Interactive Surfaces & Spaces

Lecture 08: ISS Technologies

© 2025 Dr. Florian Echtler

Aalborg University

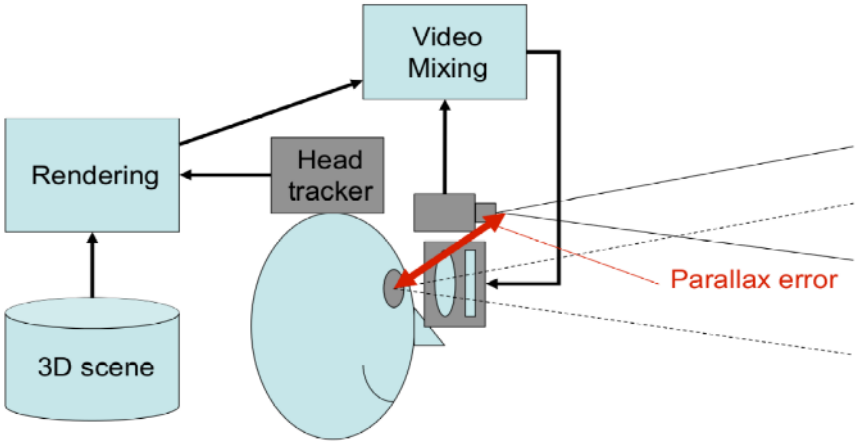
ISS Technologies

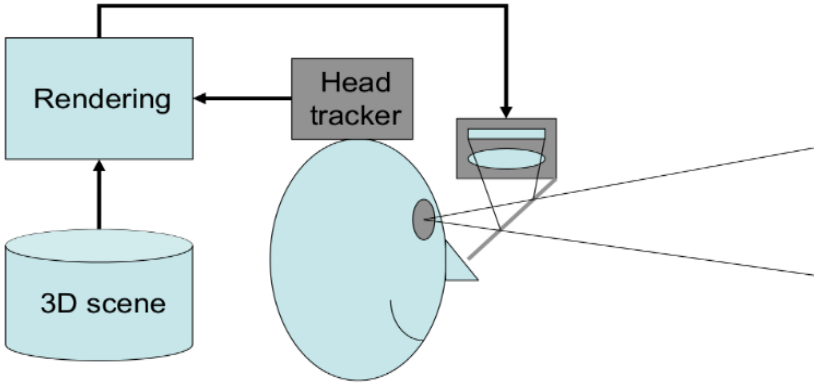
- Output
  - Head-Mounted Displays (HMDs)
  - Screens & 3D glasses
  - Sound & active Tangibles
- Input
  - Touch & Tangibles
  - Depth Cameras
  - 6 DoF Tracking

Head-Mounted Displays (HMDs)

Image source (FU): LMU Lecture by J. Wagner

- Used for interactive *spaces*
- See lecture 5 for details





Screens

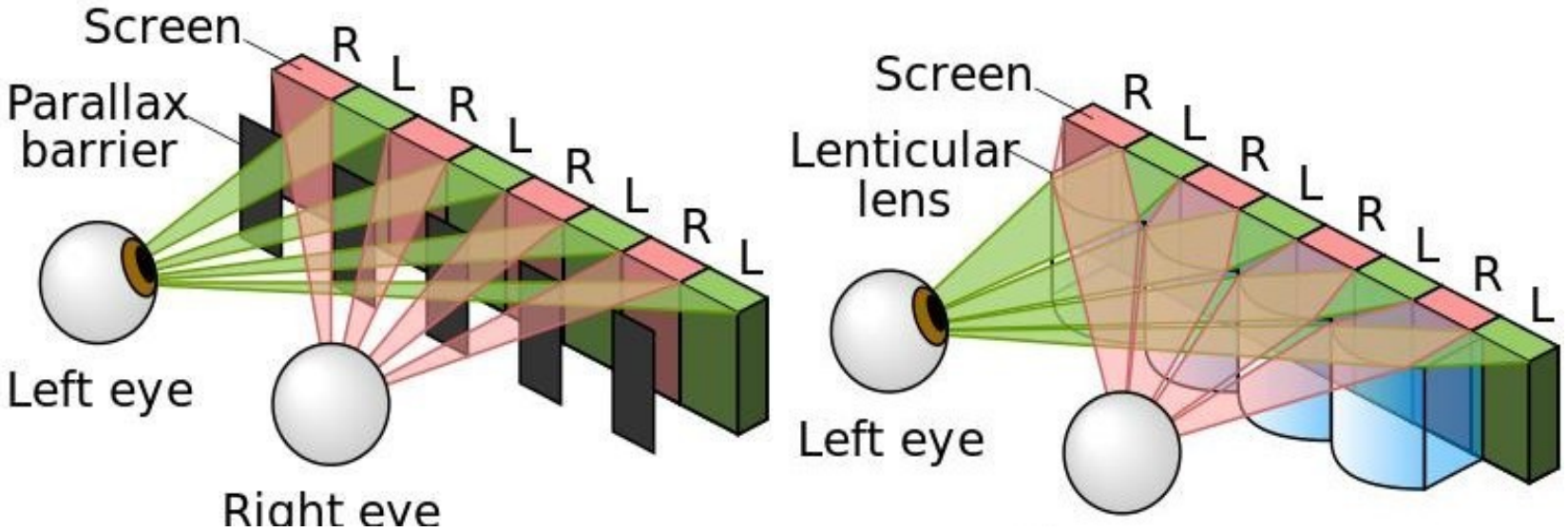
Image source (CC): https://www.flickr.com/photos/arselectronica/52845672454/

- Used for interactive *surfaces*
  - LCD/OLED displays
  - Projection screens

Stereoscopic screens (1)

Image source (CC): https://en.wikipedia.org/.../File:Parallax_barrier_vs_lenticular_screen.svg

- Need some way to get different pictures to left & right eye
- Autostereoscopic (below): mostly single-user



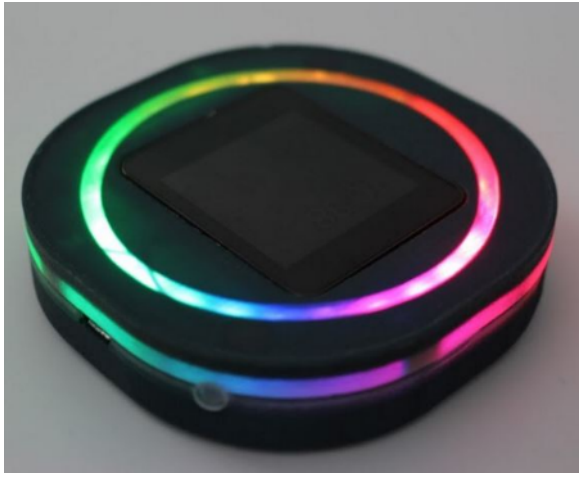
Stereoscopic screens (2)

Image source (CC): https://commons.wikimedia.org/wiki/File:ASUS_LCD_Shutter_glasses.jpg

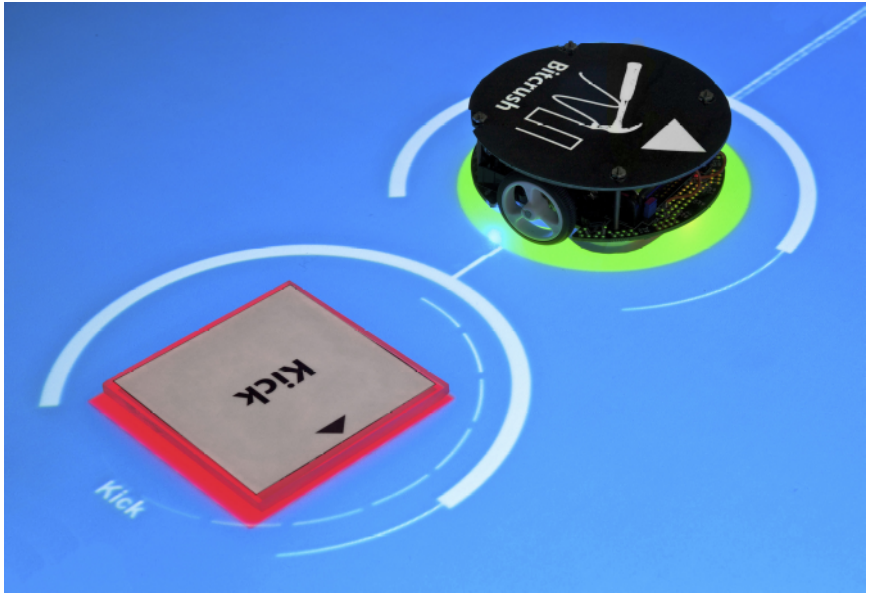
- Alternative: shutter (left) or polarizing glasses (right)
- Requires special projector and/or screen

Active Tangibles

- Tangible objects can also be output devices
- Use sound, light, motion, …



Actibles

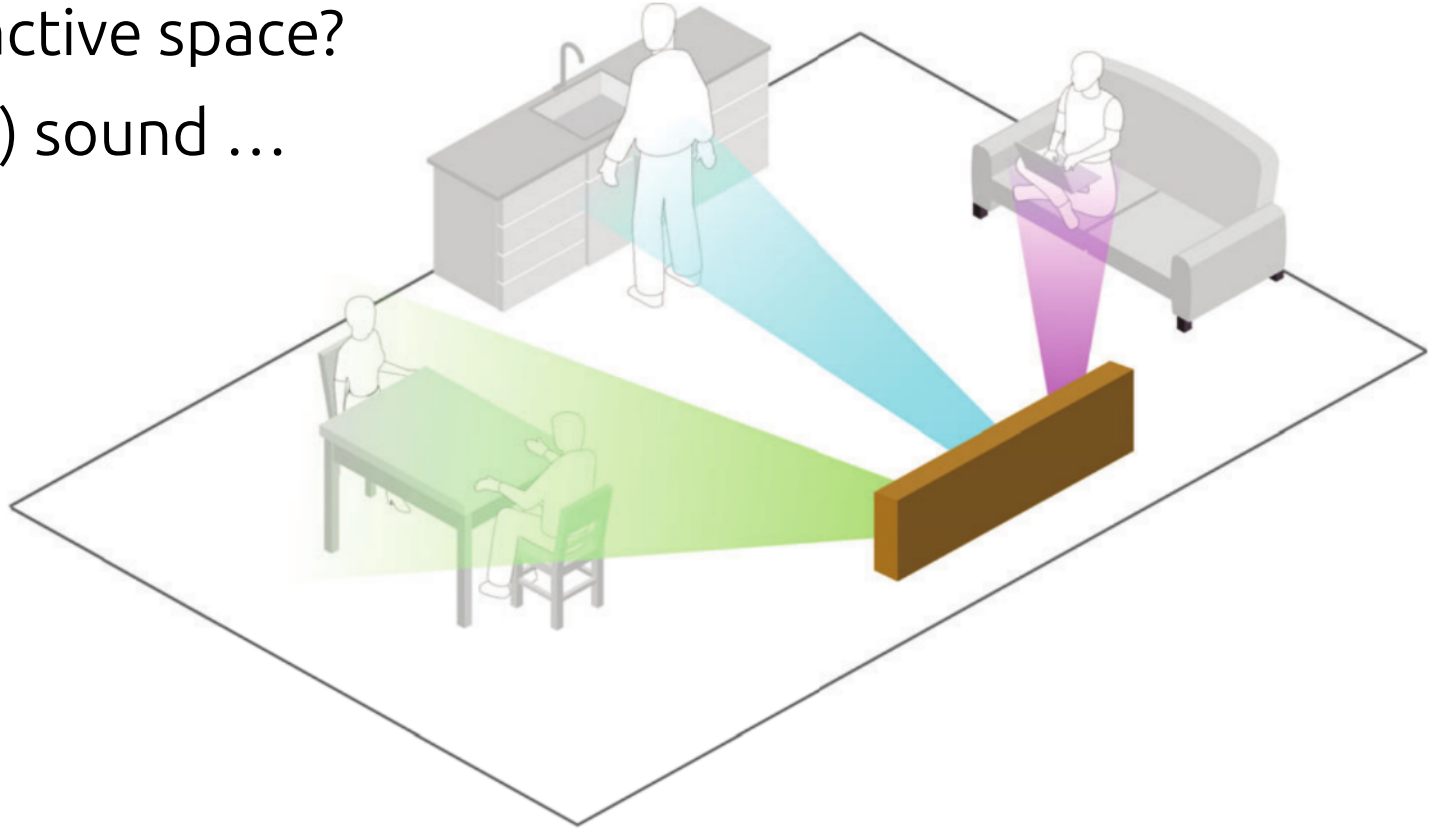


Tangible Bots

Sound zones

Image source (FU): https://vbn.aau.dk/da/.../personal-audio-through-sound-zones-interaction-and-experience/

- Is this an interactive space?
- Just (steerable) sound …

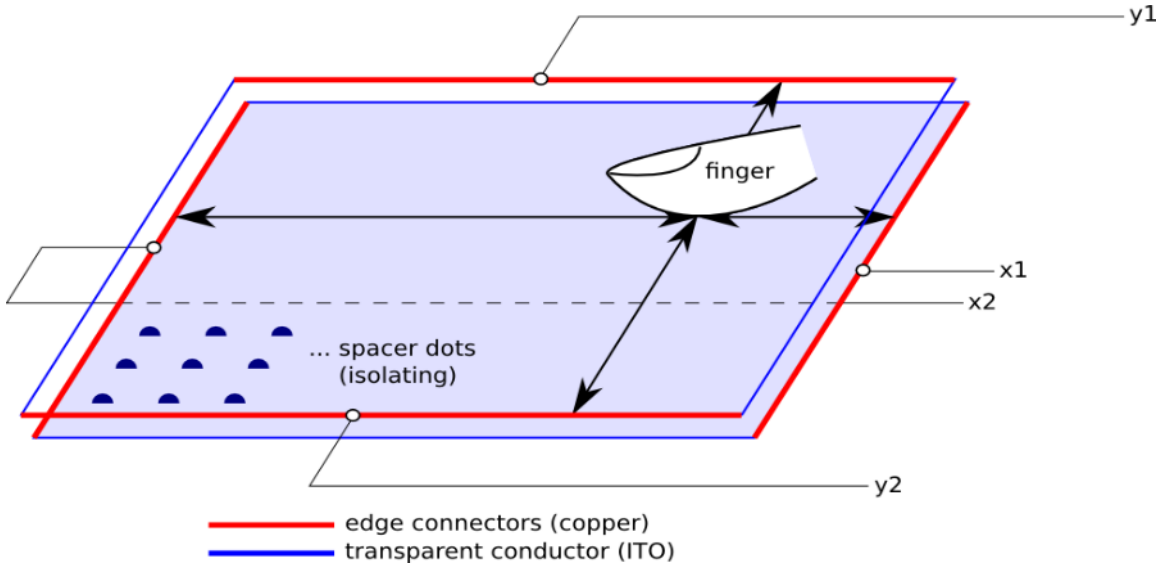


Touch & Tangible Technologies

- Resistive
- Capacitive
- Optical

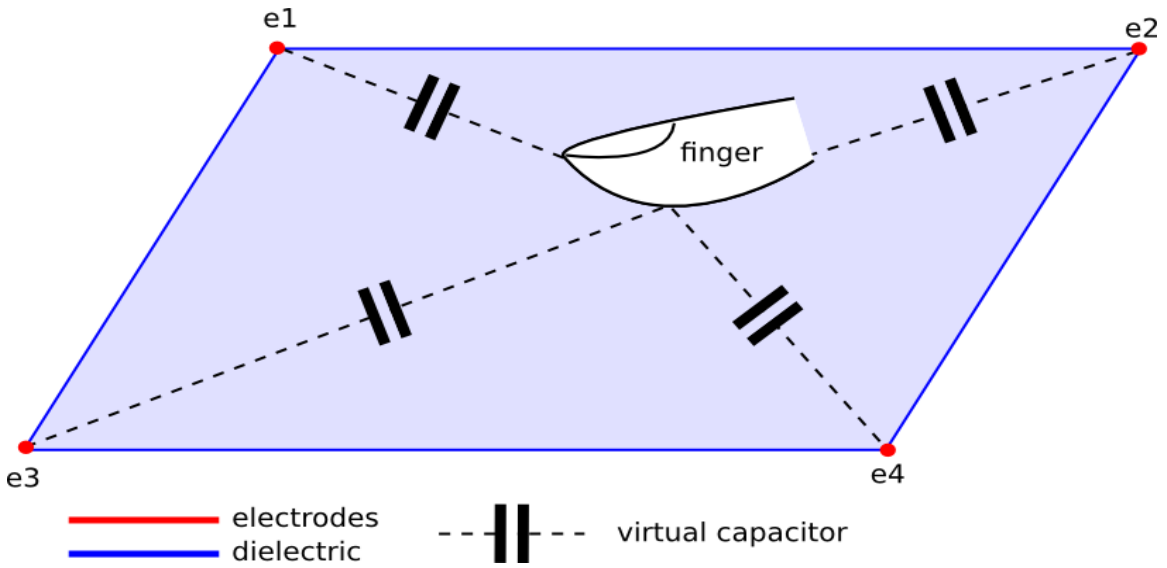
Touch – technology: resistive

- Cheap, low-end technology (no multitouch)
  - Two conductive layers separated by spacers
  - Can be used with gloves, pens, …



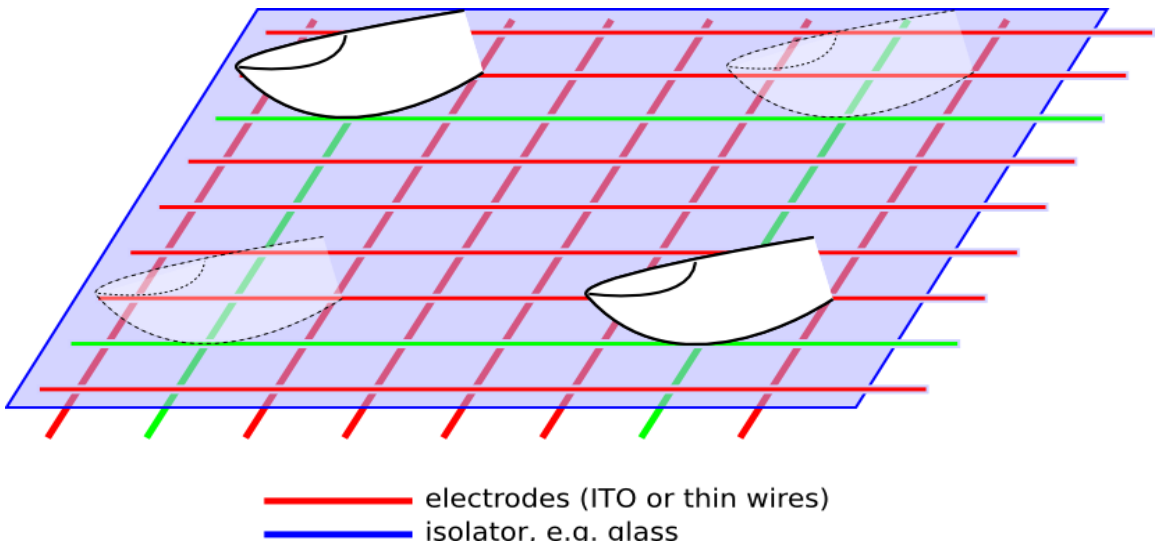
Touch – technology: capacitive (1)

- Common in POS terminals etc. (no multitouch)
- Robust, simple (but not usable with gloves)



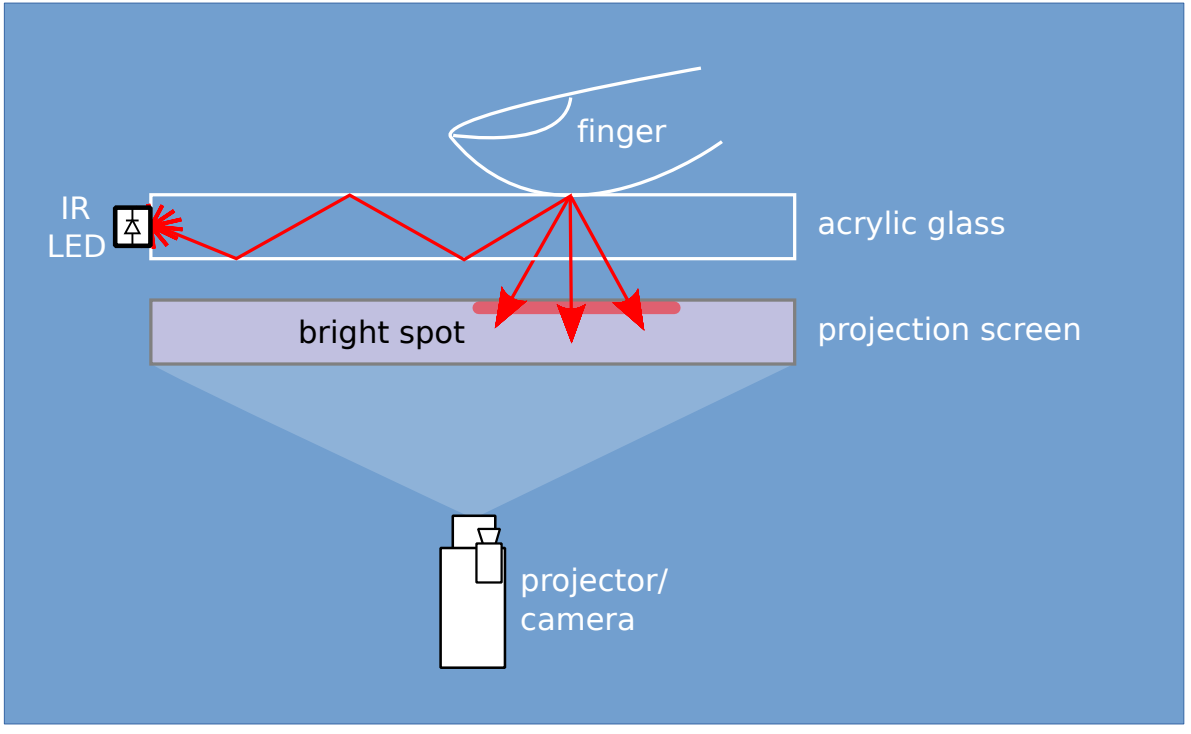
Touch – technology: capacitive (2)

- “Projected capacitive” - two variants:
  - Mutual capacitance: each row/column measured
  - Self capacitance: each crossing measured indiv.



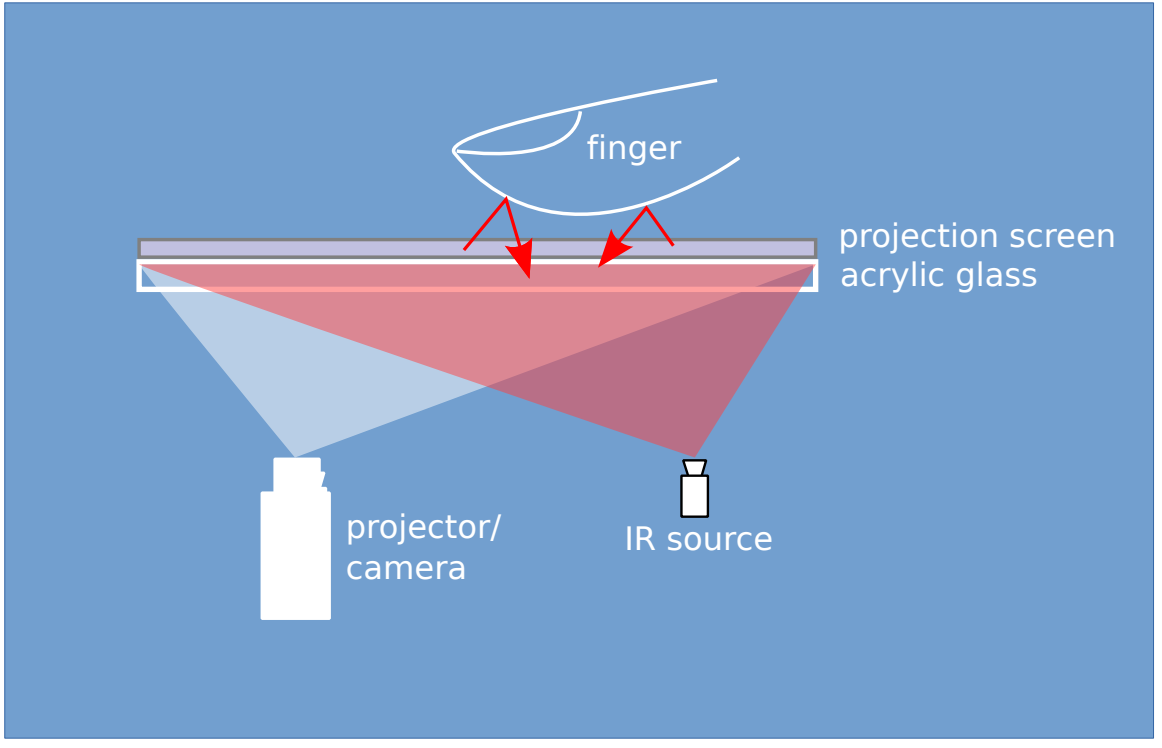
Touch – technology: optical (1)

- Frustrated Total Internal Reflection (FTIR)



Touch – technology: optical (2)

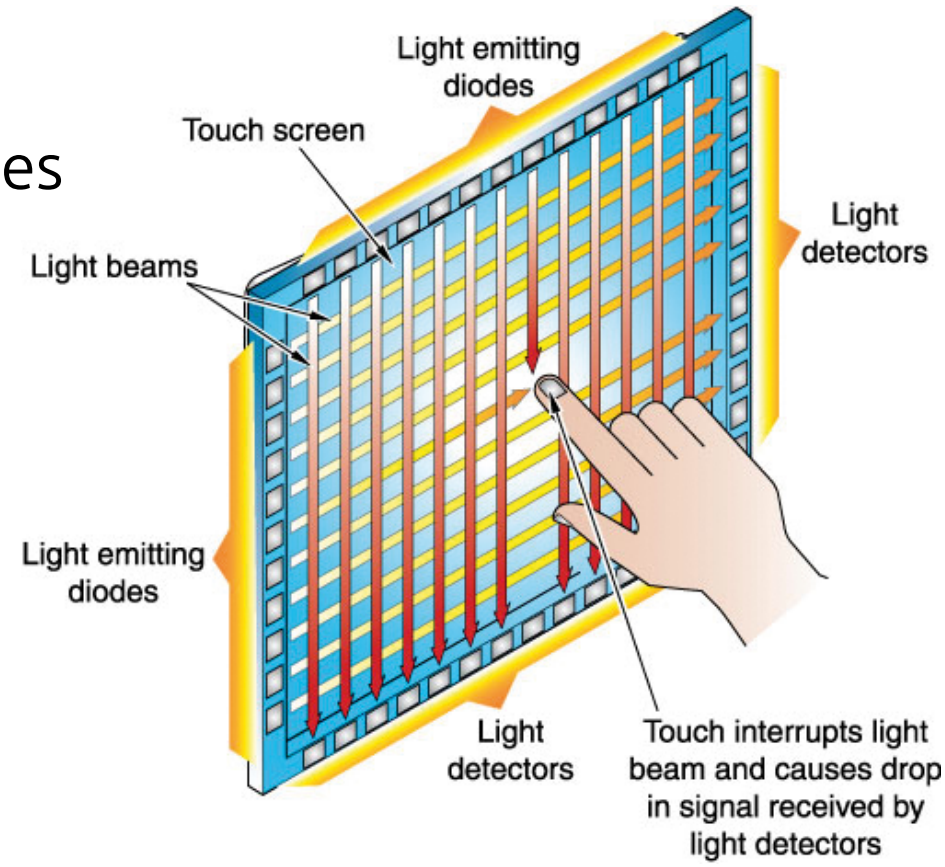
- Diffuse Illumination (DI)



Touch – technology: optical (3)

Image source (FU): https://www.canvys.com/custom-touch-screen-monitors/touch-screen-infrared-touch/

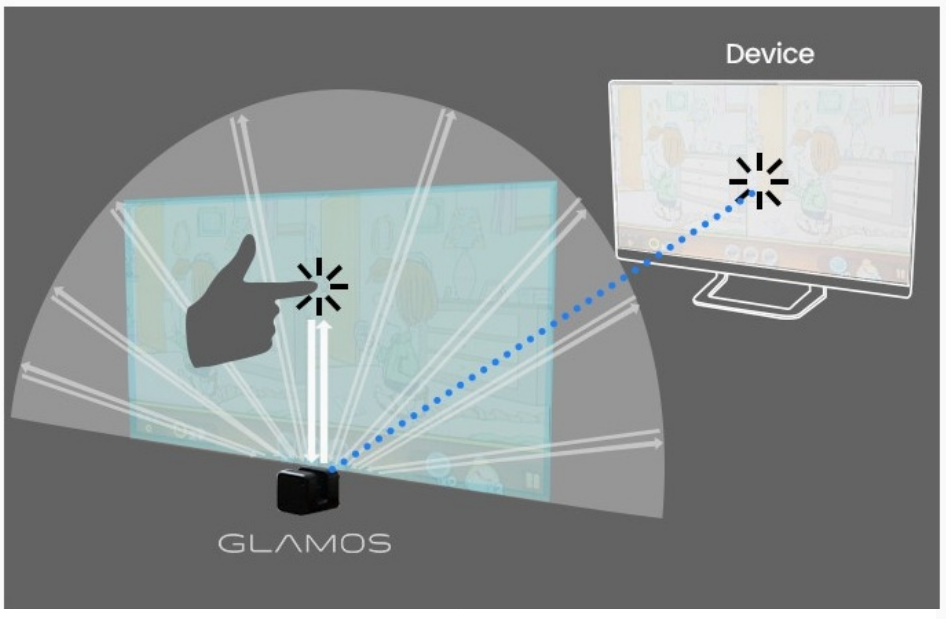
- Grid-based variants:
- IR grid + sensors on opposite sides

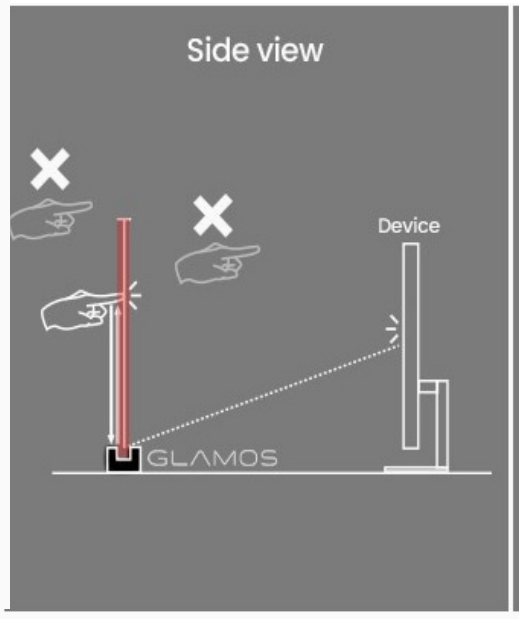


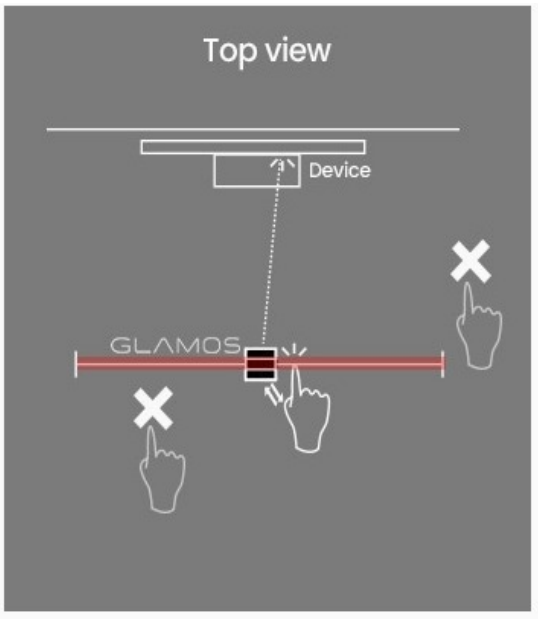
Touch – technology: optical (4)

Image source (FU): https://www.yankodesign.com/2020/01/28/this-tiny-lidar-sensor-...

- Laser rangefinder(s)
- Suitable for very large screens

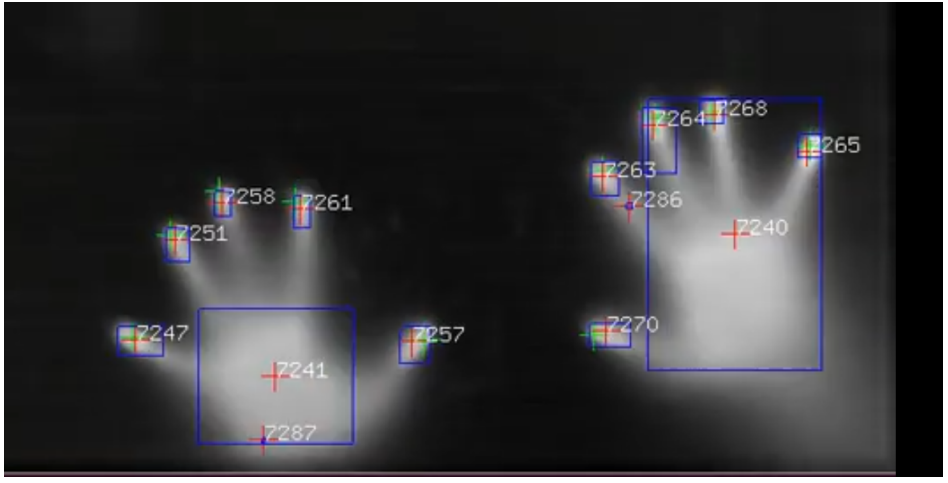






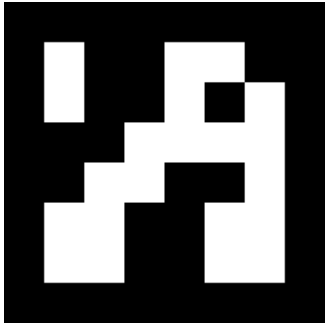
Touch – technology: optical (5)

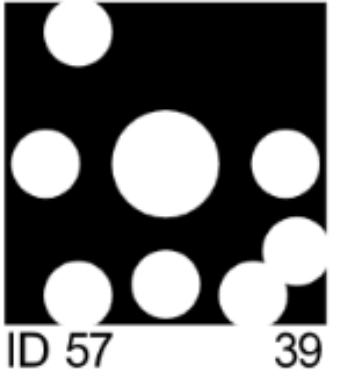
- In-cell sensing - custom LCD screen with light-sensitive pixels
- Only one device: Samsung SUR40
- Pro: can detect hands, fingers, tokens, …
- Con: sensitive to stray light



Tangible technologies: optical (1)

- Passive tokens: ReacTIVision, AruCo, ByteTag, …
- Camera-based system required

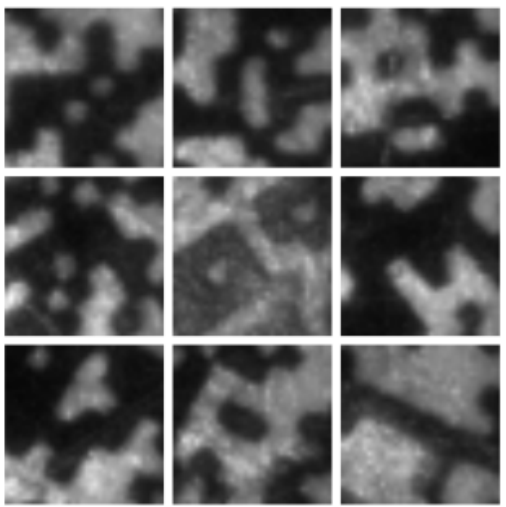




Tangible technologies: optical (2)

Image source (CC): https://dl.acm.org/doi/abs/10.1145/3532719.3543259

- „Inside-out“ tracking with active tangibles
- Needs no camera, but special surface pattern



Pen/stylus input

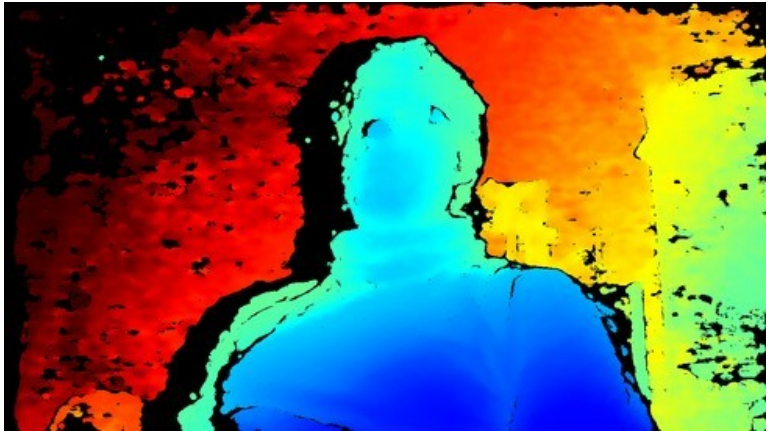
- Most touch technologies can detect pens
  - Simulate „fingertip“
- Special cases:
  - Apple Pencil, Wacom pens:

    capacitive or wireless data channel to screen for angle, pressure, …
  - Anoto pens (discontinued), Tiptoi:

    camera in tip, scans invisible pattern on paper

Depth Cameras

- „Regular“ camera: color values for each pixel
- Depth camera: distance values for each pixel (usually visualized with color map)



Geometry-based DCs

- Fundamental principle:
  - Create two „views“ of scene
  - Match scene points between views
  - Determine 2 angles for each scene point
  - Trigonometry happens
  - Receive distance

Geometry: Stereo Matching

- Method 1: two images of scene

  → *stereo matching* of corresponding pixels

  → ideally only needed on horizontal *scanline*

  - Examples: Occipital Structure Sensor, Intel Realsense D4xx

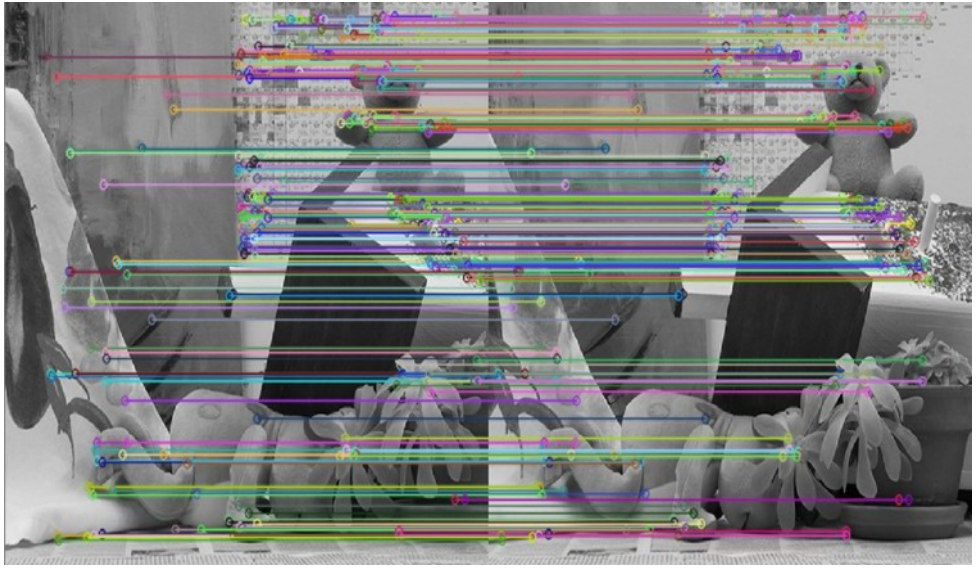


Image Source (FU): https://www.mtbs3d.com/phpbb/viewtopic.php?f=138&t=18055

Geometry: Speckle Pattern

- Method 2.1: *speckle* pattern = random dots
  - Random, but previously known pattern
  - „Patches“ of pattern can be matched
    → depth res. lower than image res.
  - Example: Kinect v1

Geometry: Stripe Pattern

- Method 2.2: Gray code = alternating stripes
  - Encodes binary ID for each pixel
  - Requires high frame rate or static scene (why?)
  - Requires IR projector
  - Example: Realsense

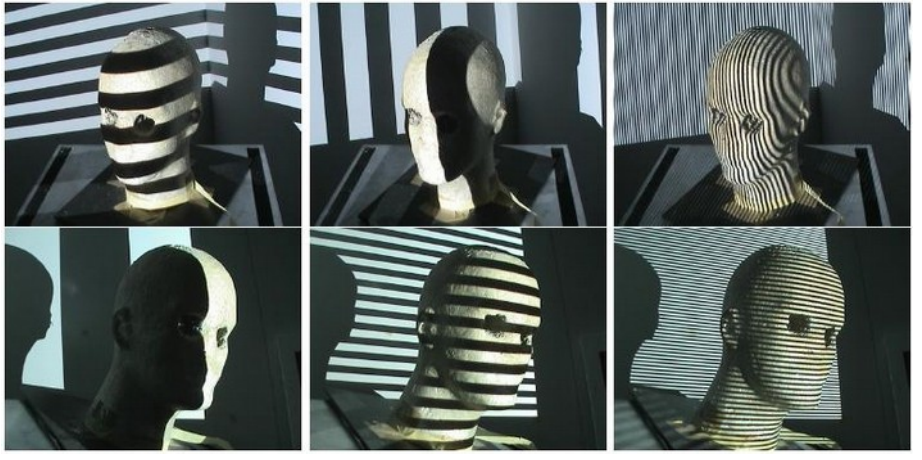


Image Source (FU): http://www.sci.utah.edu/~gerig/.../CS6320-CV-S2012-StructuredLight-II.pdf

Time-of-Flight/ToF DCs

- Fundamental principle:
  - Emit (infrared) flash
  - Measure time until reflected light arrives
  - (very simple) Math happens
  - Receive distance
- Example: Kinect v2

Time-of-Flight DCs

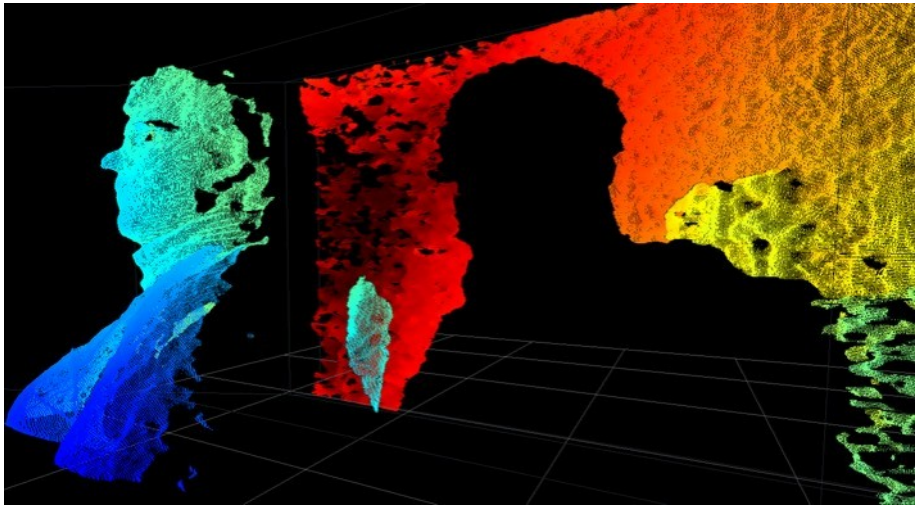
- Solution: measure *phase* difference (not time)
  - Create clock signal with freq. f which modulates …
    a) IR emitter („flash“)
    b) sensitivity of light sensor
  - Result is phase difference
  - Calculation with c and f gives distance

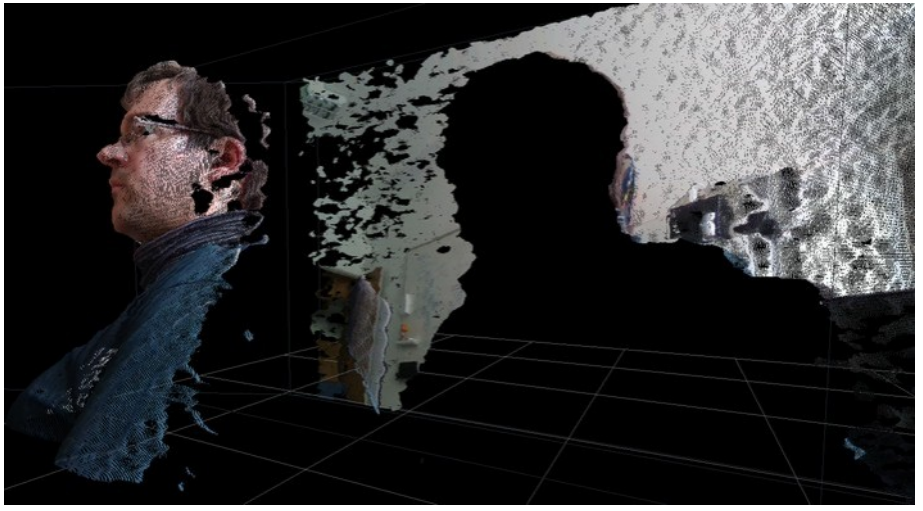
Depth-Color Alignment

- most depth cameras also have a “plain” color cam
- Problem: how to find corresponding color value for each depth pixel (or vice versa)?
- Requires intrinsic and extrinsic camera parameters (?)
  - Intrinsic: field of view, distortion, focal length, …
  - Extrinsic: translation, rotation w.r.t. origin

Point Clouds

- Each depth pixel → 3D point (x, y, z)
- Plus color → (x, y, z, r, g, b)

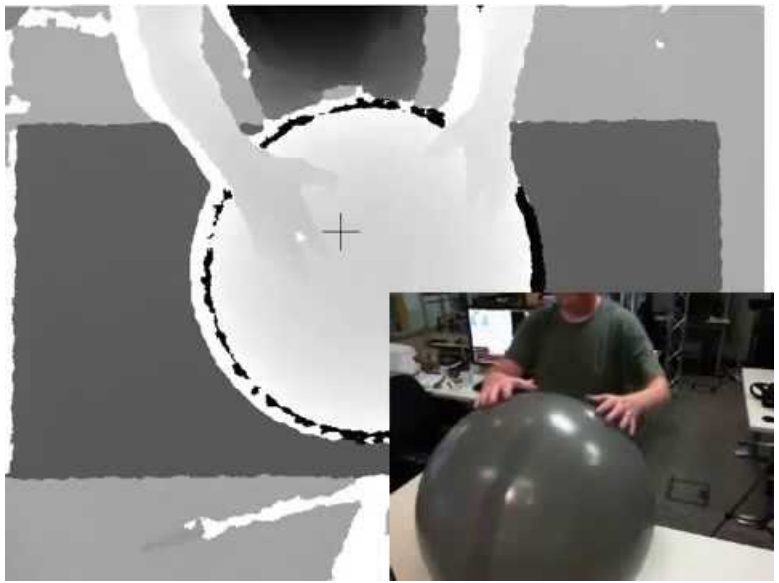




Depth cameras for ISS?

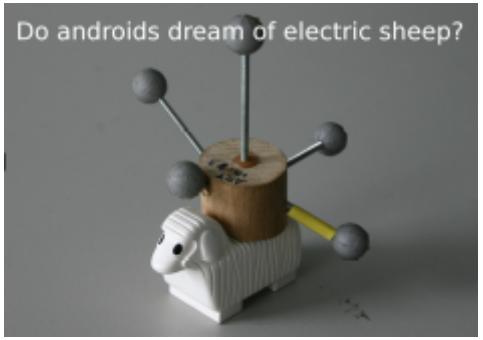
Image source (FU): https://www.youtube.com/watch?v=w6RtfrczmYQ

- Use as „filter“: ignore the surface itself, only detect objects
- No planar surface necessary



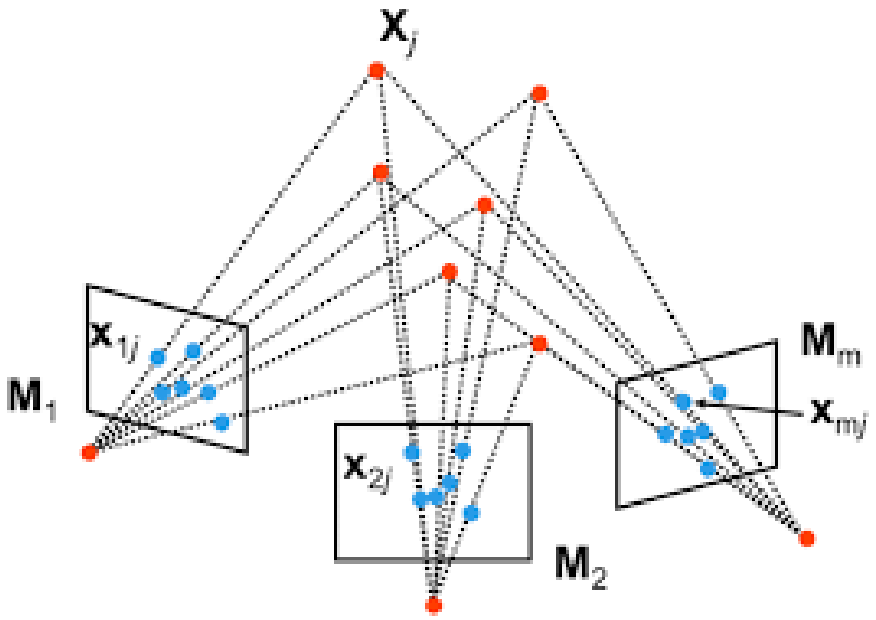
6DoF Tracking (1)

- Uses 2 or more cameras, asymmetric targets
- Reflectors (Vicon, ART) or LEDs (Meta, HTC)
- Note: HTC Vive has sensors on tracker, light sources in room



6DoF Tracking (2)

- Intersection of rays from >= 2 cameras → 3D points
- Unique point distances → target ID & orientation



The End