Head Mounted Display Optics I

Gordon Wetzstein
Stanford University

EE 267 Virtual Reality

Lecture 7

stanford.edu/class/ee267/

Logistics

- HW3 is probably the longest homework, so get started asap if you have not done so already
- hardware kits will be handed out in TA office hours this week

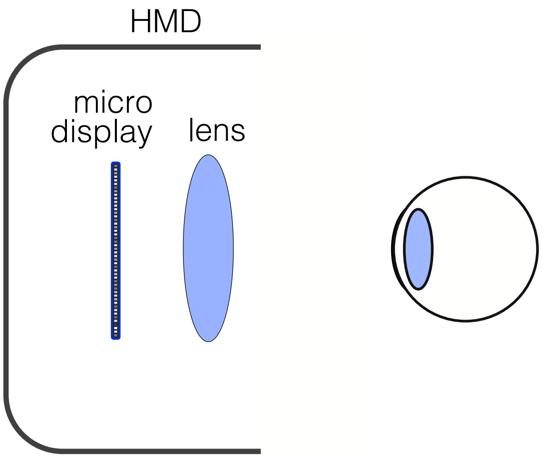
Lecture Overview

1. stereo rendering for HMDs
2. field of view and visual field
3. lens distortion correction using GLSL
4. overview of microdisplay technology

Stereo Rendering for HMDs

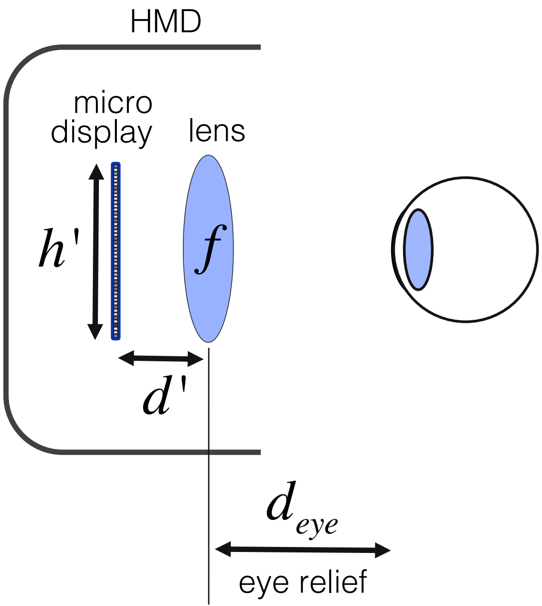
All Current-generation VR HMDs are "Simple Magnifiers"

Image Formation



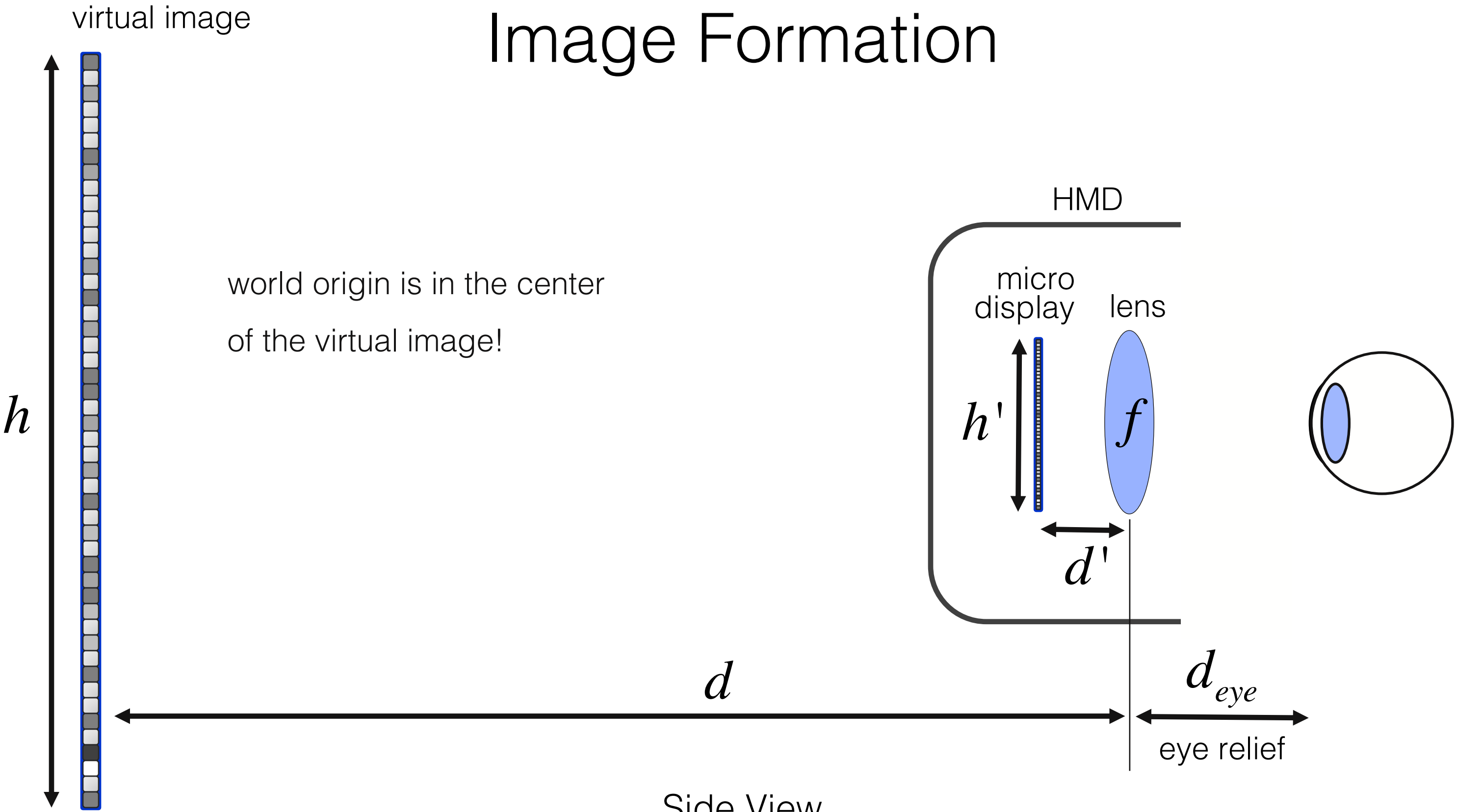
Side View

Image Formation



Side View

Image Formation

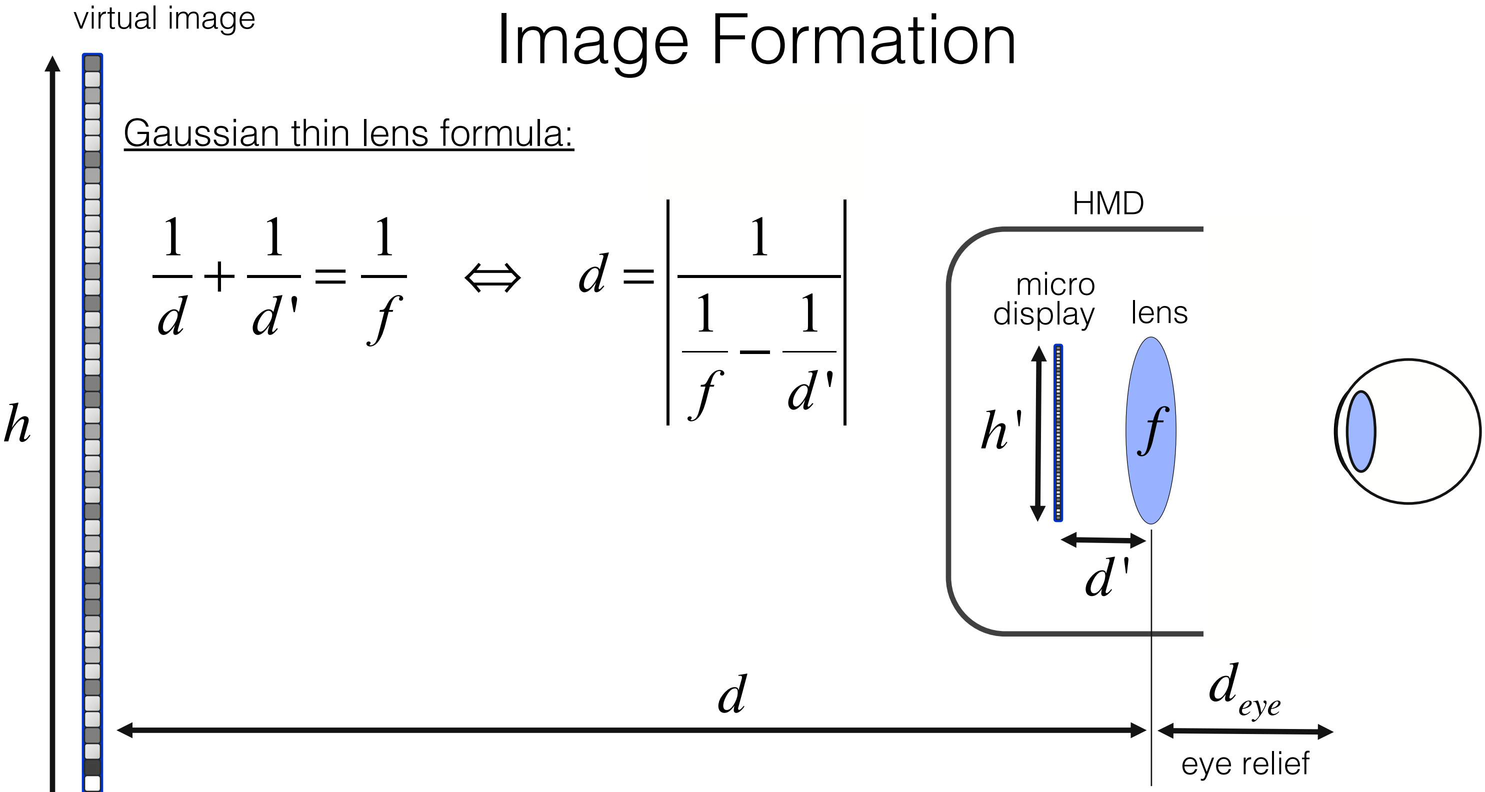


Side View

Image Formation

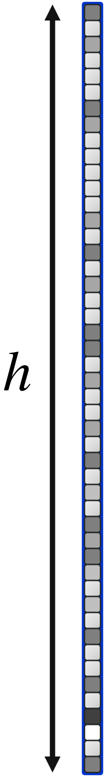
Gaussian thin lens formula:

$$\frac{1}{d} + \frac{1}{d'} = \frac{1}{f} \quad \Leftrightarrow \quad d = \left| \frac{1}{\frac{1}{f} - \frac{1}{d'}} \right|$$



Side View

Image Formation

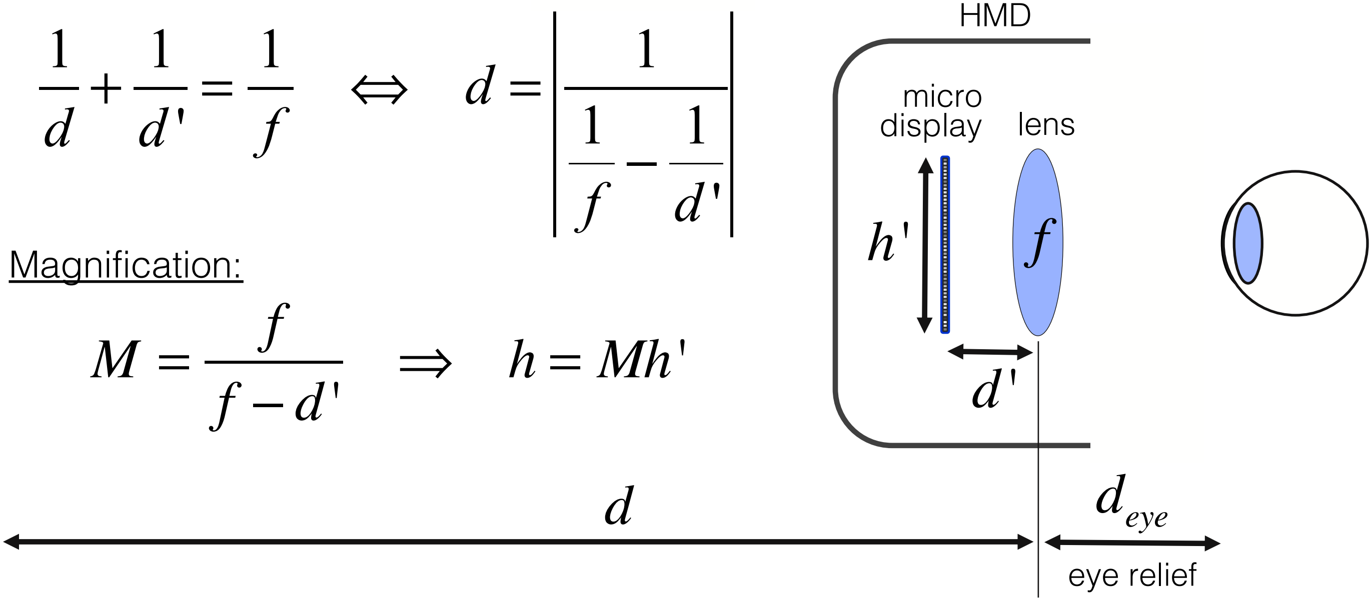


Gaussian thin lens formula:

$$\frac{1}{d} + \frac{1}{d'} = \frac{1}{f} \quad \Leftrightarrow \quad d = \left| \frac{1}{\frac{1}{f} - \frac{1}{d'}} \right|$$

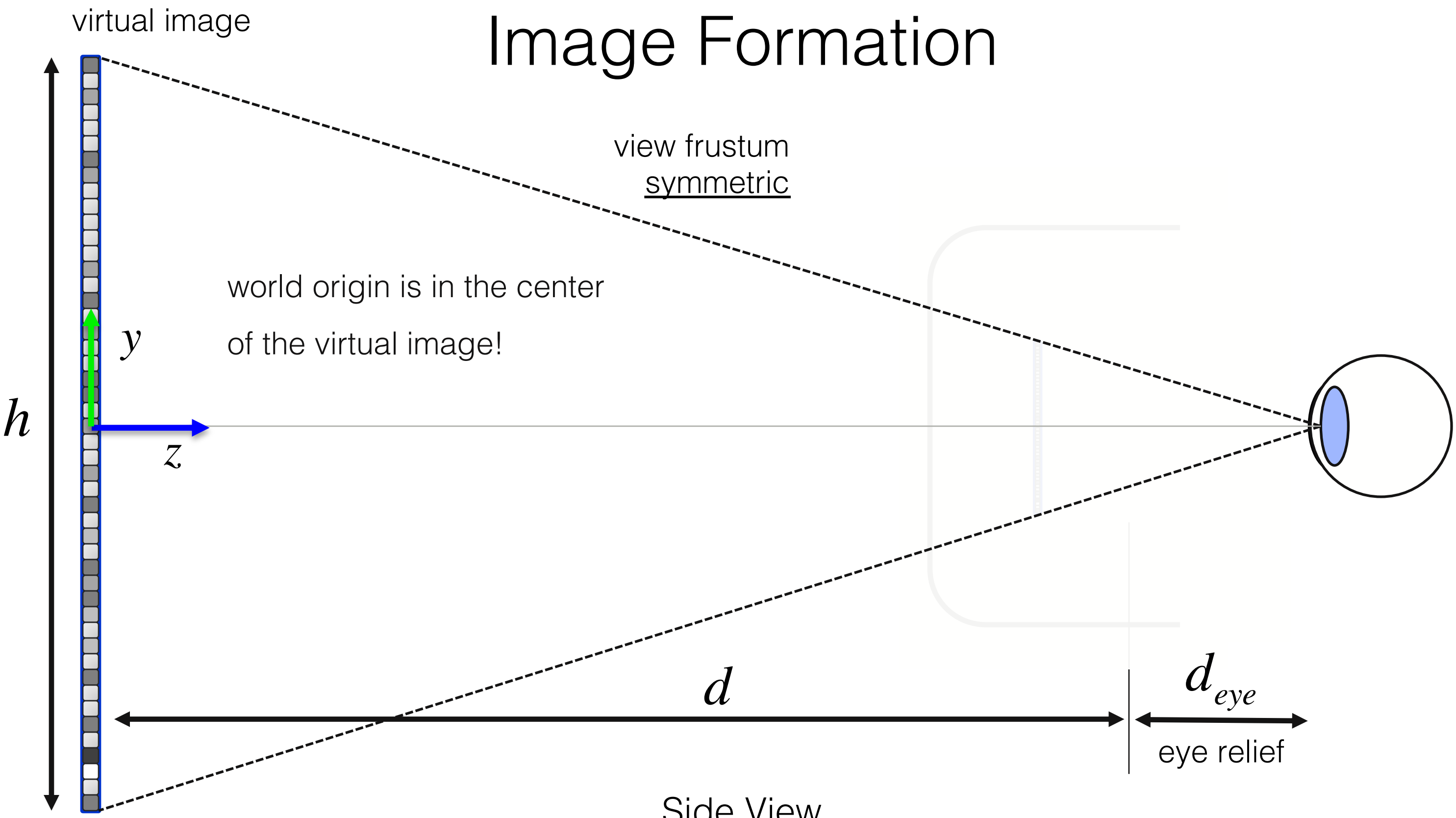
Magnification:

$$M = \frac{f}{f - d'} \quad \Rightarrow \quad h = Mh'$$



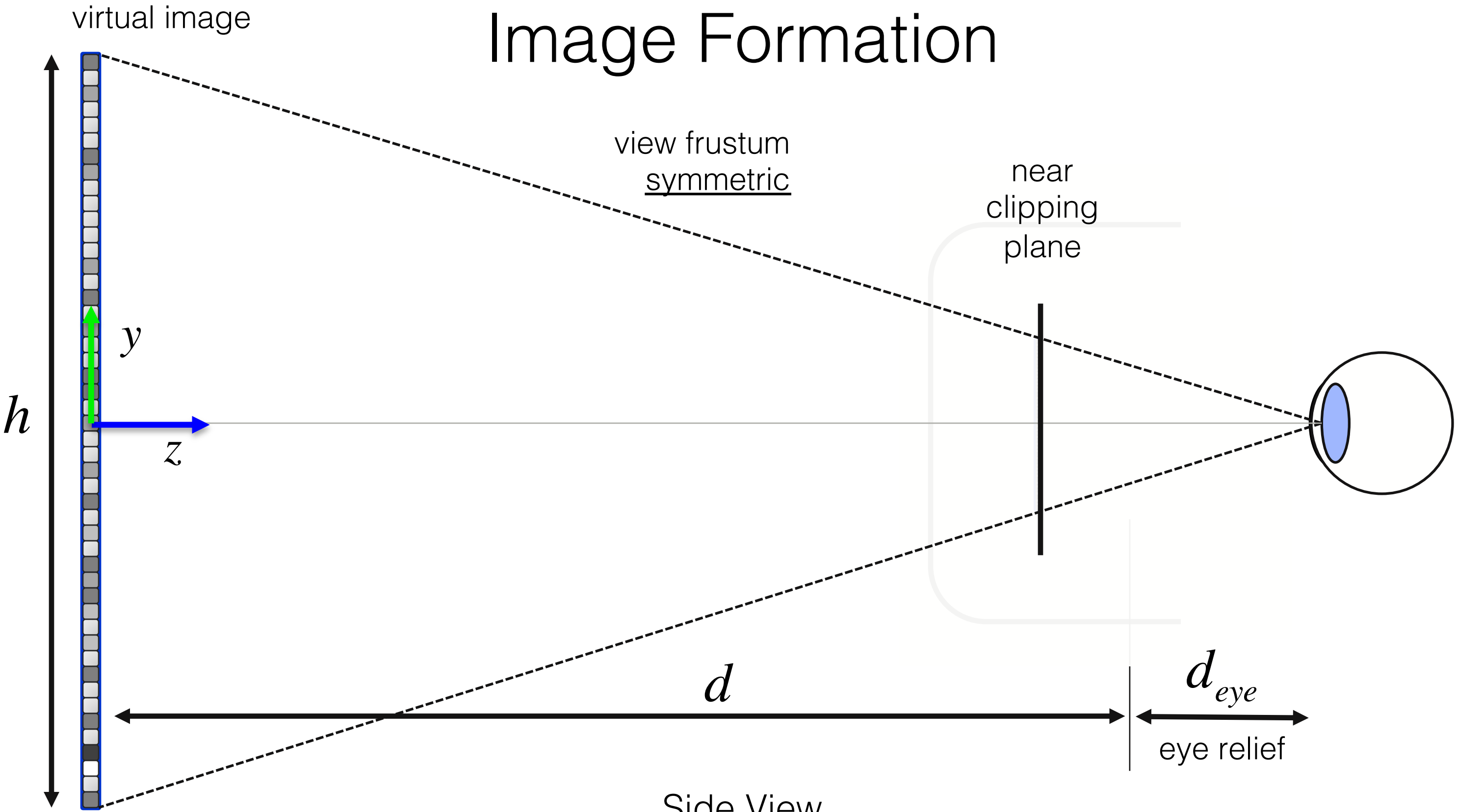
Side View

Image Formation



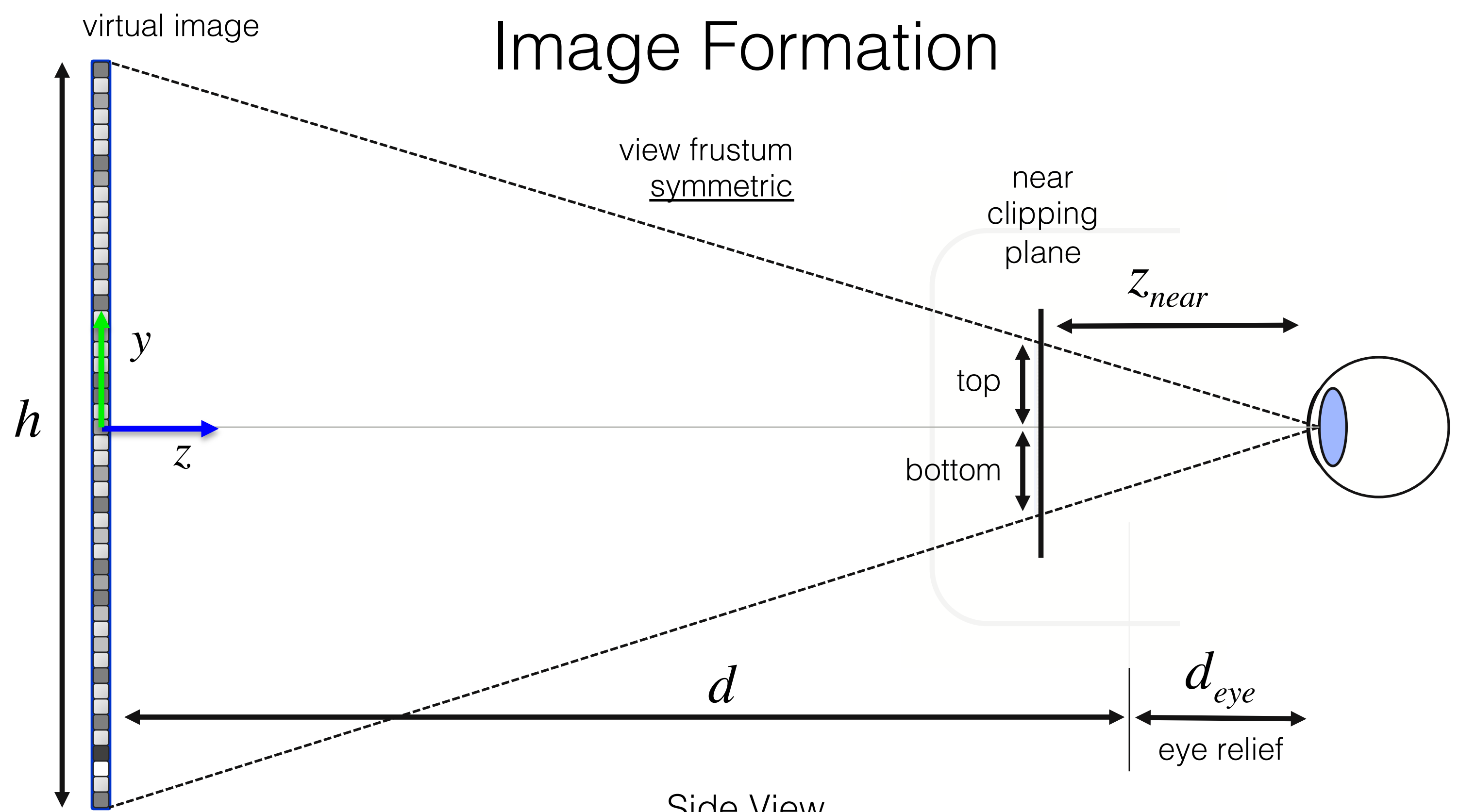
Side View

Image Formation



Side View

Image Formation



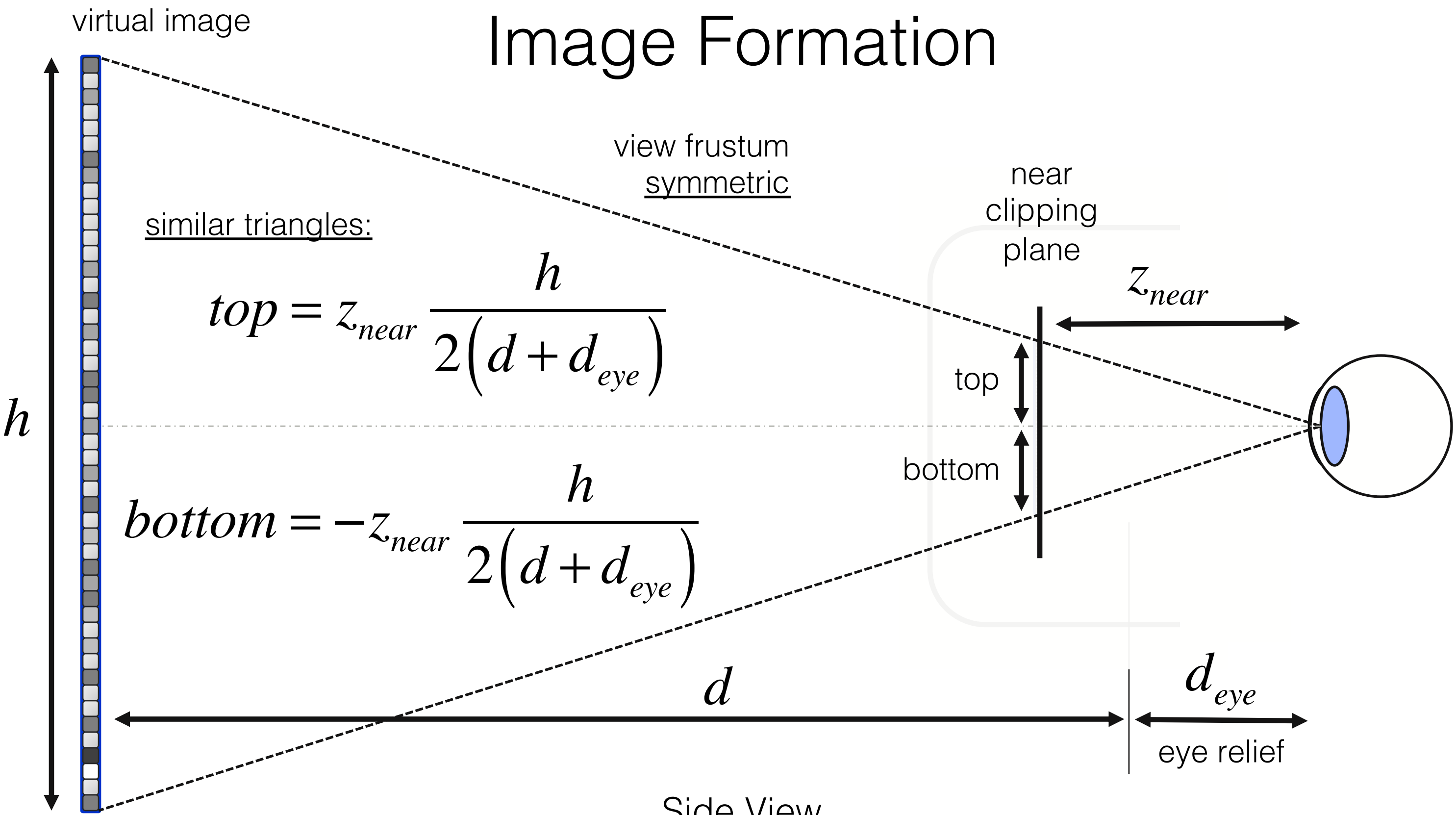
Side View

Image Formation

similar triangles:

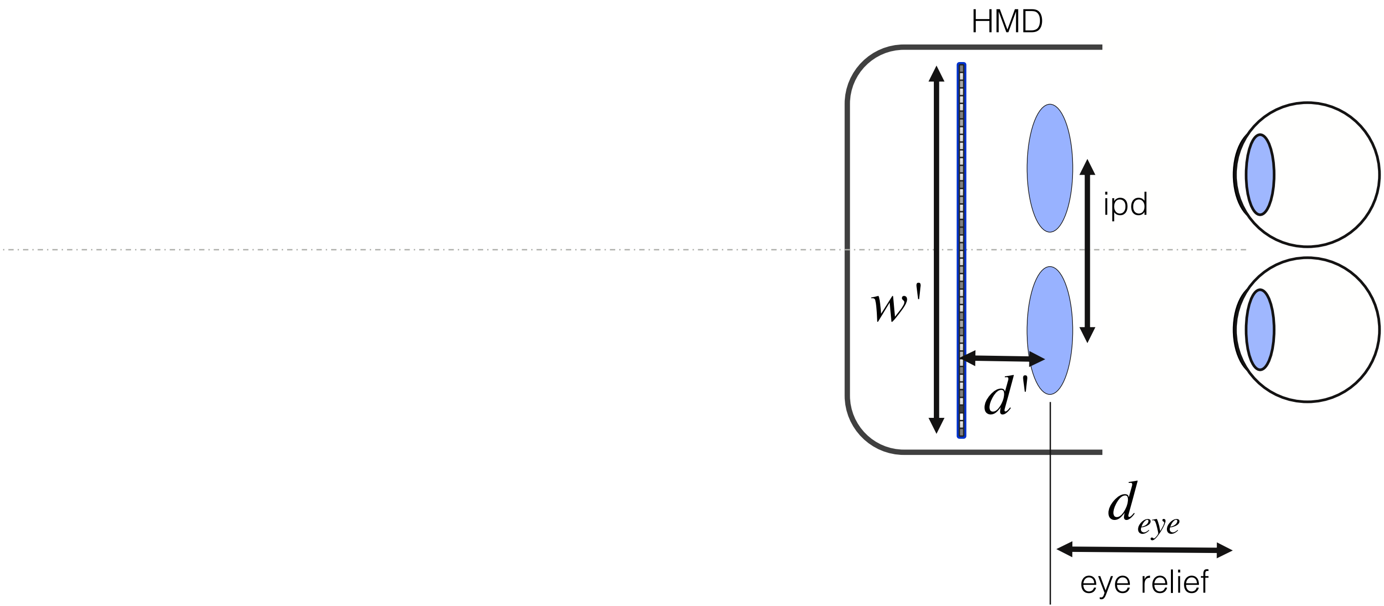
$$top = z_{near}\frac{h}{2\left(d + d_{eye}\right)}$$

$$bottom = -z_{near}\frac{h}{2\left(d + d_{eye}\right)}$$



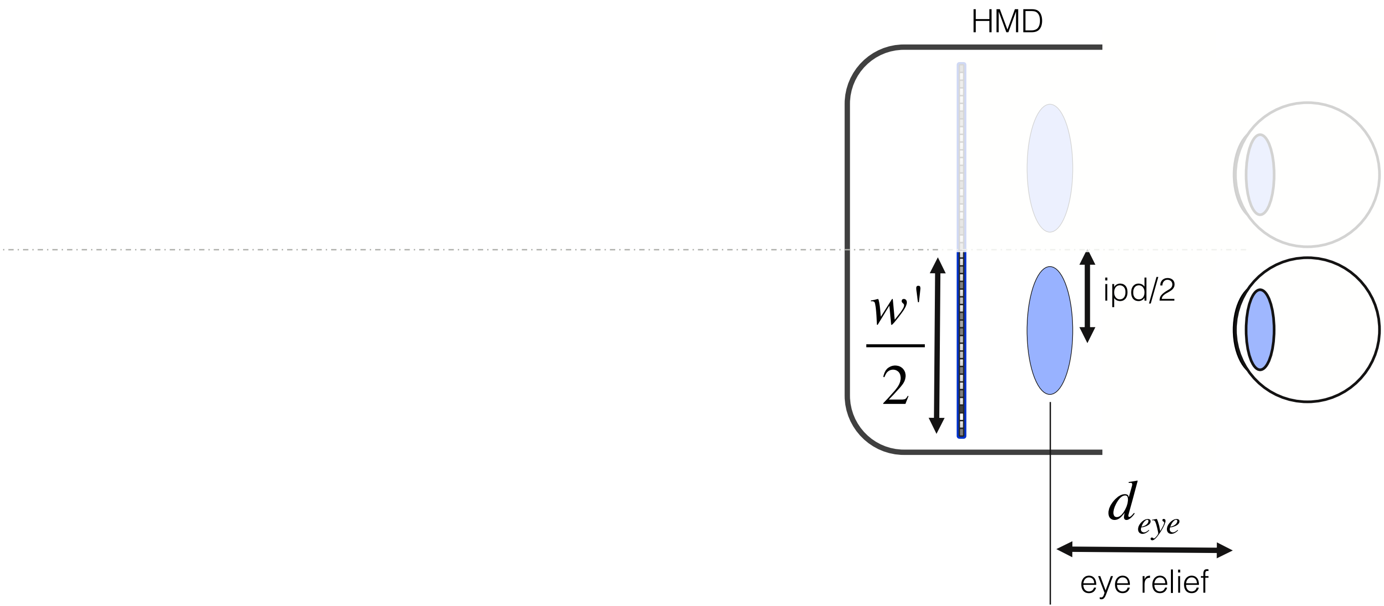
Side View

Image Formation



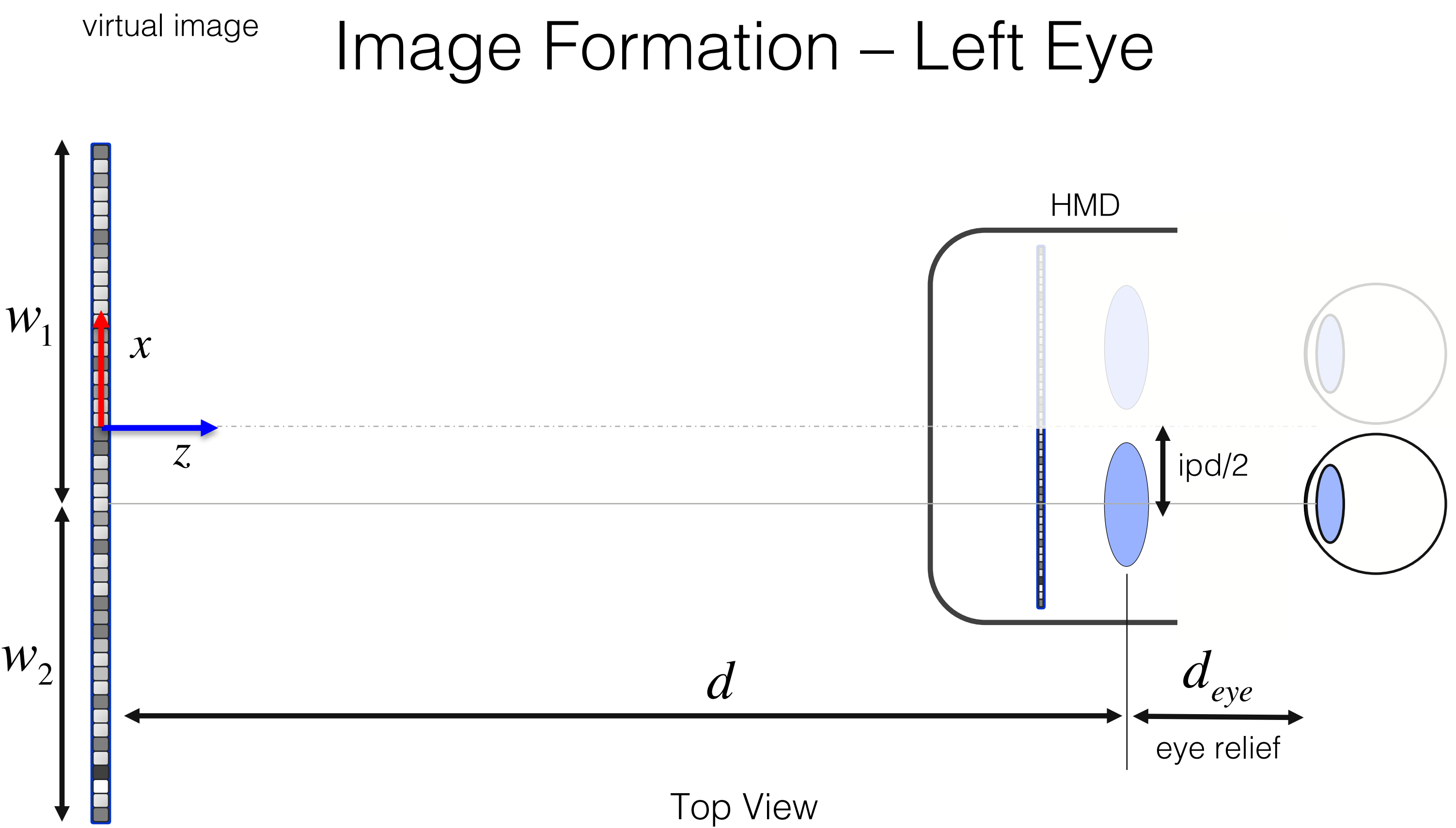
Top View

Image Formation – Left Eye



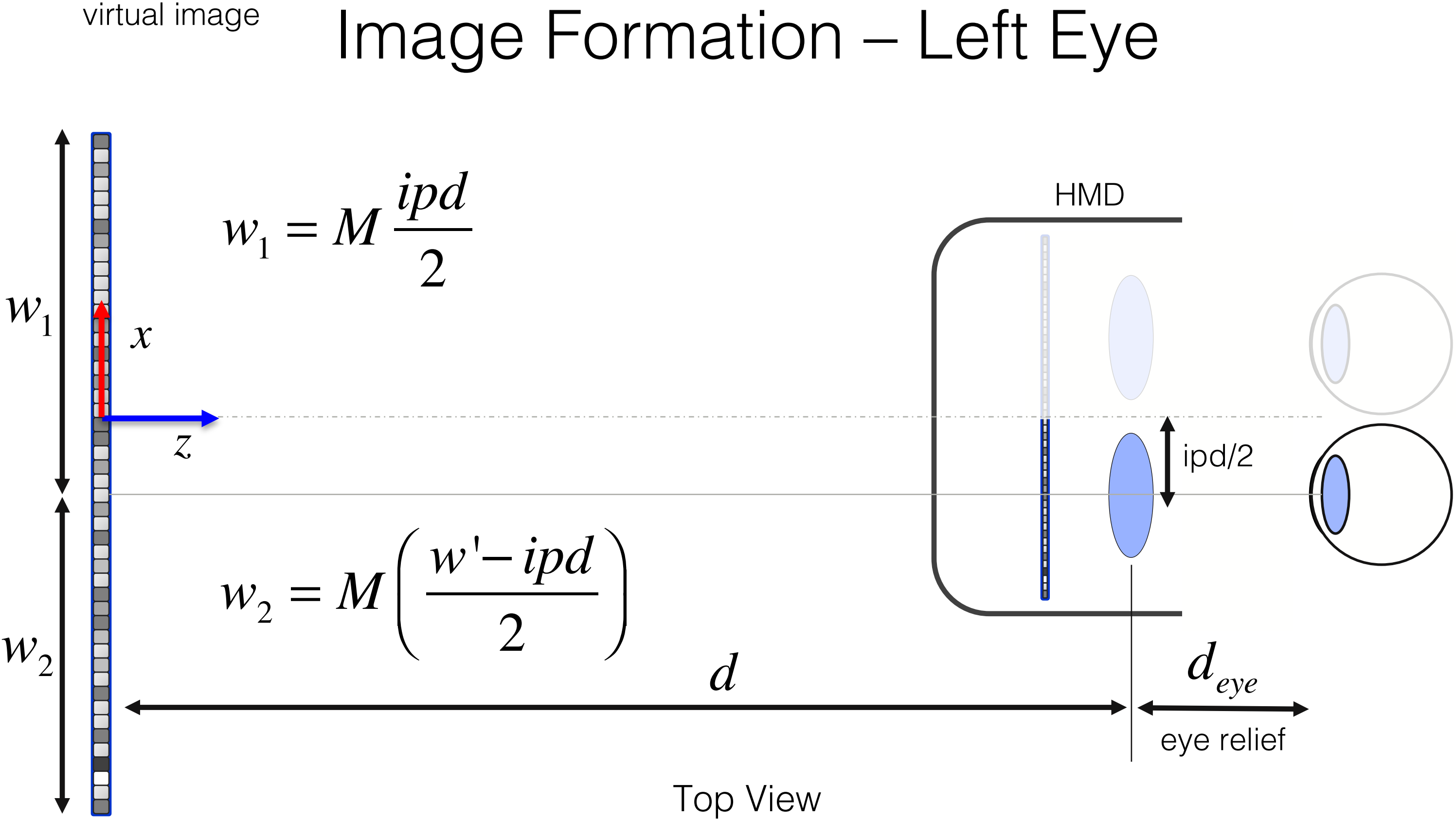
Top View

Image Formation – Left Eye



Top View

Image Formation – Left Eye



$$w_1 = M\frac{ipd}{2}$$

$$w_2 = M\left(\frac{w' - ipd}{2}\right)$$

w_1

w_2

d_{eye}

Image Formation – Left Eye

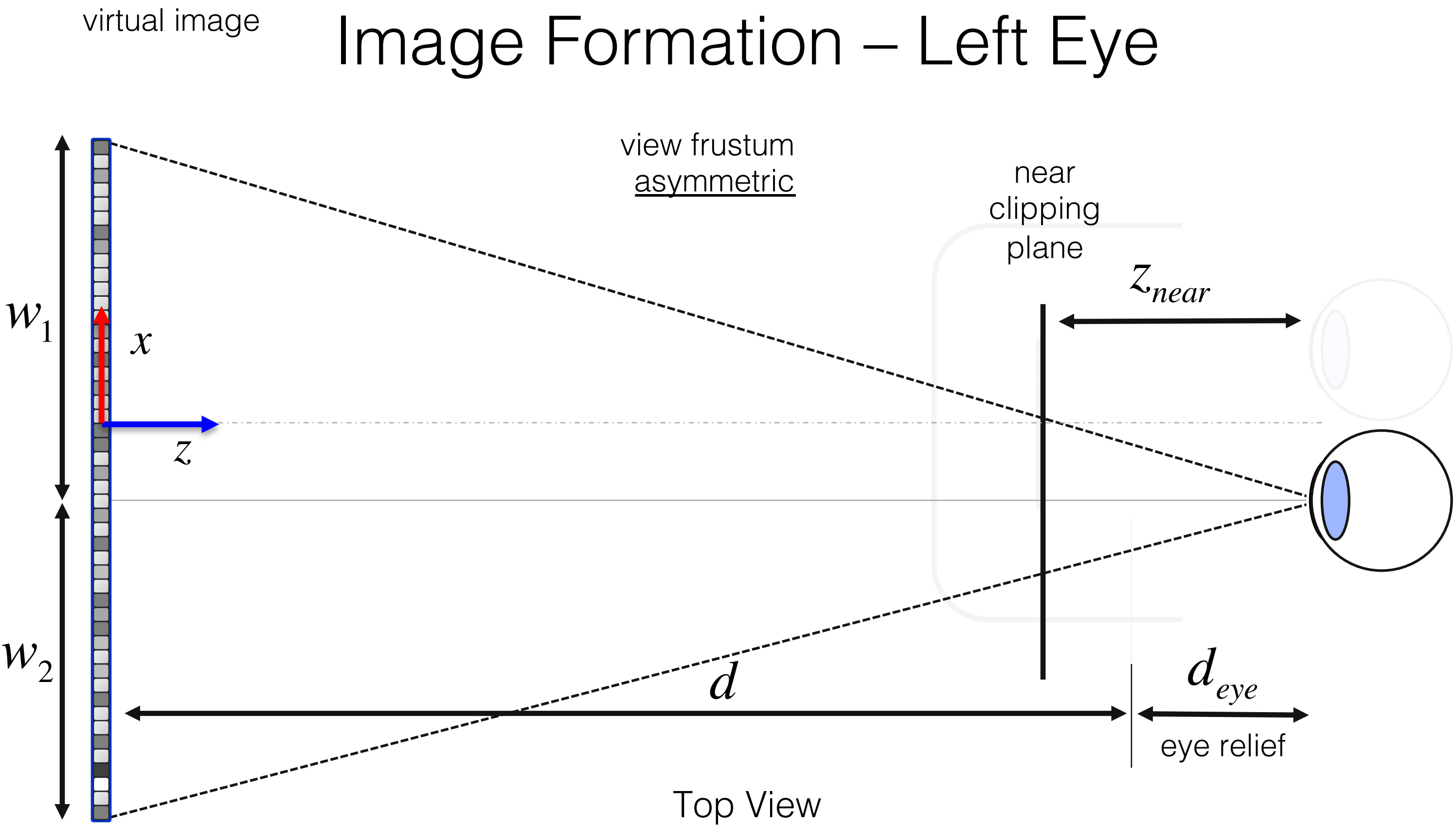


Image Formation – Left Eye

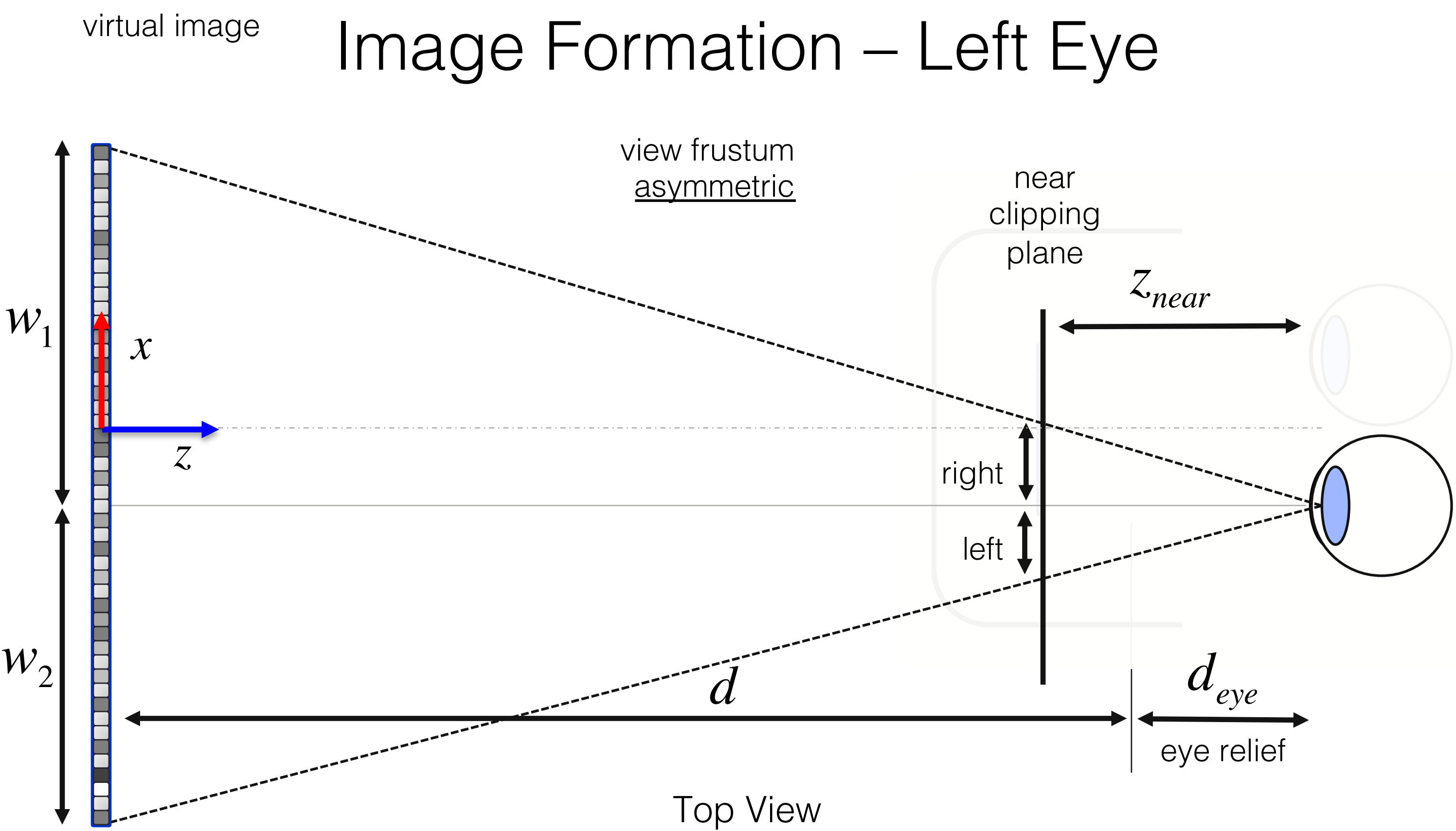


Image Formation – Left Eye

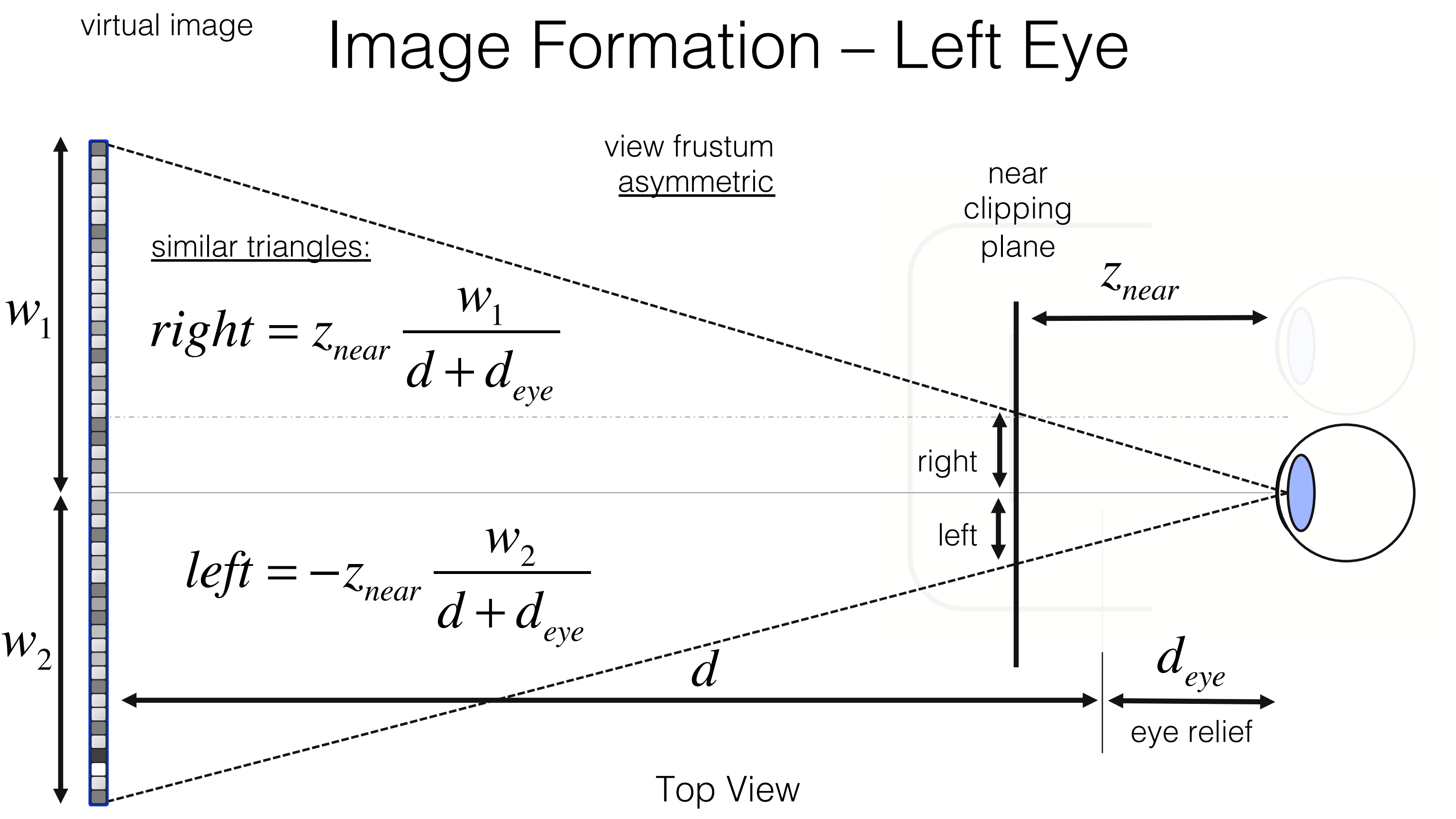
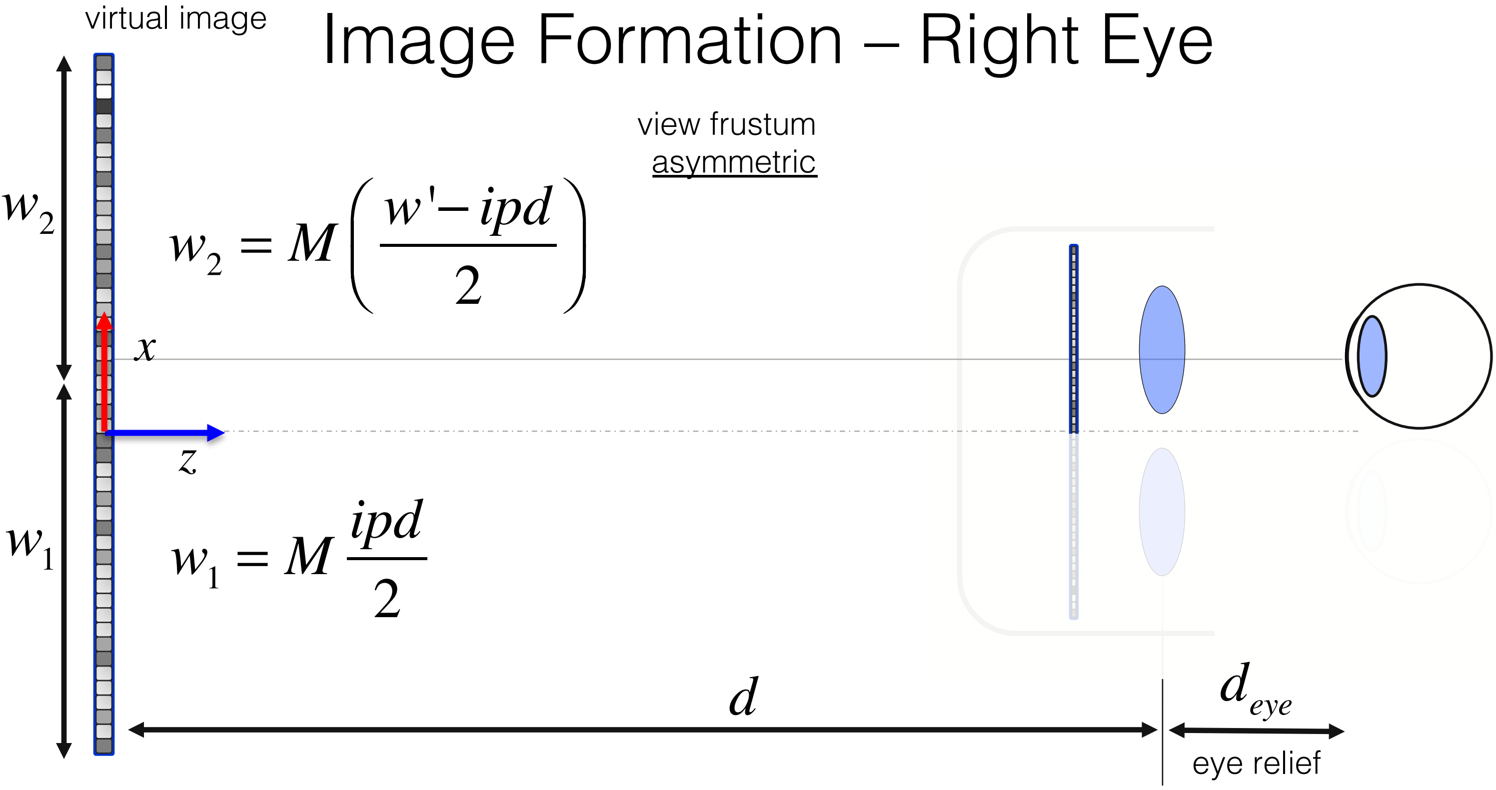


Image Formation – Right Eye

view frustum
asymmetric

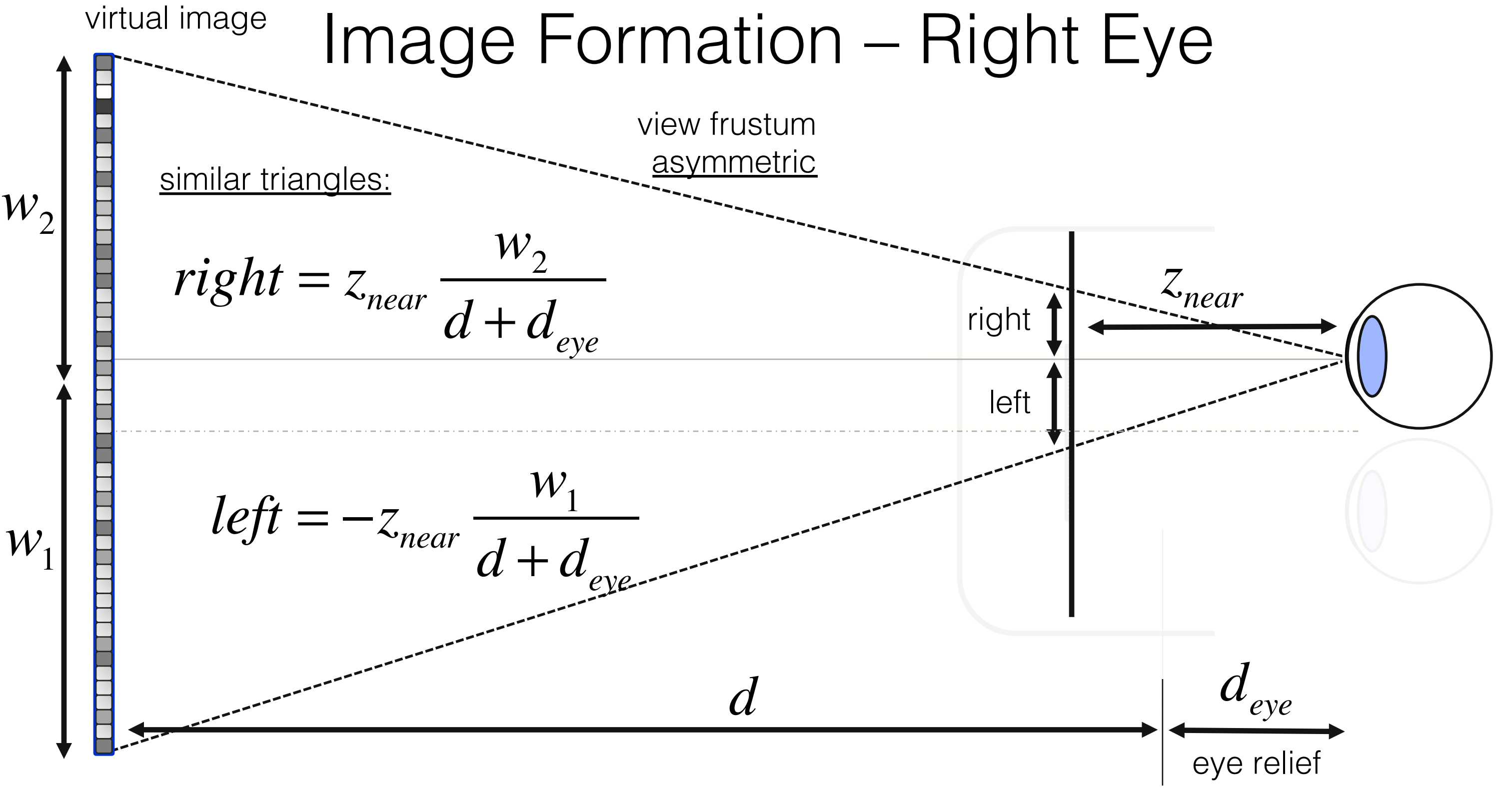
$$w_2 = M\left(\frac{w' - ipd}{2}\right)$$

$$w_1 = M\,\frac{ipd}{2}$$



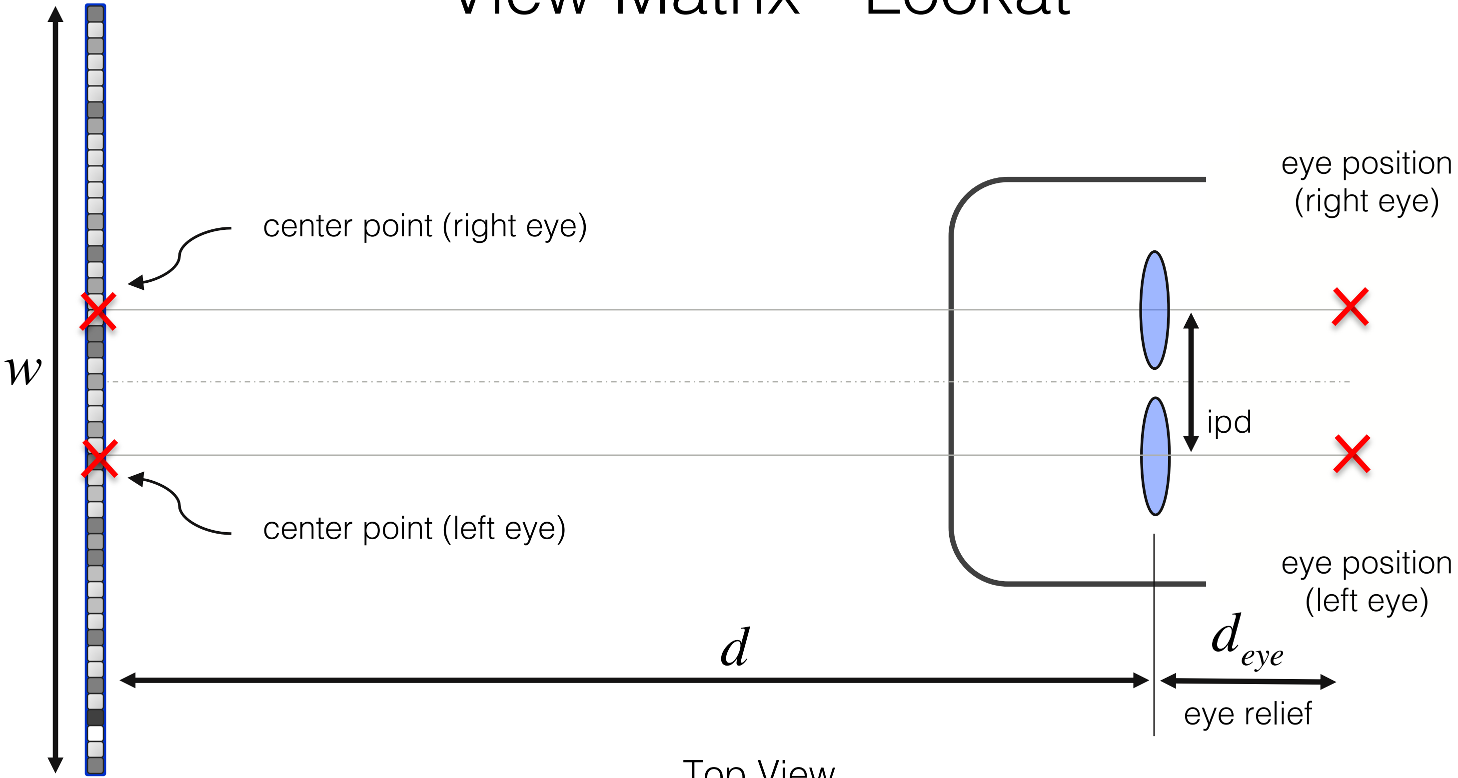
Top View

Image Formation – Right Eye



Top View

View Matrix - Lookat



Top View

Prototype Specs – View-Master Deluxe VR Viewer

- roughly follows Google Cardboard 2.0:
    - lenses focal length: 40 mm
    - lenses diameter: 34 mm
    - inter-lens distance: 64 mm
    - screen to lens distance: 39 mm
    - eye relief: 18 mm
- Topfoison 6” LCD: width 132.5 mm, height 74.5 mm; 1920x1080 px OR
- Topfoison 5.5” LCD: width 120.96 mm, height 68.03 mm; 1920x1080 px

Image Formation

- use these formulas to compute the perspective matrix in WebGL

- you can use:

```
THREE.Matrix4().makePerspective(left,right,top,bottom,near,far)

THREE.Matrix4().lookAt(eye,center,up) - attention: this only does
                                         rotation, not the translation,
                                         which is required in addition
                                         to the rotation!
```

- that's all you need to render stereo images on the HMD

Image Formation for More Complex Optics

- especially important in free-form optics, off-axis optical configurations & AR

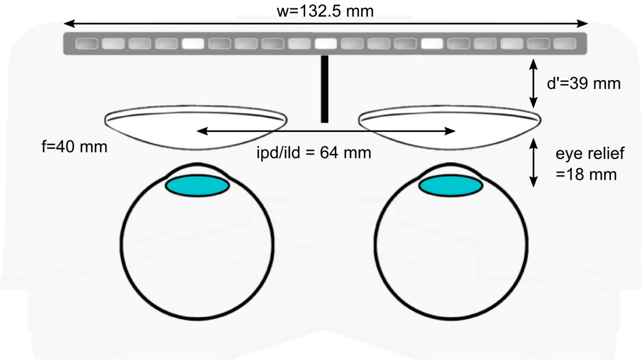
- use ray tracing – some nonlinear mapping from view frustum to microdisplay pixels

- much more computationally challenging & sensitive to precise calibration; our HMD and most magnifier-based designs will work with what we discussed so far

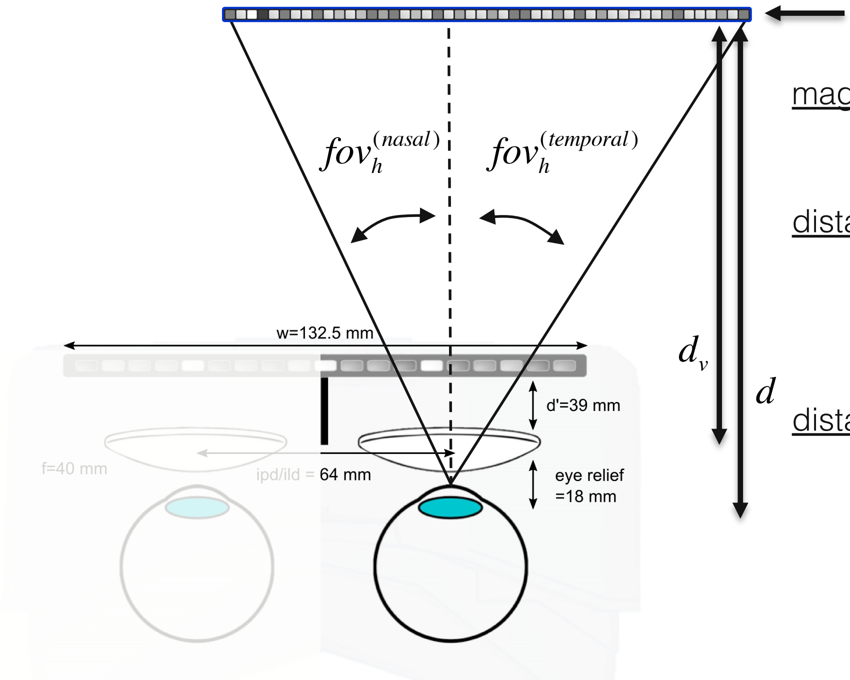
Field of View and Visual Field

Example Calculations for Field of View

- use Google Cardboard 2 lenses (f=40mm, d'=39mm, interpupillary/interlens distance = 64mm, eye relief = 18mm)
- Topfoison 6" LCD panel (132.5 x 74.5 mm)



Example Calculations for Field of View



magnification: $M = \frac{f}{f-d'} = 40$

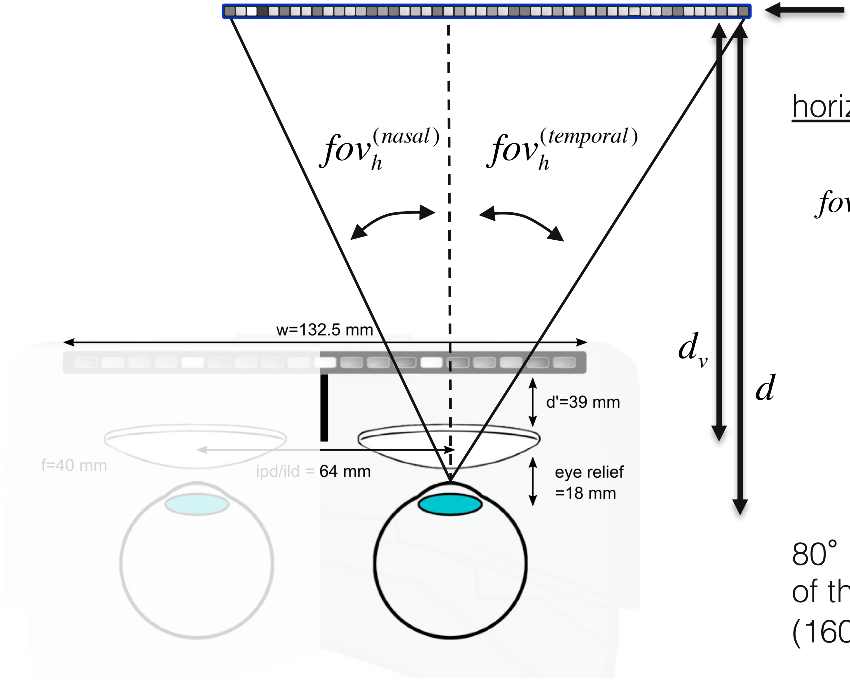
distance lens-virtual image:

$$d_v = \left| \frac{1}{\frac{1}{f} - \frac{1}{d'}} \right| = 1{,}560\,mm$$

distance eye-virtual image:

$$d = d_v + d_{eye} = 1{,}578\,mm$$

Example Calculations for Field of View



horizontal field of view:

$$fov_h = fov_h^{(nasal)} + fov_h^{(temporal)}$$

$$= \tan^{-1}\left(\frac{M\, {}^{ipd}\!/_{2}}{d}\right) + \tan^{-1}\left(\frac{M\, {}^{(w-ipd)}\!/_{2}}{d}\right)$$

$$= 39^\circ + 41^\circ = \boxed{80^\circ}$$

80° horizontal field of view is approx. 50% of the horizontal visual field of a single eye (160° total)

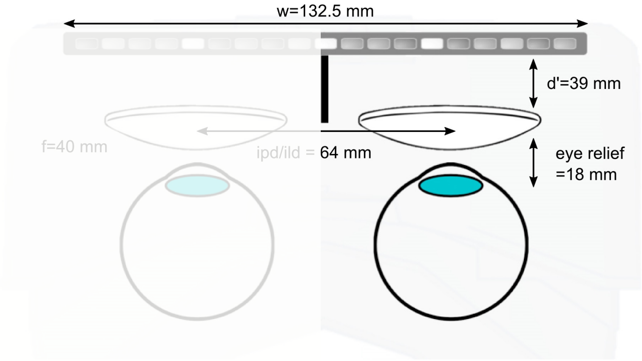
Example Calculations for Field of View

virtual image of right display side

vertical field of view:

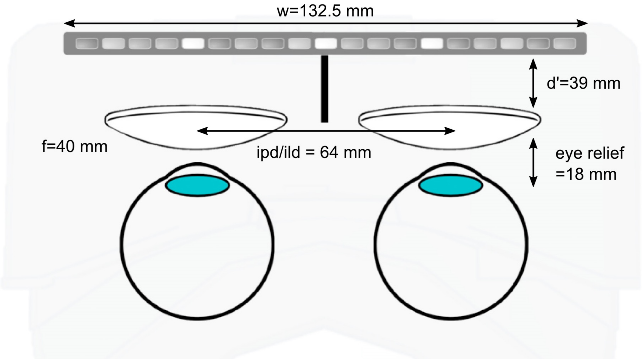
$$fov_v = fov_v^{(\text{superior})} + fov_h^{(\text{inferior})}$$

$$= 2\tan^{-1}\left(\frac{M\, h/2}{d}\right) = 87^\circ$$



87° vertical field of view is approx. 64% of the vertical visual field of a single eye (135° total)

Example Calculations for Field of View



total monocular field of view of both eyes:

$$fov_h^{(total)} = 2\,fov_h^{(\text{temporal})} = 82^\circ$$

82° monocular field of view is approx. 41% of the full monocular visual field of both eyes (200° total)

binocular field of view of both eyes:

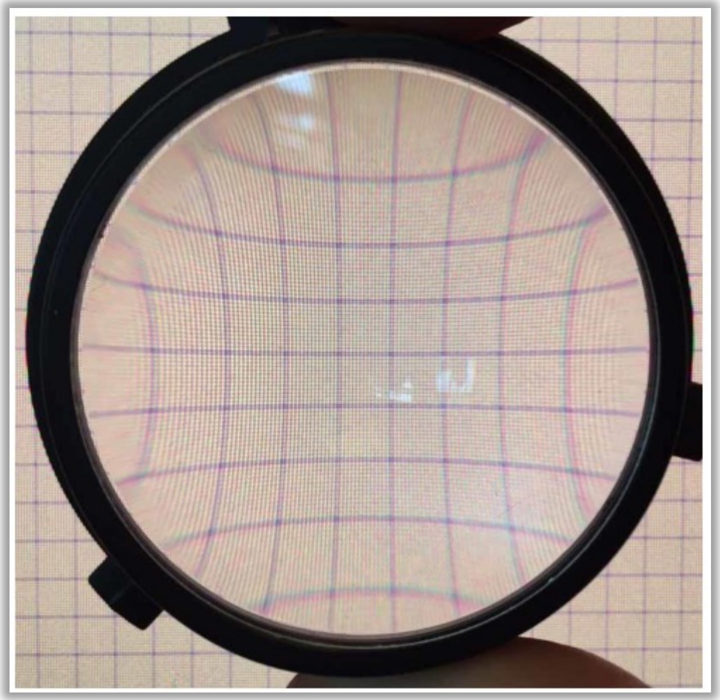
$$fov_h^{(total)} = 2\,fov_h^{(\text{nasal})} = 78^\circ$$

78° binocular field of view is approx. 65% of the binocular visual field of both eyes (120° total)

Lens Distortion Correction

All lenses introduce image distortion, chromatic aberrations, and other artifacts – we need to correct for them as best as we can in software!

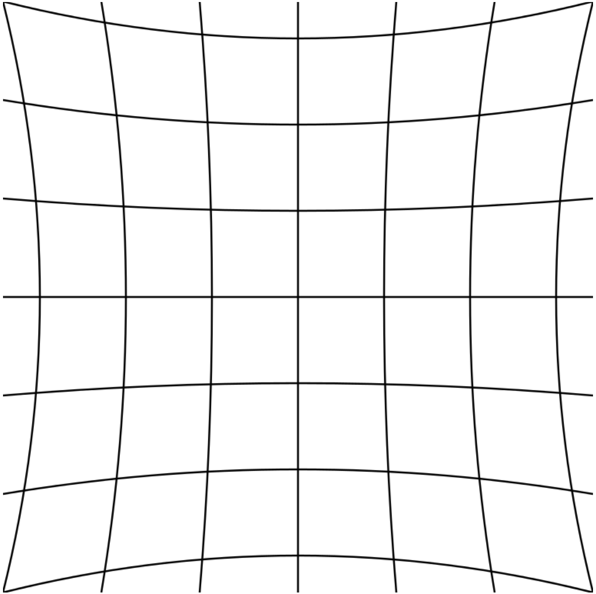
Lens Distortion



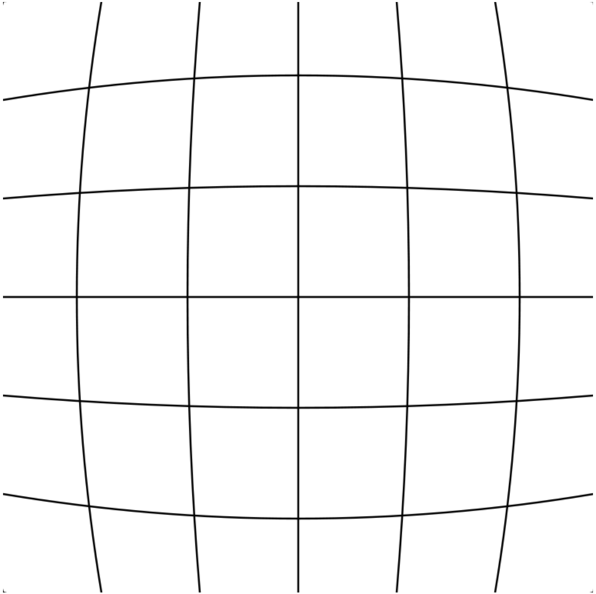
- grid seen through HMD lens
- lateral (xy) distortion of the image
- chromatic aberrations: distortion is wavelength dependent!

image from: https://www.slideshare.net/Mark_Kilgard/nvidia-opengl-in-2016

Lens Distortion

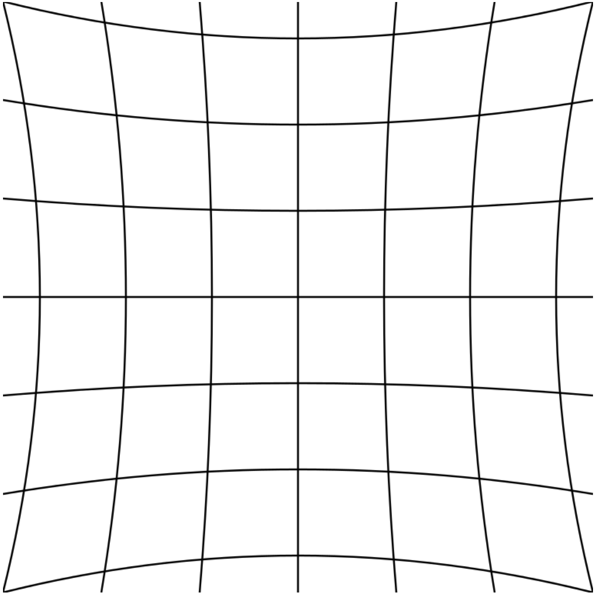


Pincussion Distortion



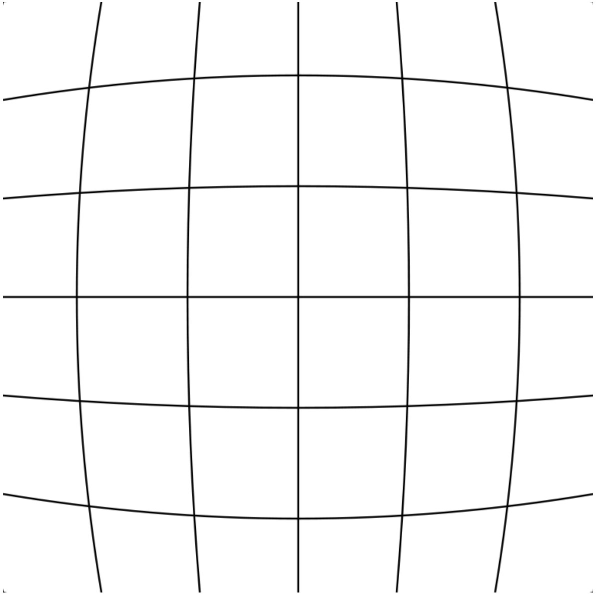
Barrel Distortion

Lens Distortion



Pincussion Distortion

<u>optical</u>



Barrel Distortion

<u>digital correction</u>

Lens Distortion

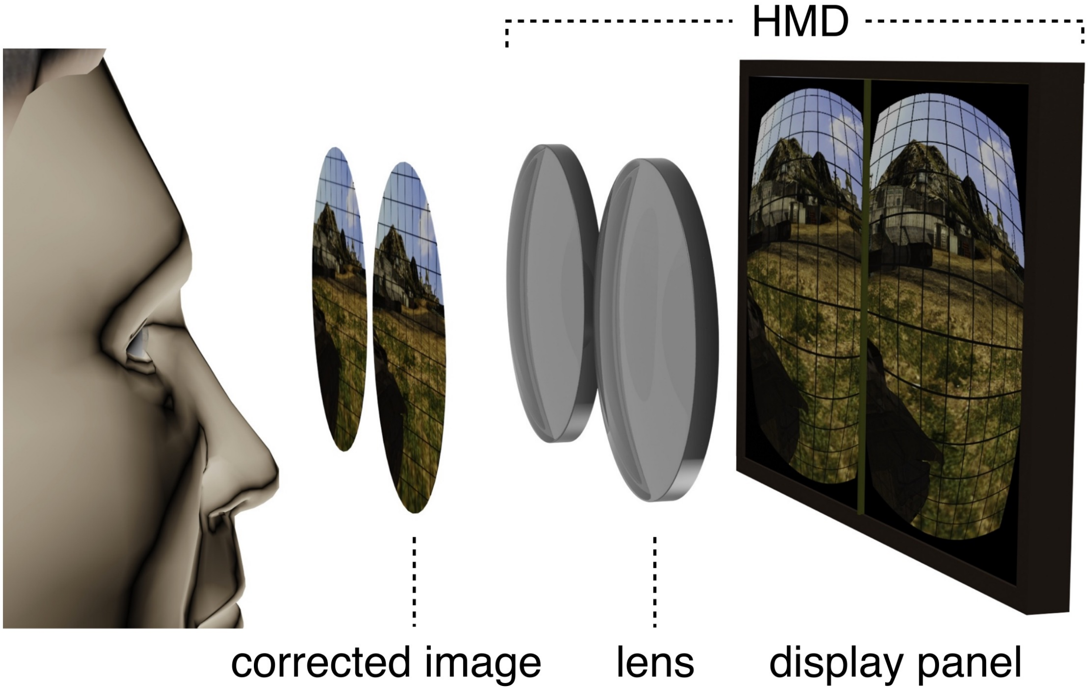
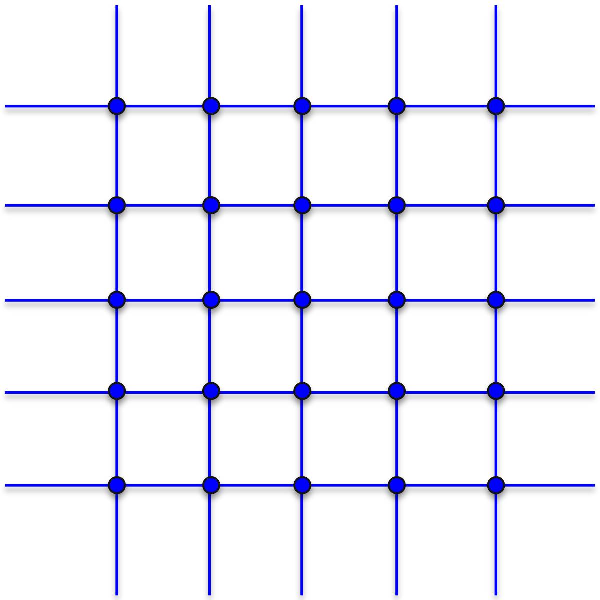


image from: https://www.slideshare.net/Mark_Kilgard/nvidia-opengl-in-2016

Lens Distortion

- x_u, y_u undistorted point



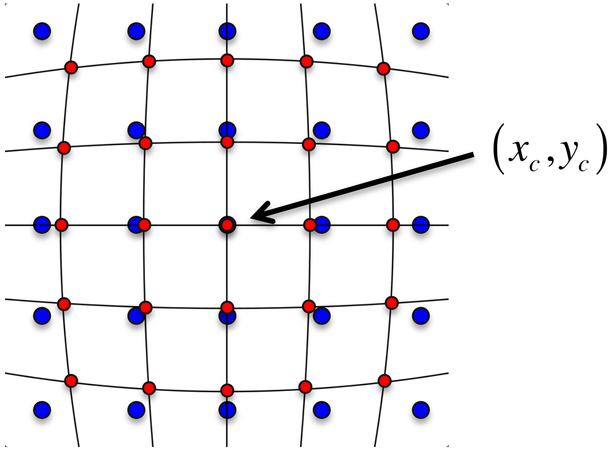
Lens Distortion

- x_u, y_u undistorted point
- $x_d \approx x_u\left(1 + K_1 r^2 + K_2 r^4\right)$

  $y_d \approx y_u\left(1 + K_1 r^2 + K_2 r^4\right)$

x_d, y_d	distorted point coordinates
K_1, K_2	distortion coefficients
r	normalized distance from center
x_c, y_c	center of optical axis

→ this is the origin, i.e. all other points are defined relative to this



Barrel Distortion

digital correction

Lens Distortion

- x_u, y_u undistorted point
- $x_d \approx x_u\left(1 + K_1 r^2 + K_2 r^4\right)$

  $y_d \approx y_u\left(1 + K_1 r^2 + K_2 r^4\right)$

x_d, y_d distorted point coordinates

K_1, K_2 distortion coefficients

r normalized distance from center

x_c, y_c center of optical axis

→ this is the origin, i.e. all other points are defined relative to this

NOTES:

- center is assumed to be the center point (on optical axis) on screen
- distortion is radially symmetric around center point
- easy to get confused!
- can implement in fragment shader (not super efficient, but easier for us)

Normalizing r

- x_u, y_u undistorted point
- $x_d \approx x_u\left(1 + K_1 r^2 + K_2 r^4\right)$

  $y_d \approx y_u\left(1 + K_1 r^2 + K_2 r^4\right)$

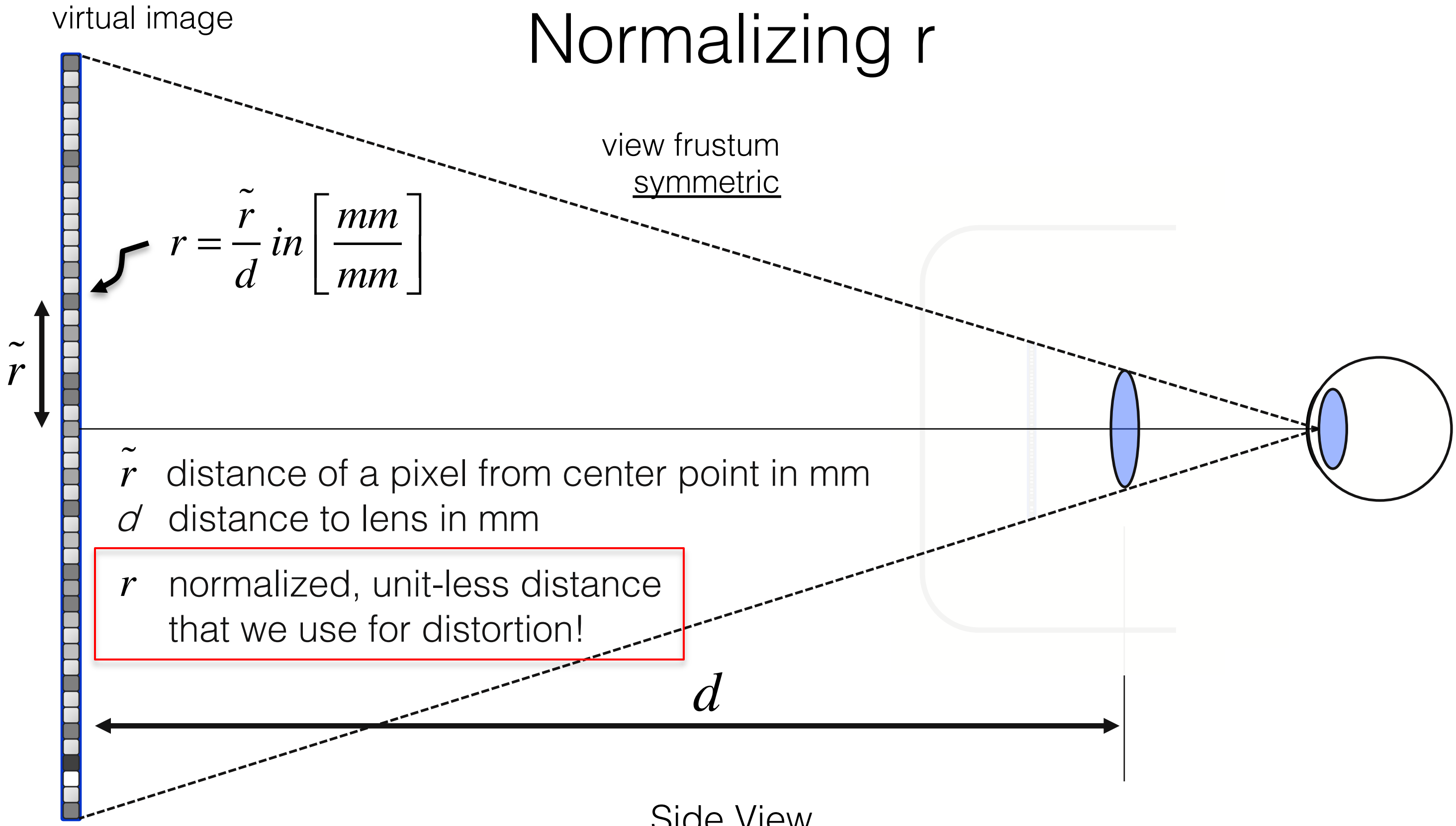
un-normalized radial distance from center:

$$\tilde{r}^2 = \left(x_u - x_c\right)^2 + \left(y_u - y_c\right)^2 \longrightarrow$$

Calculate $\tilde{r}$ in metric units, e.g. mm. Need physical size of the pixels of your screen for this!

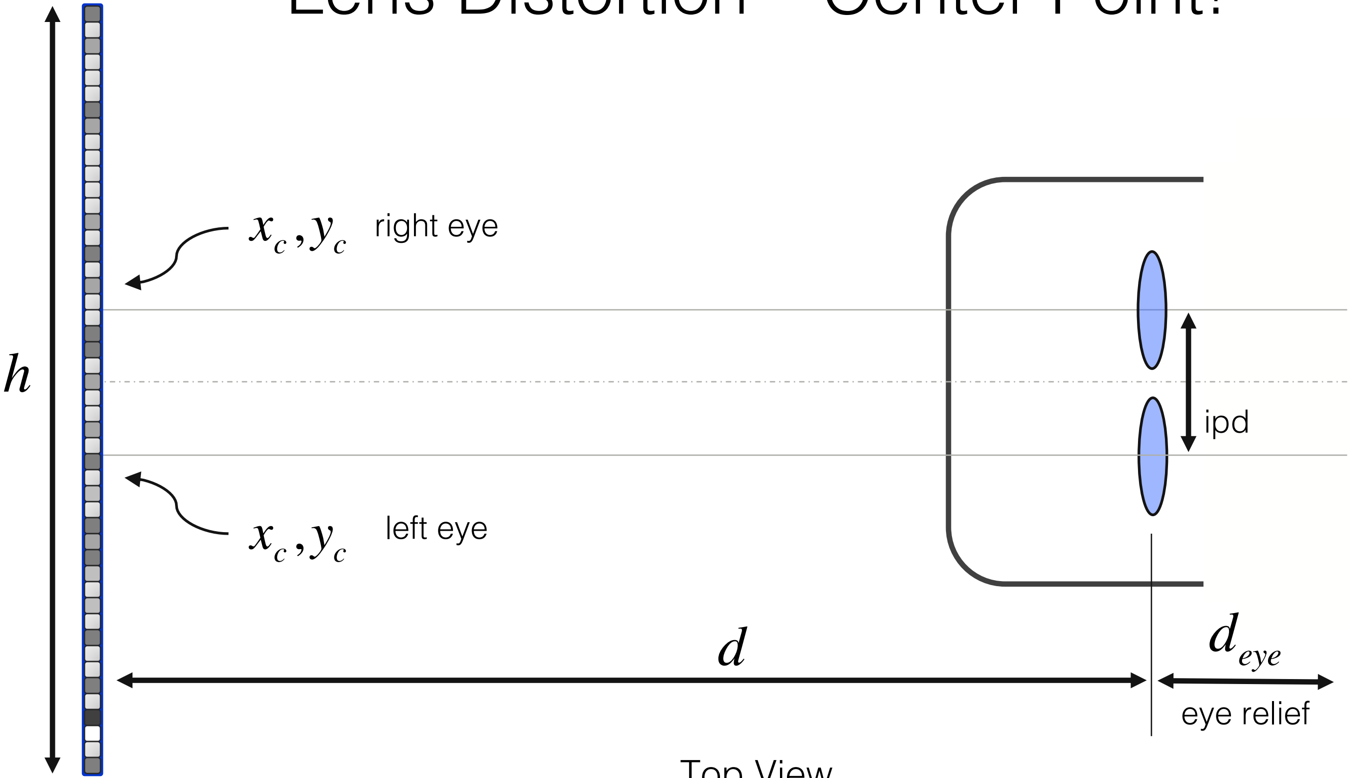
x_c, y_c center

Normalizing r



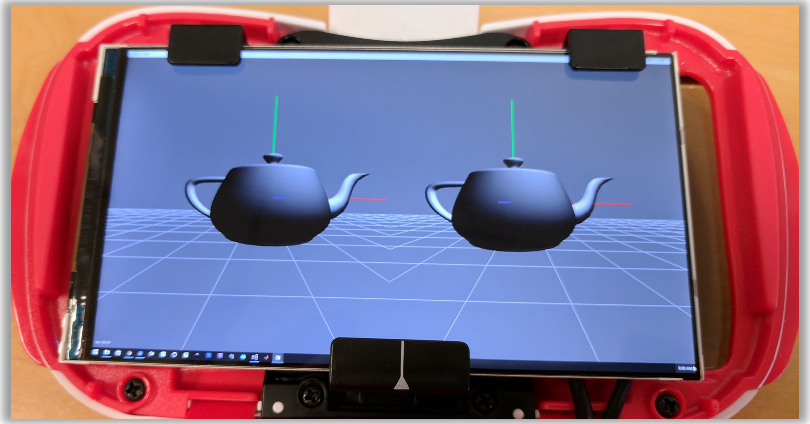
Side View

Lens Distortion – Center Point!

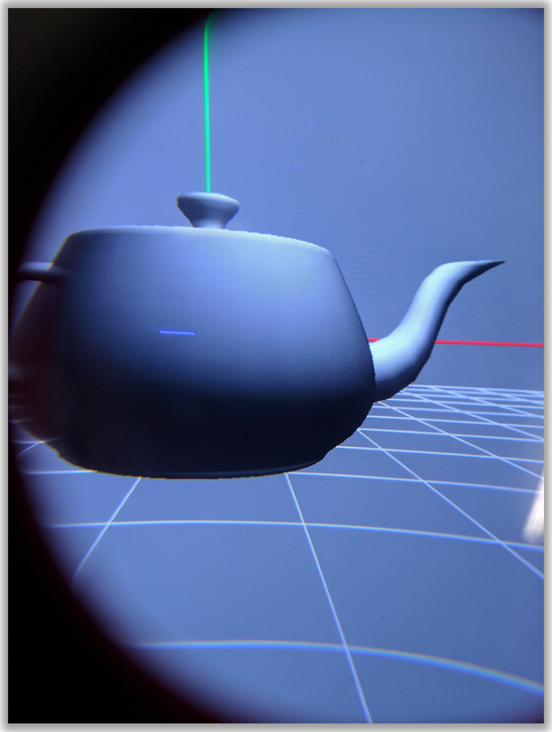


Top View

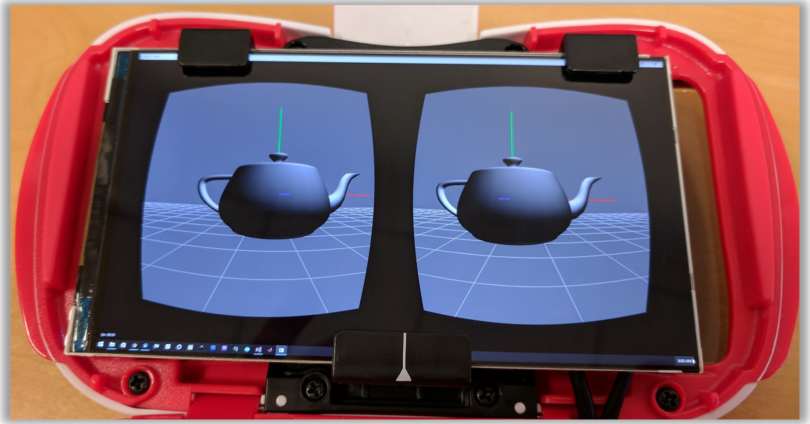
Lens Distortion Correction Example



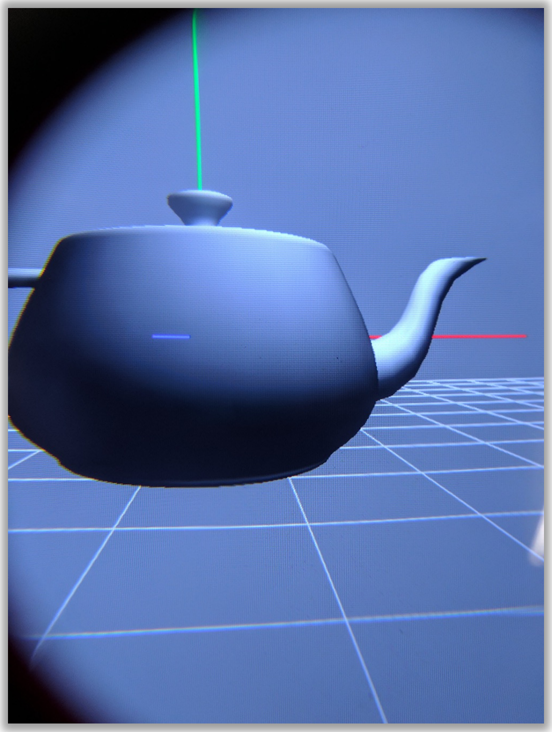
stereo rendering without lens distortion correction



Lens Distortion Correction Example

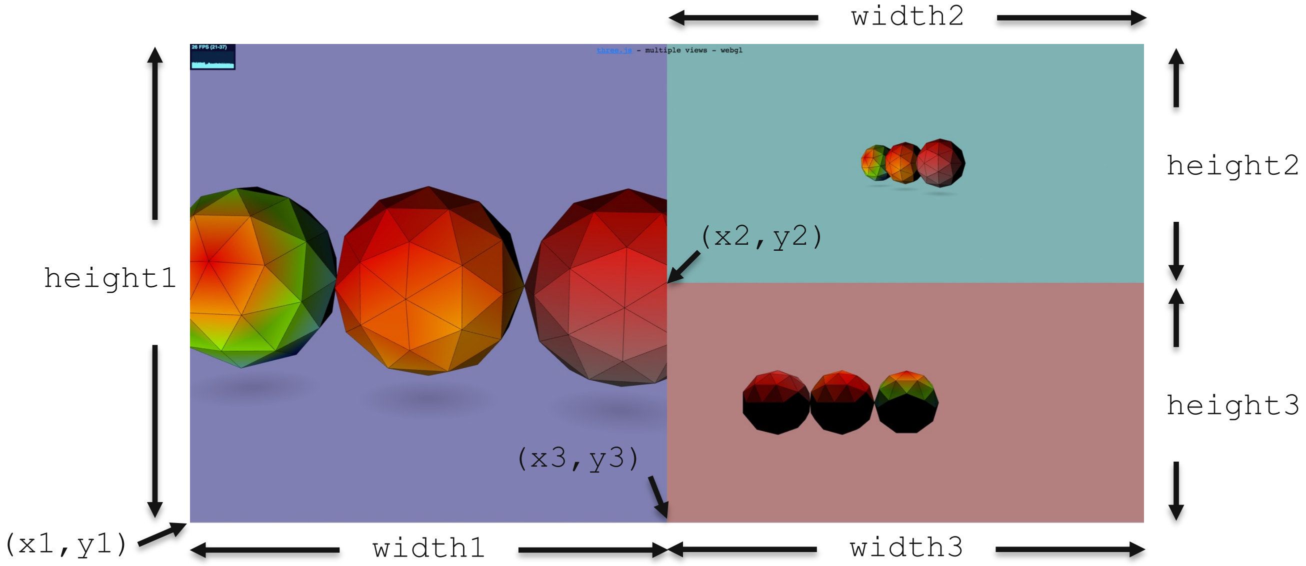


stereo rendering <u>with</u> lens distortion correction



How to Render into Different Parts of the Window?

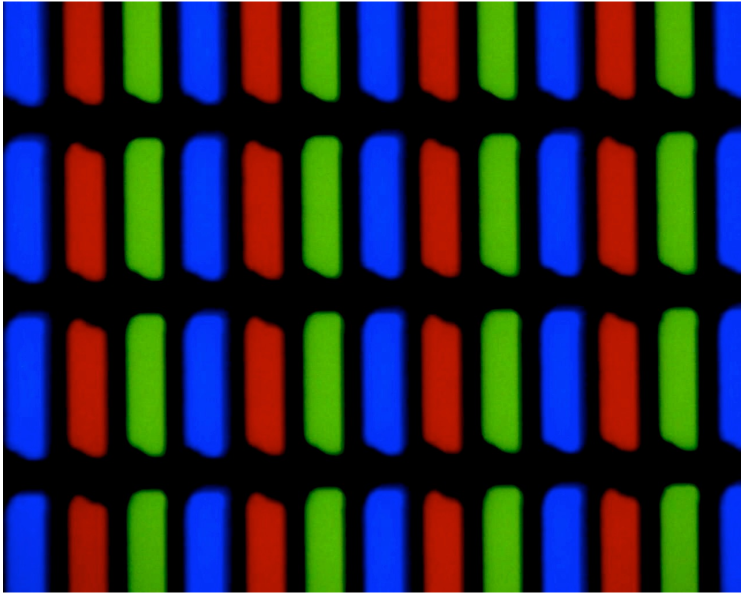
- `WebGLRenderer.setViewport(x,y,width,height)`
- `x,y` lower left corner; width, height viewport size



Overview of Microdisplays

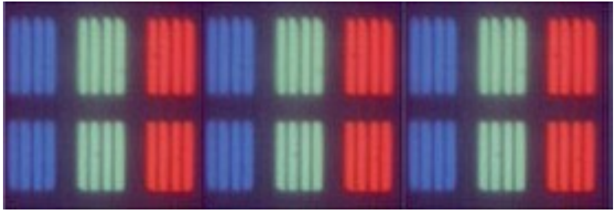
Liquid Crystal Display (LCD) - Subpixels

wikipedia

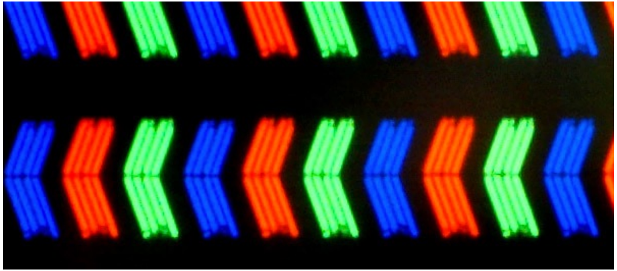


TN subpixels

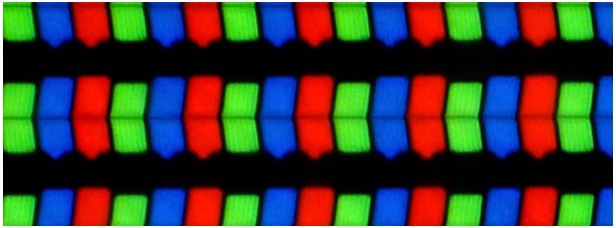
IPS



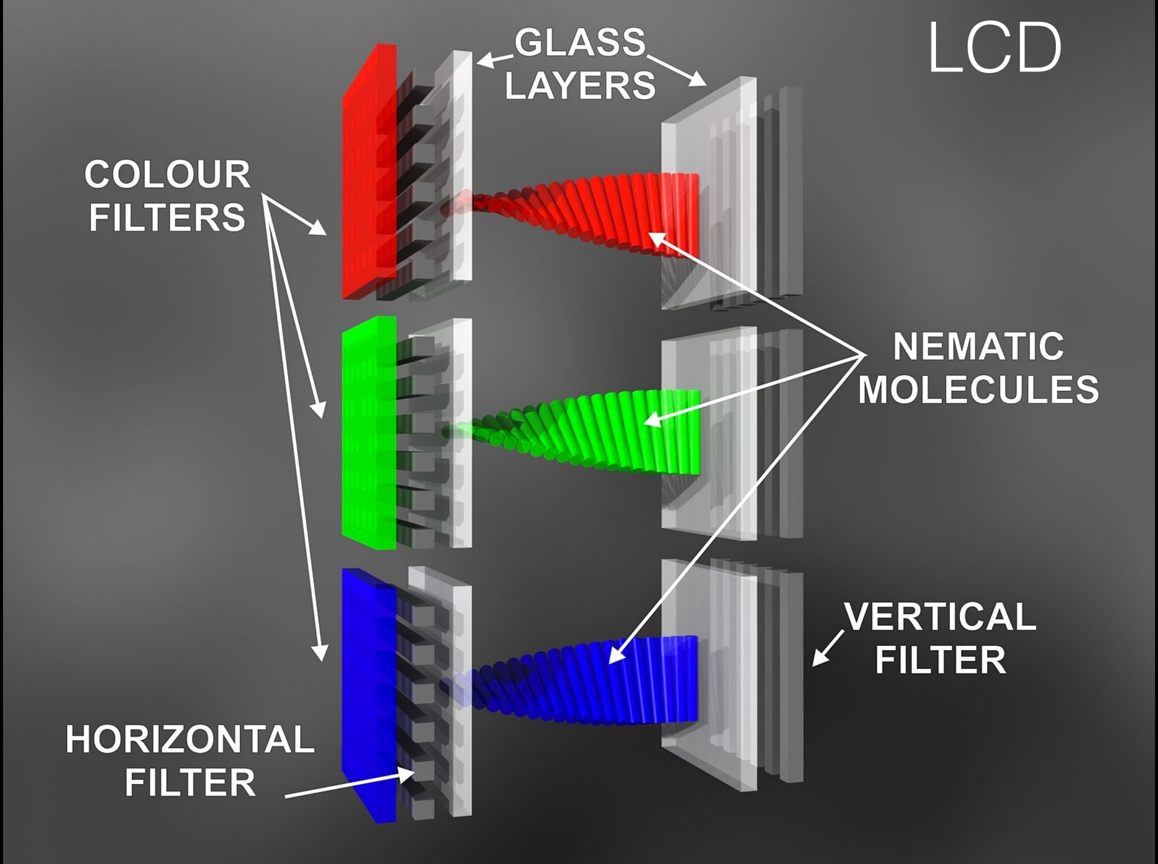
S-IPS



IPS



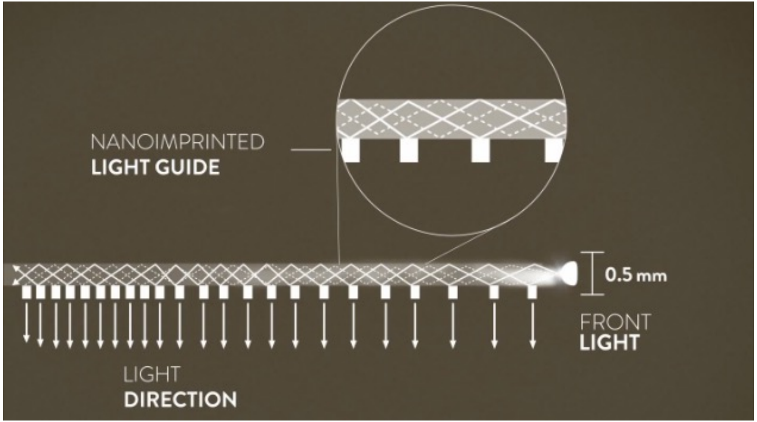
LCD

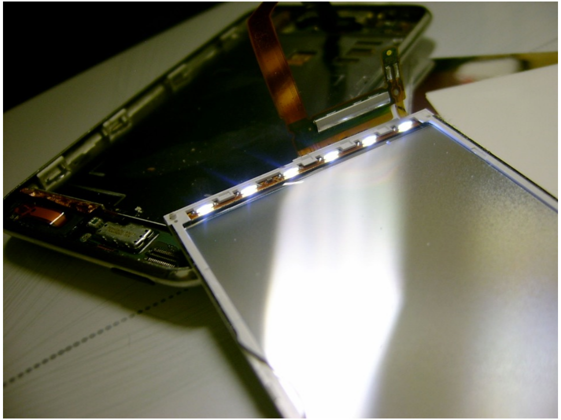


wikipedia

LCD Backlight

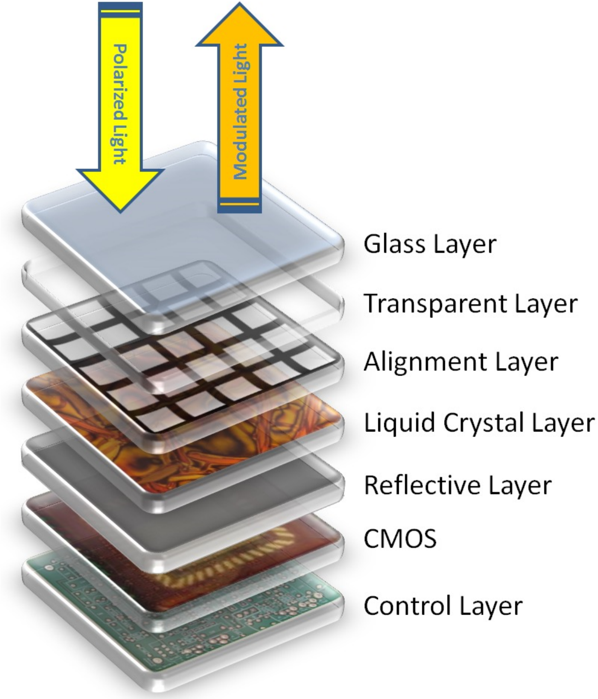
extremetech.com





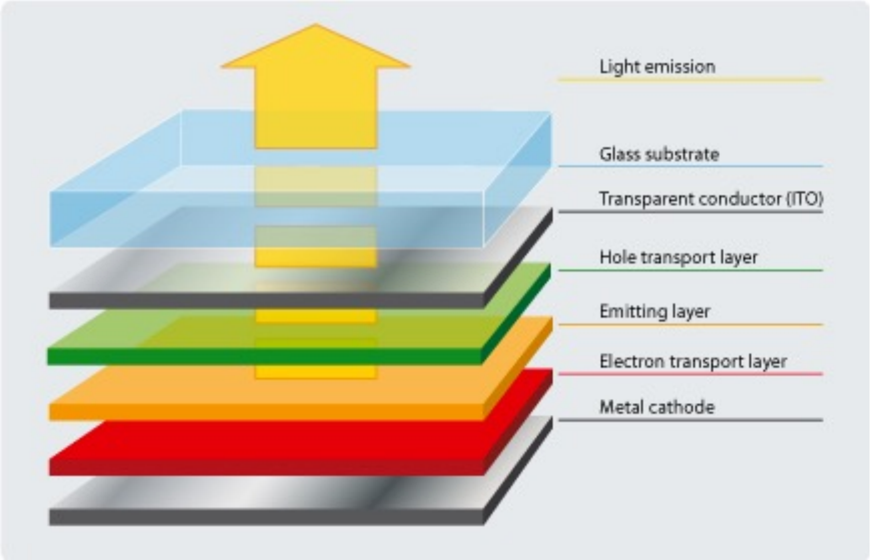
wikipedia

Liquid Crystal on Silicon (LCoS)



- basically a reflective LCD
- standard component in projectors and head mounted displays
- used e.g. in google glass

Organic Light Emitting Diodes (OLED)

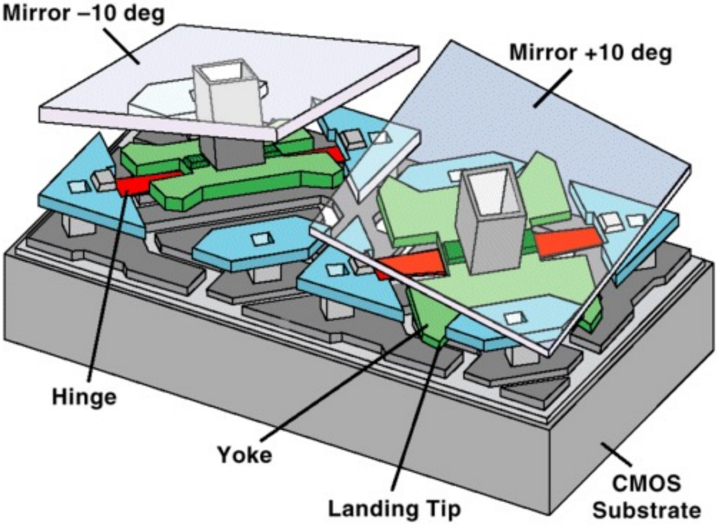


lightrabbit.co.uk

- Self emissive
- Lower persistence (can turn on and off faster than LCD/LCoS, which is great for VR)
- used e.g. VR-compatible phones, like Google's Pixel

Digital Micromirror Device (DMD)

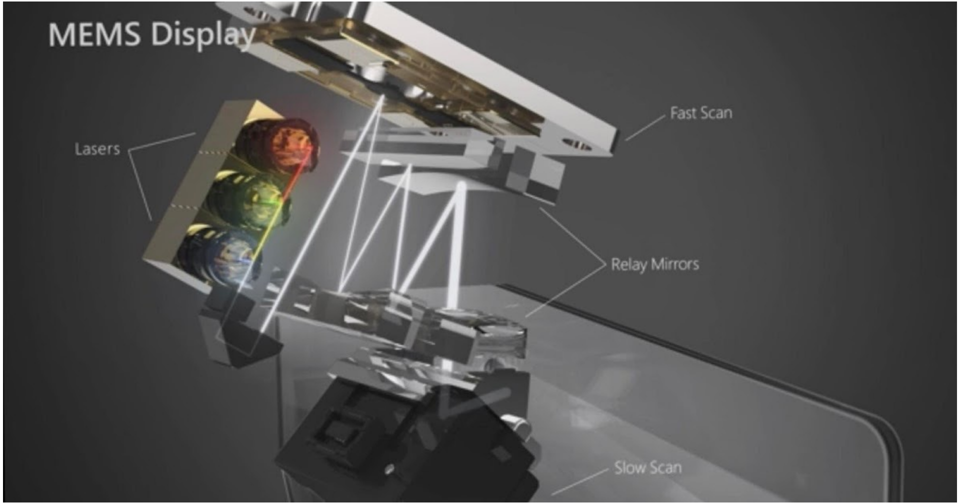
- developed by TI
- MEMS device
- binary states (e.g. +/- 10 degrees)
- gray-level through pulse width modulation (PWM)
- Super-fast (10-20 kHZ binary display
- More light efficient than LCD/LCoS!



Texas Instruments

Laser Beam Scanning (LBS)

- Used e.g. by Microsoft Hololens 2 (shown)
- 1 laser source (per color) with single 2D or two 1D MEMS mirrors scanning at some fixed rate



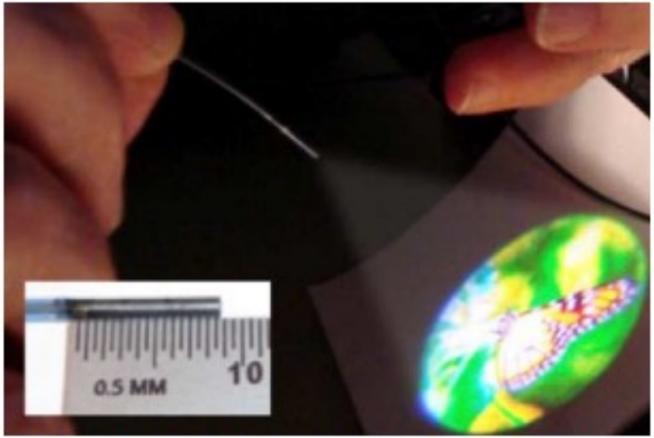
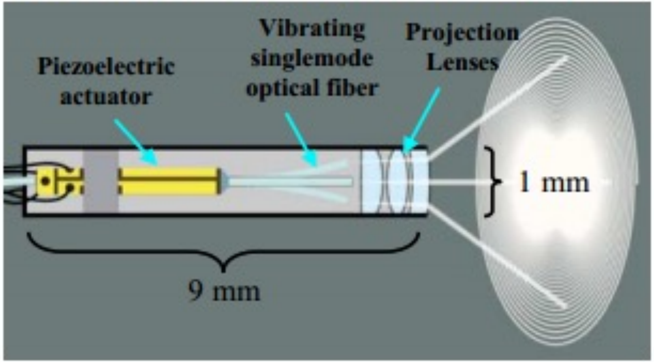


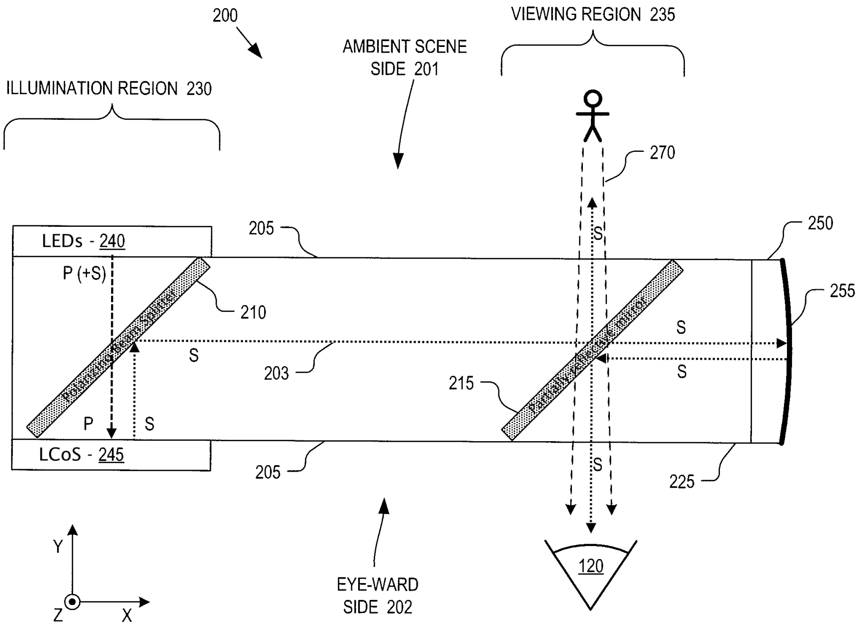
Figure 1. 1 mm x 9 mm scanning fiber projector.



B. T. Schowengerdt, R. Johnston, C.D. Melville, E.J. Seibel. 3D Displays Using Scanning Laser Projection. SID 2012.

Next Lecture: HMD Displays Optics II

- advanced VR & AR optics



drawing from Google Glass patent