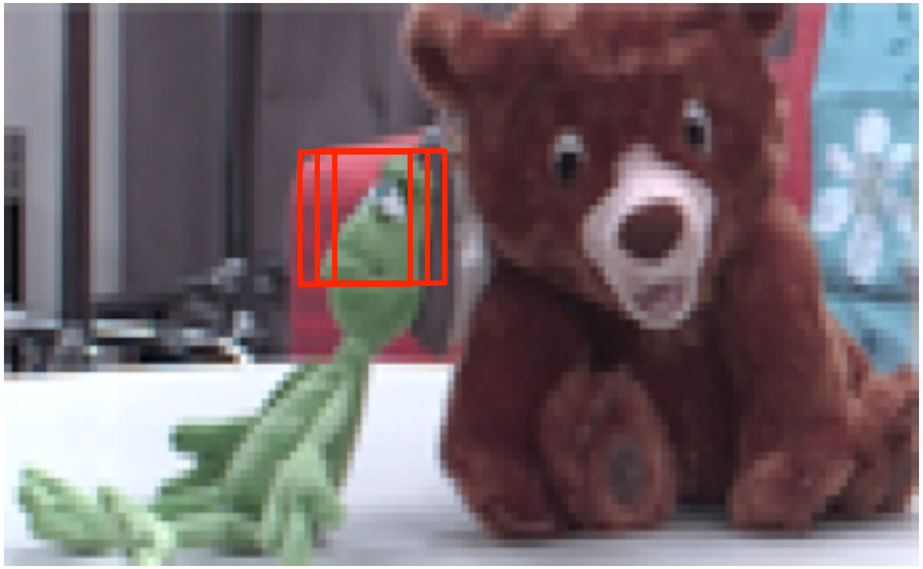
Hierarchical disparity



Left, level 1:

Recursively propagate matches

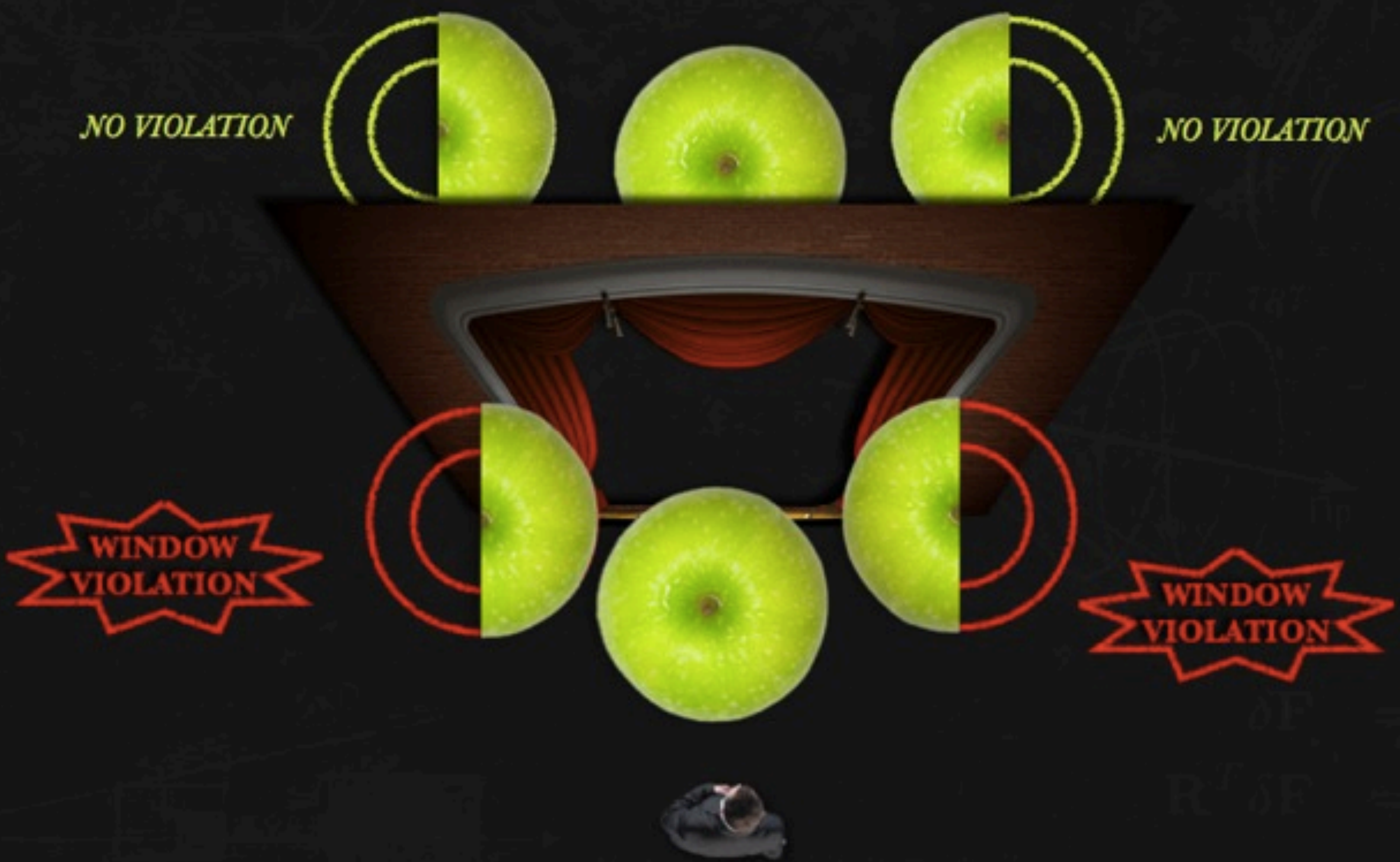
Right, level 1:



Window Violation

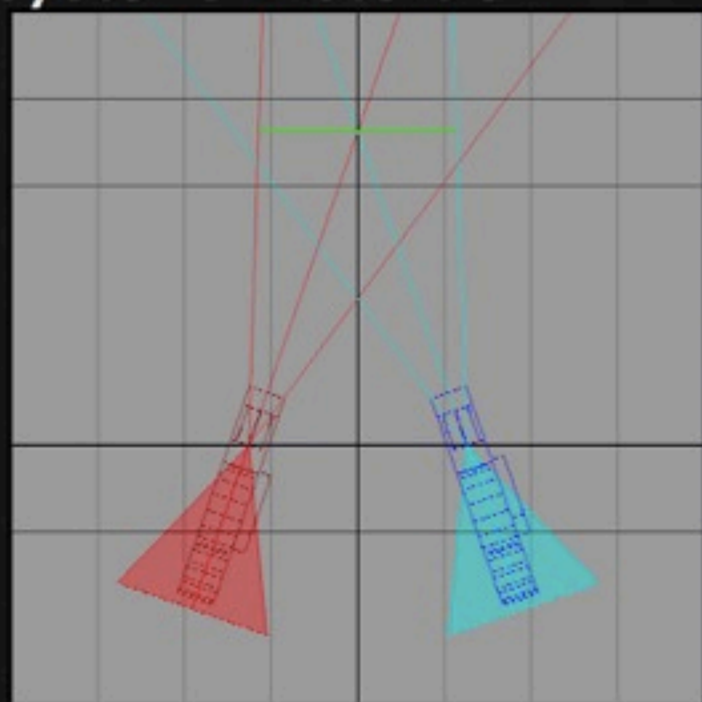
Window Violation

Window Violation



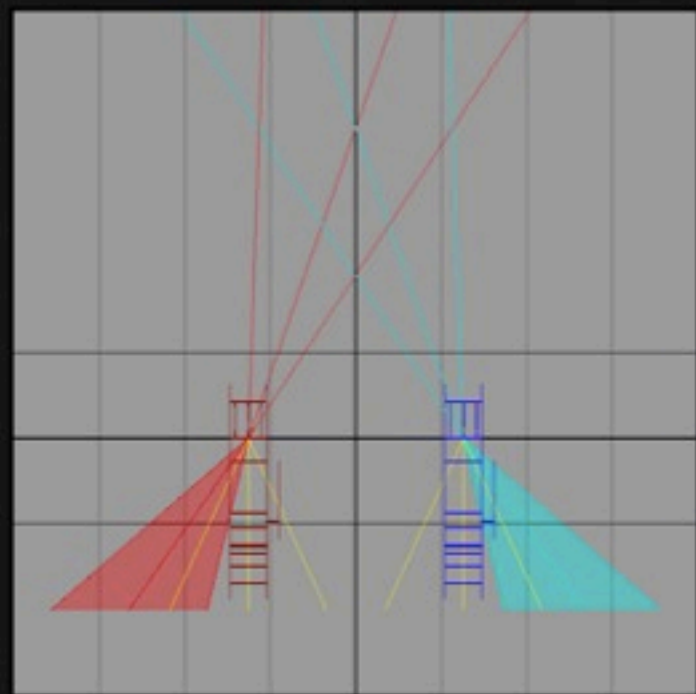
Toe-In vs. HIT Mode

Toe-in Mode: Convergence created by camera rotation (common in live action) - Causes Keystone Distortion.



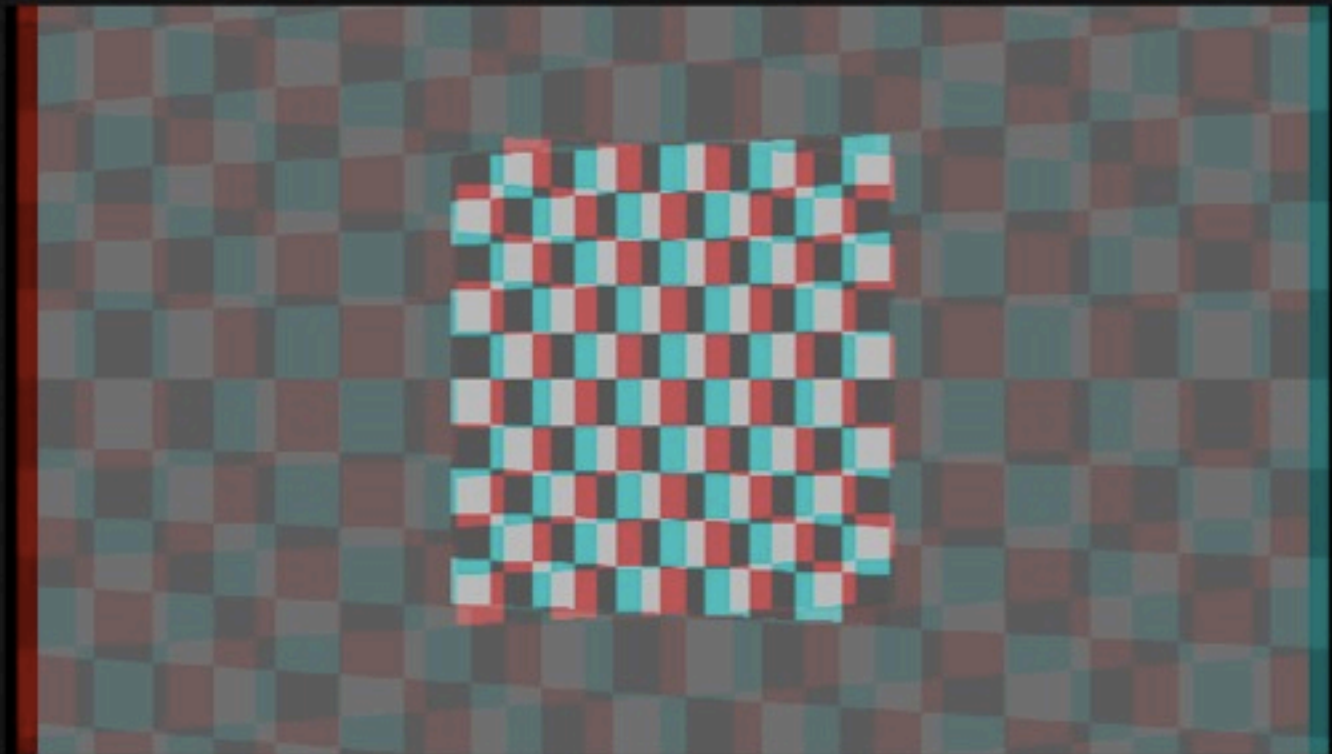
Toe-In vs. HIT Mode

- HIT (Horizontal Image Translation) Mode: Cameras remain parallel, convergence is created by offsetting the film back - No keystoning



Toe-In Mode

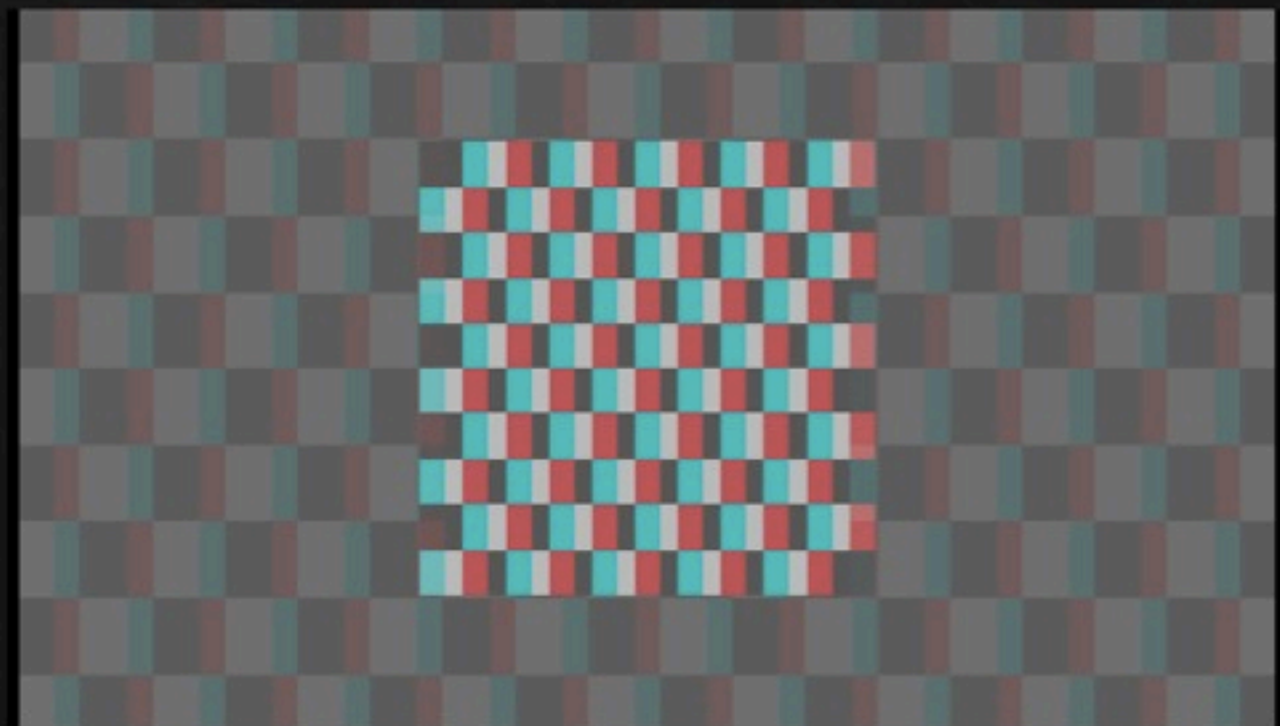
- Vertical Keystoning (ouch!)



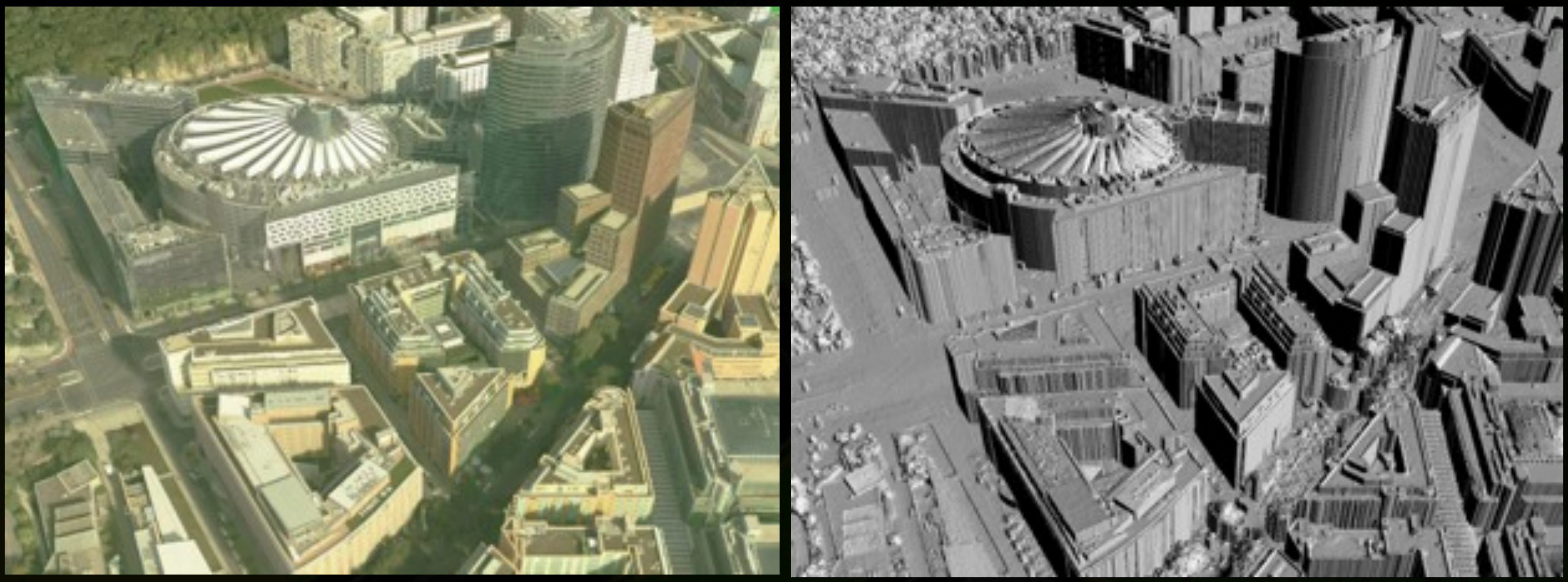
NVID

HIT Mode

- No Vertical Keystoning



3D Reconstruction from aerial images



- Stereo cameras are mounted on an airplane to obtain a terrain map
- Images taken from http://www.robotic.de/Heiko.Hirschmueller/

3D Reconstruction of Cities

- City of Dubrovnik reconstructed from images taken from Flickr in a fully automatic way
  - S. Agarwal, N. Snavely, I. Simon, S. Seitz and R. Szeliski “Building Rome in a Day”, ICCV, 2009

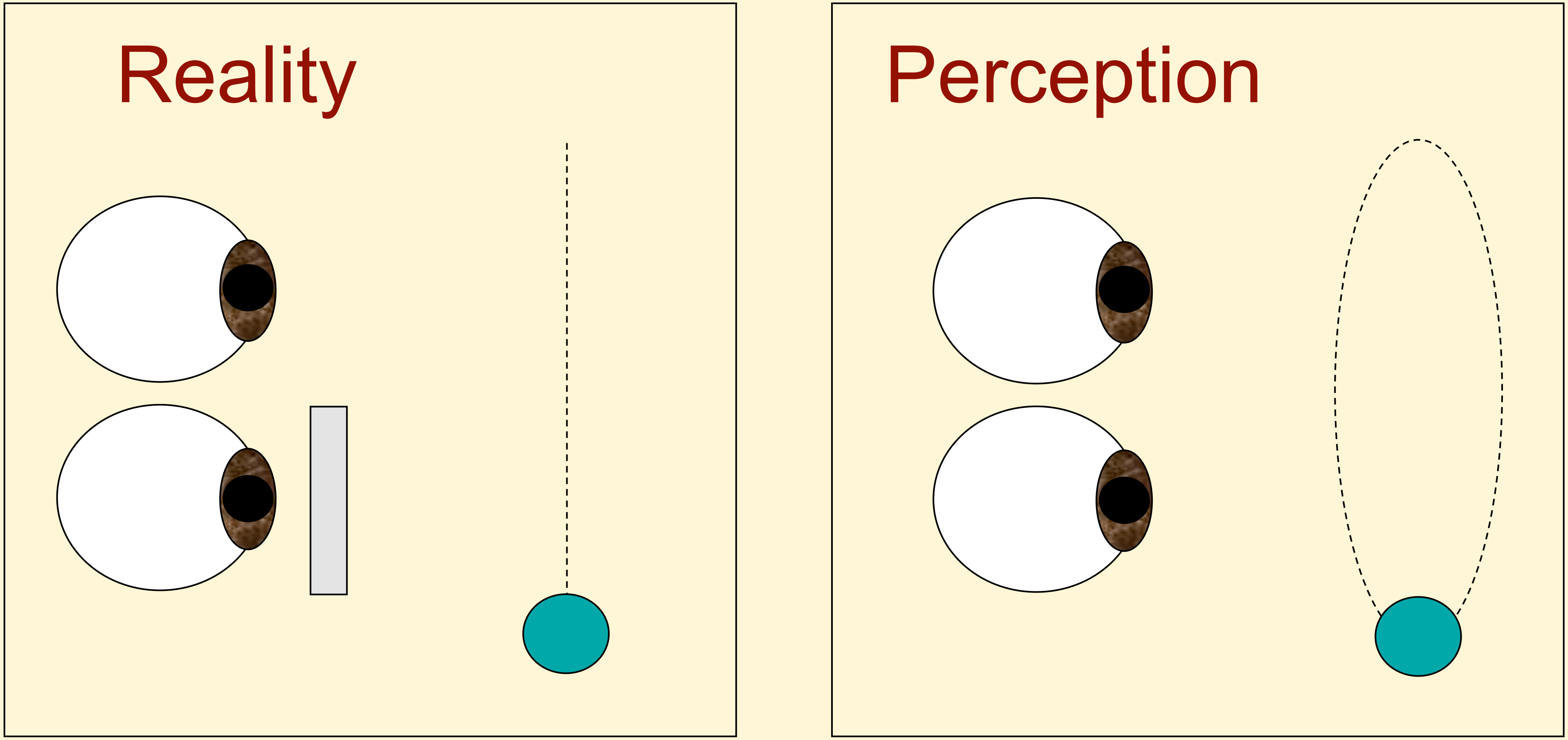
The Pulfrich effect
cover (only) the left eye with sunglasses

The neuronal basis of the Pulfrich effect in primate

Jenny Read
Bruce Cumming

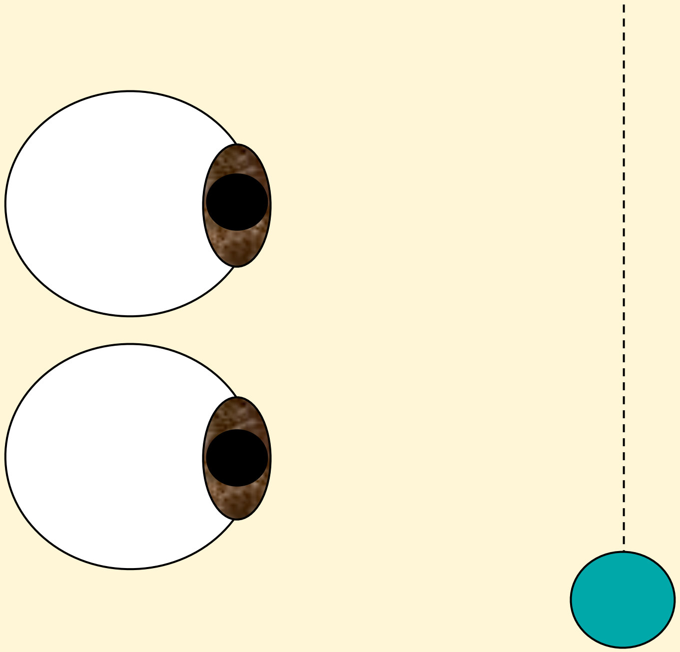
Laboratory of Sensorimotor Research
National Eye Institute
National Institutes of Health
Bethesda, Maryland

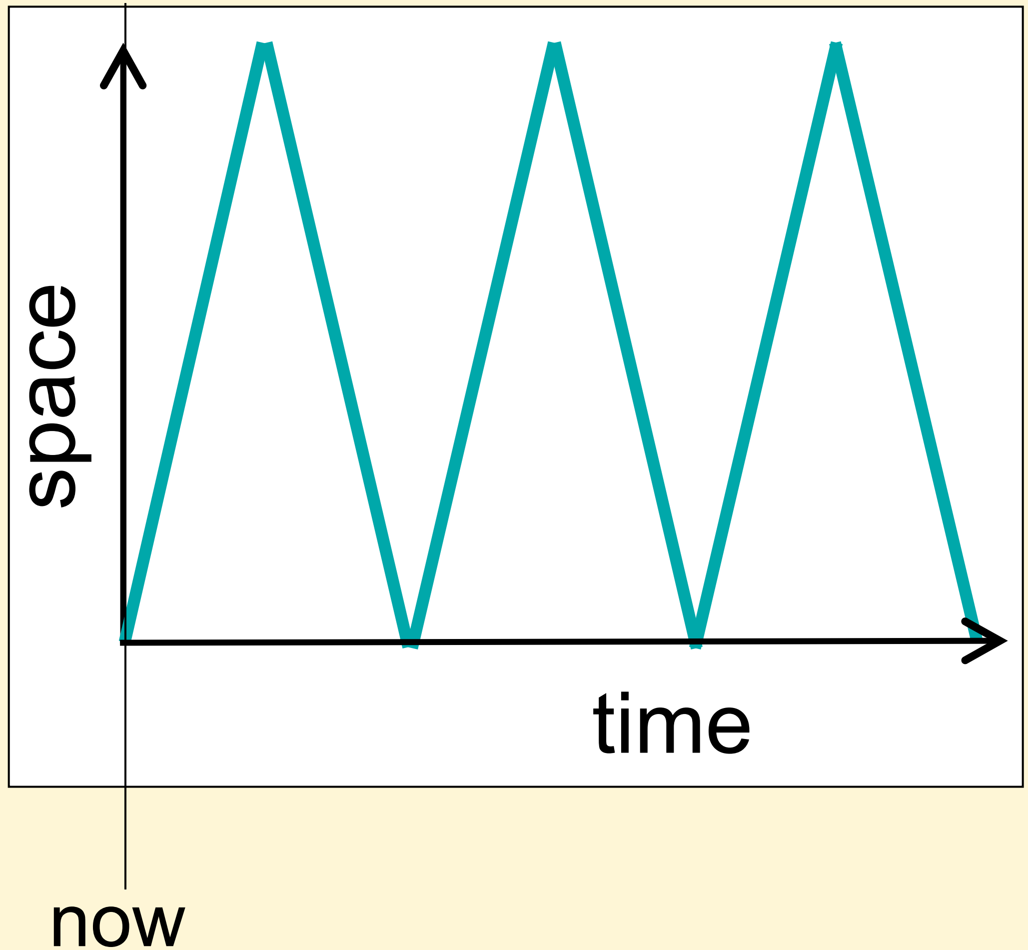
The Pulfrich effect



- Illusory perception of a moving object when one eye's image is delayed

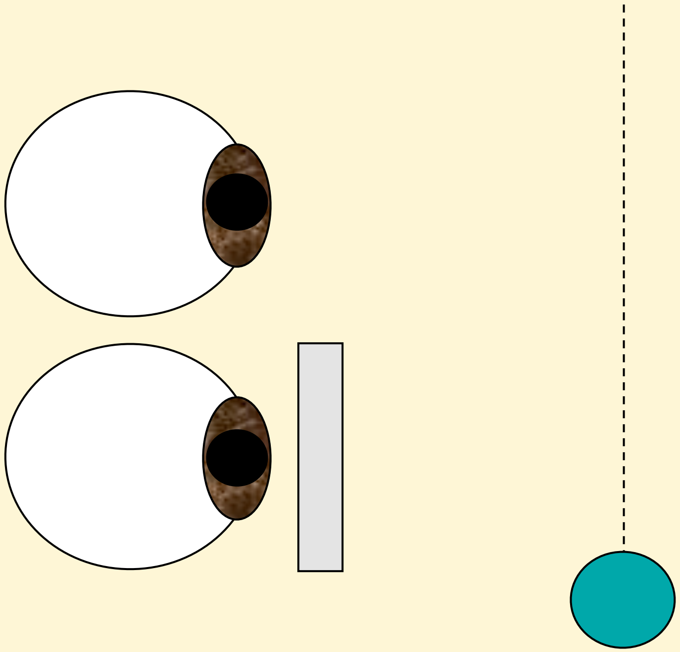
Space-time diagram

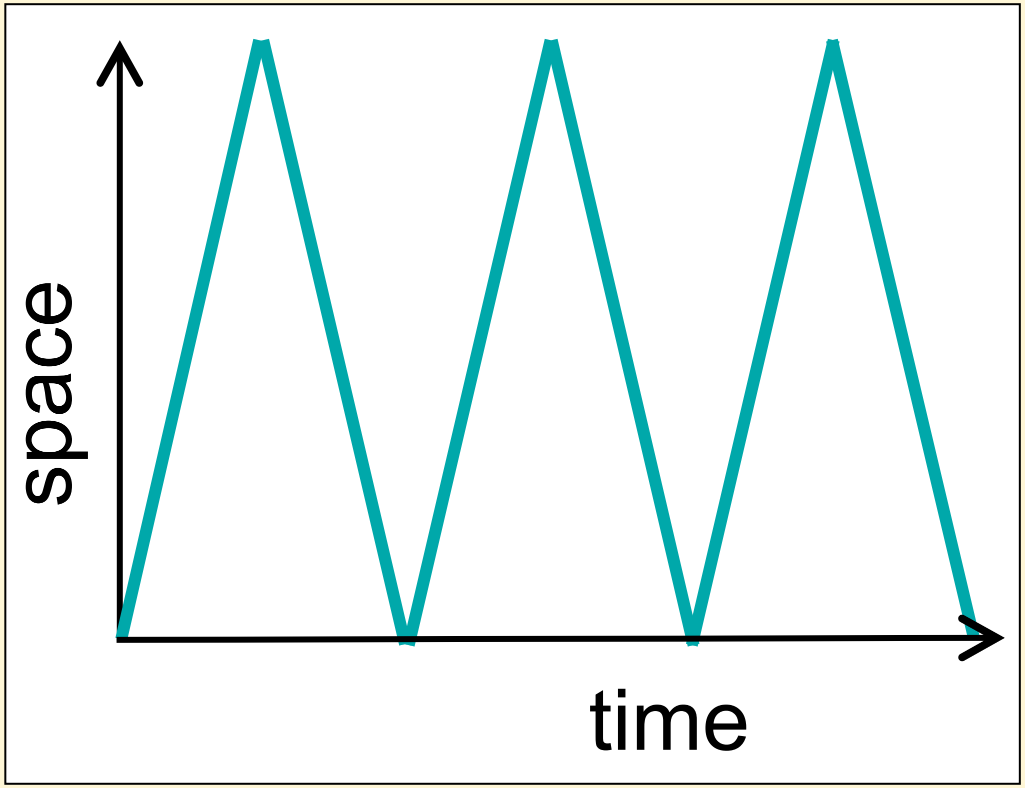




- Moving object

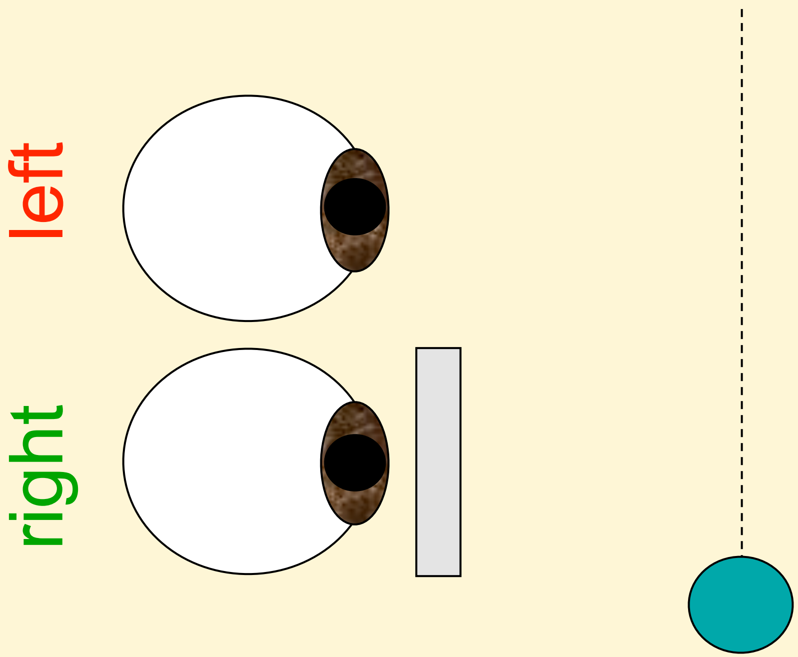
The Pulfrich effect

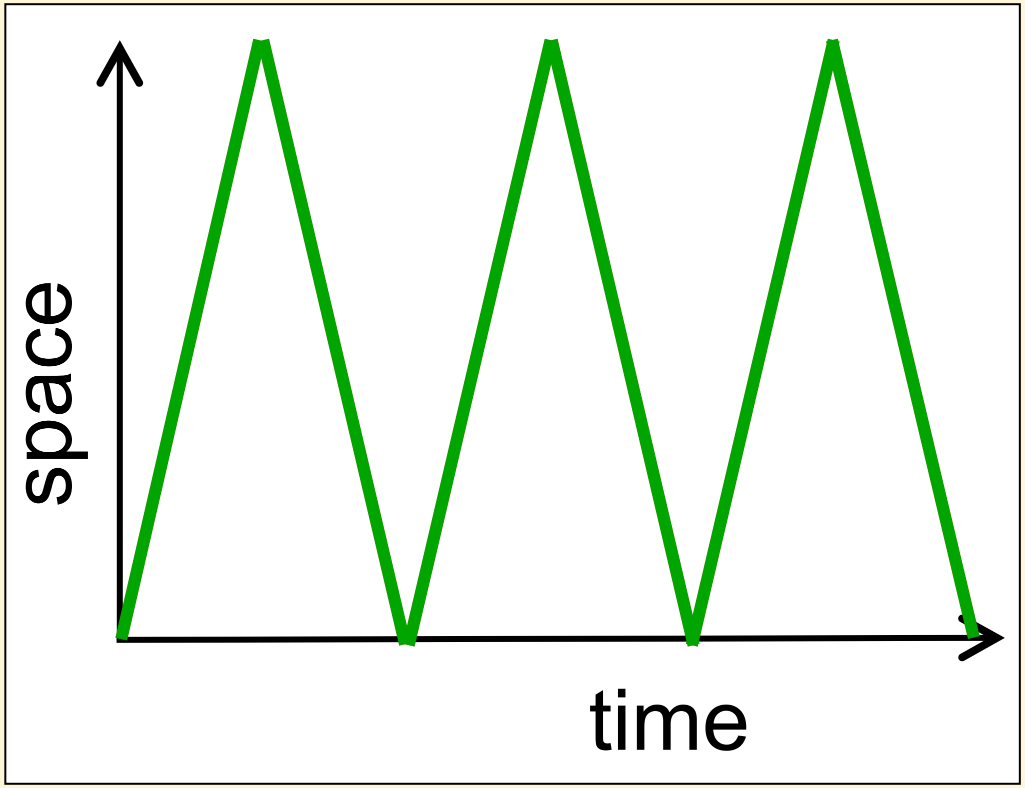




- Moving object
- A delay is introduced in one eye’s image

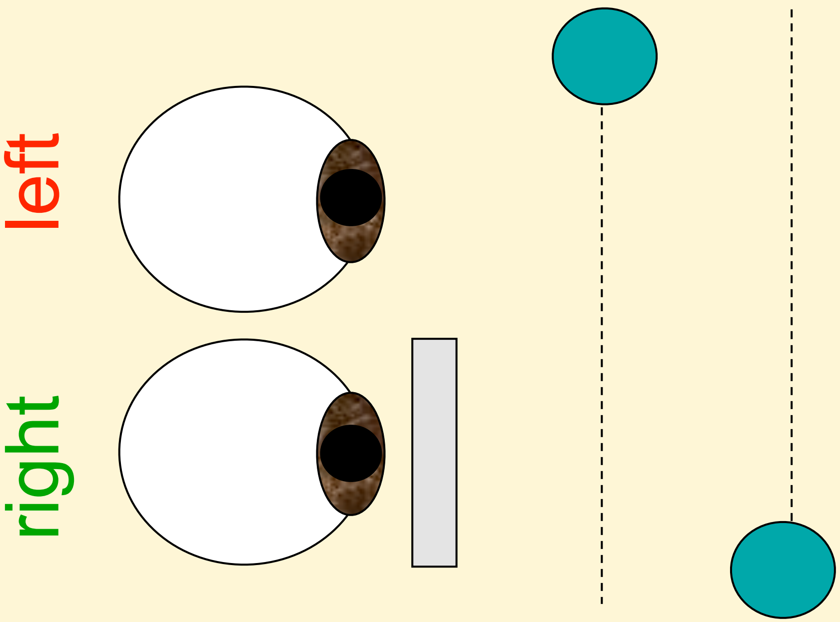
The Pulfrich effect

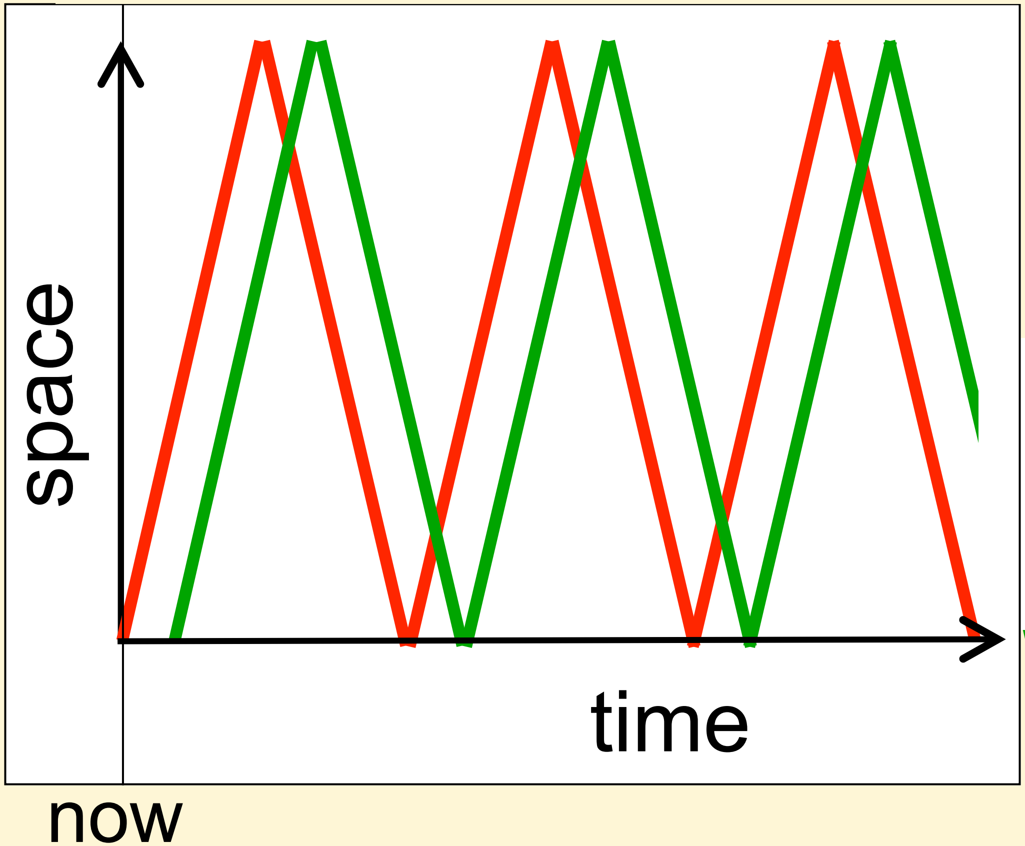




- Moving object
- A delay is introduced in one eye’s image

The Pulfrich effect





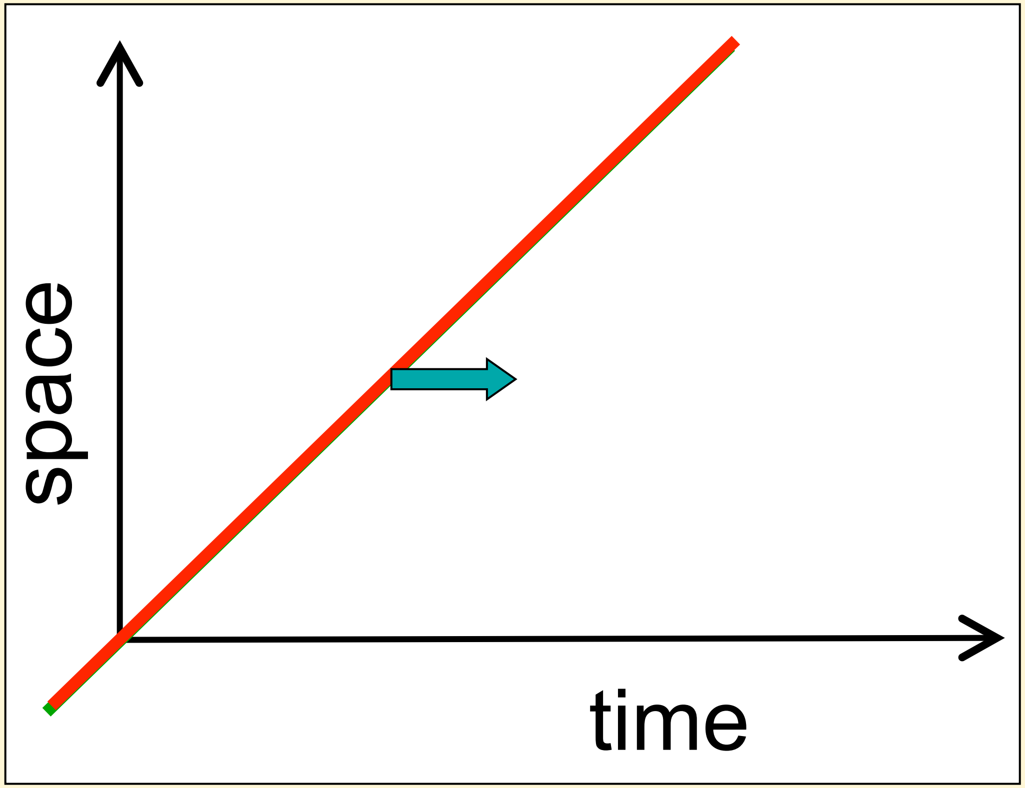
- Moving object
- A delay is introduced in one eye's image
- The object is perceived as moving in depth

Classical explanation

- Temporal delay is geometrically equivalent to spatial disparity

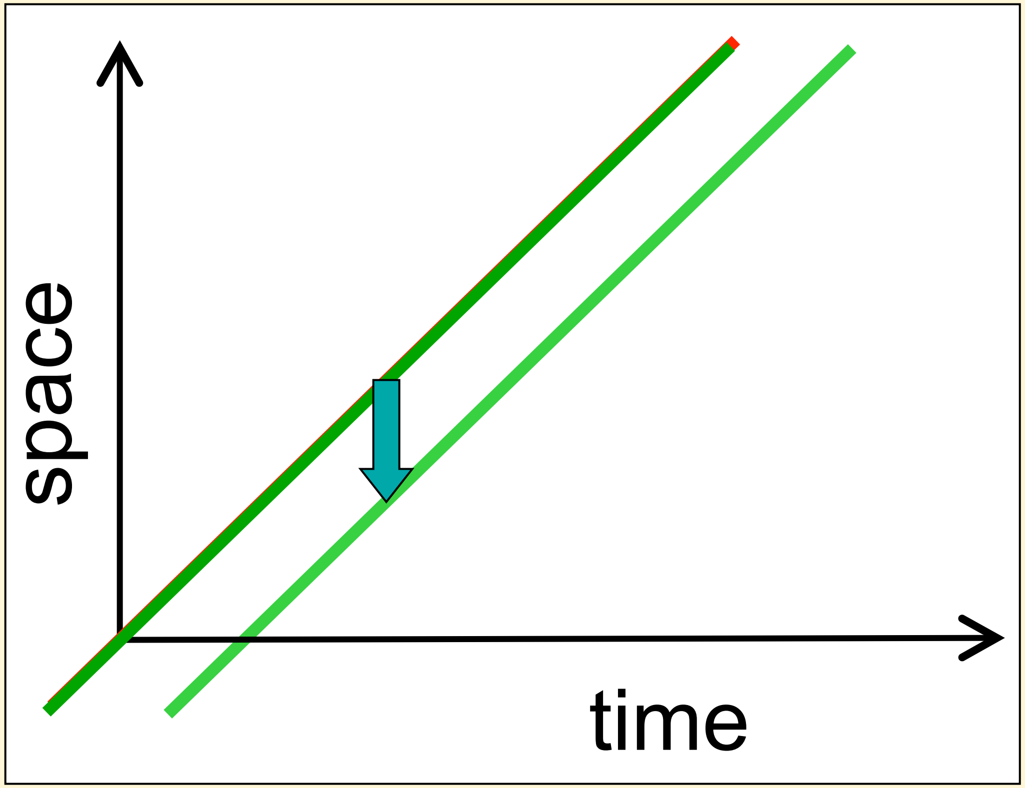
Temporal delay is geometrically equivalent to spatial disparity

Temporal delay



Temporal delay is geometrically equivalent to spatial disparity

Spatial disparity



Classical explanation

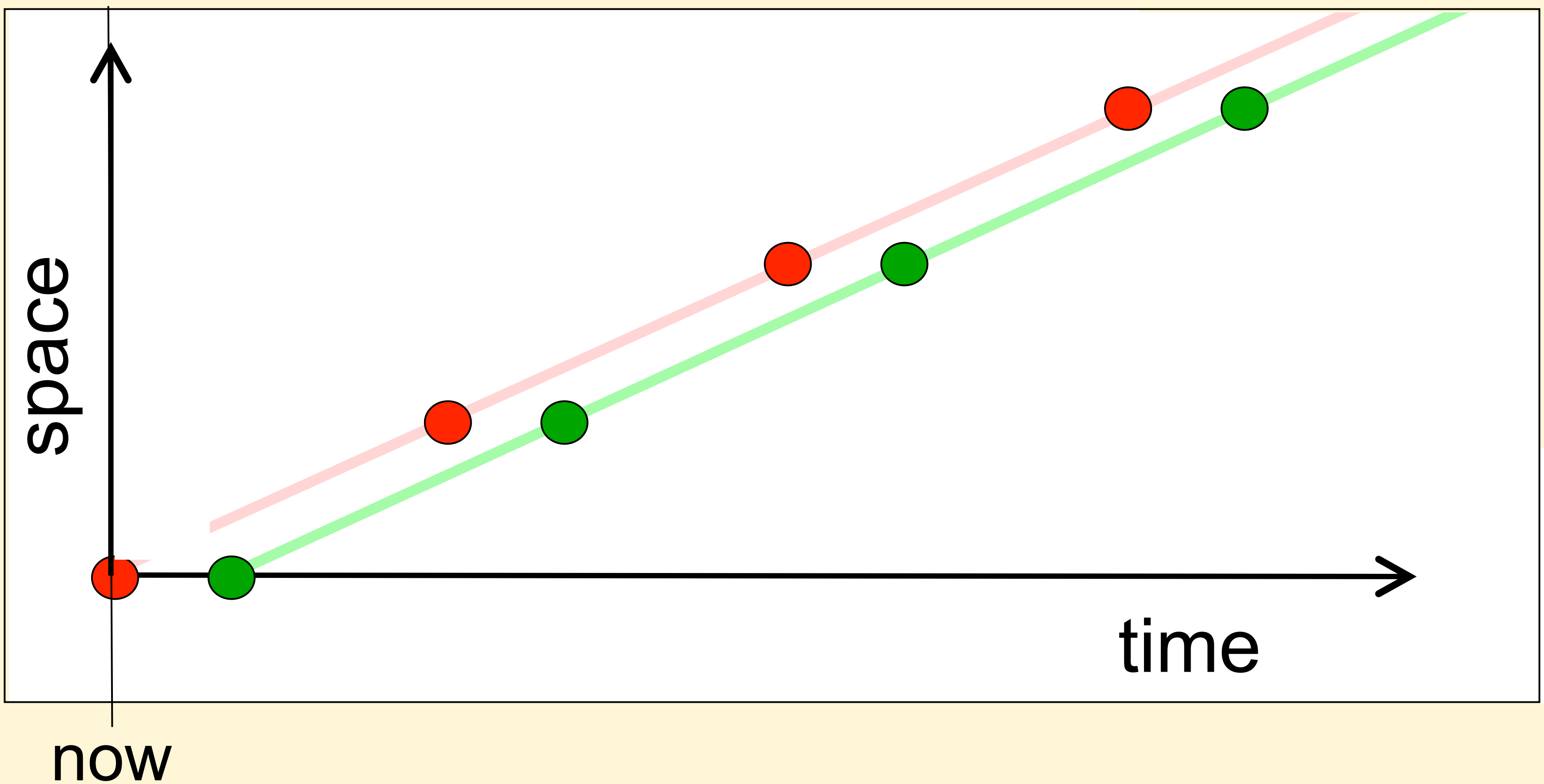
- Temporal delay is geometrically equivalent to spatial disparity
  - so Pulfrich stimulus activates mechanisms which usually process spatial disparity

But: the classic explanation doesn't seem to work for the stroboscopic Pulfrich effect

(Lee, 1970)

Stroboscopic Pulfrich effect

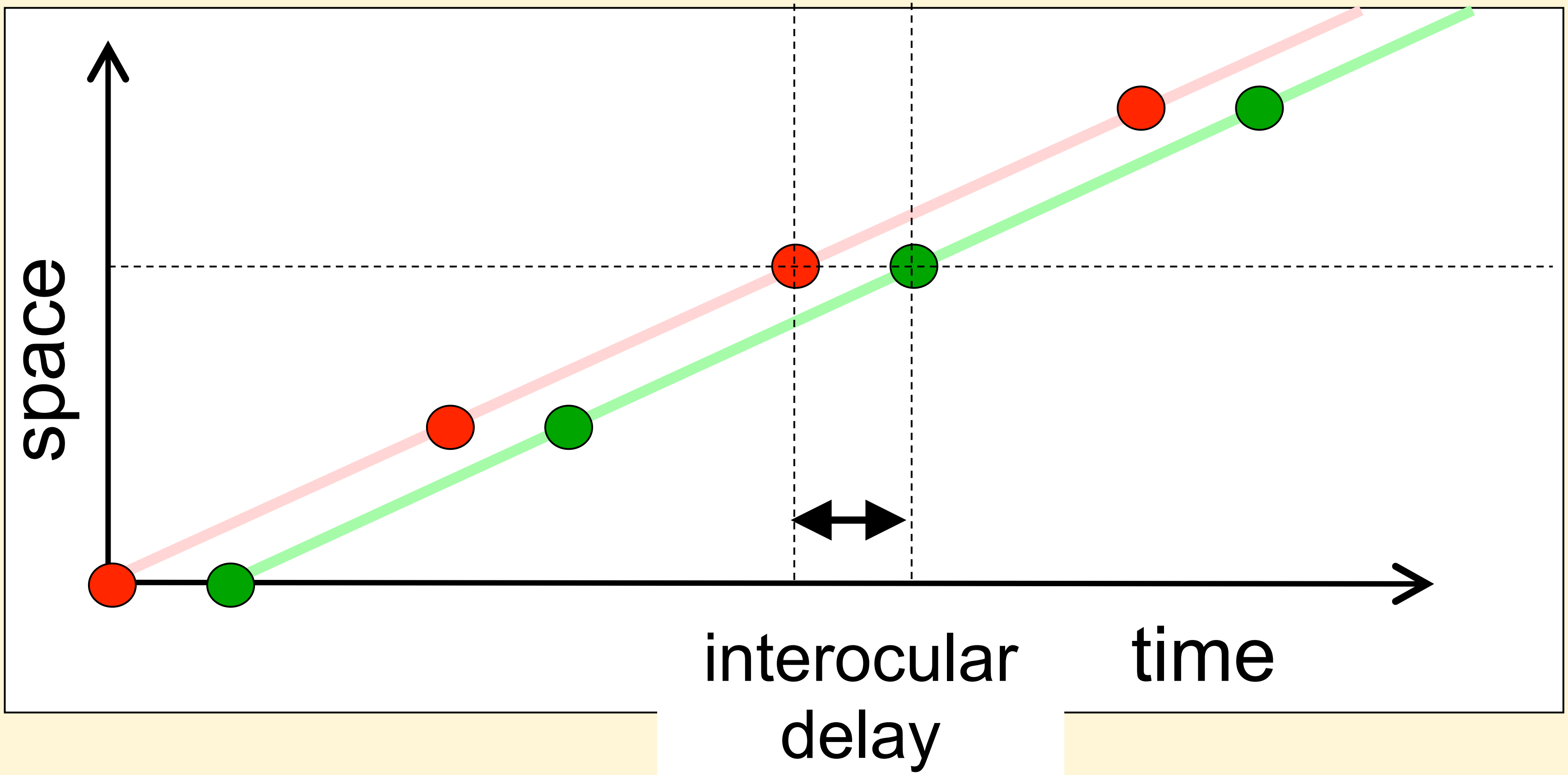
Flashing stimulus, one eye lagging the other



Stroboscopic Pulfrich effect

Flashing stimulus, one eye lagging the other

No spatial disparity, purely temporal delay



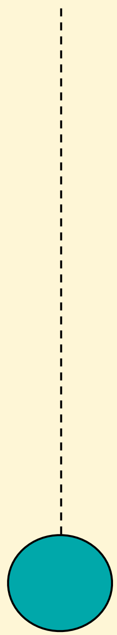
Neuronal basis of the Pulfrich effect

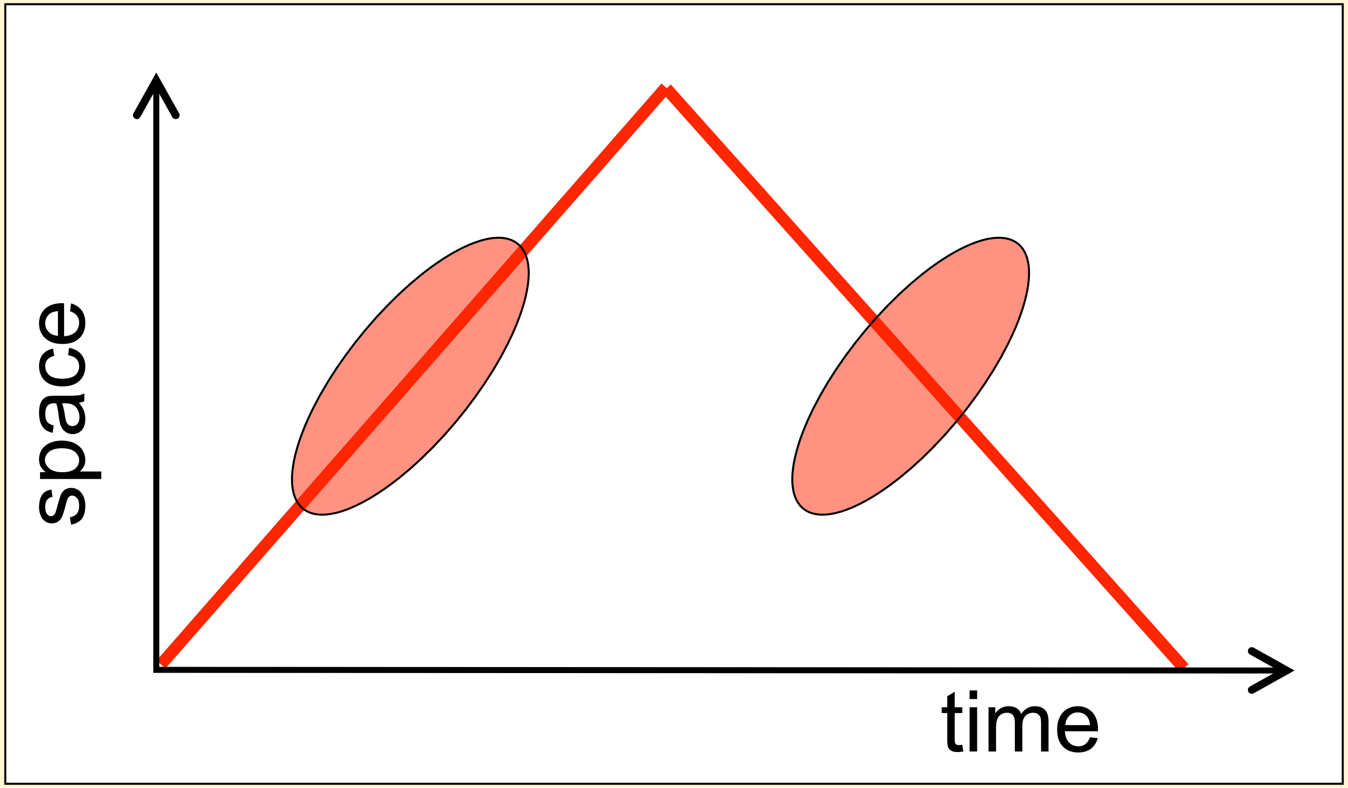
- Suggestion (Qian & Anderson, 1997):
  - the stimulus activates cells sensitive both to
    - direction of motion and
    - interocular disparity

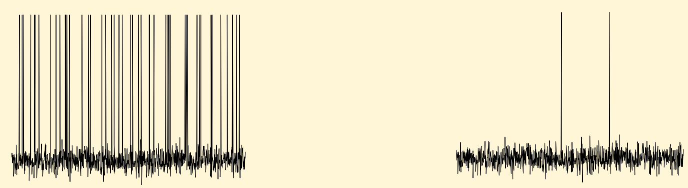
⇒joint encoding of motion and disparity

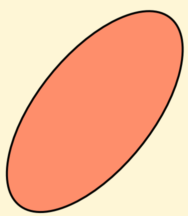
- implies receptive fields which are inseparable (tilted) in space and time

Direction-selective cells have receptive fields which are tilted in space and time

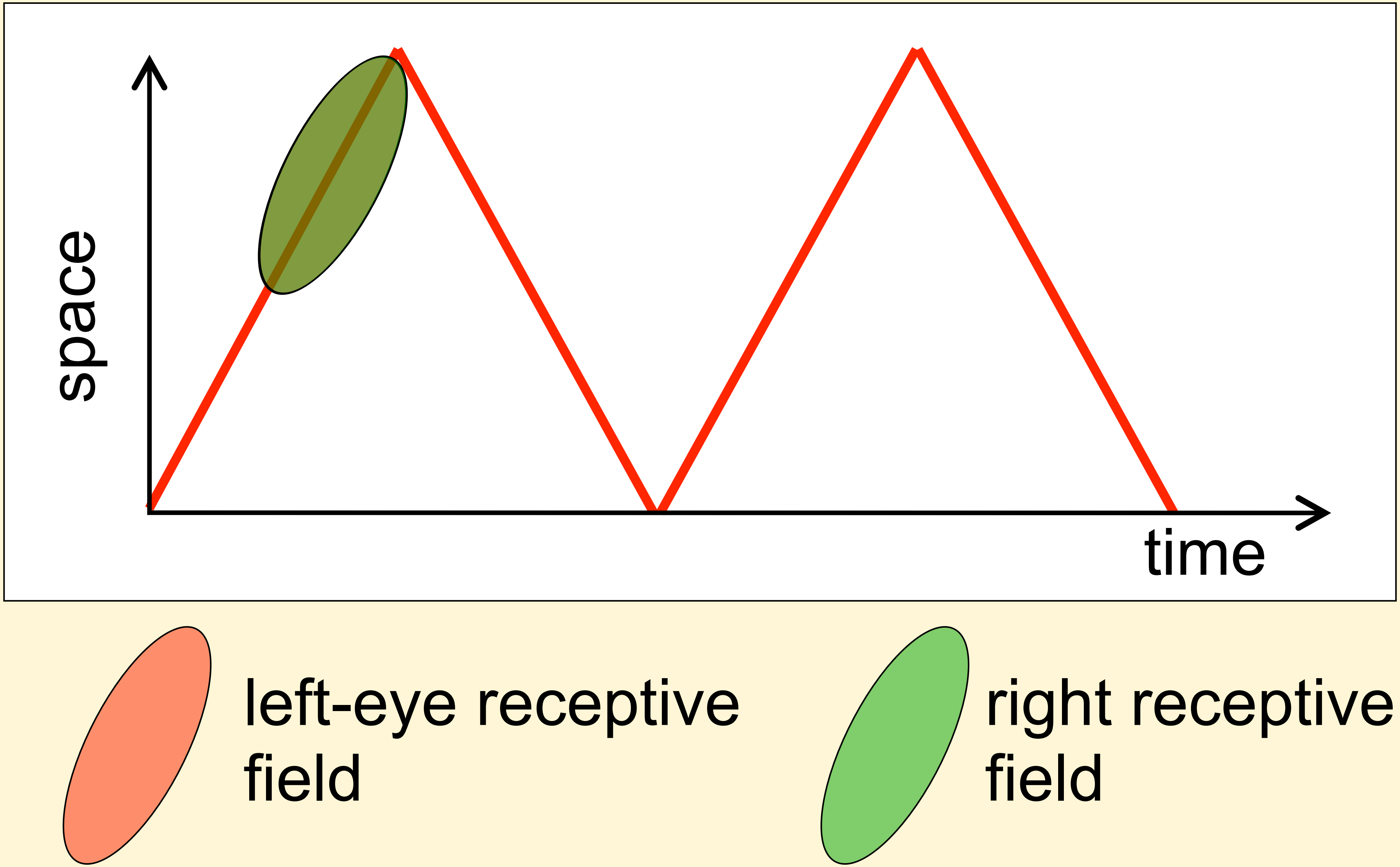




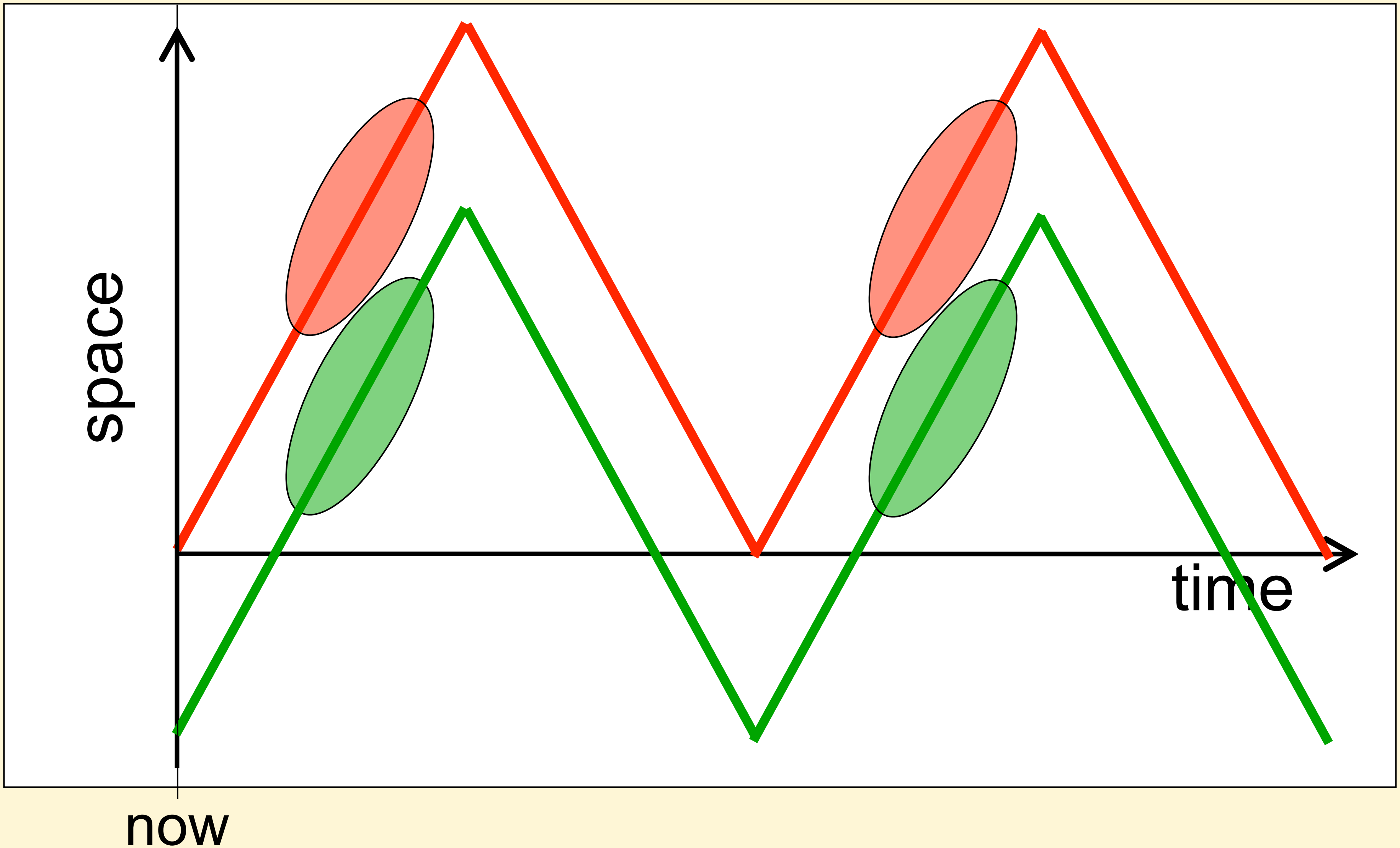




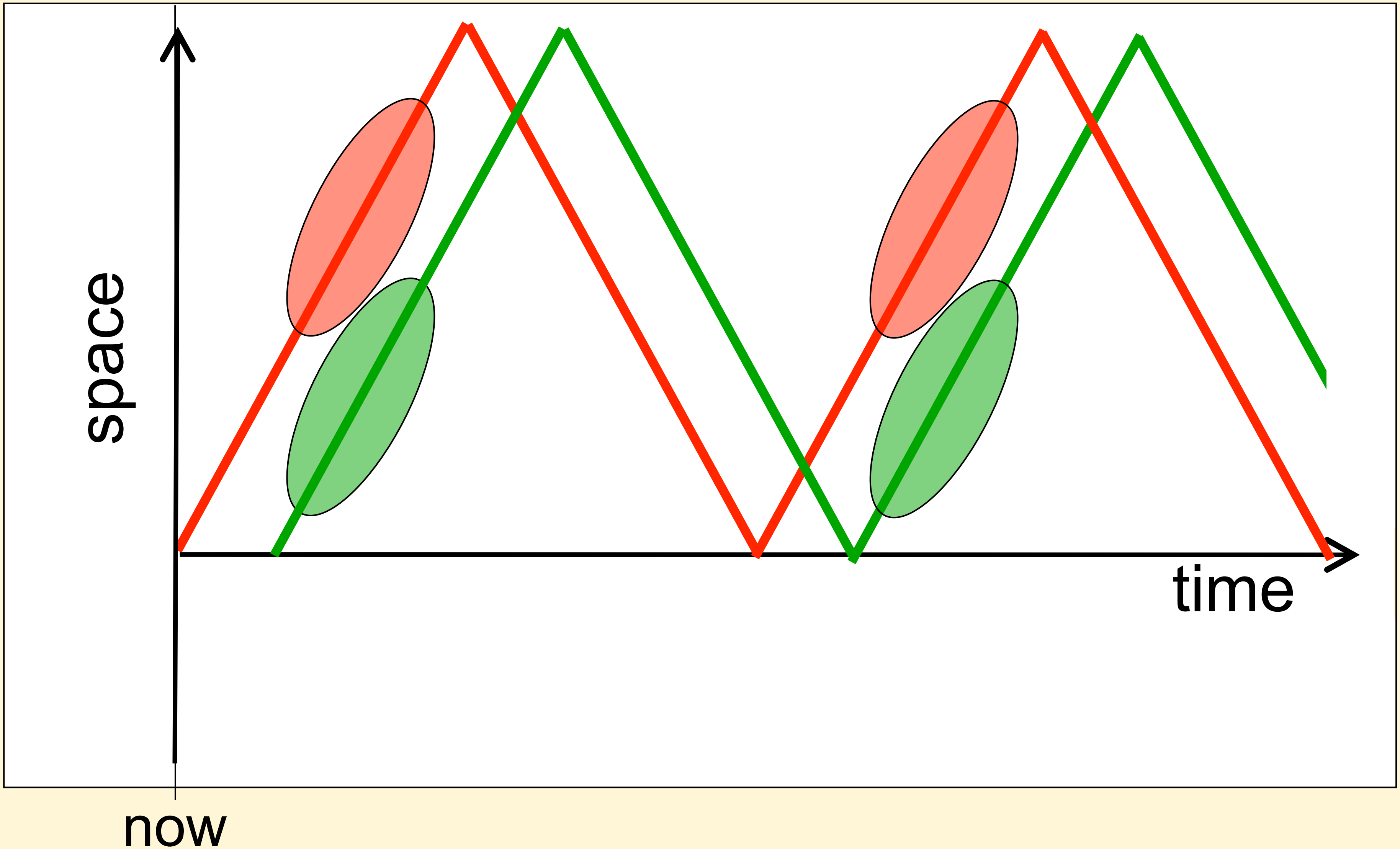
Receptive fields tuned to disparity *and* direction of motion



Designed to respond to a moving object with disparity…



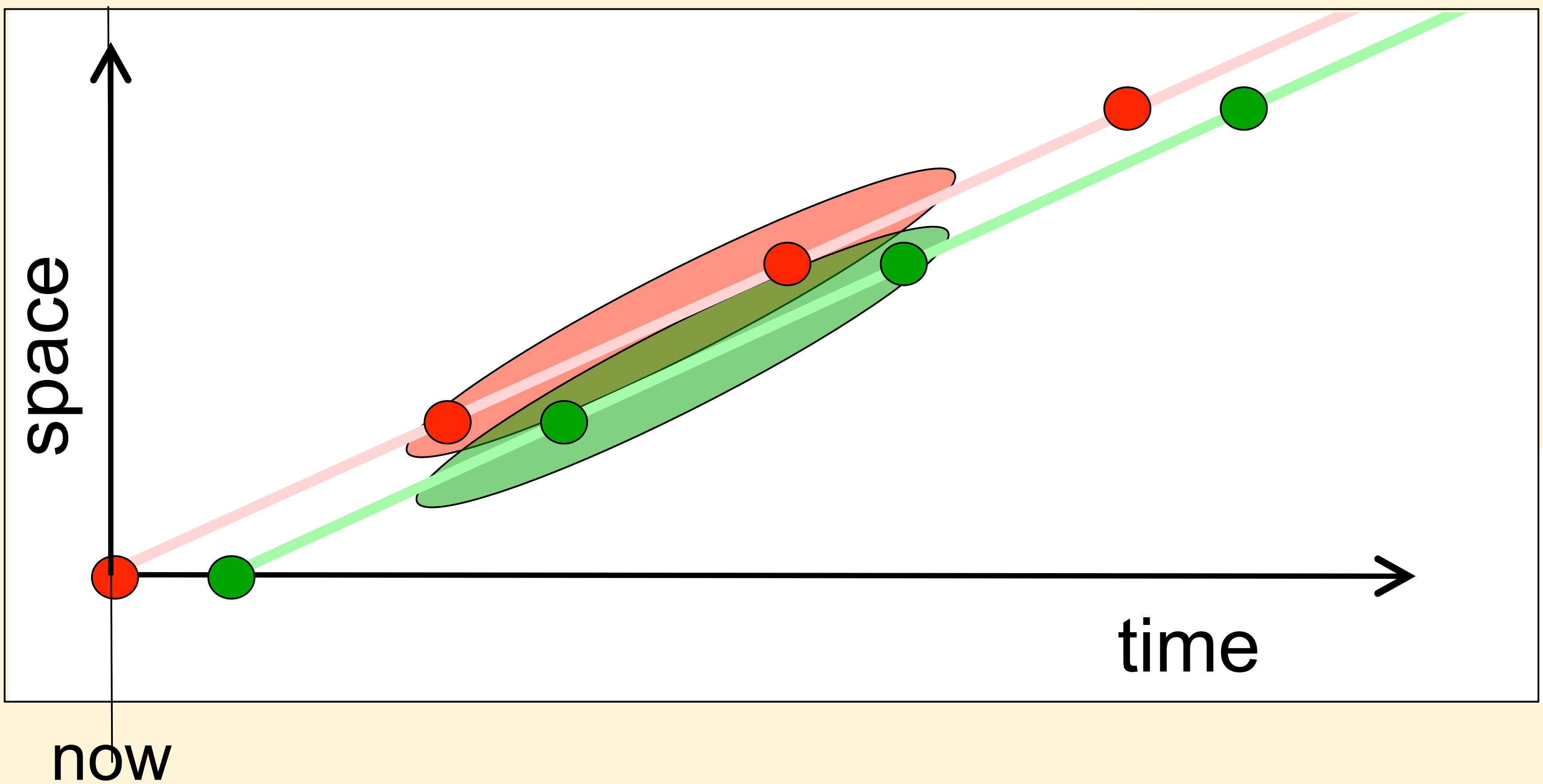
…will also respond to a moving object with zero disparity but a temporal delay



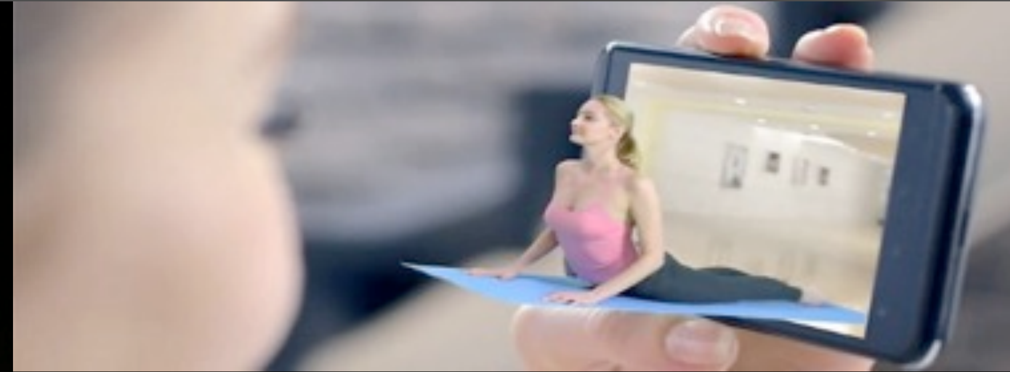
Stroboscopic Pulfrich effect

No spatial disparity, purely temporal delay

Stroboscopic stimulus activates tilted RFs

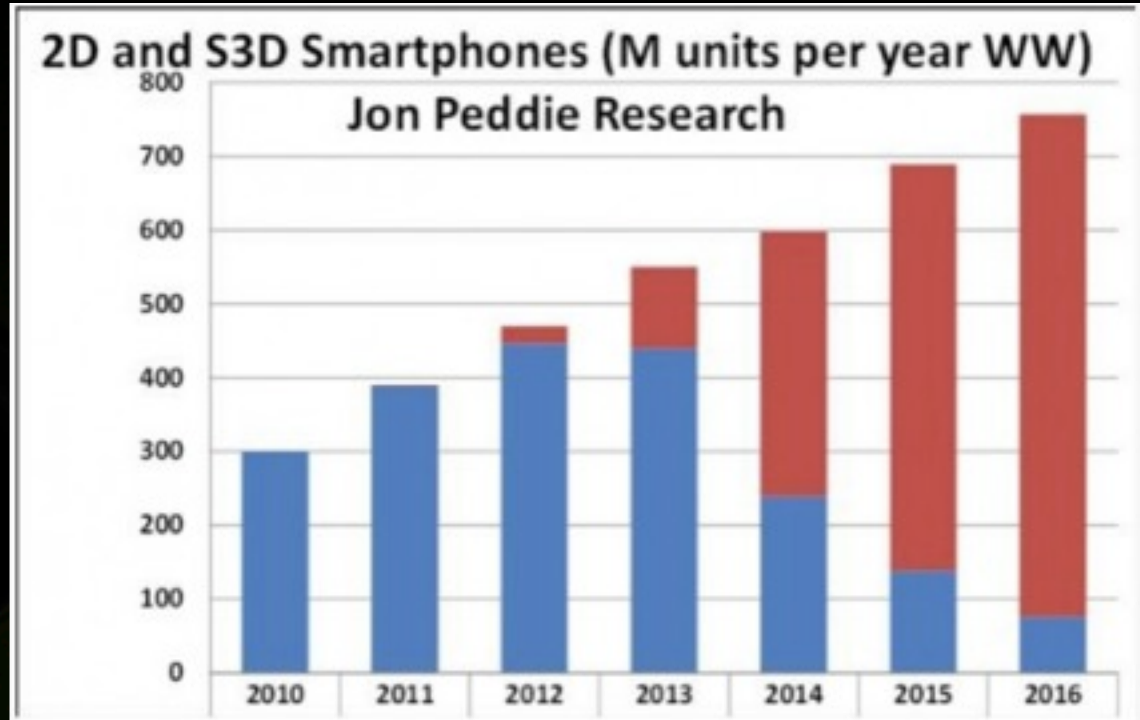


Stereo camera phones



At least Jon Peddie believes in 3D

- It's not just about games; the addition of two cameras will enable gesture-based controls, advanced AR applications, visualization with depth, and real-world data capture



80% of smartphones will have S3D capture and display capability by 2015 (Source: Jon Peddie Research)

Online comment by "ralphg":
3D tvs failed. 3D cinema is limping. 3D digital cameras have not taken off. Popup books are not more popular than flat paper ones.
...
As for smartphones, in the real world, people use them simply as portable computers: check email, send texts, make phone calls, and a few other tasks. No 3D in that.