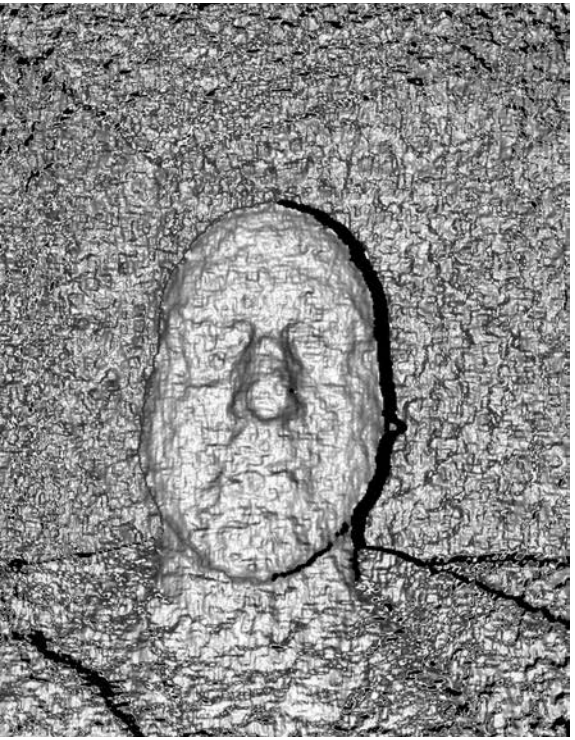
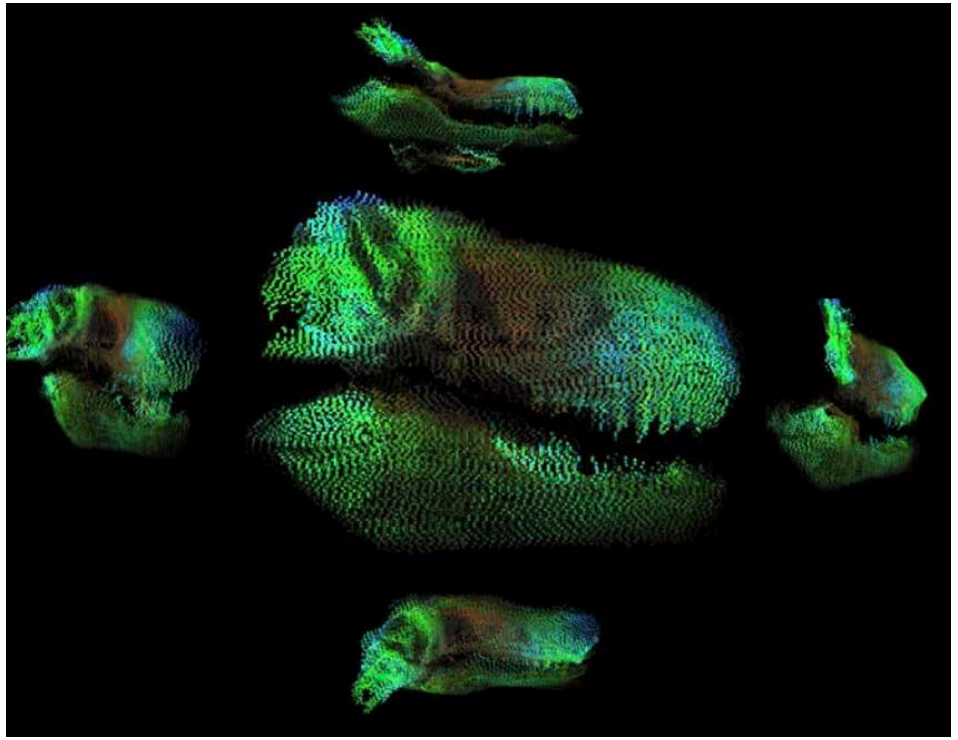
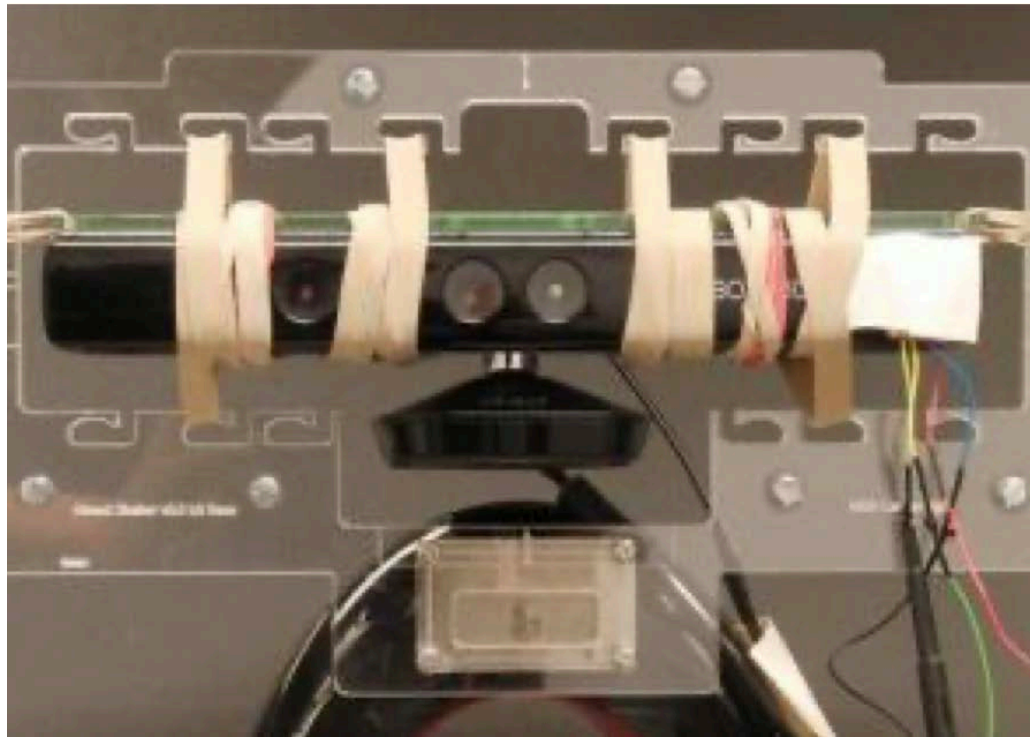
Visual Computing I:

Interactive Computer Graphics and Vision







Practical Depth Sensing and Working with Range Data

Stefanie Zollmann and Tobias Langlotz

Last time..

The Fundamental Matrix

Notes:

$$p_1 = K_1[I\,|\,0]x$$
$$p_2 = K_2[R\,|\,t]x$$

- One way to express $\mathbf{F}$ is

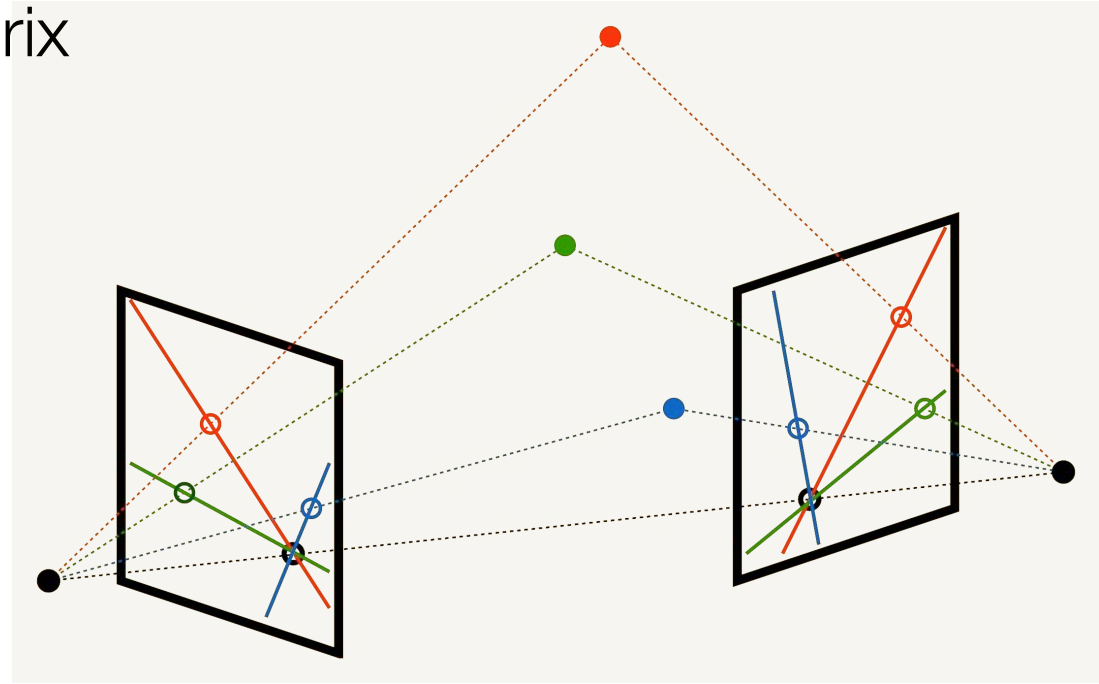
$$\mathbf{F} = \mathbf{K}_2^{-T}[\mathbf{t}]_\times \mathbf{R}\mathbf{K}_1^{-1}$$

- Here
  - $\mathbf{K}_1$ and $\mathbf{K}_2$ are the calibration matrices of the two cameras
  - $\mathbf{R}$ and $\mathbf{t}$ are the rotation and translation between them
  - $[\mathbf{t}]_\times$ is the matrix form of the cross product with $\mathbf{t}$:

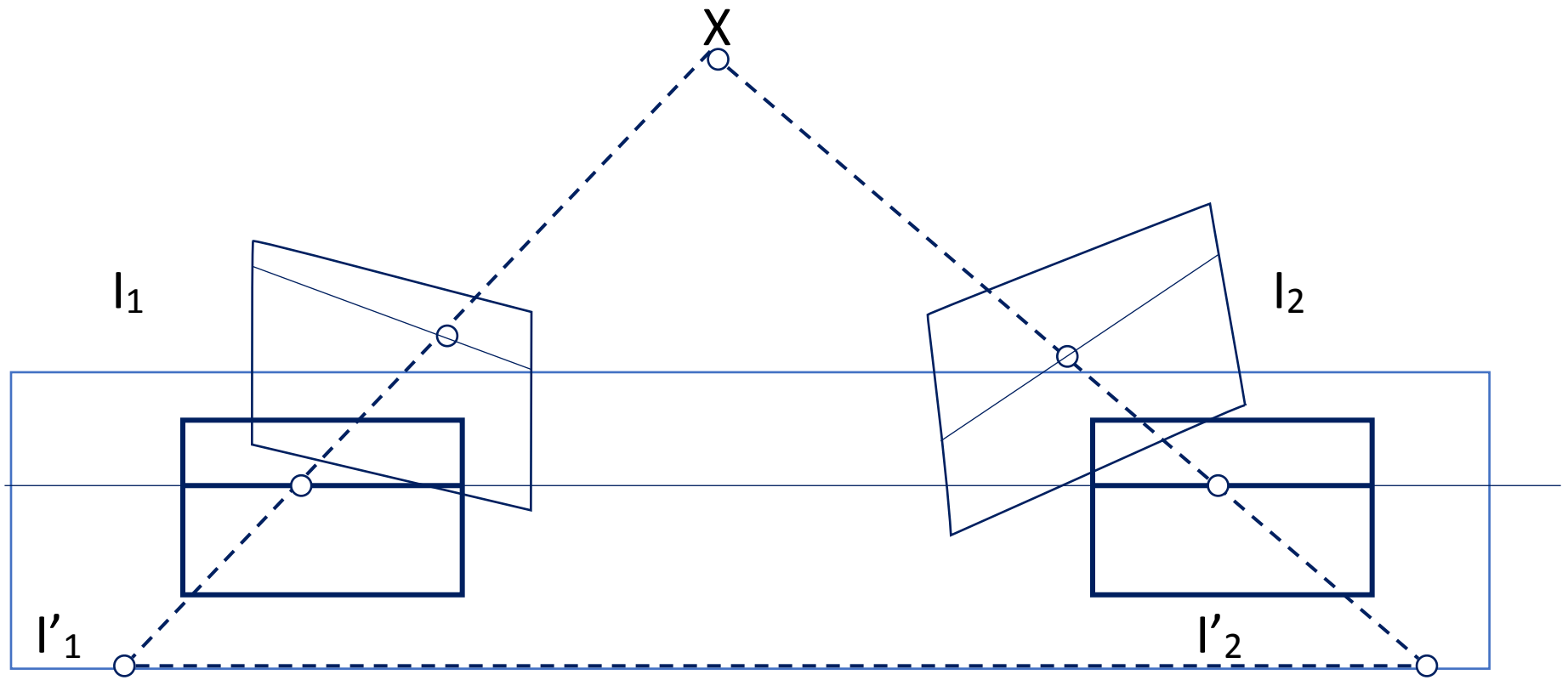
$$[t]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}$$

The Essential Matrix

- For calibrated cameras
  - We can factor out $\mathbf{K}_1$ and $\mathbf{K}_2$ in $\mathbf{F} = \mathbf{K}_2^{-T}[\mathbf{t}]_\times \mathbf{R}\mathbf{K}_1^{-1}$
    - This gives us the essential matrix
      $$\mathbf{E} = \mathbf{K}_2^T \mathbf{F} \mathbf{K}_1 = [\mathbf{t}]_\times \mathbf{R}$$
  - $\mathbf{E}$ has only five DoF:
    - 3 for 3D rotation
    - 3 for 3D translation
    - Less 1 for unknown scale



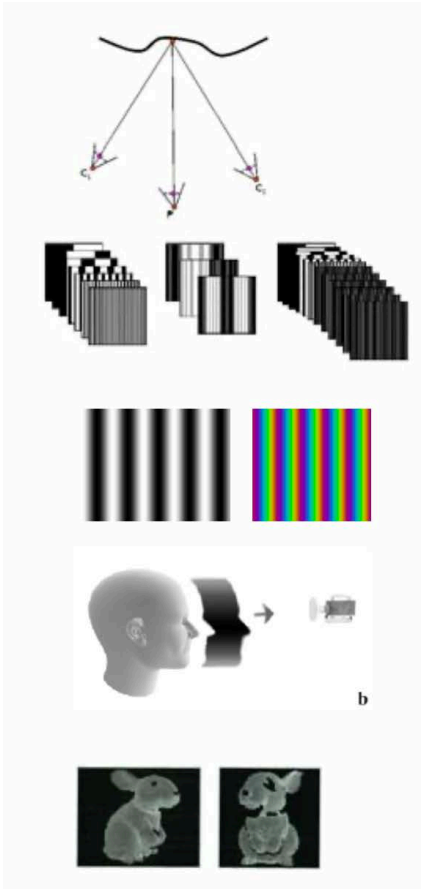
Stereo Rectification

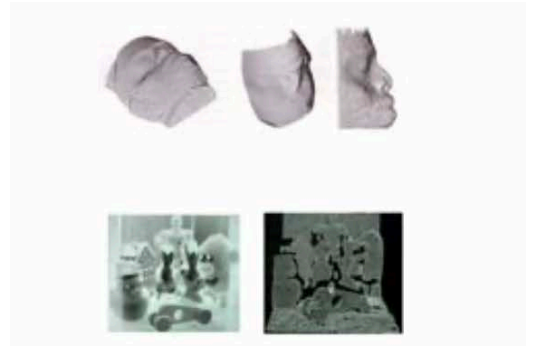


Today:
Practical Depth Sensing and Range Data

Overview

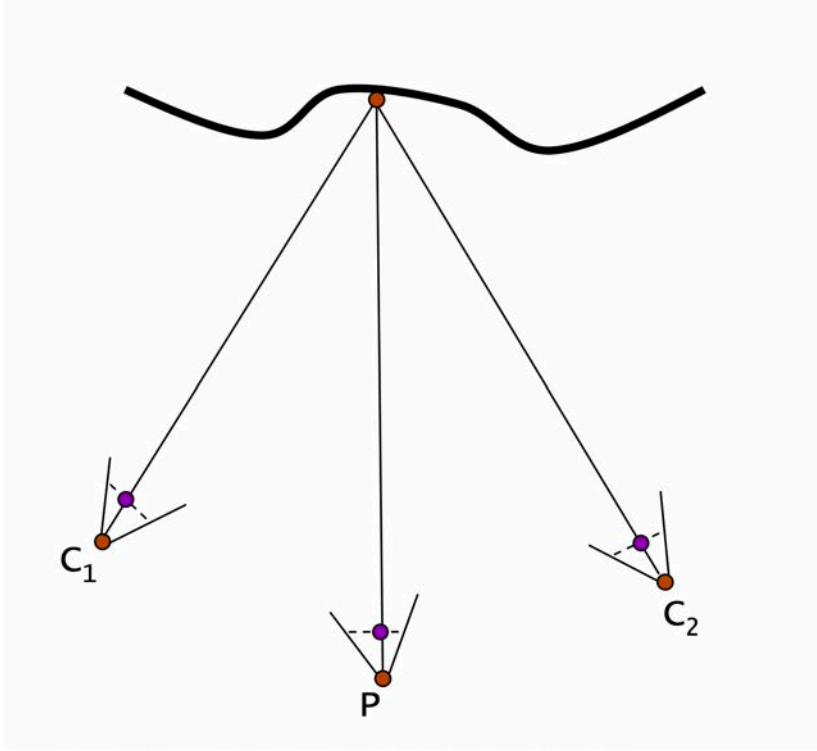
- Range Scanning
  - Passive:
    - Stereo or Multi-Camera
    - Active:
    - Structured Light Projection
    - Optical Time-of-Flight
      - Direct ToF
      - Indirect ToF
- Registration of Depth Maps
- Processing Depth Maps

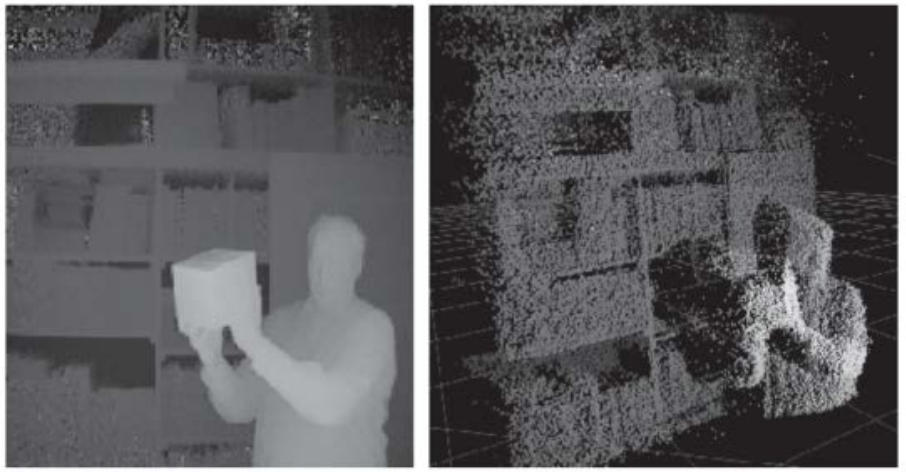




Range Scanning

- Range sensors:
  - Passive: >=2 cameras (stereovision)
  - Active: >=1 cameras + >=1 projectors (structured light)
- Result is depth information given as:
  - Range image / depth map store depth at pixel positions from a specific view (RGBD image)
  - A point cloud is a 3D dataset of discrete (X,Y,Z) surface points



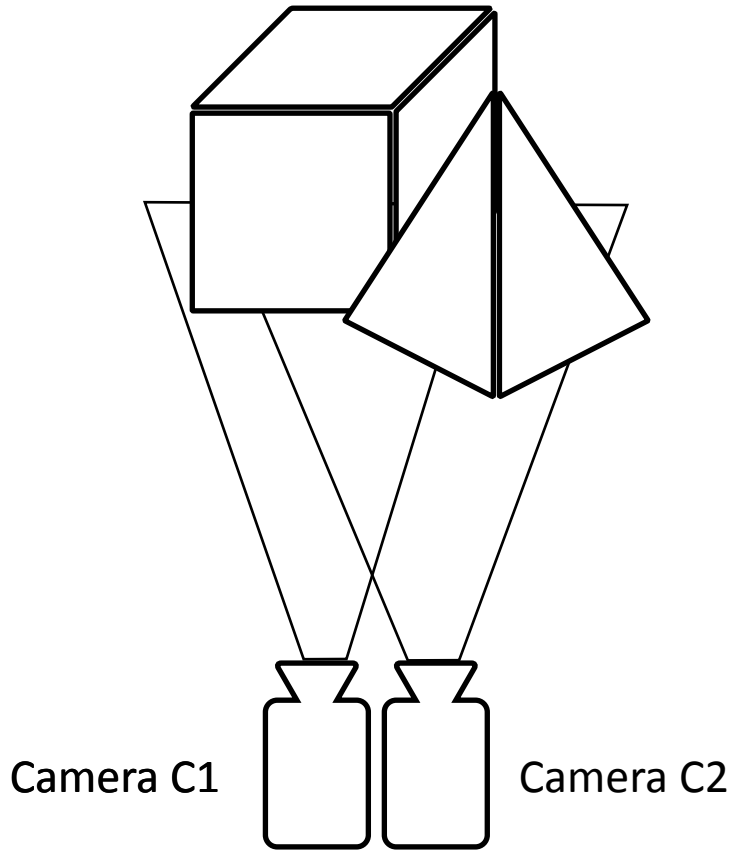


Depth Map Point cloud

Passive Range Scanning

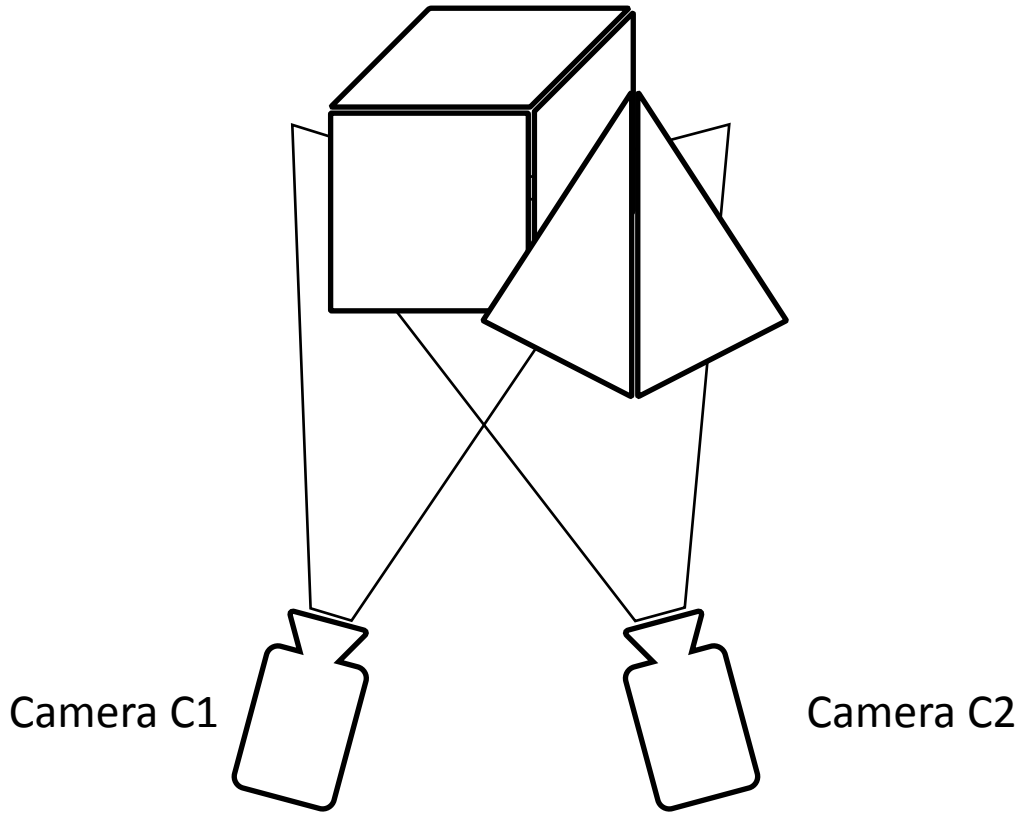
Passive Range Scanning

- Two or more cameras
  - Stereo camera
  - Two cameras
  - One camera takes several photos from different perspectives
  - Multiple cameras
- Which one is the easiest and why?



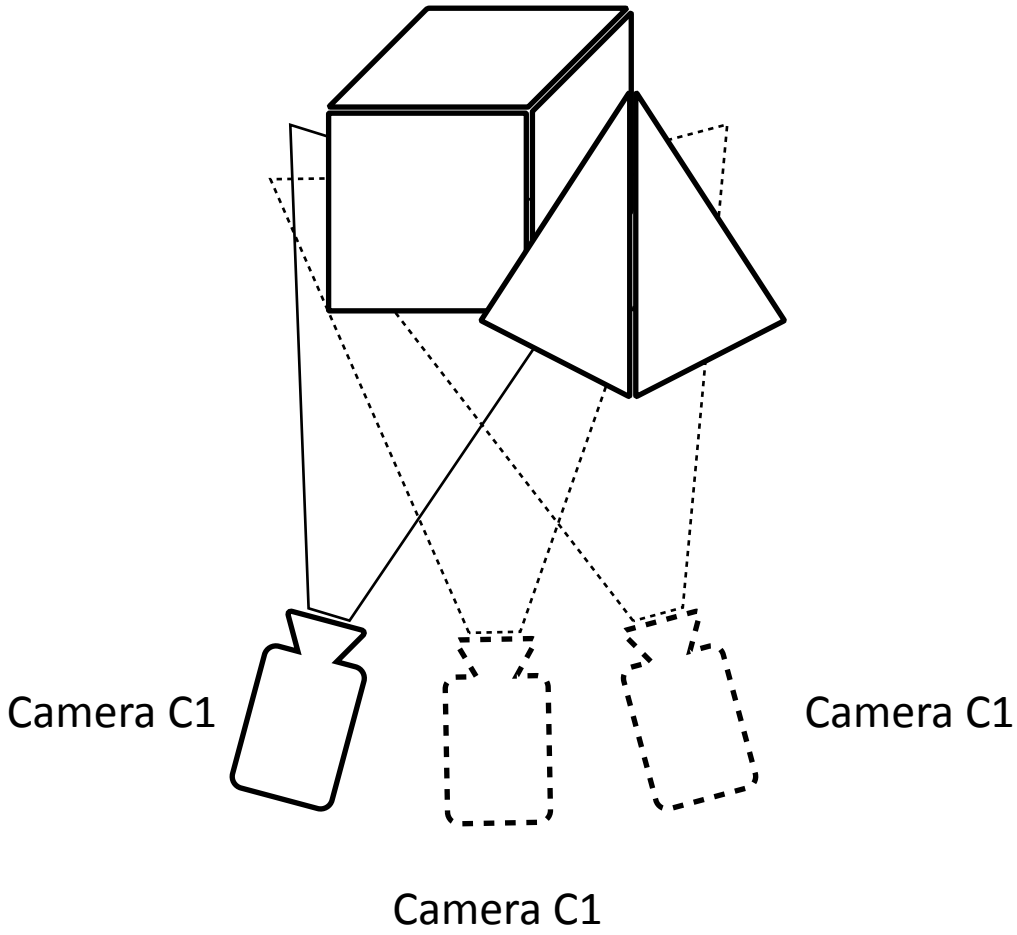
Passive Range Scanning

- Two or more cameras
  - Stereo camera
  - Two cameras
  - One camera takes several photos from different perspectives
  - Multiple cameras
- Which one is the easiest and why?



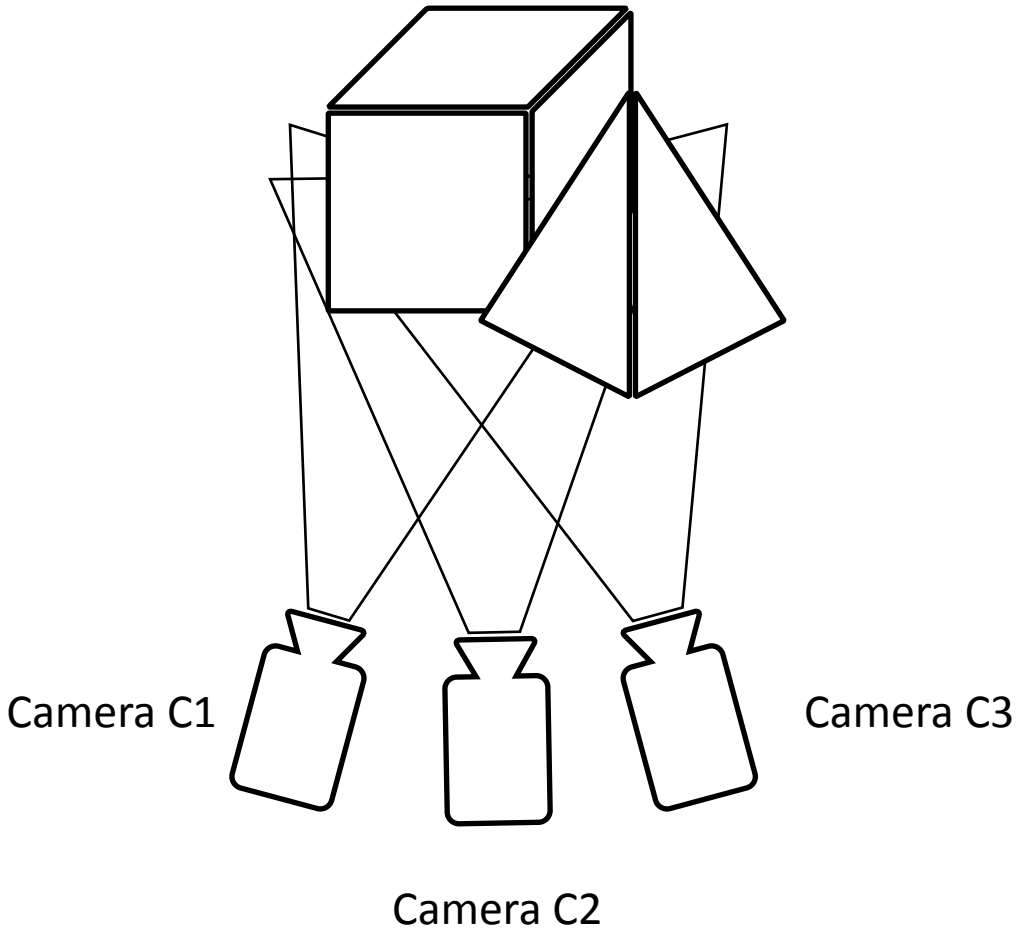
Passive Range Scanning

- Two or more cameras
  - Stereo camera
  - Two cameras
  - One camera takes several photos from different perspectives
  - Multiple cameras
- Which one is the easiest and why?



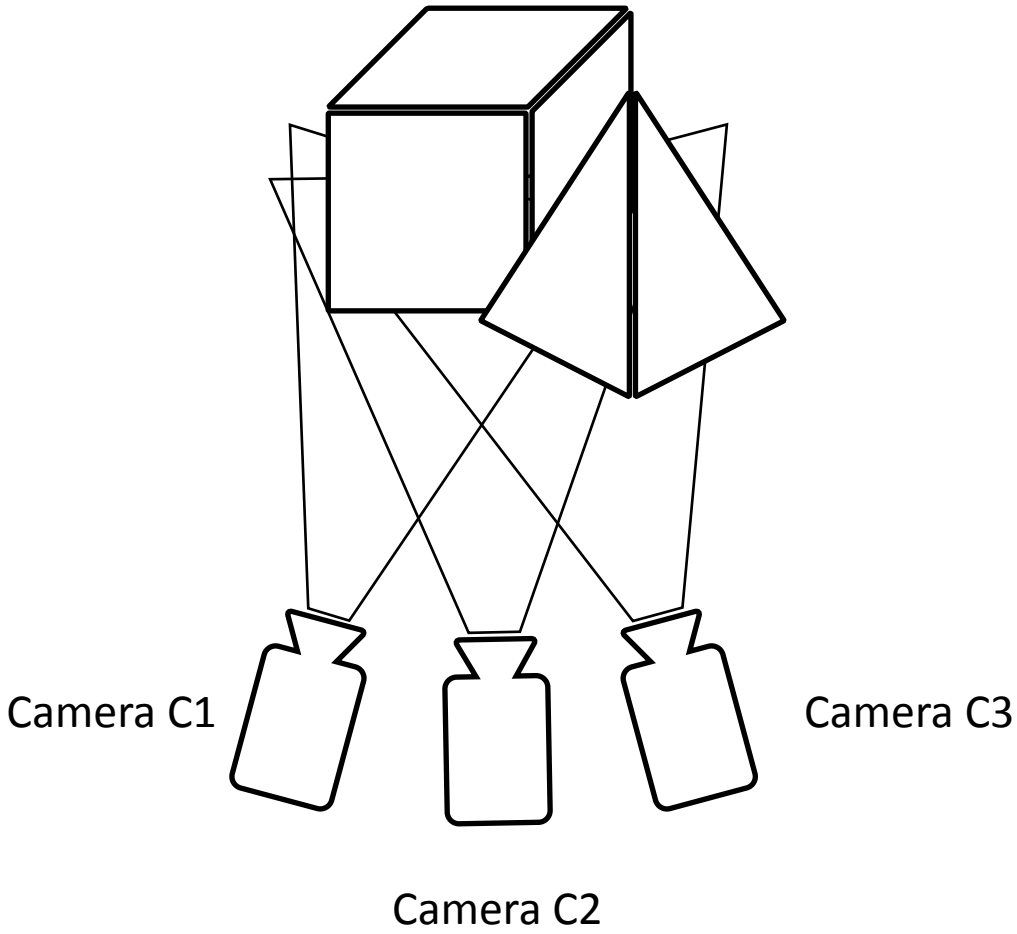
Passive Range Scanning

- Two or more cameras
  - Stereo camera
  - Two cameras
  - One camera takes several photos from different perspectives
  - Multiple cameras
- Which one is the easiest and why?



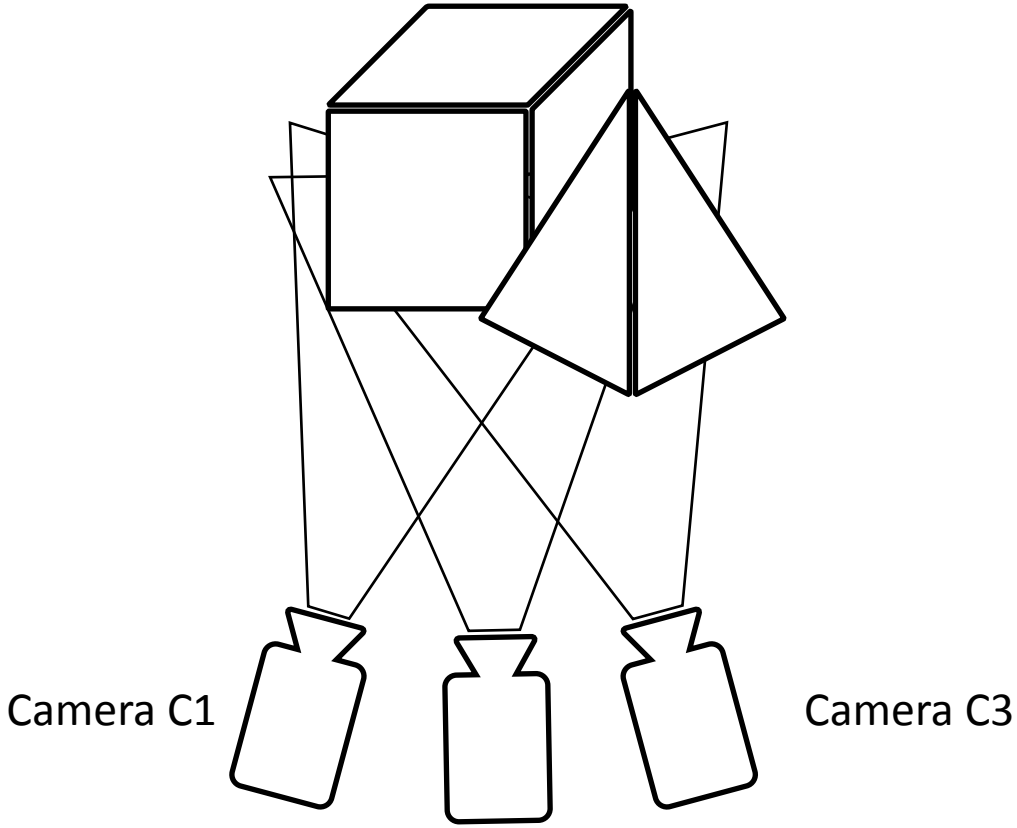
Passive Range Scanning

- Two or more cameras
  - Stereo camera
  - Two cameras
  - One camera takes several photos from different perspectives
  - Multiple cameras
- Which one is the easiest and why?
- What is the main challenge?



Passive Range Scanning

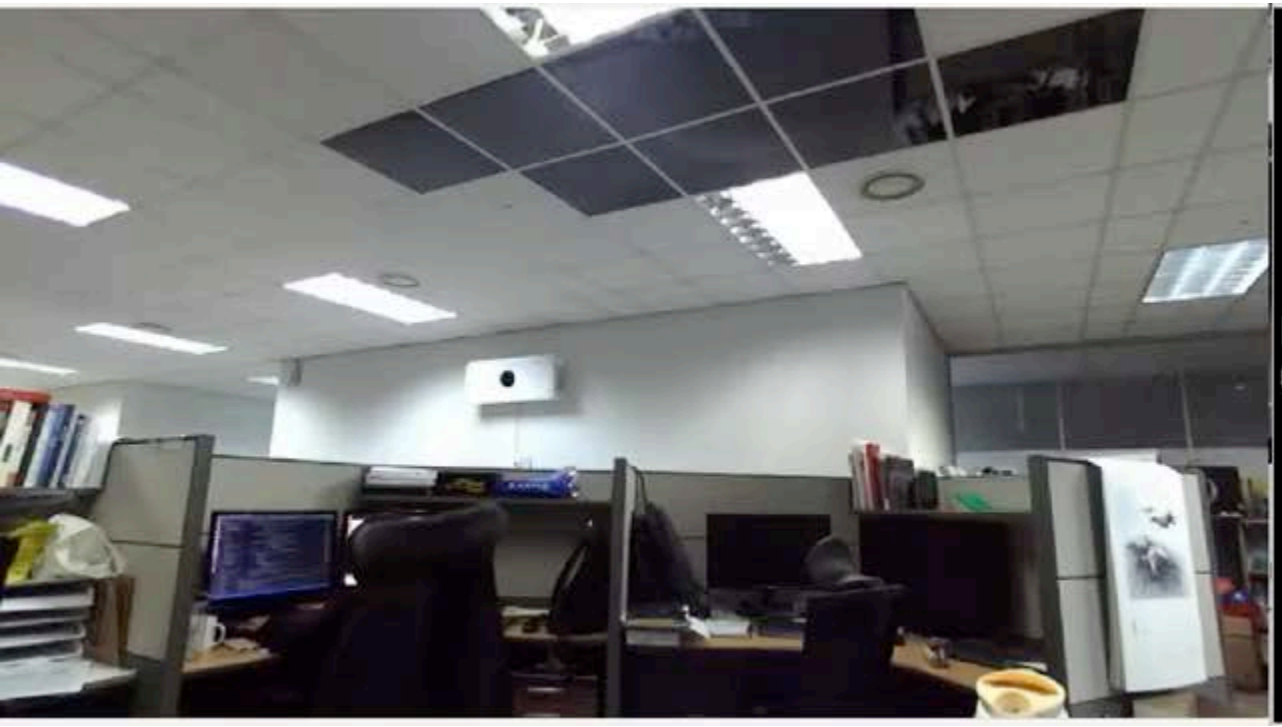
- Two or more cameras
  - Stereo camera
  - Two cameras
  - One camera takes several photos from different perspectives
  - Multiple cameras
- Which one is the easiest and why?
- What is the main challenge?

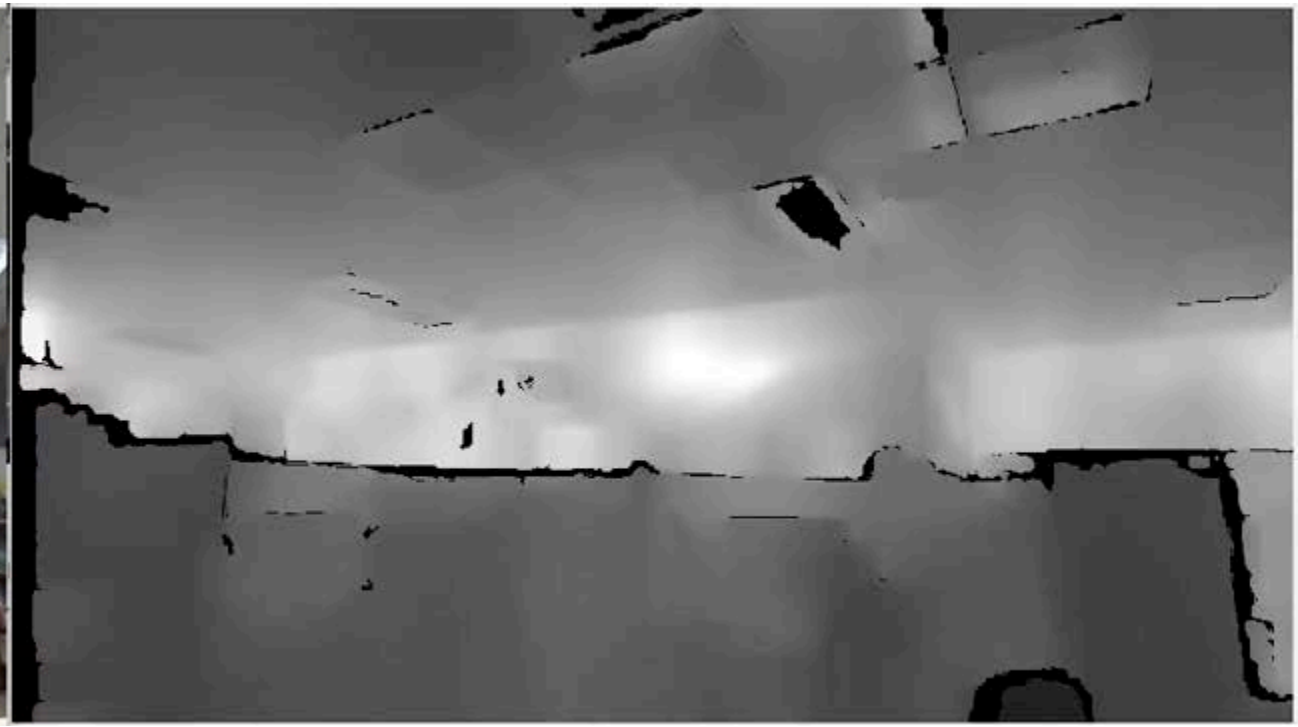


PointGrey Bumble Bee

Zed Mini

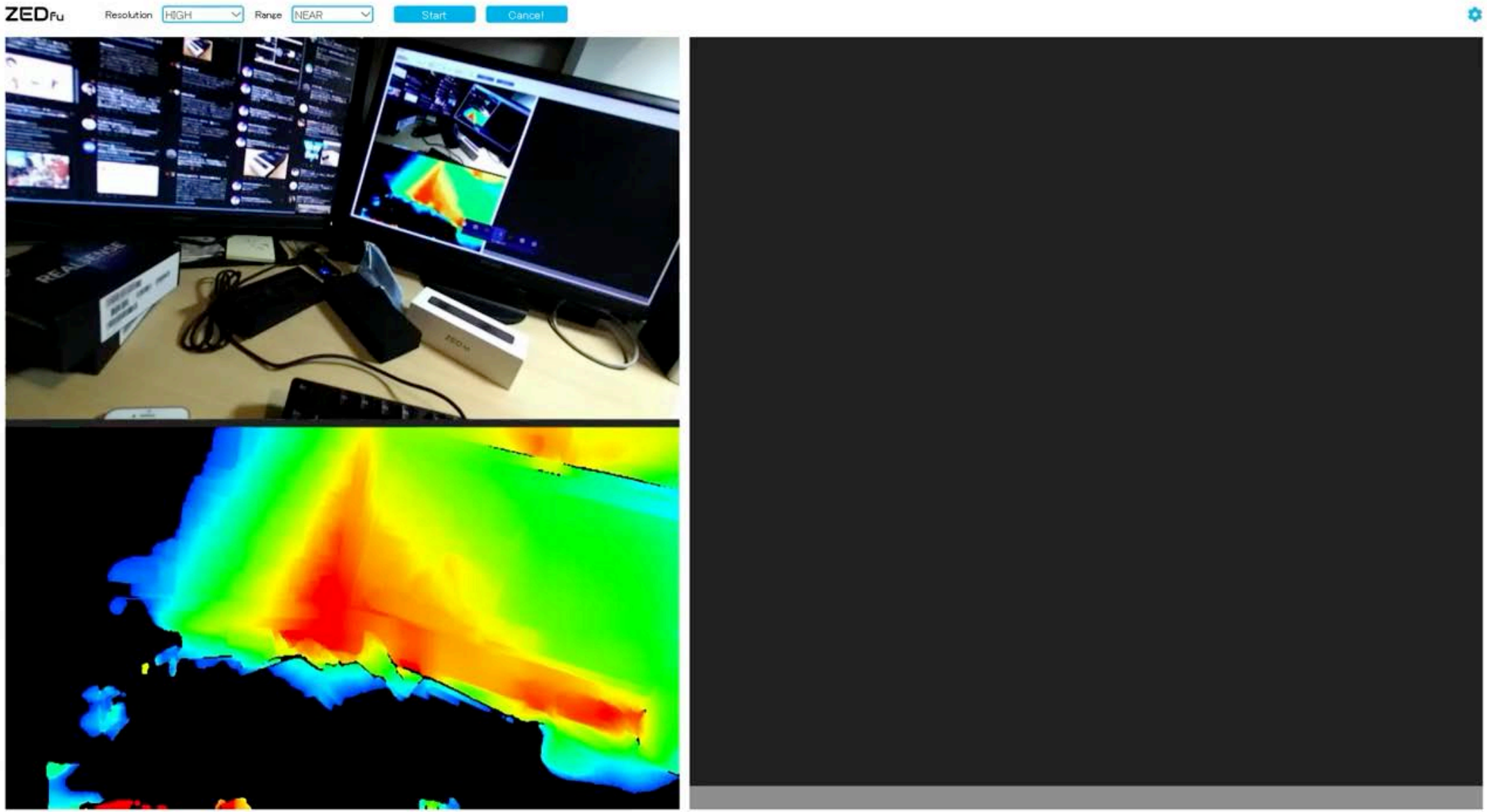
Passive range scanning





Youtube Taeyeon Kim

Passive range scanning

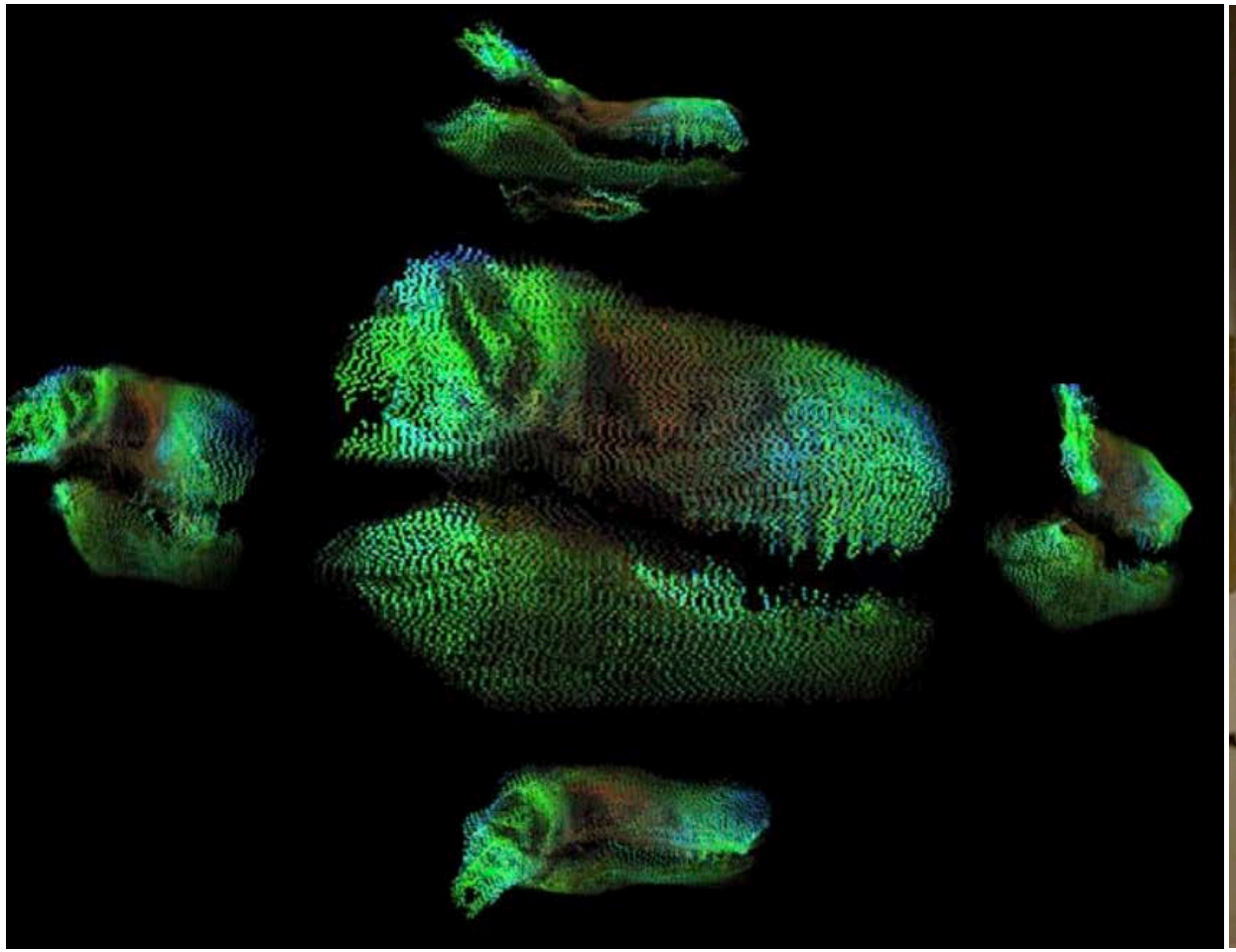


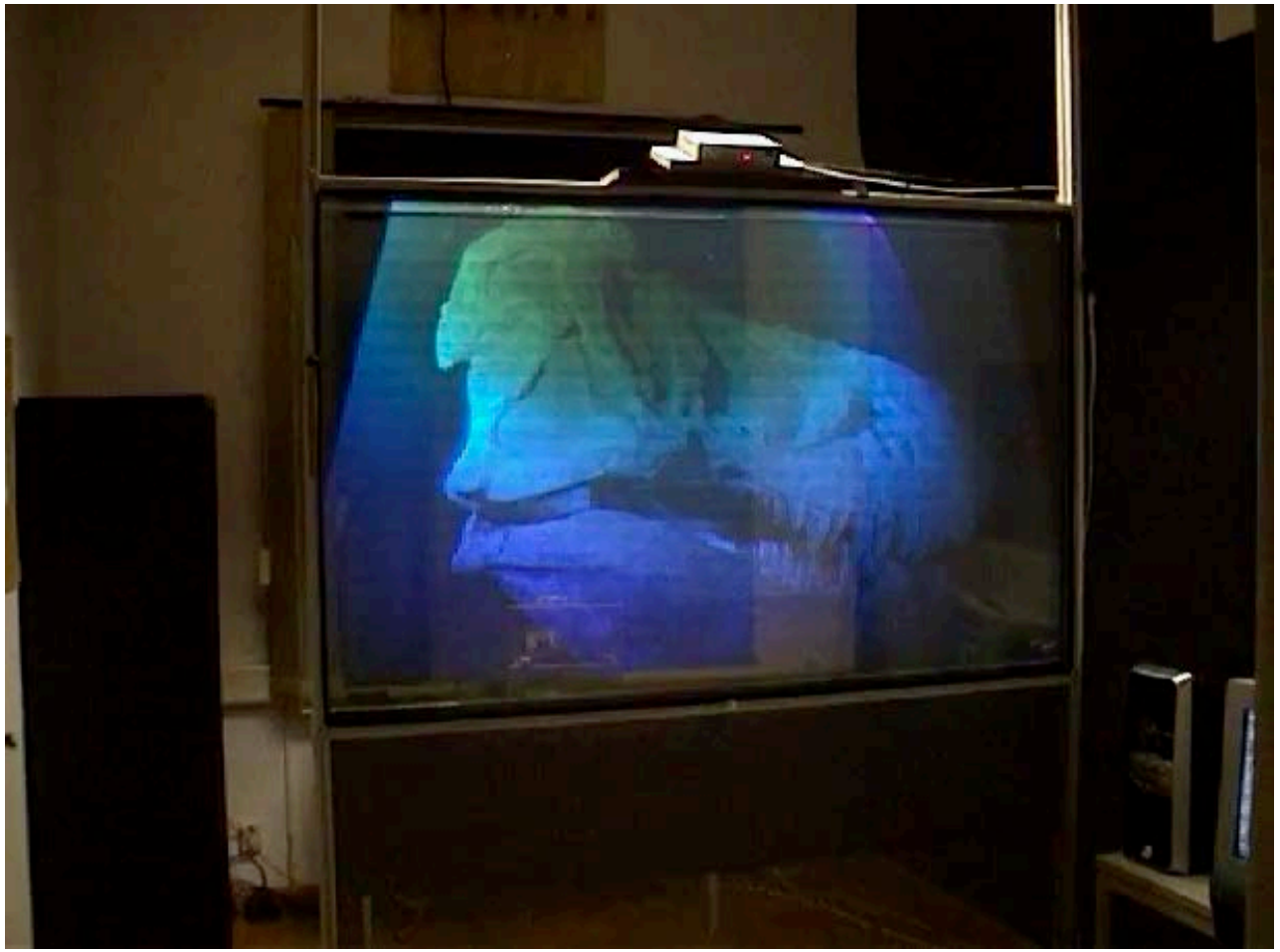
Youtube Atsizumi

Passive range scanning

Vexcel (YouTube Video: Skyline Software Systems)

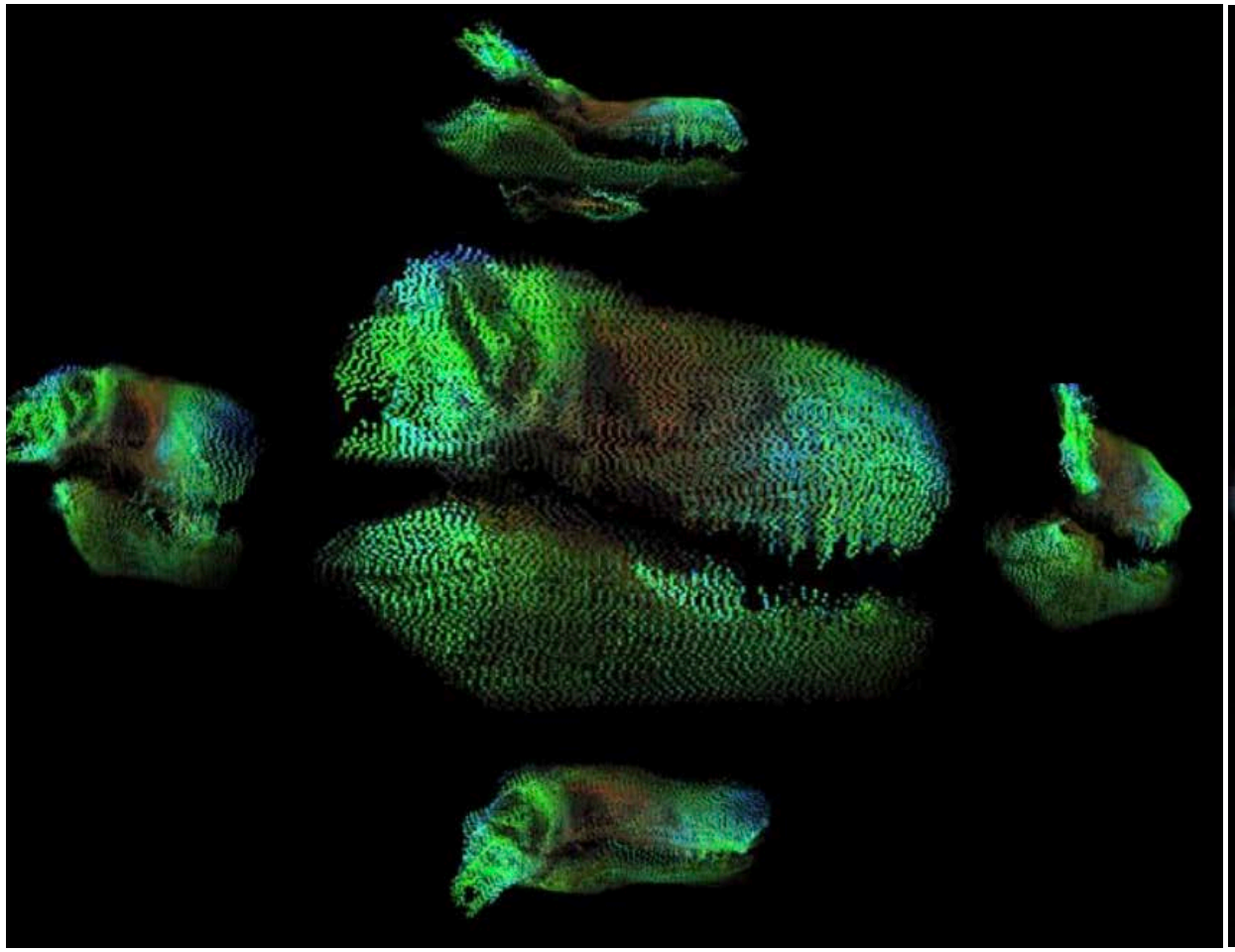
Passive range scanning

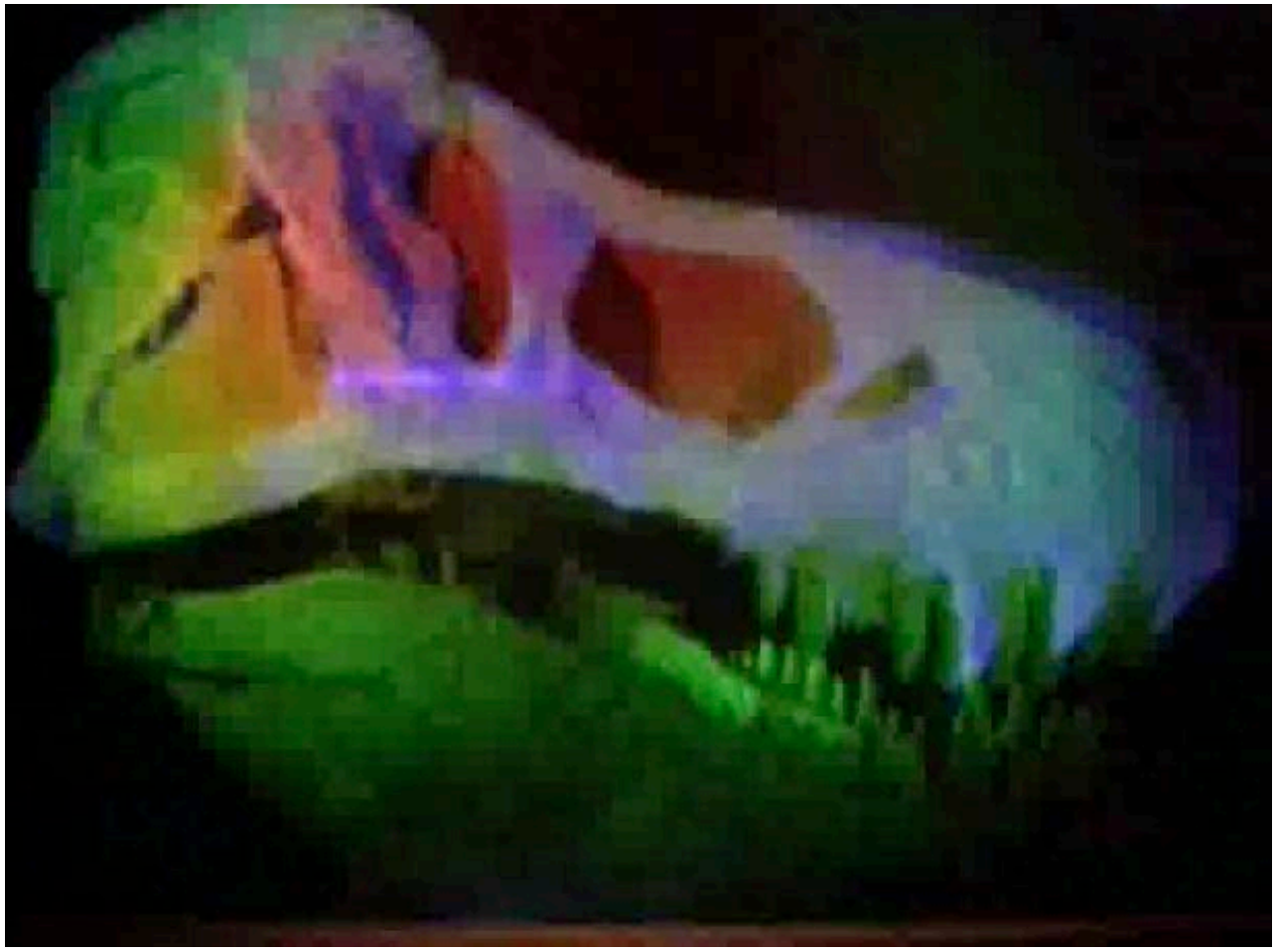




Holographics

Passive range scanning



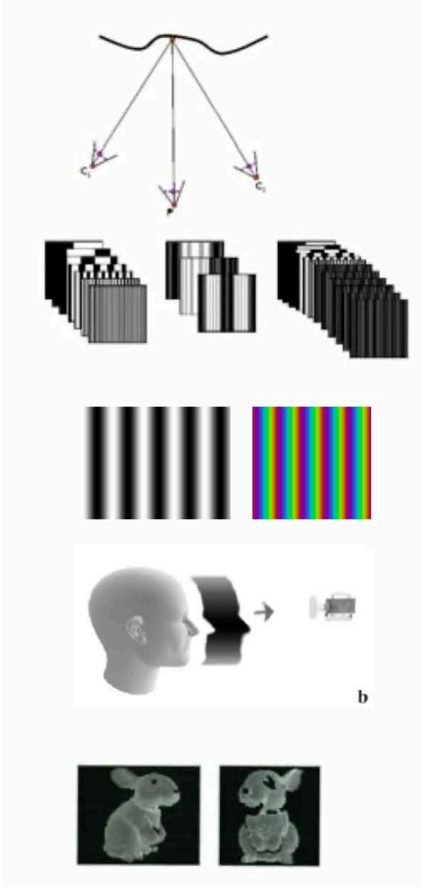


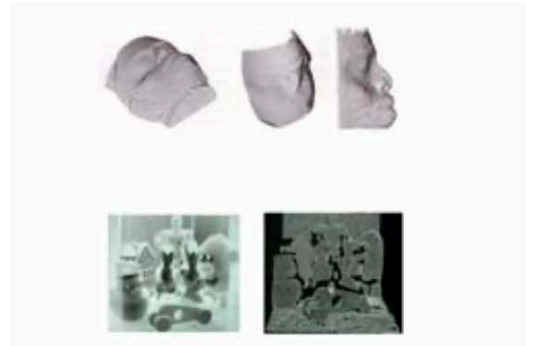
Holographics

Active Range Scanning

Overview

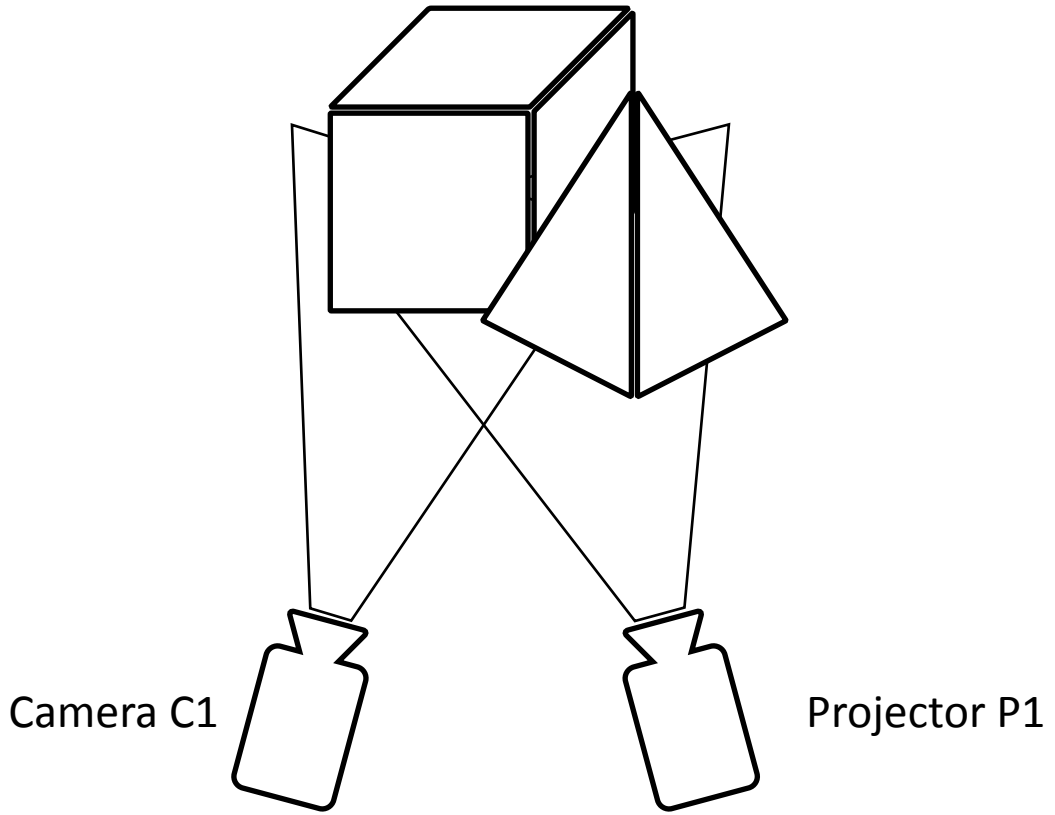
- Range Scanning
  - Passive:
    - Stereo or Multi-Camera
  - Active:
    - Structured Light Projection
    - Optical Time-of-Flight
      - Direct ToF
      - Indirect ToF
- Registration of Depth Maps
- Processing Depth Maps





Active Range Scanning

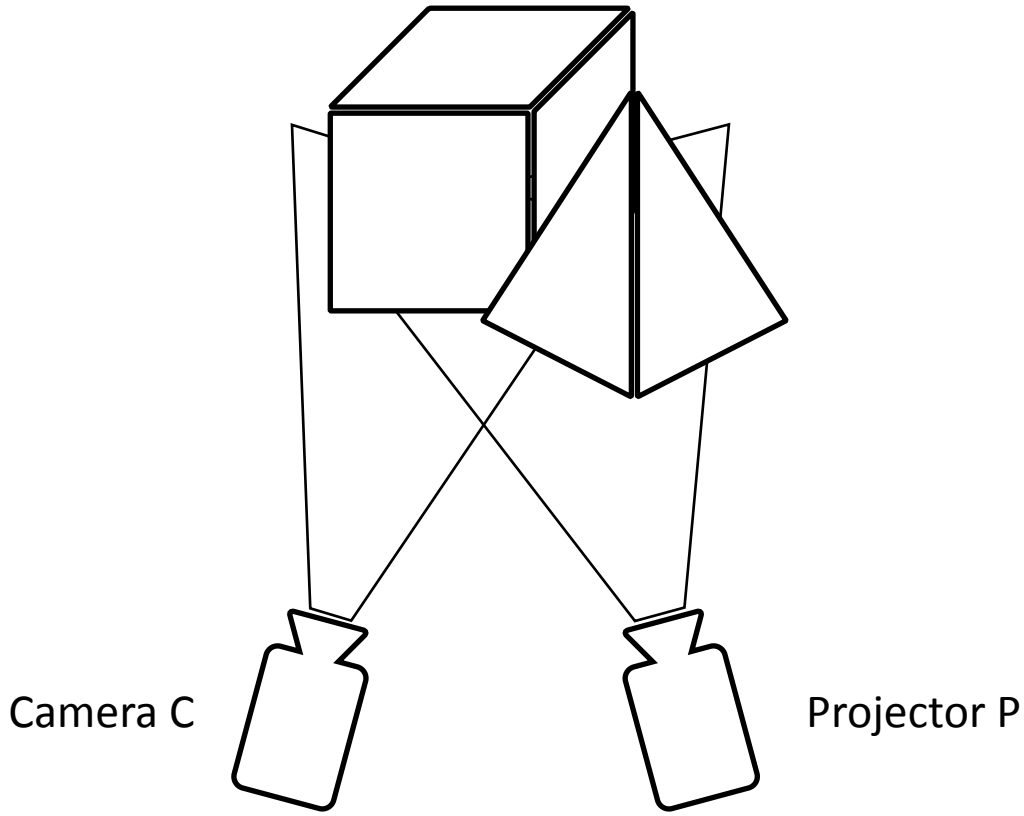
- Why active range scanning?
  - It avoids the correlation problem
- Replace one camera with a projector / light emitter
  - Structured Light Projection
  - Optical Time-of-Flight
    - Direct ToF
    - Indirect ToF

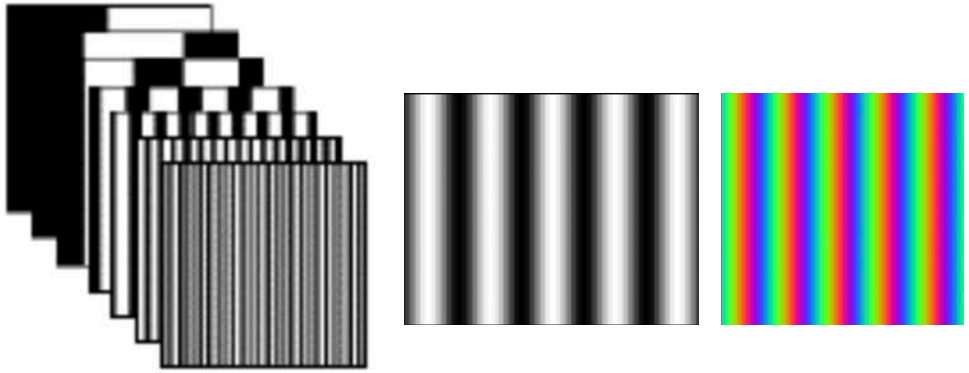


Structured Light Projection

Structured Light Projection

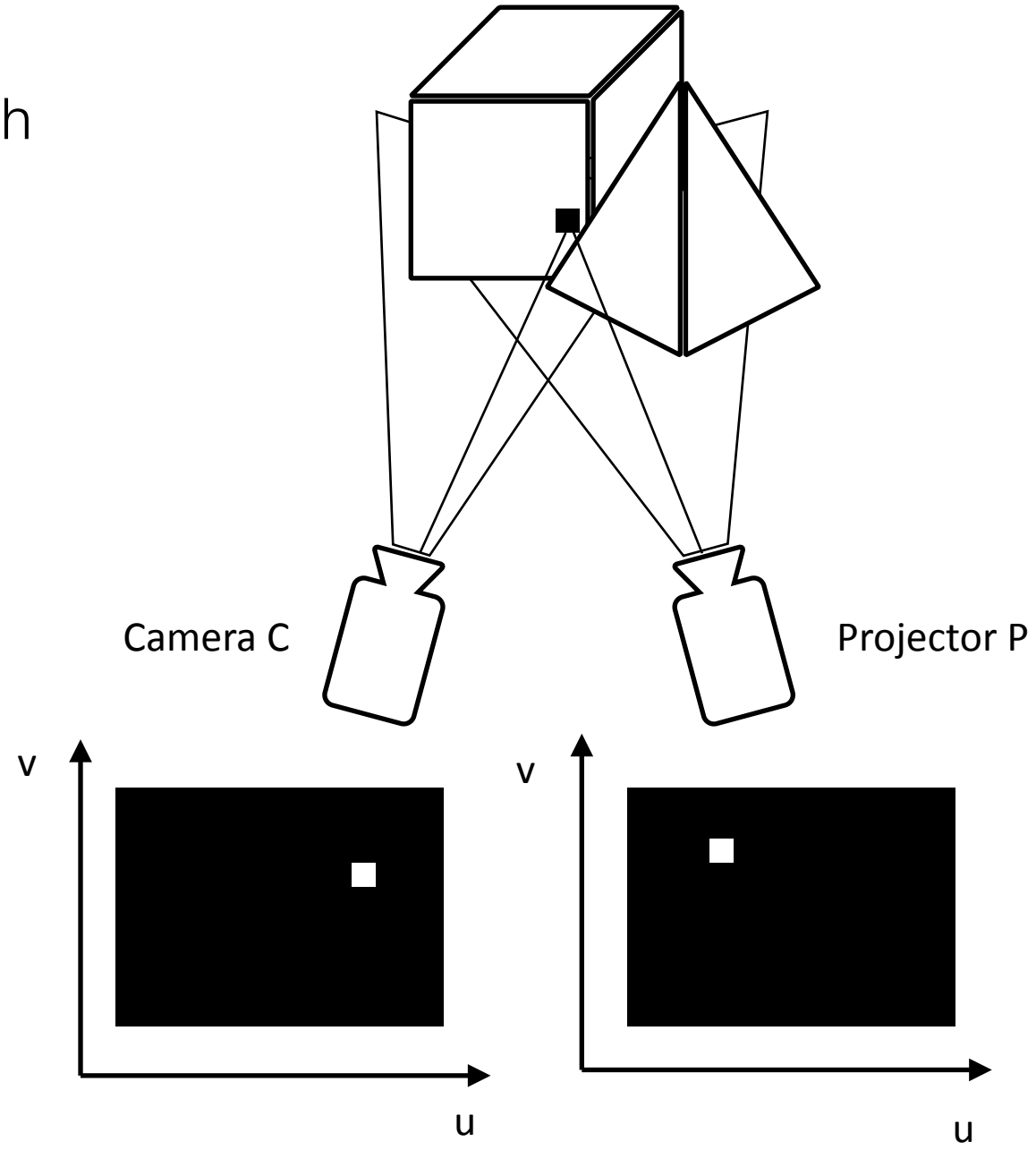
- Many structured light techniques exist that follow the two objectives:
  - Determine projector-camera correspondences with as few images as possible
  - Be robust against surface modulation and noise
- Codes include:
  - Binary codes, such as the Gray code (most common): <100 images, quite robust
  - Intensity codes that apply cosine patterns and phase shifts: 6 images, but are not robust against strong absorptions
  - Color coding: <3 images, but very fragile (does not work on coloured surfaces)





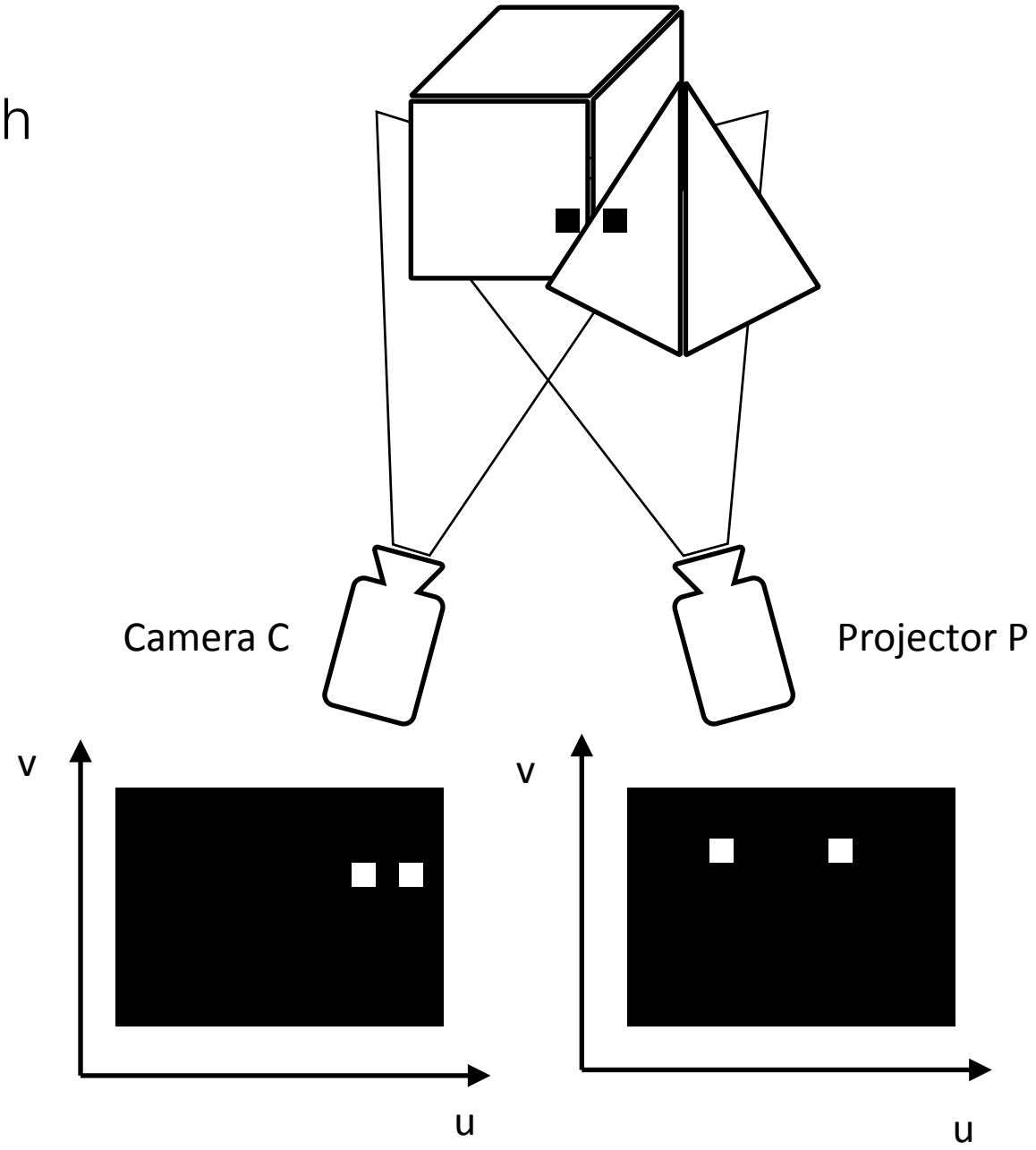
Structured Light Projection

- Let's assume 1 projector + 1 camera (both are calibrated)
- The simplest light pattern is projecting a point at $[u_P, v_P]$ that can be found at $[u_C, v_C]$. This has to be repeated for all points
- What is the problem?
  - Too slow!
- What is a better approach?



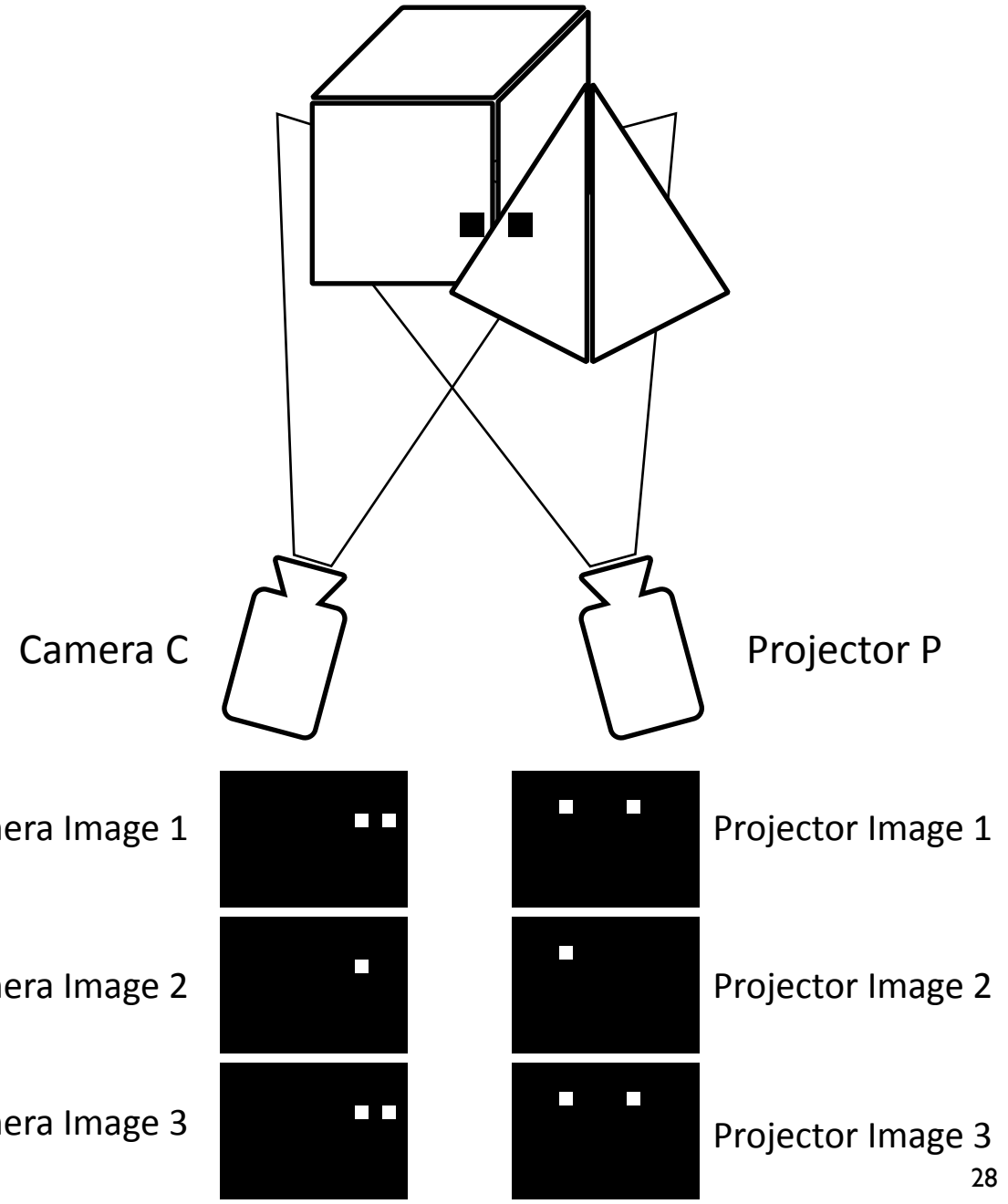
Structured Light Projection

- Let's assume 1 projector + 1 camera (both are calibrated)
- The simplest light pattern is projecting a point at $[u_P, v_P]$ that can be found at $[u_C, v_C]$. This has to be repeated for all points
- What is the problem?
  - Too slow!
- Projecting two points!
  - But how to distinguish?



Structured Light Projection

- Let's assume 1 projector + 1 camera (both are calibrated)
- The simplest light pattern is projecting a point at $[u_P, v_P]$ that can be found at $[u_C, v_C]$. This has to be repeated for all points
- What is the problem?
  - Too slow!
- Projecting two points!
  - Project unique "code"
- But how to scale up?

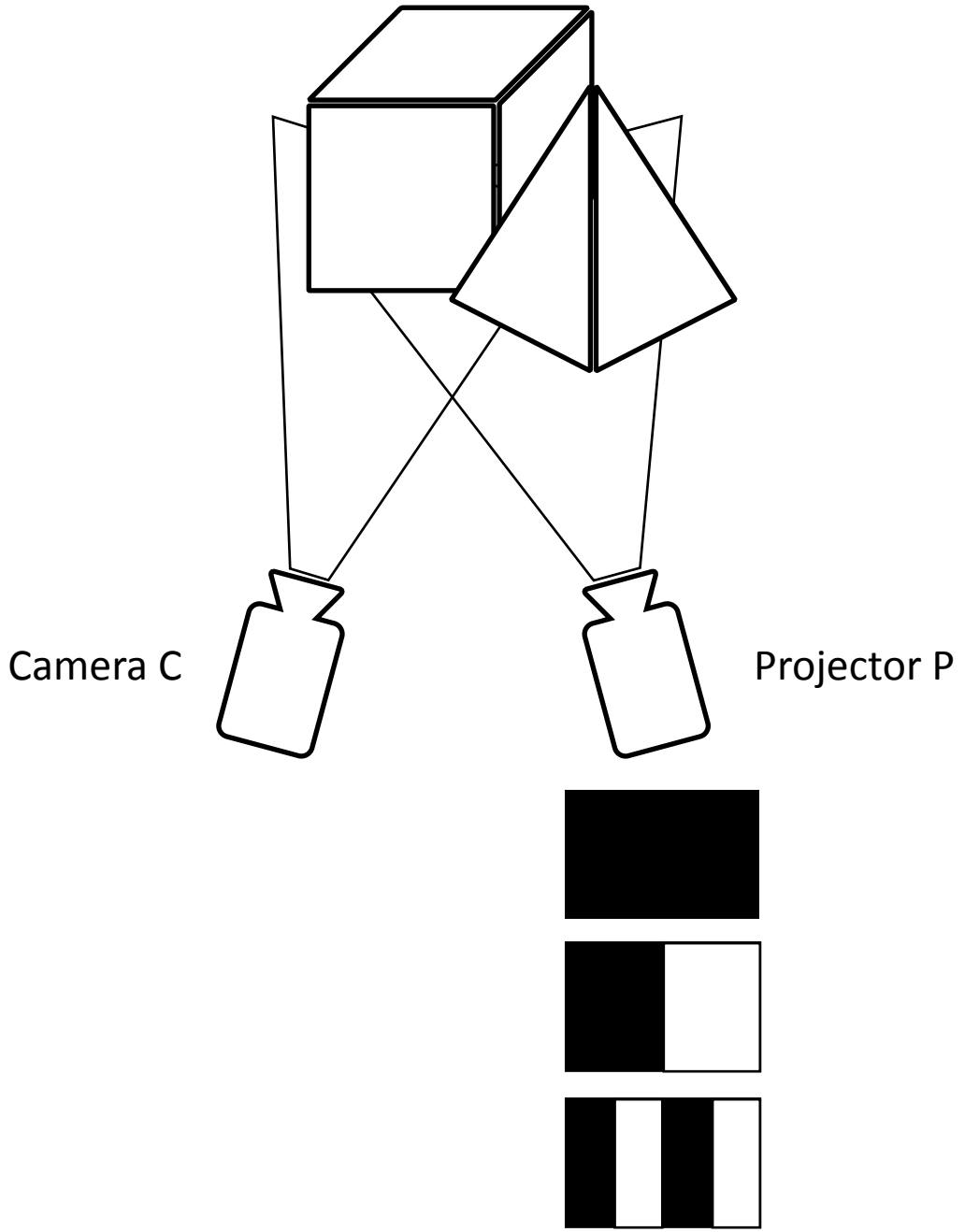


Structured Light Projection

Binary calibration

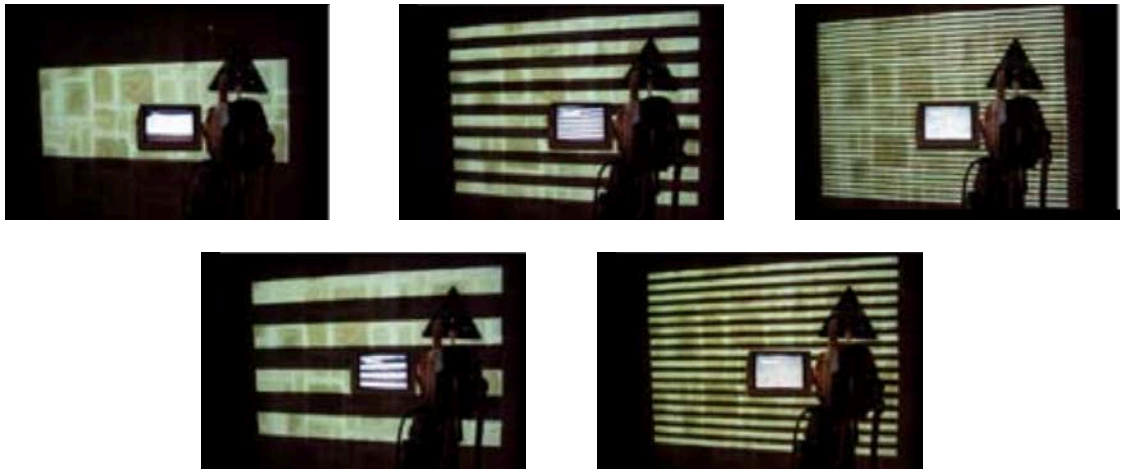
Structured Light Projection

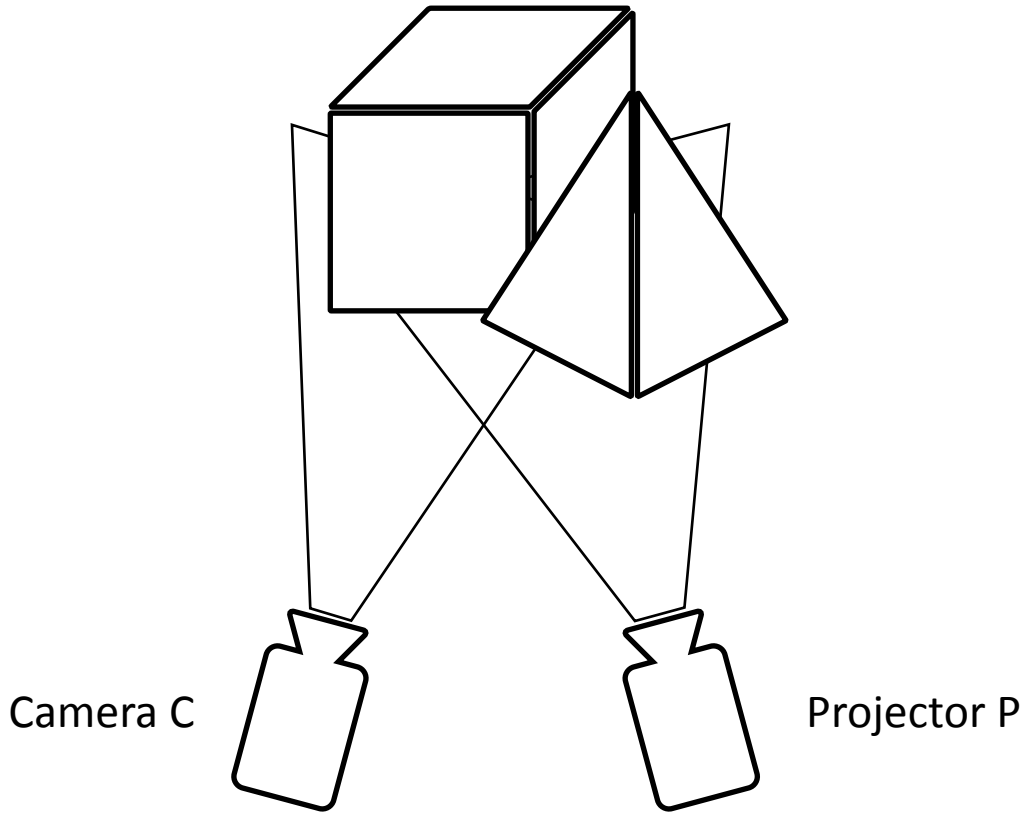
- Let's further optimise:
  - Time multiplexing with line strip scanning
  - Each projected pixel codes a unique ID (e.g. Binary pattern)
  - How many images for 1024px resolution?

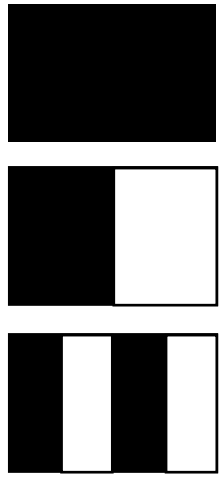


Structured Light Projection

- Let's further optimise:
  - Time multiplexing with line strip scanning
  - Each projected pixel codes a unique ID (e.g. Binary pattern)
  - How many images for 1024px resolution?



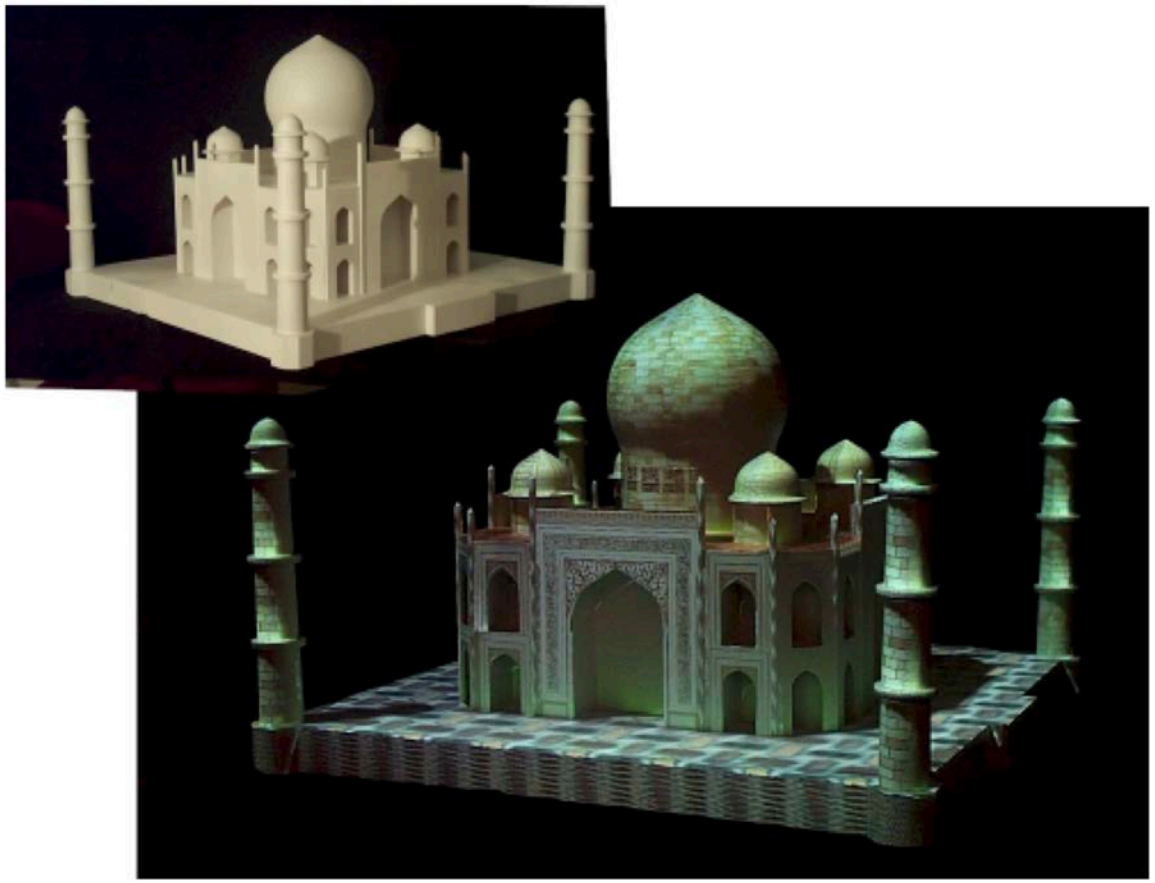


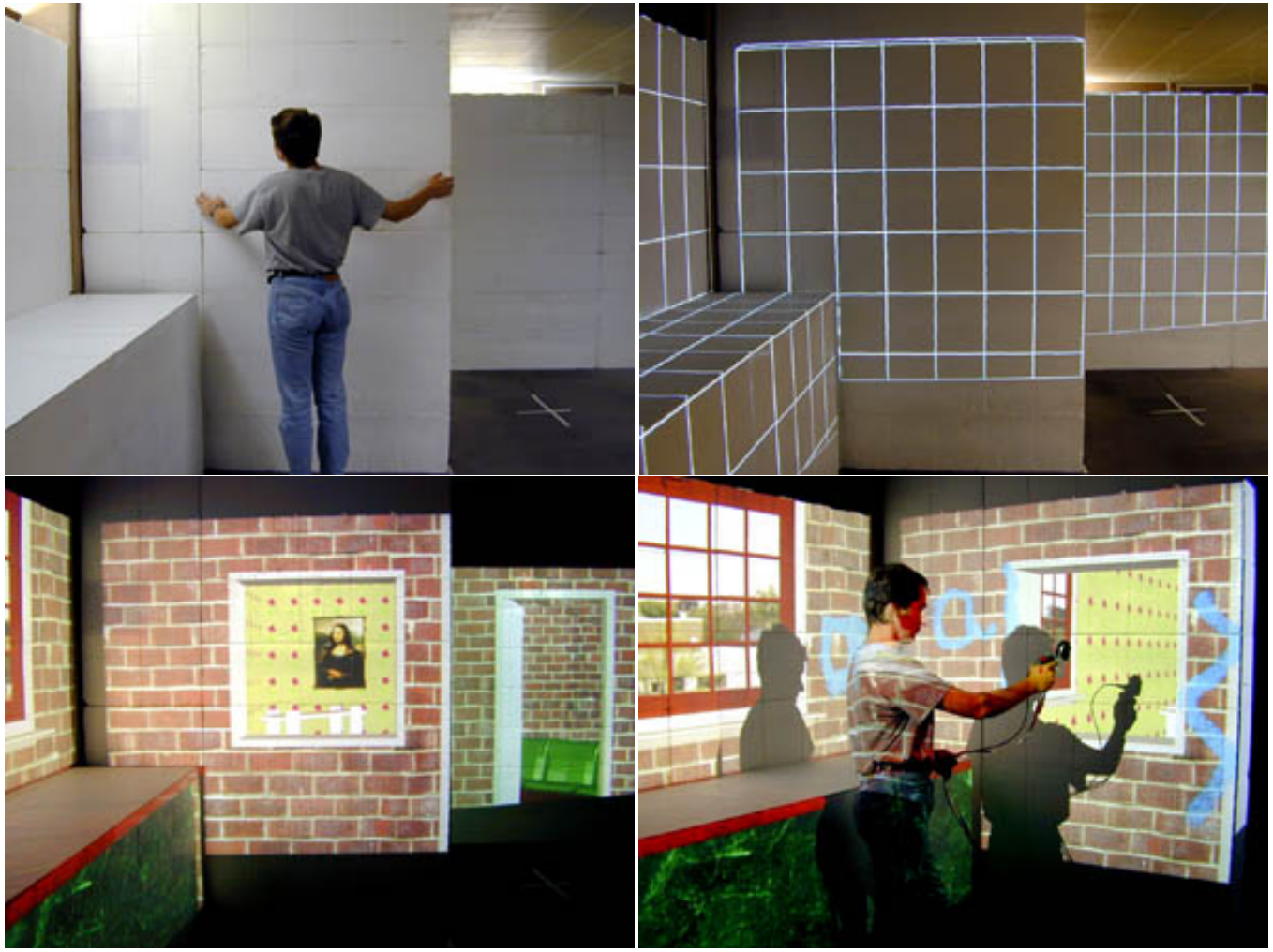


Structured Light Projection

Smart projectors

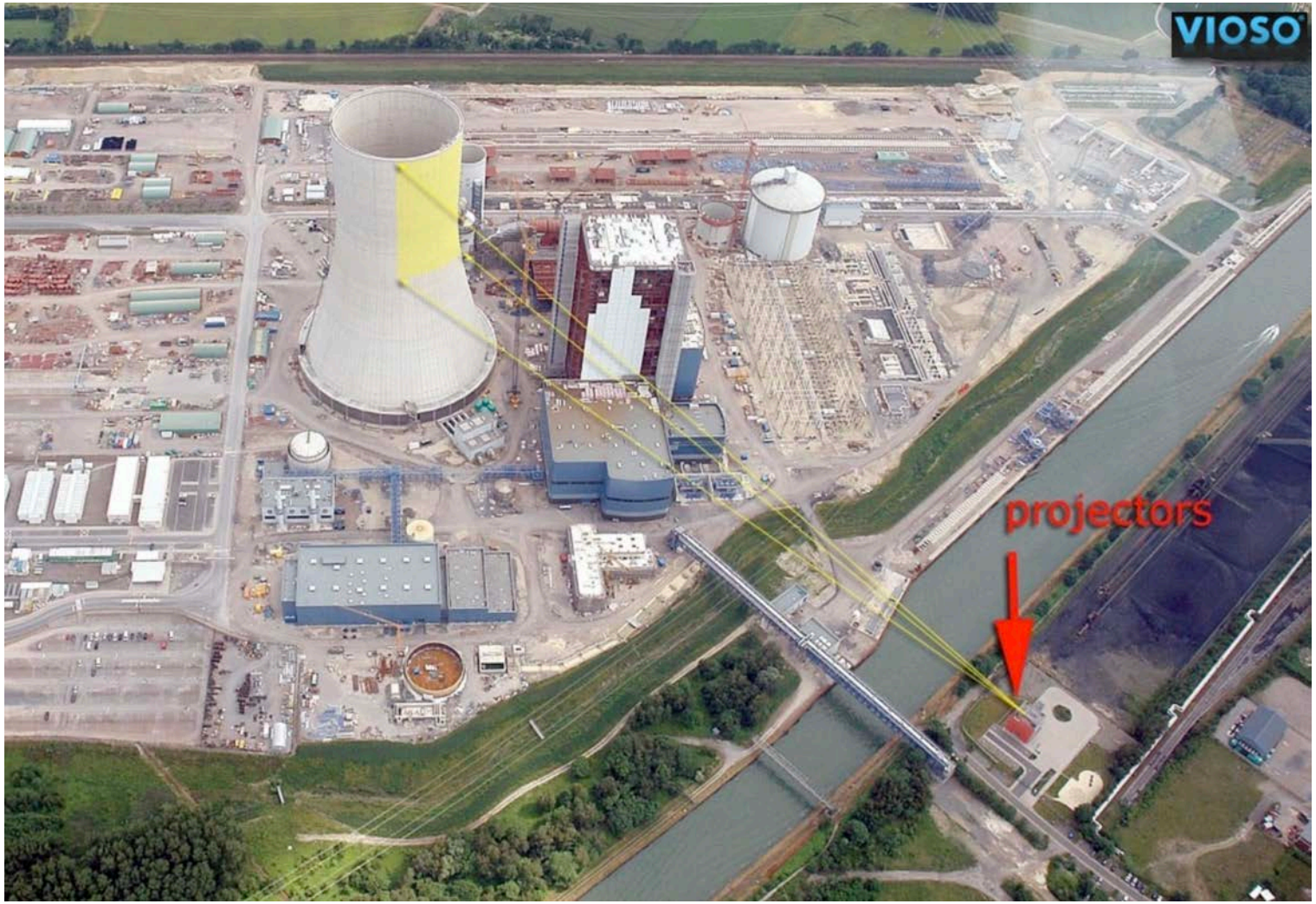
Structured Light Projection





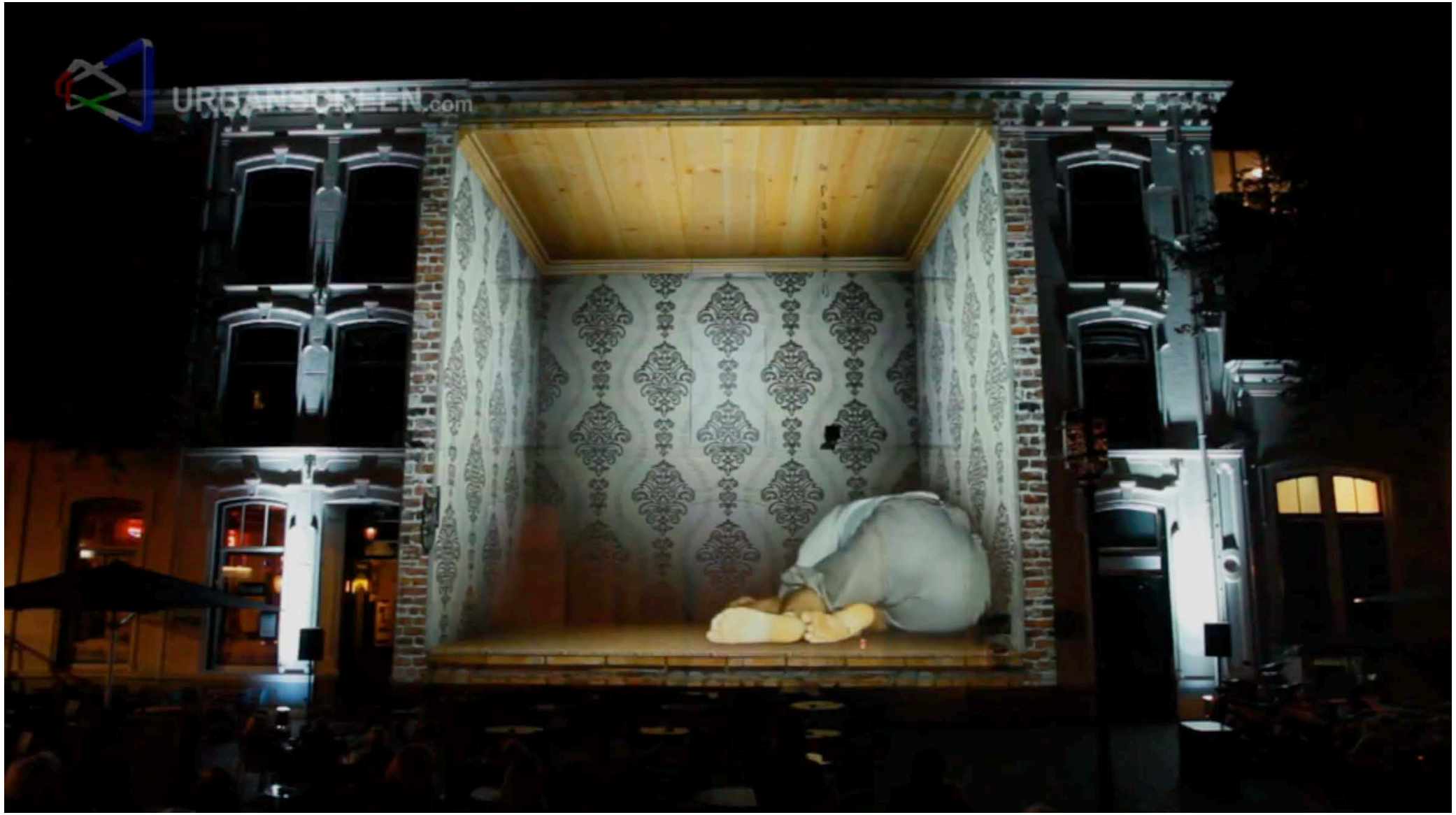
Raskat et al. “Shaderlamps”

Structured Light Projection



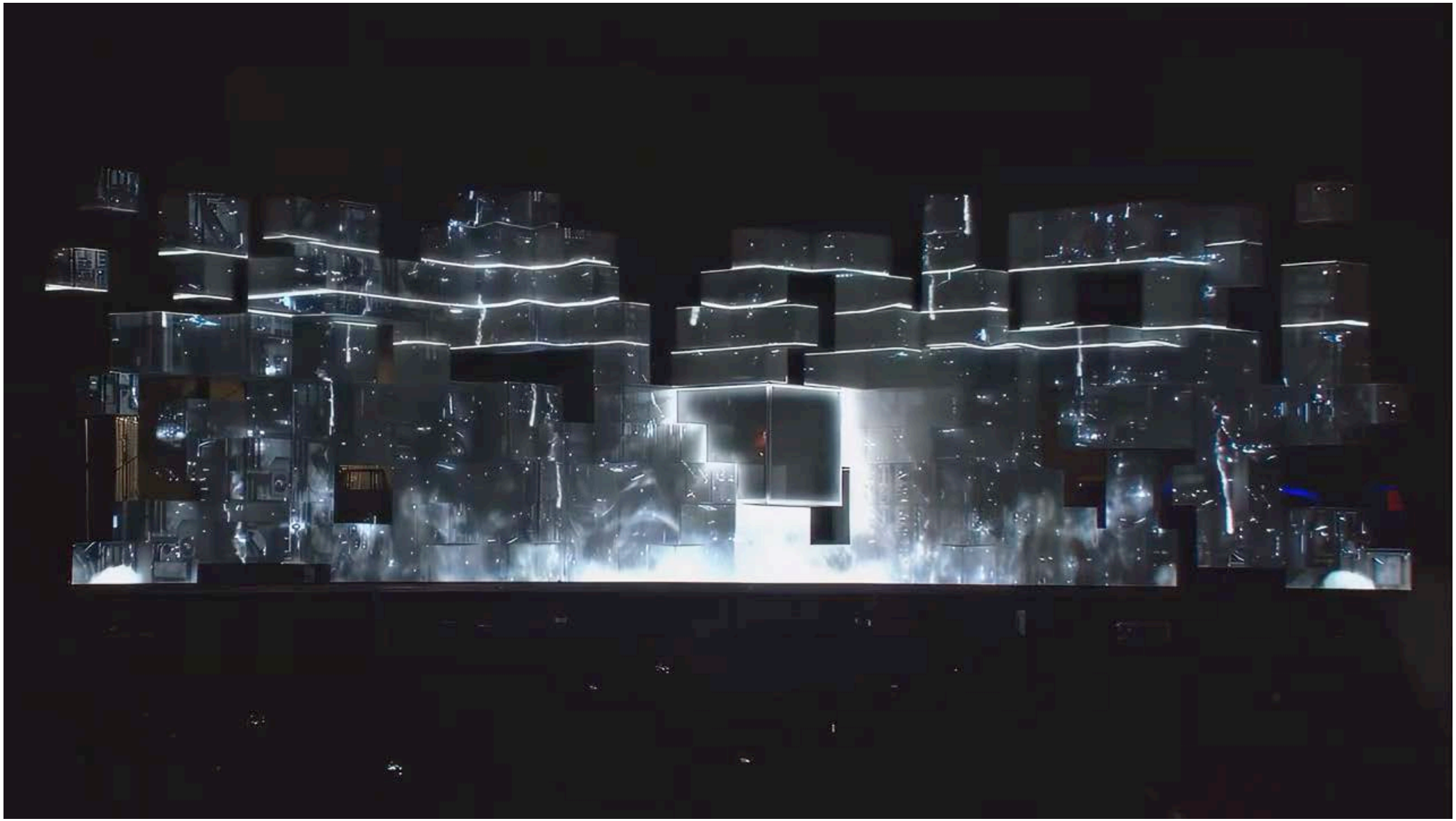
Smart projectors / Vioso

Structured Light Projection



Smart projectors / Vioso

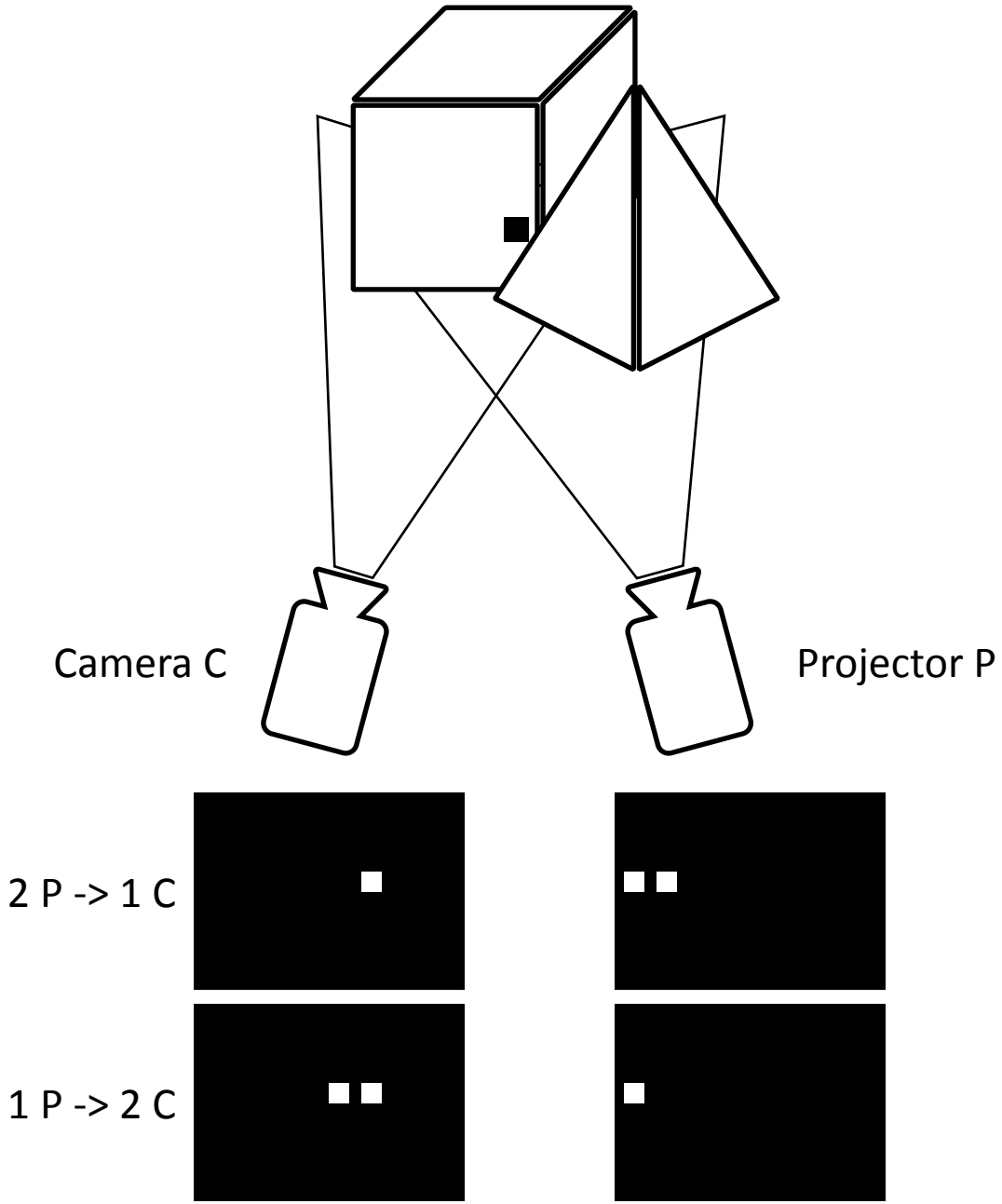
Structured Light Projection



Amob Tobin

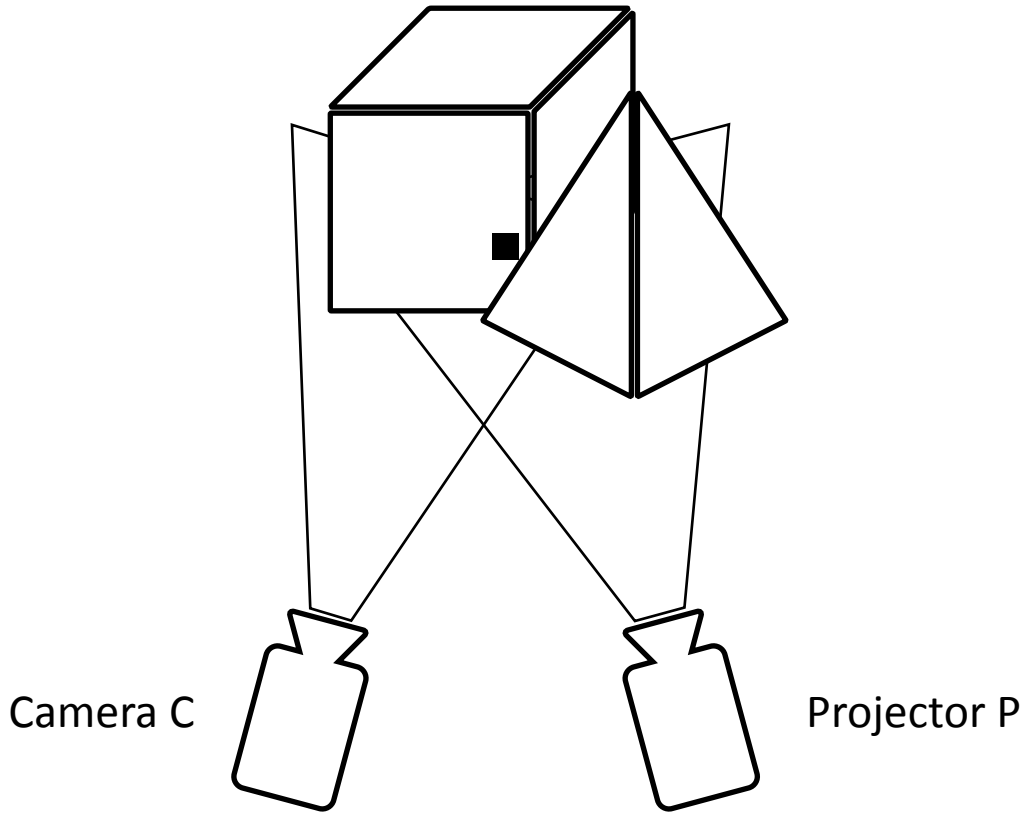
Structured Light Projection

- Other issues:
  - Results usually not in 1-to-1 mappings!
    - Multiple projector pixels can map on same camera pixels / multiple camera pixels can be addressed by single projector pixel
  - Solution?
    - Multiple entries must be averaged



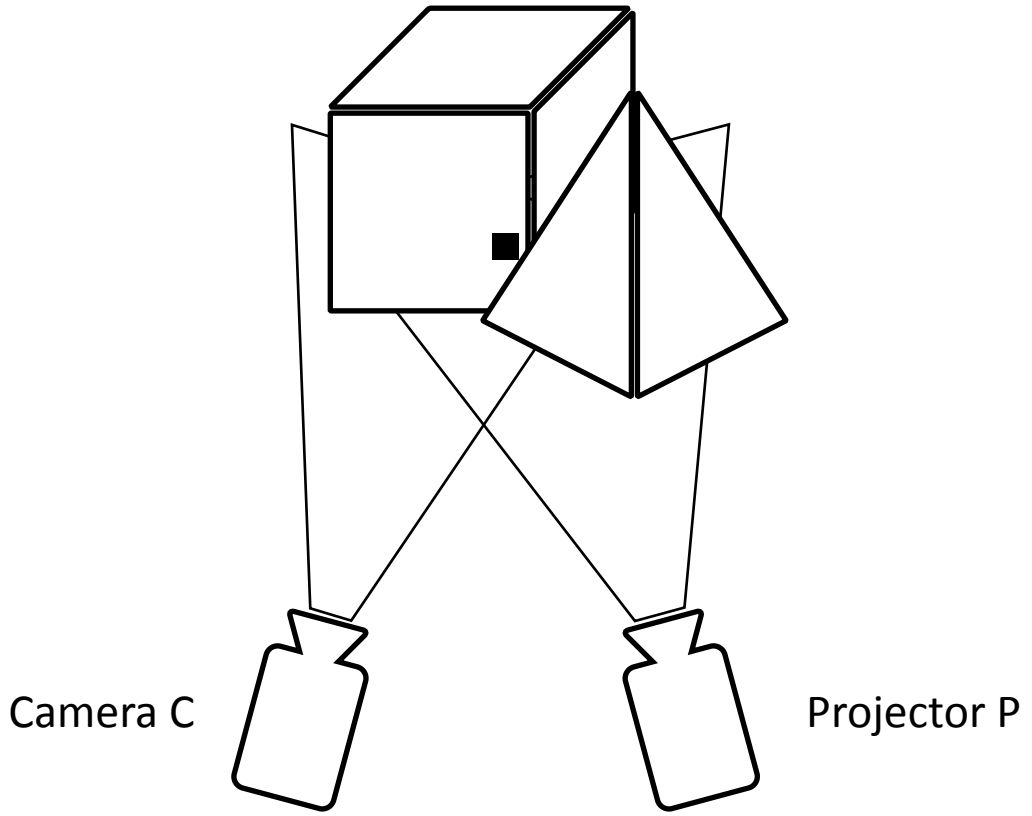
Structured Light Projection

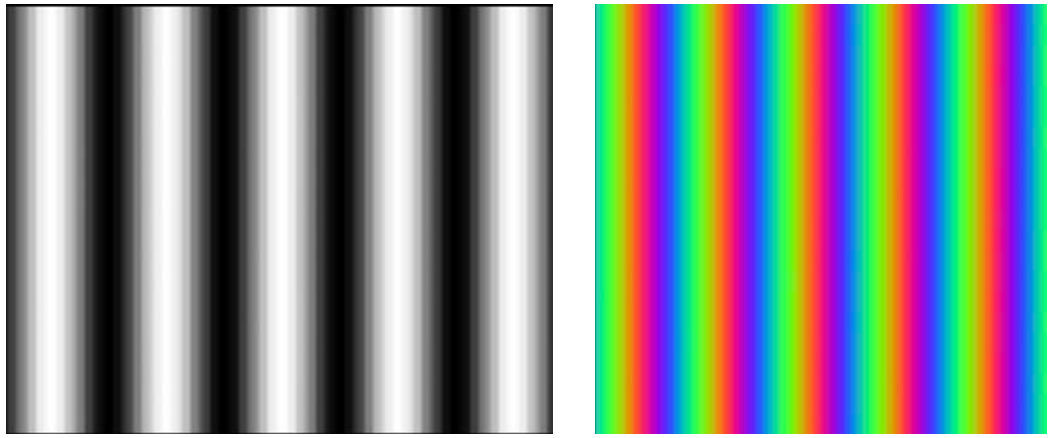
- Other issues:
  - Results usually not in 1-to-1 mappings!
    - Multiple projector pixels can map on same camera pixels / multiple camera pixels can be addressed by single projector pixel
  - Solution?
    - Multiple entries must be averaged
- Any other ideas for improvements?



Structured Light Projection

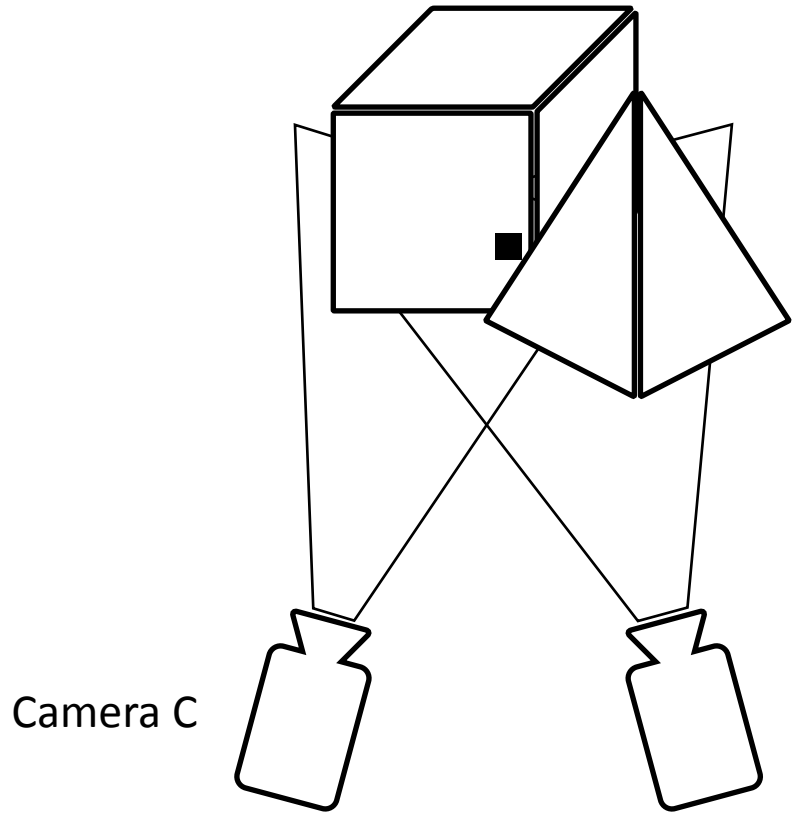
- Other issues:
  - Results usually not in 1-to-1 mappings!
    - Multiple projector pixels can map on same camera pixels / multiple camera pixels can be addressed by single projector pixel
  - Solution?
    - Multiple entries must be averaged
- Any other ideas for improvements?
  - Encode more information into structured projection
  - For example: colours and intensities

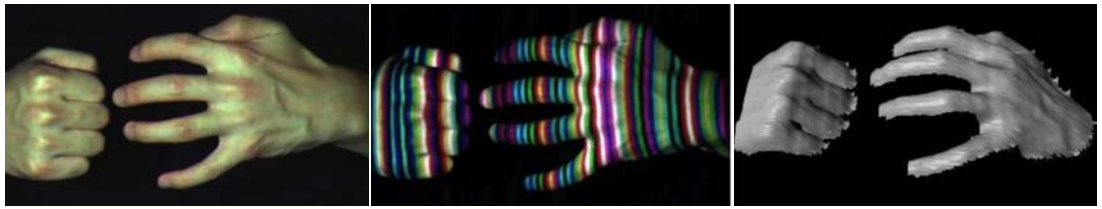




Structured Light Projection

- Other issues:
  - Results usually not in 1-to-1 mappings!
    - Multiple projector pixels can map on same camera pixels / multiple camera pixels can be addressed by single projector pixel
  - Solution?
    - Multiple entries must be averaged
- Any other ideas for improvements?
  - Encode more information into structured projection
  - For example: colours and intensities

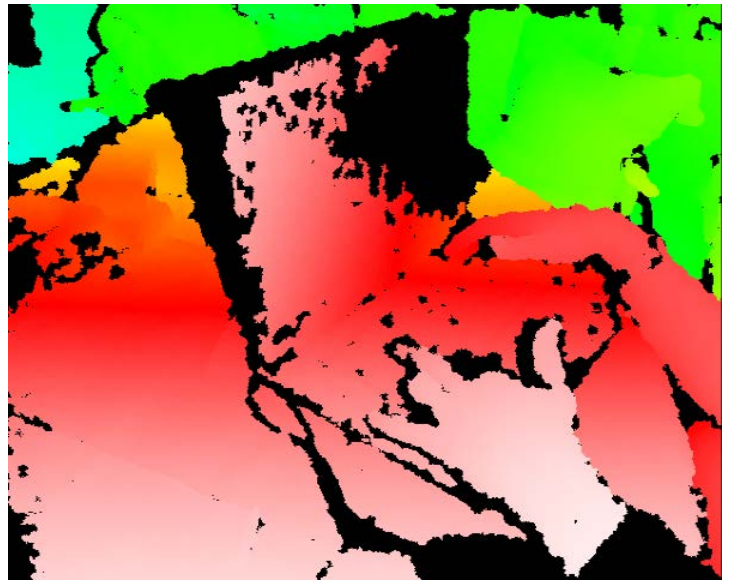




Structured Light Projection

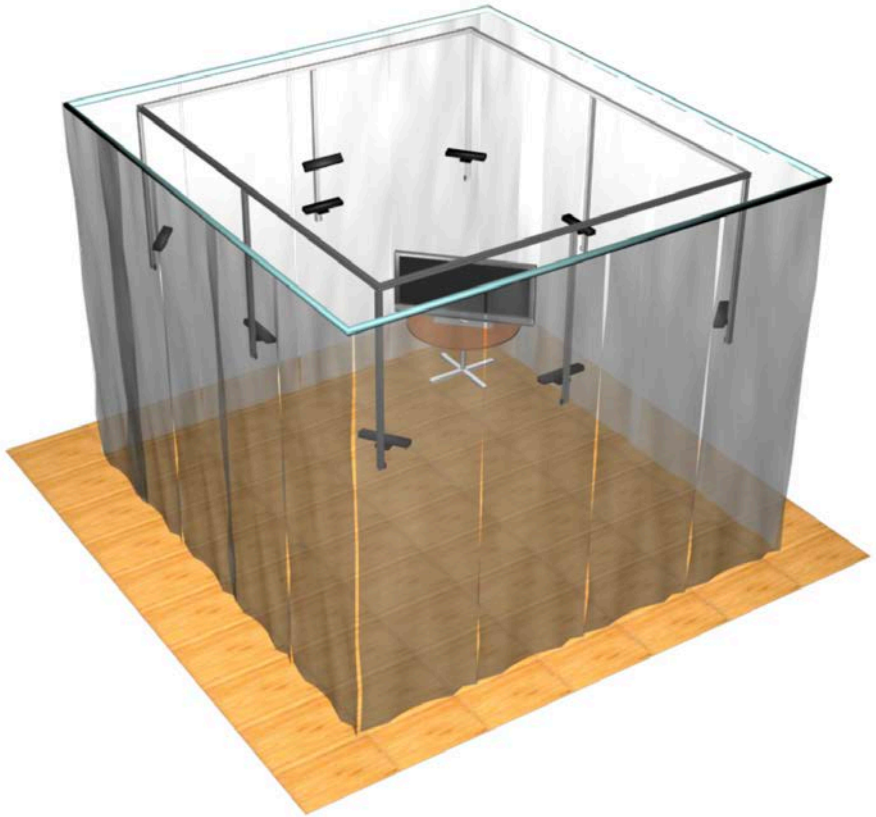
https://fts.gr/structured-light-3d-scanner/

Structured Light Projection



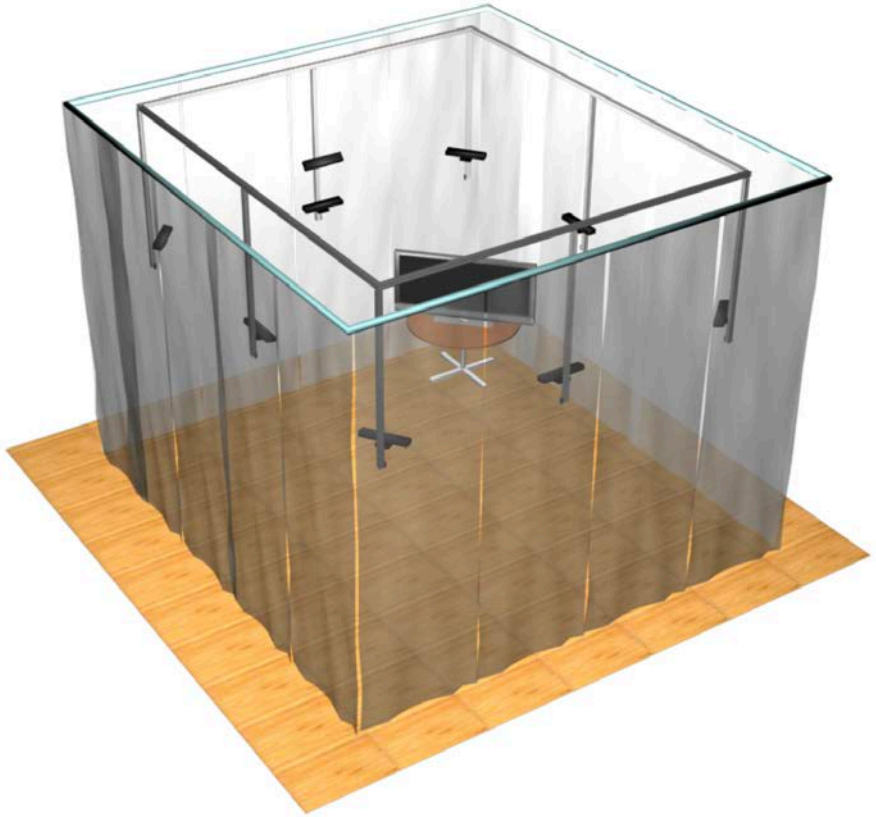
MS Kinect 1 (PrimeSense sensor, not MS Kinect 2 uses ToF discussed later)

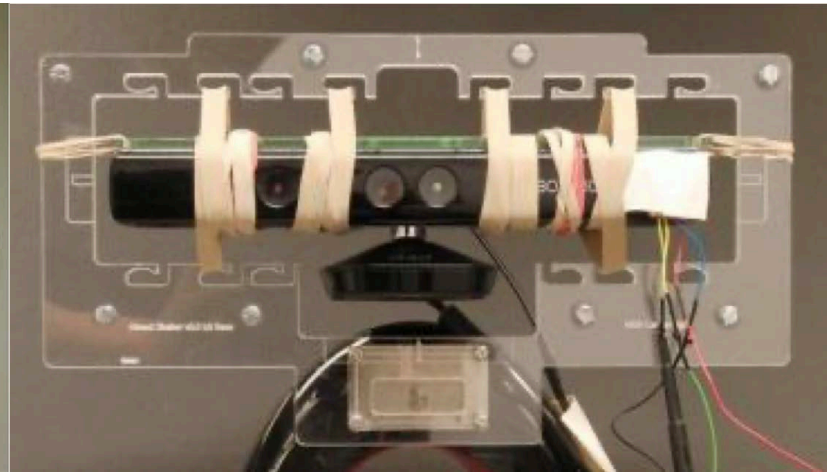
Structured Light Projection



Maimone and Fuchs “Reducing Interference Between Multiple Structured Light Depth Sensors Using Motion”

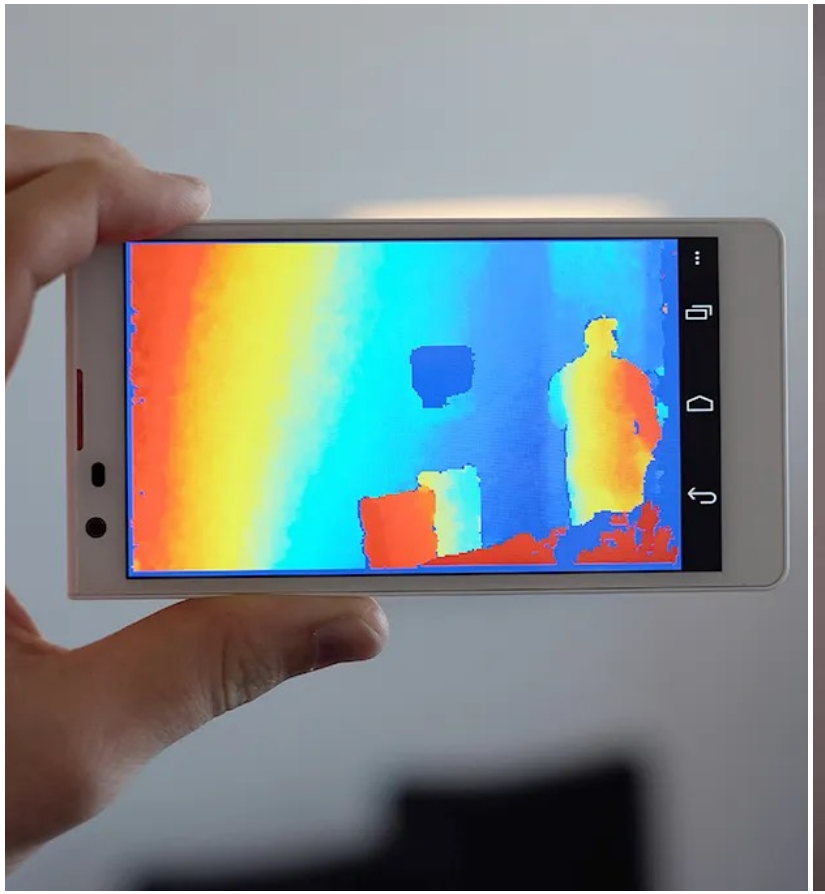
Structured Light Projection





Maimone and Fuchs “Reducing Interference Between Multiple Structured Light Depth Sensors Using Motion”

Structured Light Projection

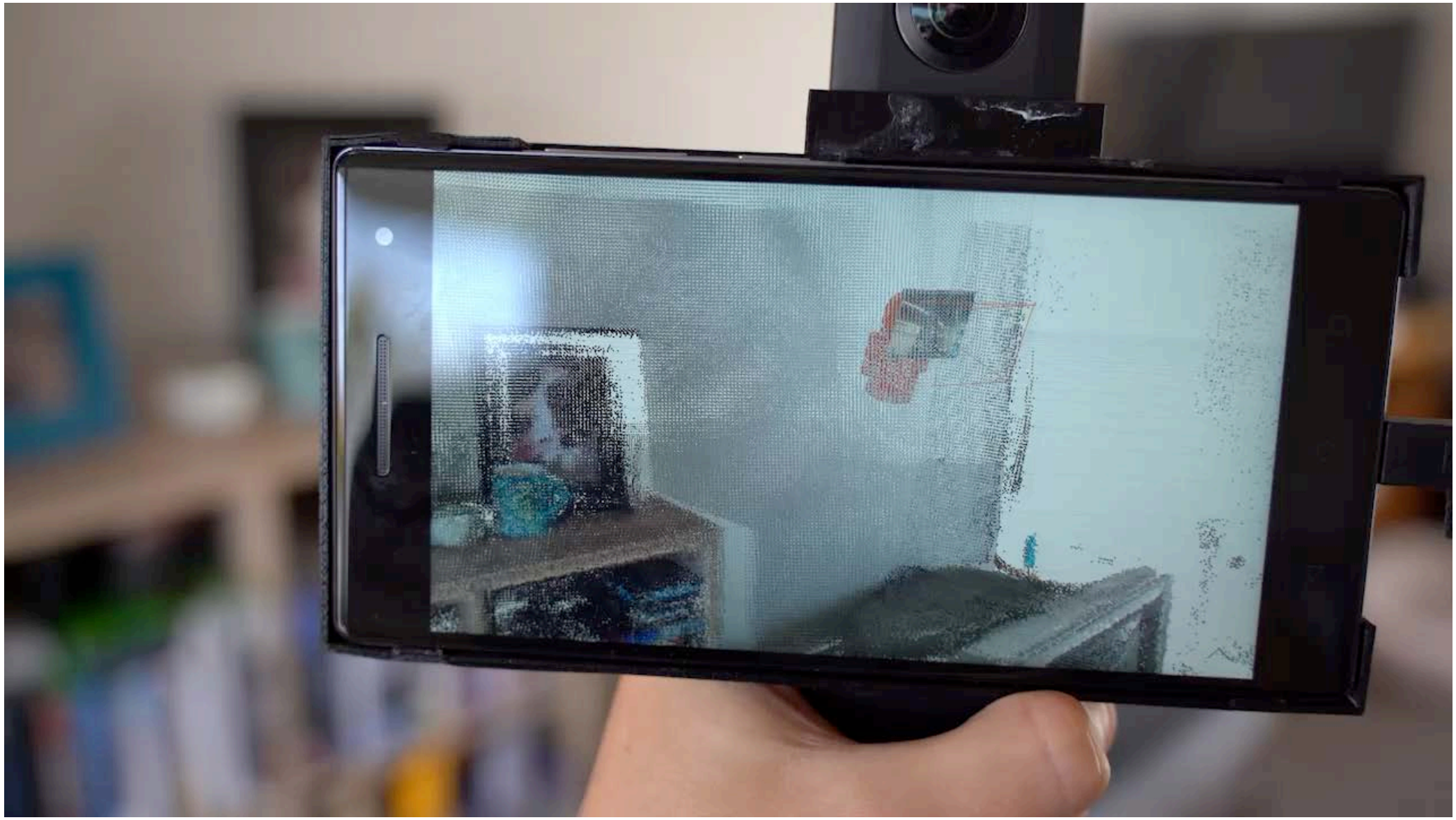


Google Project Tango (also using ToF discussed later)

Structured Light Projection

Mobile Portation

Structured Light Projection

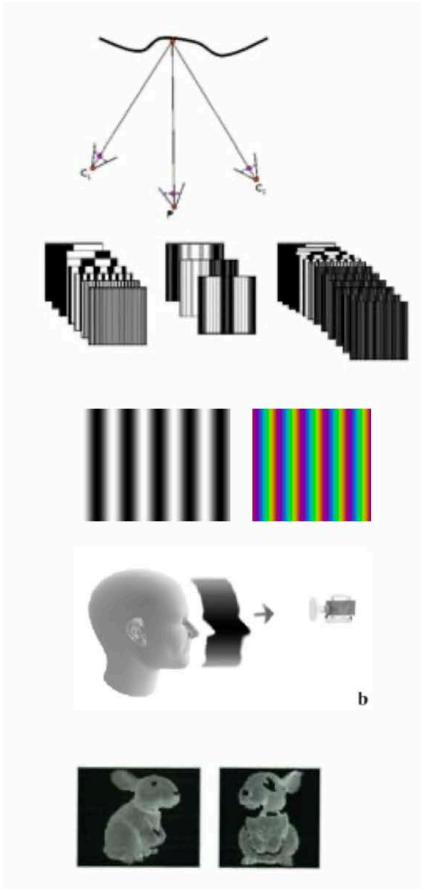


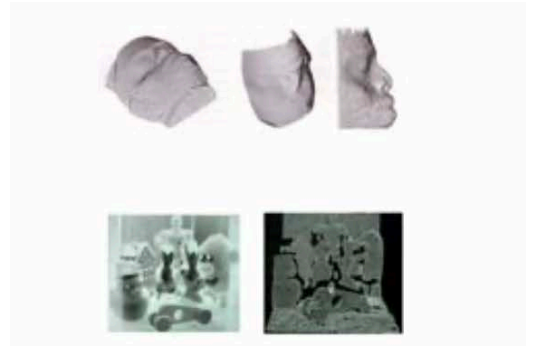
Mobile Portation

Time of Flight (ToF)

Overview

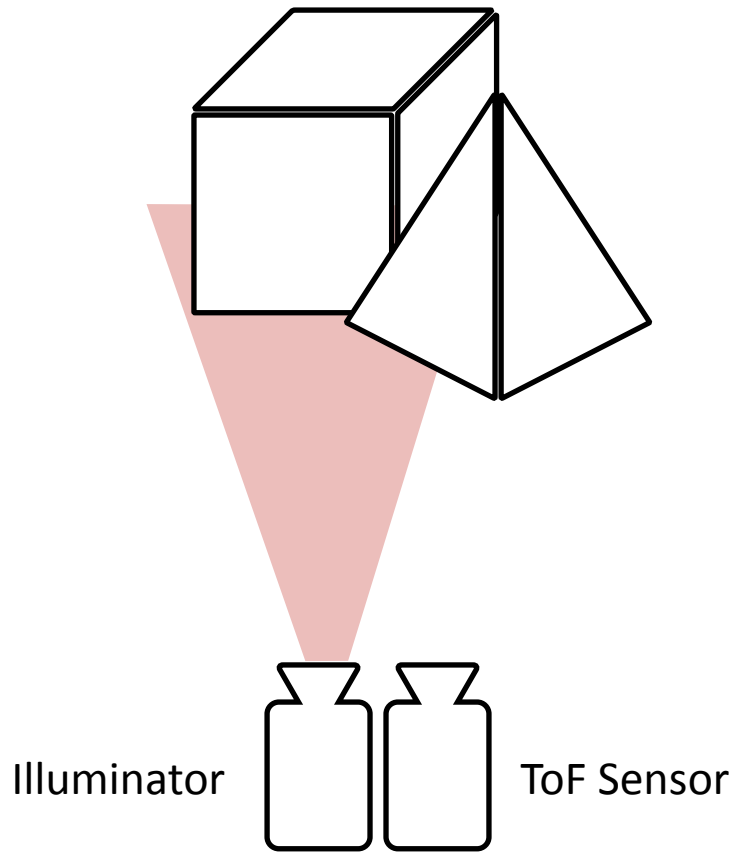
- Range Scanning
  - Passive:
    - Stereo or Multi-Camera
  - Active:
    - Structured Light Projection
    - Optical Time-of-Flight
      - Direct ToF
      - Indirect ToF
- Registration of Depth Maps
- Processing Depth Maps





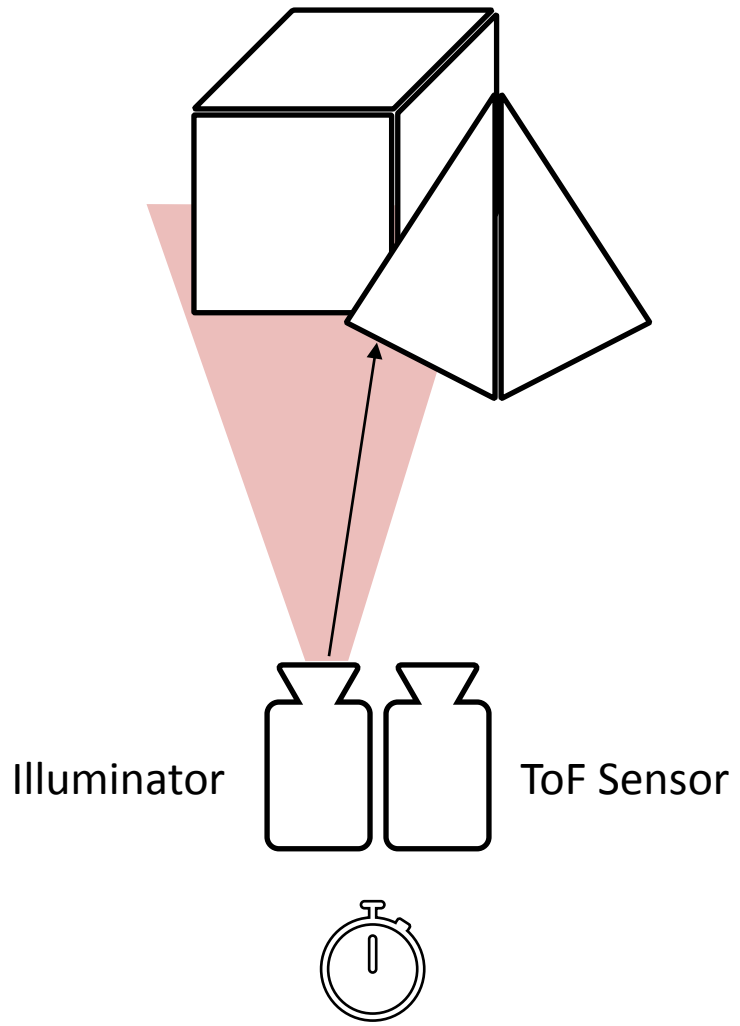
Optical Time-of-Flight (ToF)

- General idea:
  - Compute depth by round-trip estimation of a light wave emitted and its reflection back to the sensor
  - $D = C \cdot \Delta t/2$
- Use low-power IR illumination for pulsed illumination
  - Can be switched on/off with <1ns
  - Generates a “light wall” that is reflected back by objects



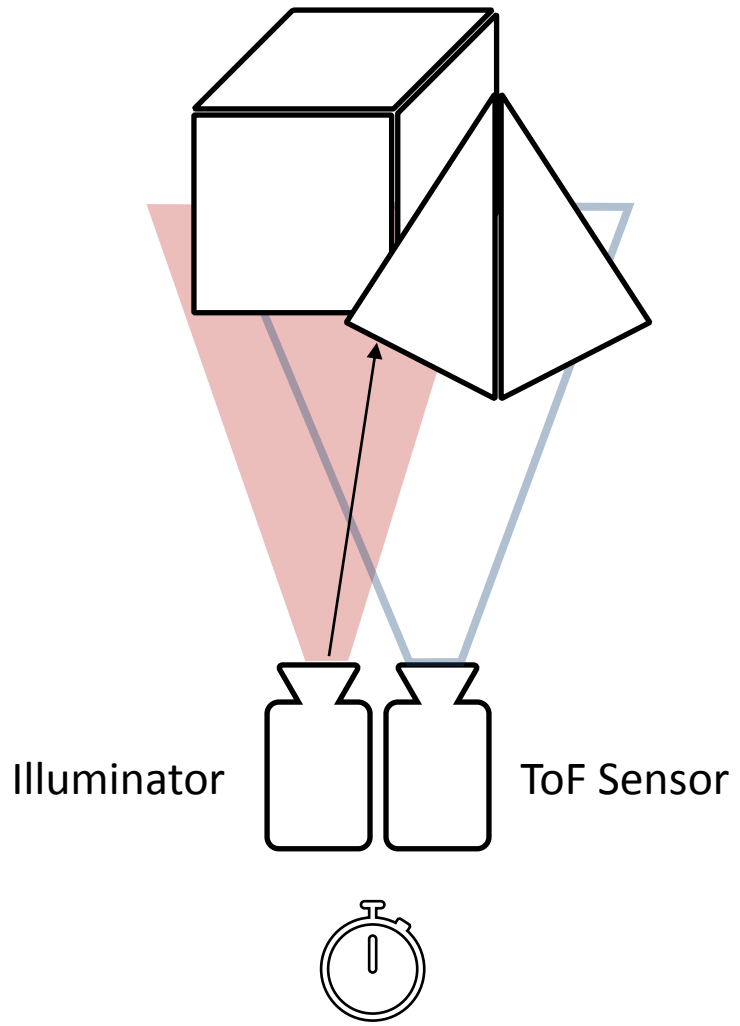
Optical Time-of-Flight (ToF)

- General idea:
  - Compute depth by round-trip estimation of a light wave emitted and its reflection back to the sensor
  - $D = C \cdot \Delta t/2$
- Use low-power IR illumination for pulsed illumination
  - Can be switched on/off with <1ns
  - Generates a “light wall” that is reflected back by objects



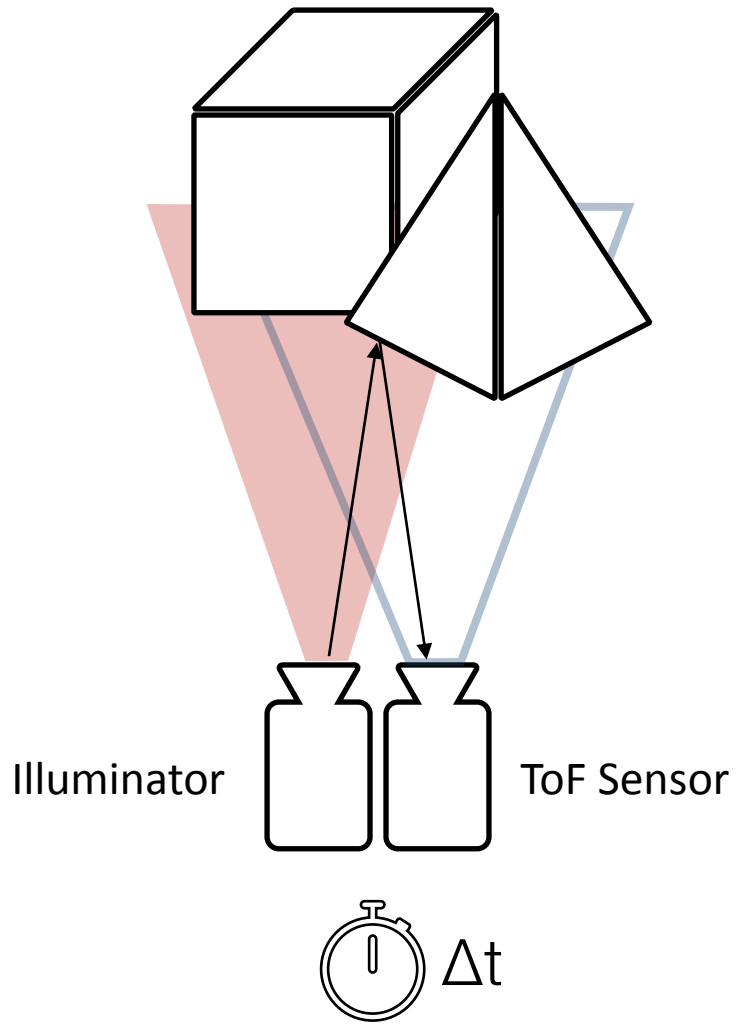
Optical Time-of-Flight (ToF)

- General idea:
  - Compute depth by round-trip estimation of a light wave emitted and its reflection back to the sensor
  - $D = C \cdot \Delta t/2$
- Use low-power IR illumination for pulsed illumination
  - Can be switched on/off with <1ns
  - Generates a “light wall” that is reflected back by objects



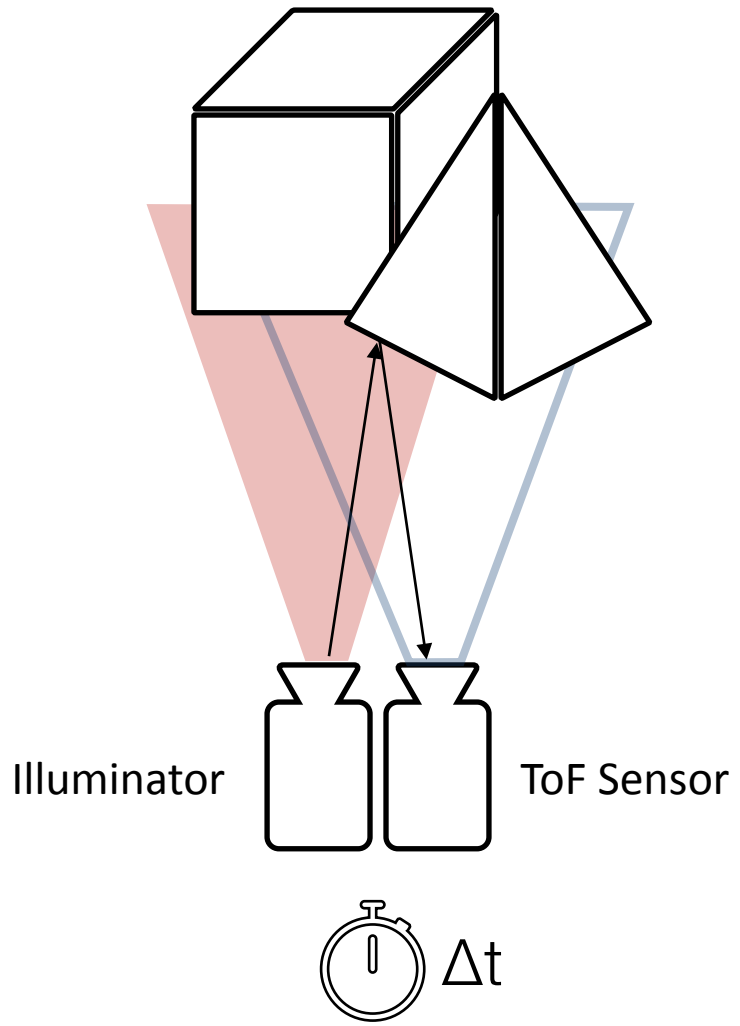
Optical Time-of-Flight (ToF)

- General idea:
  - Compute depth by round-trip estimation of a light wave emitted and its reflection back to the sensor
  - $D = C \cdot \Delta t/2$
- Use low-power IR illumination for pulsed illumination
  - Can be switched on/off with <1ns
  - Generates a “light wall” that is reflected back by objects



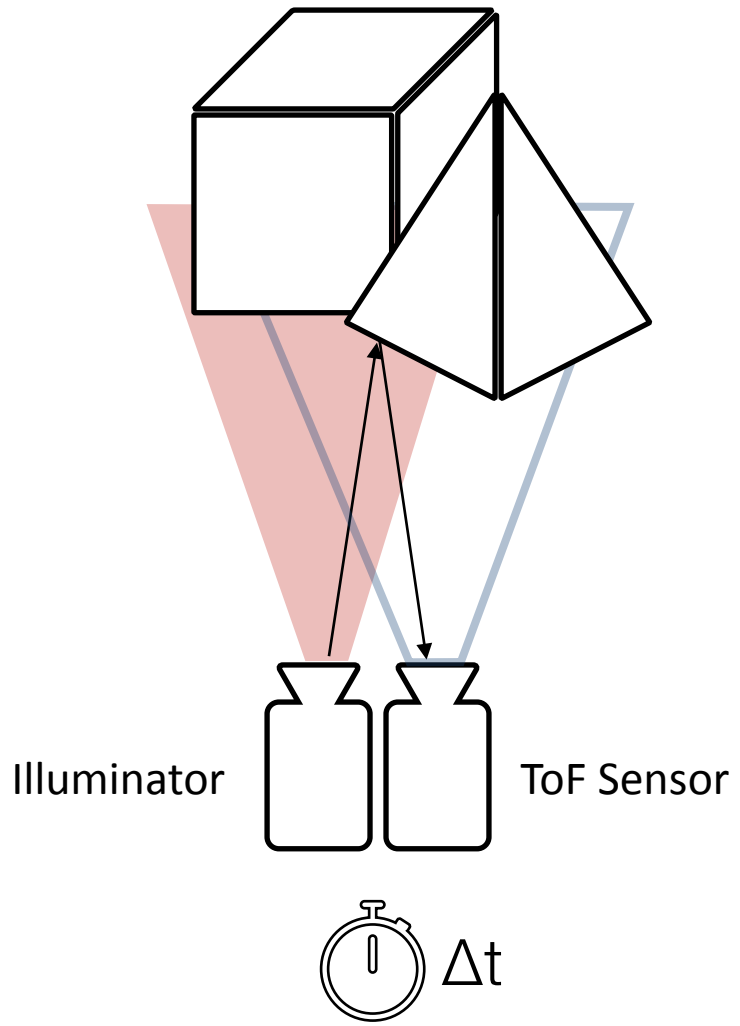
Optical Time-of-Flight (ToF)

- General idea:
  - Compute depth by round-trip estimation of a light wave emitted and its reflection back to the sensor
  - $D = C \cdot \Delta t/2$
  - $\Delta t = 2D/C$
  - $D=1m\ \Delta t = 6.6*10^{-09}\ s$
  - $D=0.01m\ \Delta t = 6.6*10^{-11}\ s$
  - $D=0.001m\ \Delta t = 6.6*10^{-12}\ s$ (pico seconds)



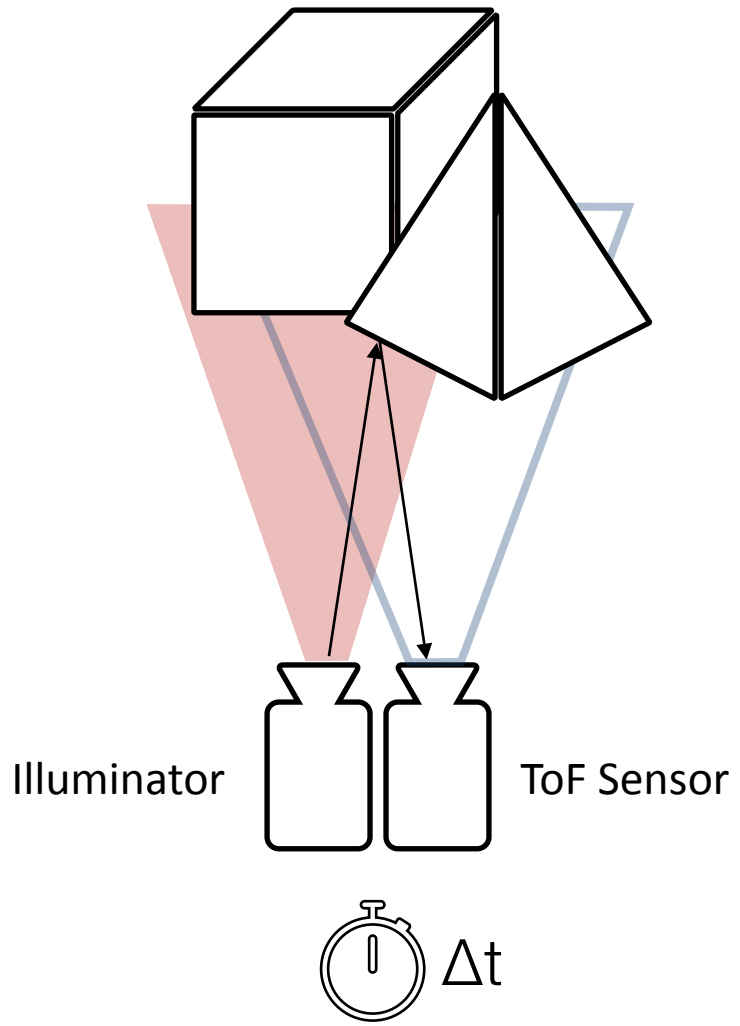
Optical Time-of-Flight (ToF)

- General idea:
  - Compute depth by round-trip estimation of a light wave emitted and its reflection back to the sensor
  - $D = C \cdot \Delta t/2$
  - $\Delta t = 2D/C$
  - $D=1m\ \Delta t = 6.6*10^{-09}\ s$
  - $D=0.01m\ \Delta t = 6.6*10^{-11}\ s$
  - $D=0.001m\ \Delta t = 6.6*10^{-12}\ s$ (pico seconds)
  - 3.3 picoseconds for light to travel 1 millimeter



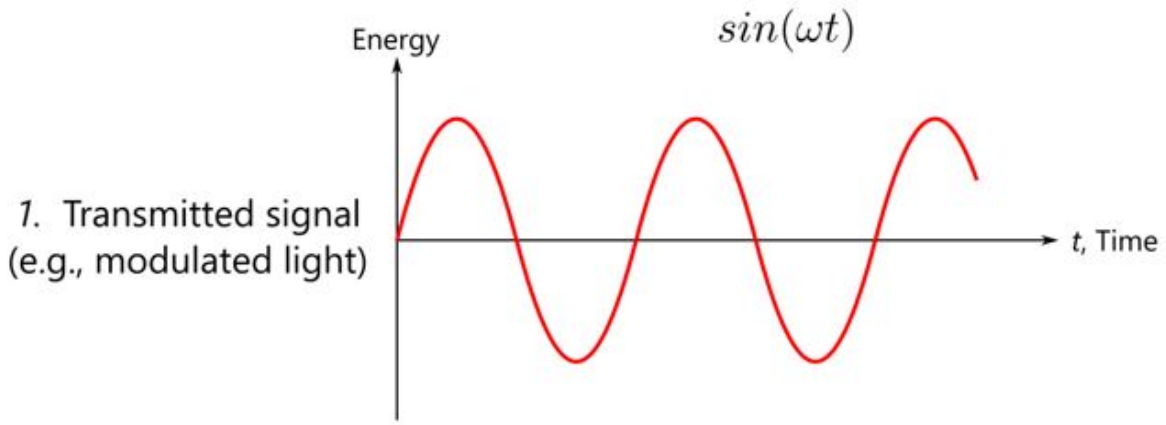
Optical Time-of-Flight (ToF)

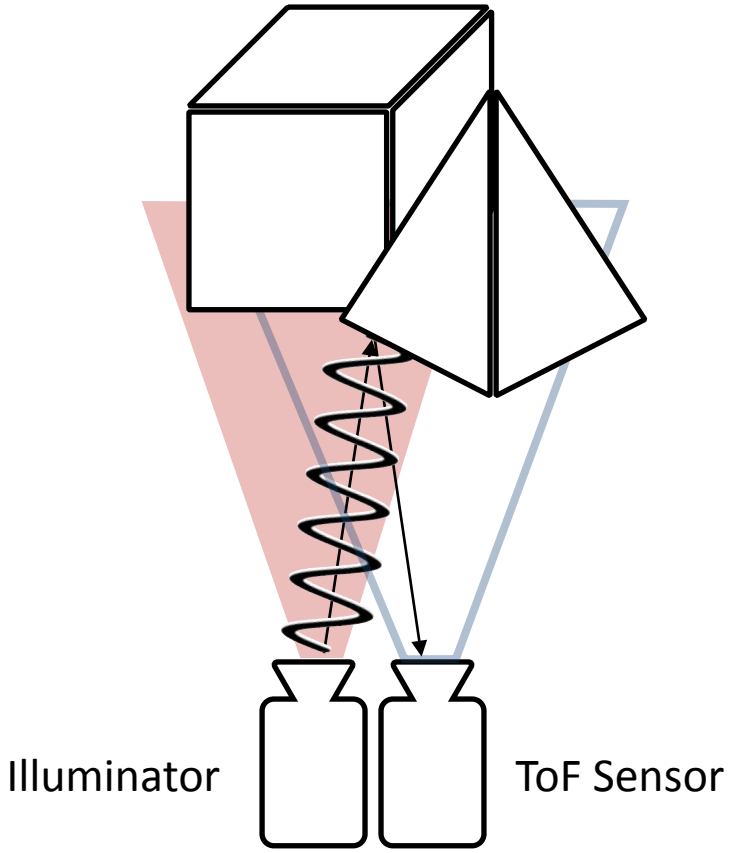
- Practical realisation:
  - Direct Time of Flight
    - Measuring actual round-trip of light pulse
    - Technical limits to measuring time affect accuracy
  - Indirect Time of Flight
    - Indirectly measuring round-trip of light pulse
    - Usually utilising phase information



Indirect Time-of-Flight (ToF)

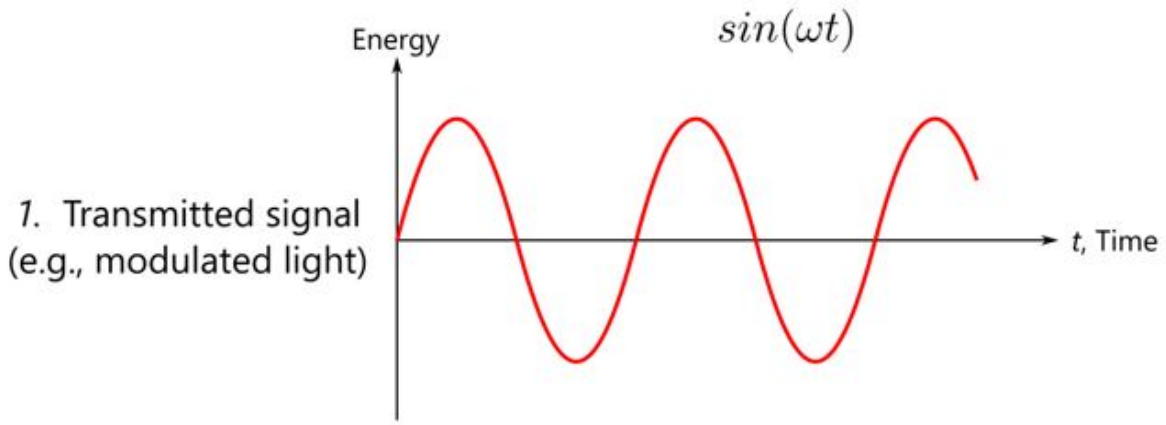
- We emit light with a periodic pattern, then calculate the phase shift in the return measurement.

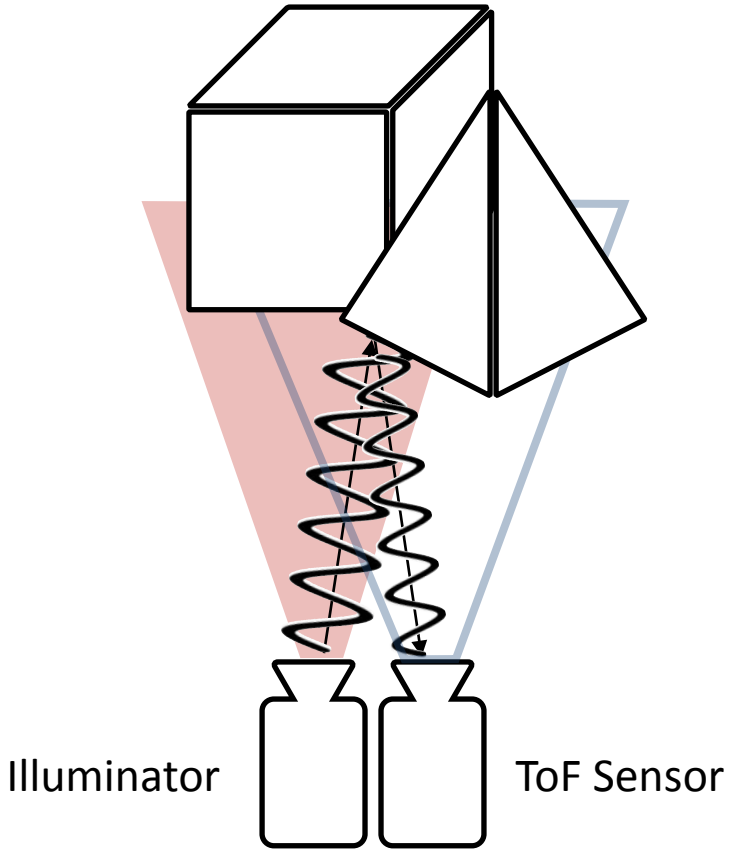




Indirect Time-of-Flight (ToF)

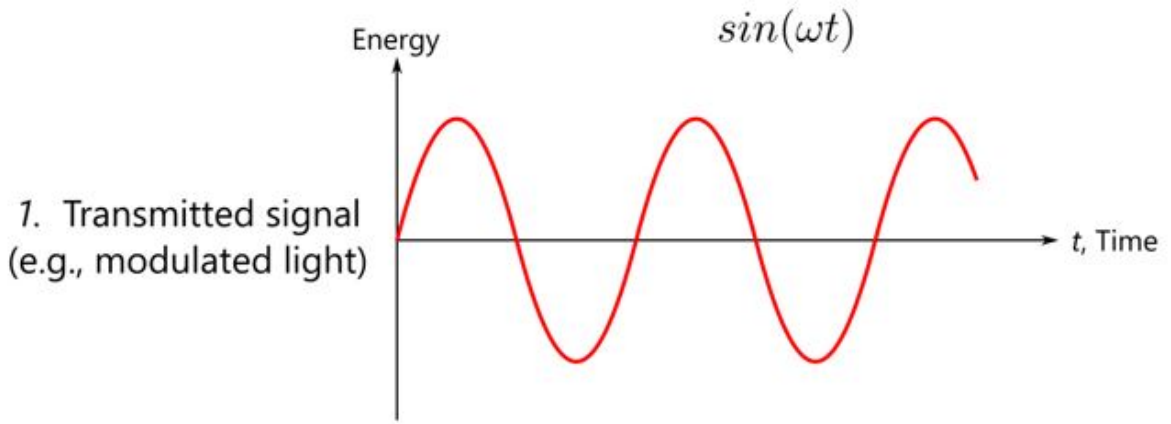
- We emit light with a periodic pattern, then calculate the phase shift in the return measurement.

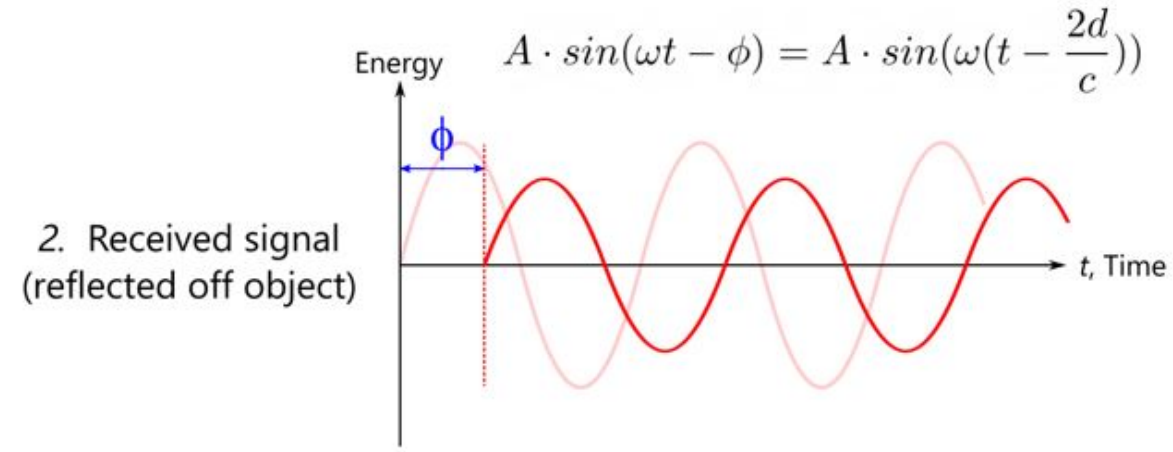


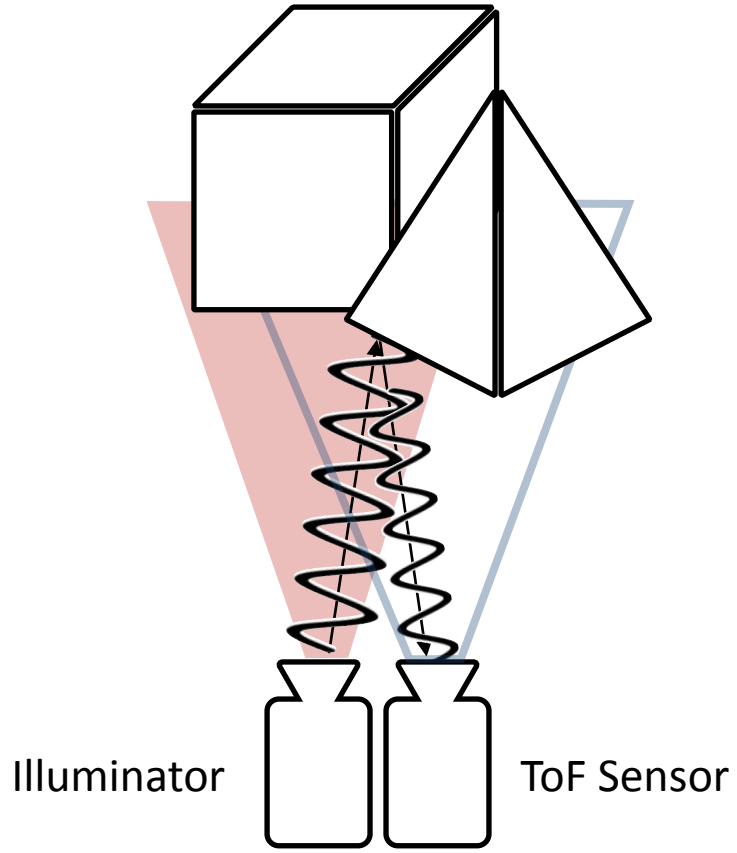


Indirect Time-of-Flight (ToF)

- We emit light with a periodic pattern, then calculate the phase shift in the return measurement.





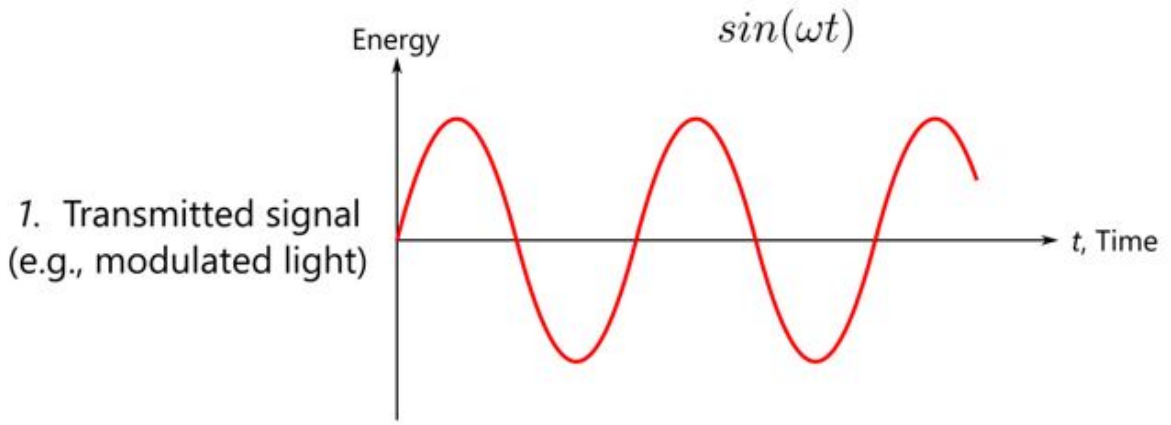


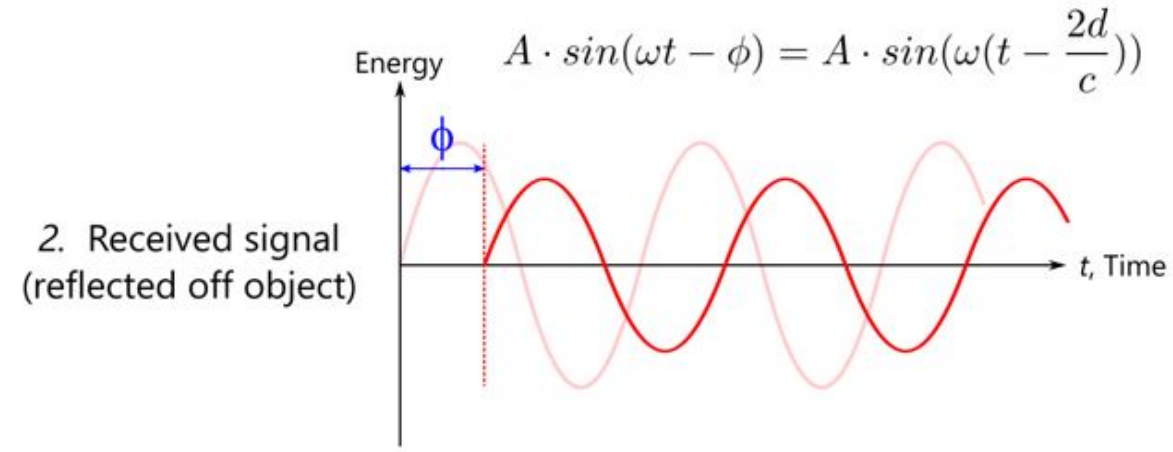
$$d = \frac{c \cdot \phi}{2 \cdot 2\pi f}$$

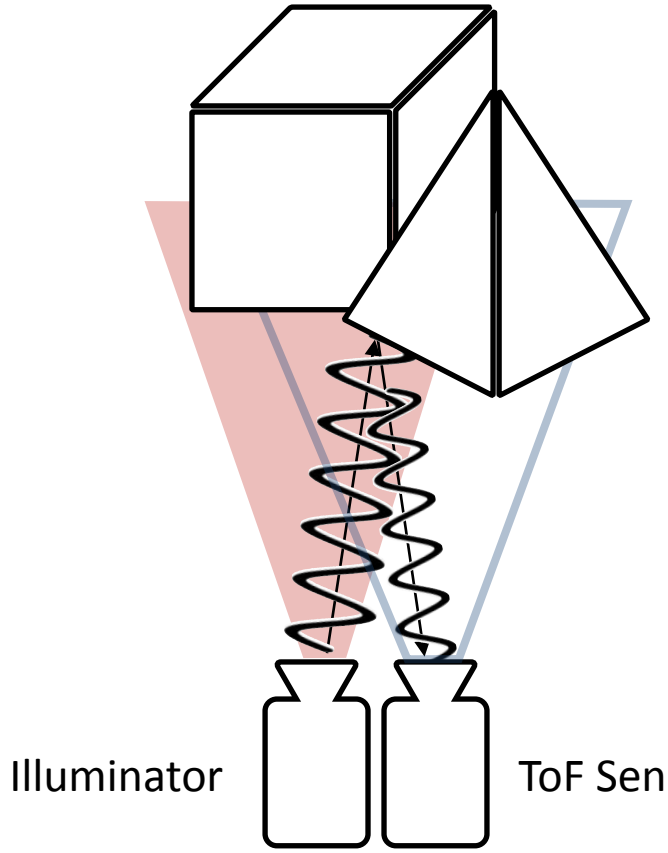
Distance to target · Speed of signal · Phase shift (in radians) · Round trip = 2 legs · Modulation frequency

Indirect Time-of-Flight (ToF)

- We emit light with a periodic pattern, then calculate the phase shift in the return measurement.







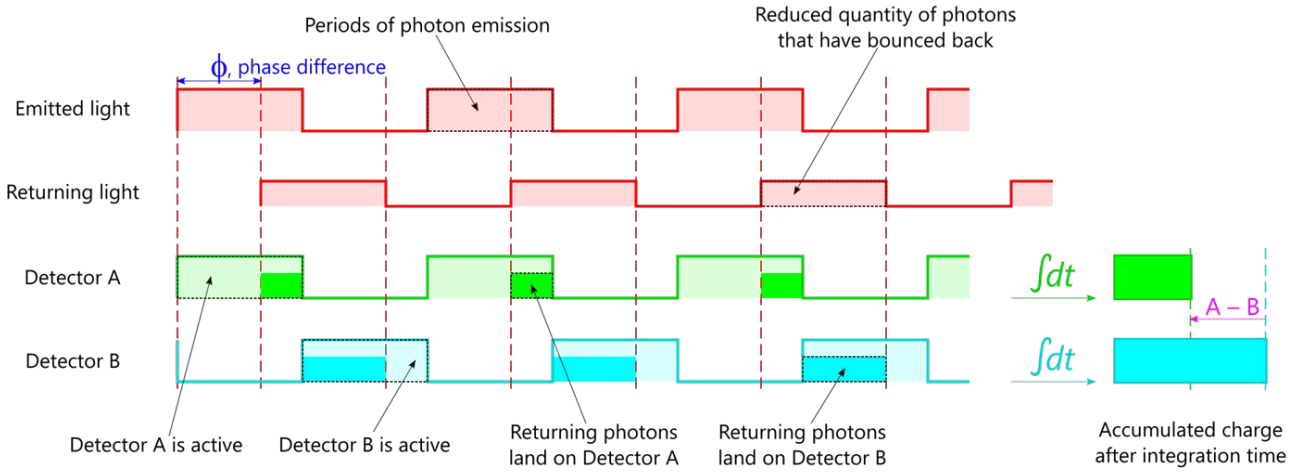
$$d = \frac{c \cdot \phi}{2 \cdot 2\pi f}$$

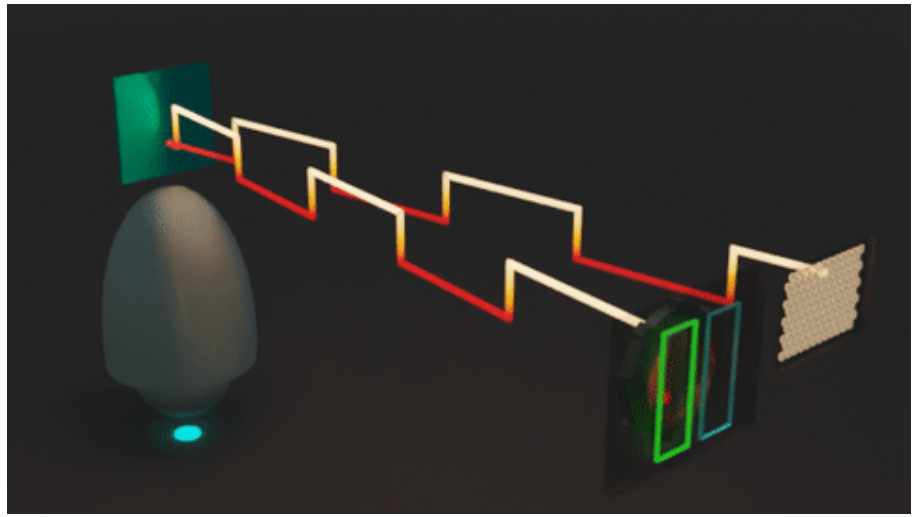
Distance to target; Speed of signal; Phase shift (in radians); Round trip = 2 legs; Modulation frequency

How to obtain?

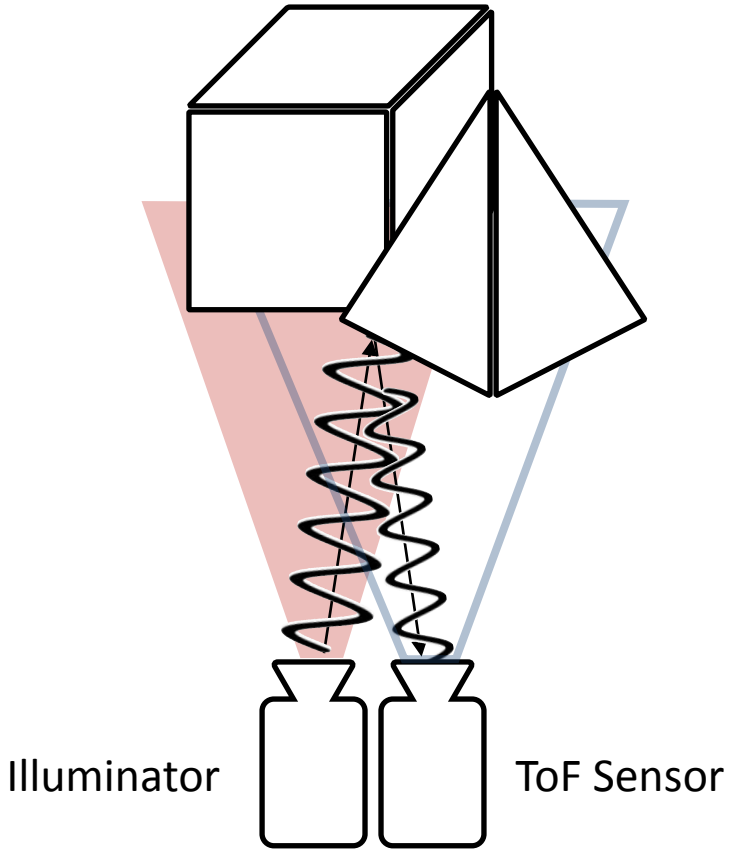
Indirect Time-of-Flight (ToF)

- Two detectors synchronised with the light emitter



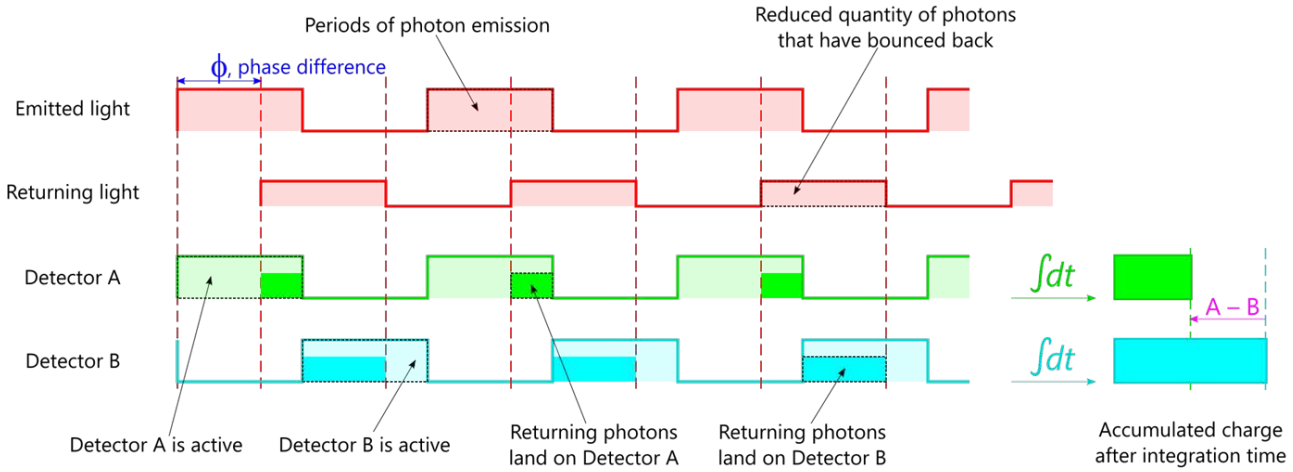


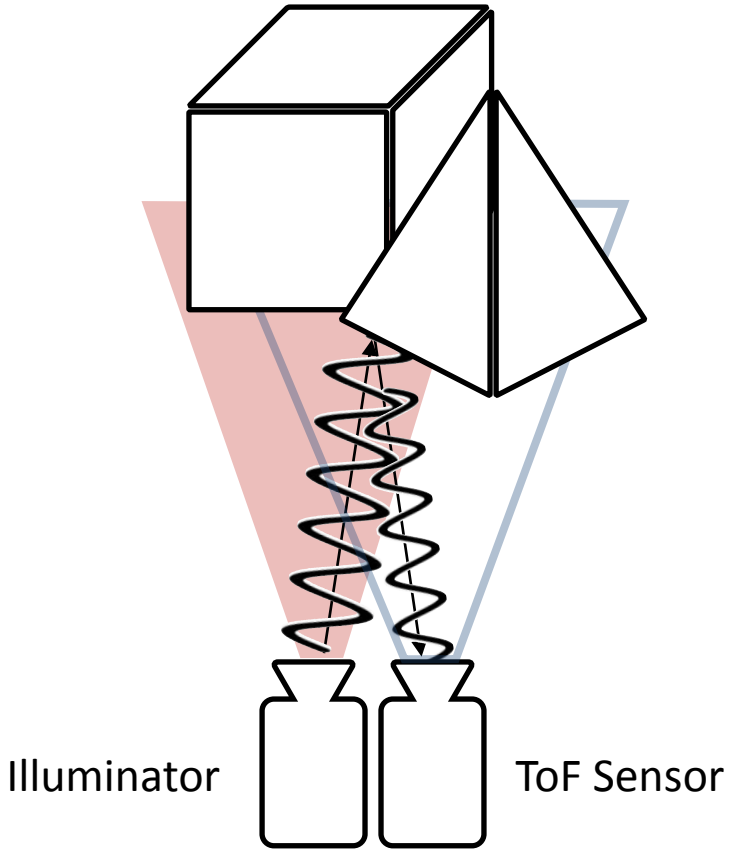
https://devblogs.microsoft.com



Indirect Time-of-Flight (ToF)

- Two detectors synchronised with the light emitter



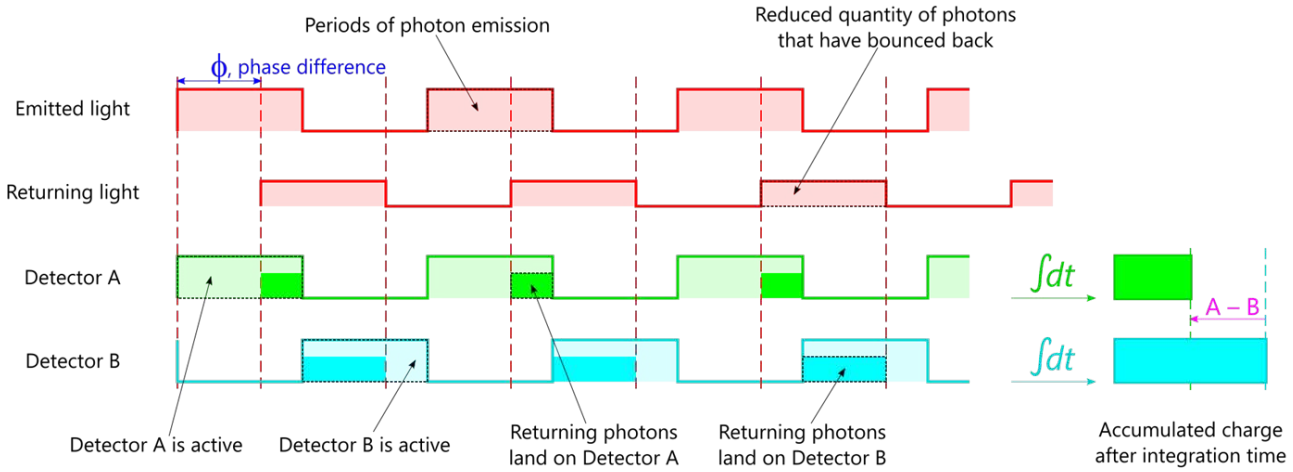


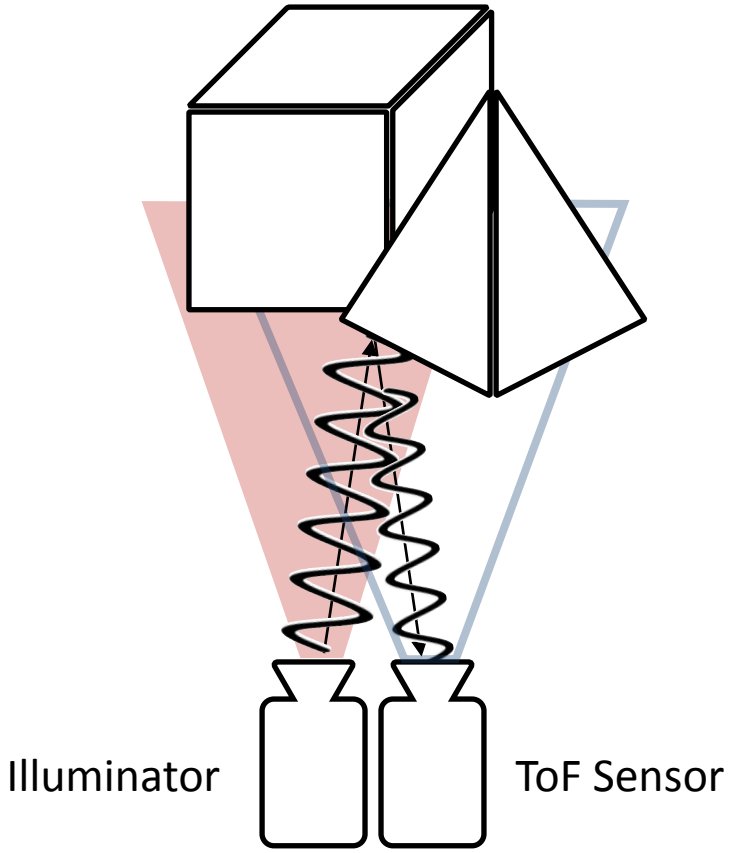
$$d = \frac{c \cdot \phi}{2 \cdot 2\pi f}$$

Distance to target: d
Speed of signal: c
Phase shift (in radians): ϕ
Round trip = 2 legs: 2
Modulation frequency: f

Indirect Time-of-Flight (ToF)

- Two detectors synchronised with the light emitter





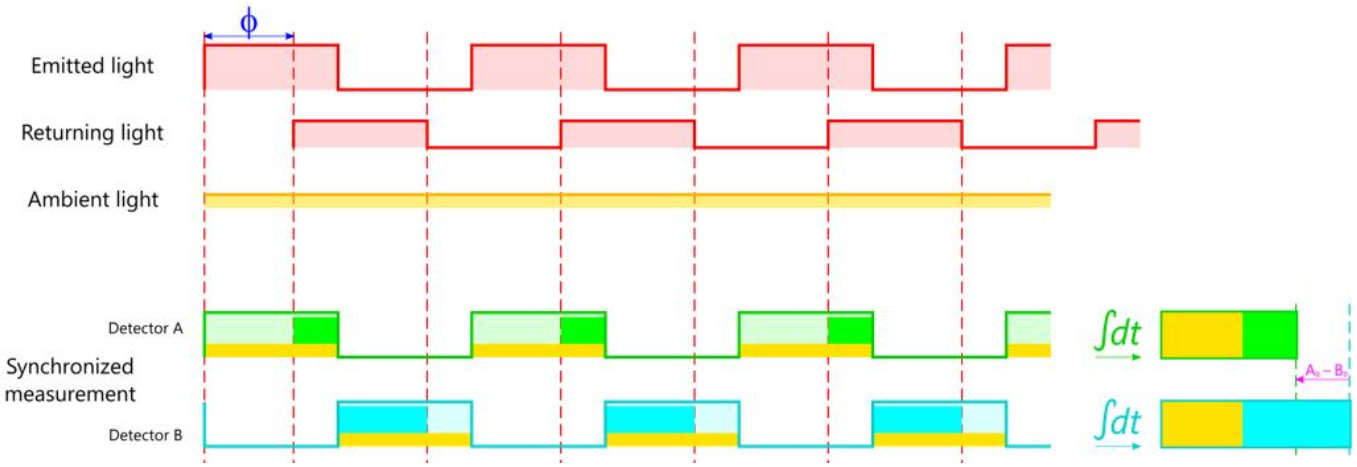
- What can go wrong?
  - Ambient light?
  - Ambiguity?

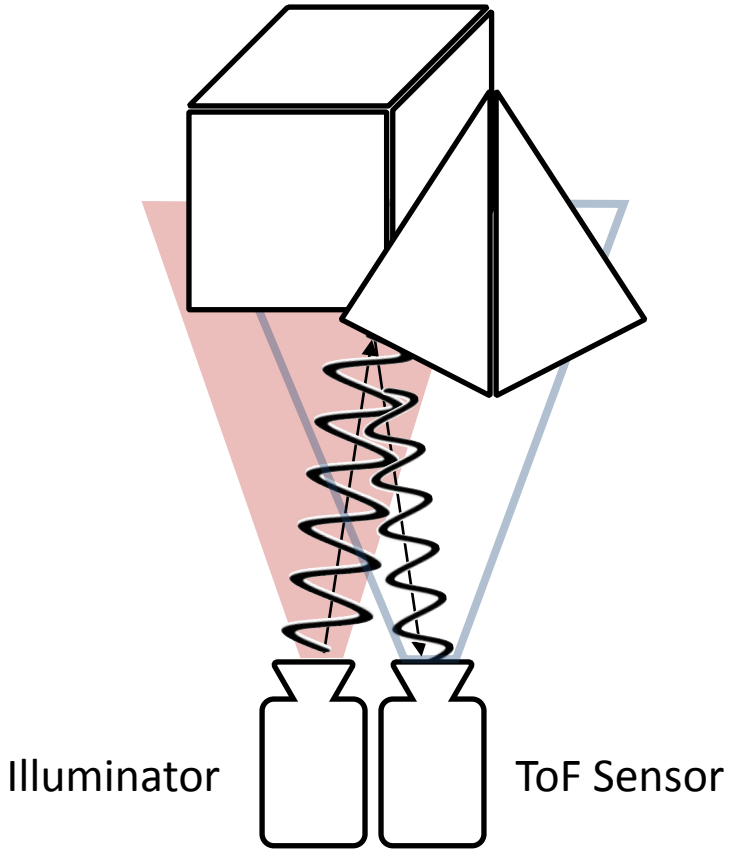
$$d = \frac{c \cdot \phi}{2 \cdot 2\pi f}$$

Distance to target; Speed of signal; Phase shift (in radians); Round trip = 2 legs; Modulation frequency

Indirect Time-of-Flight (ToF)

- Ambient light:
  - Usually not a problem



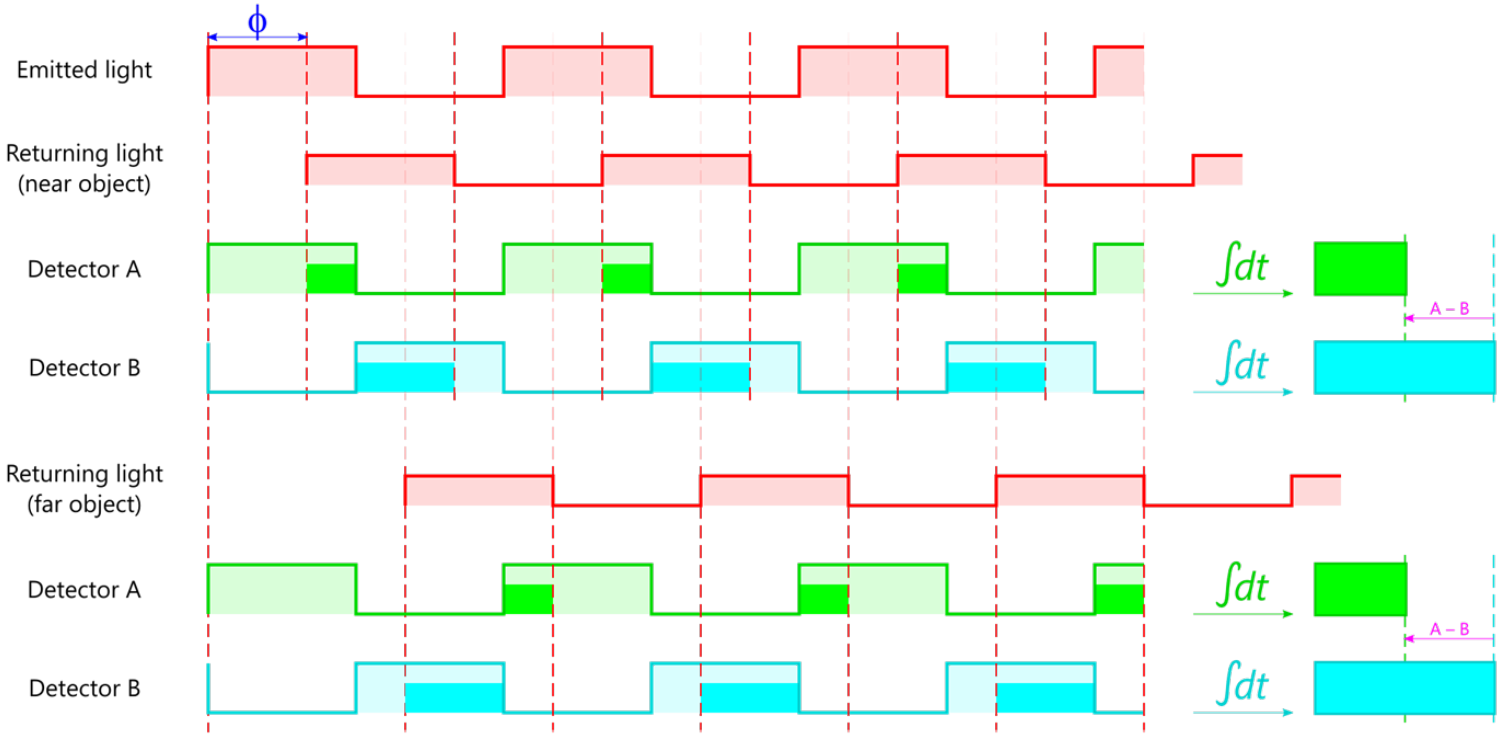


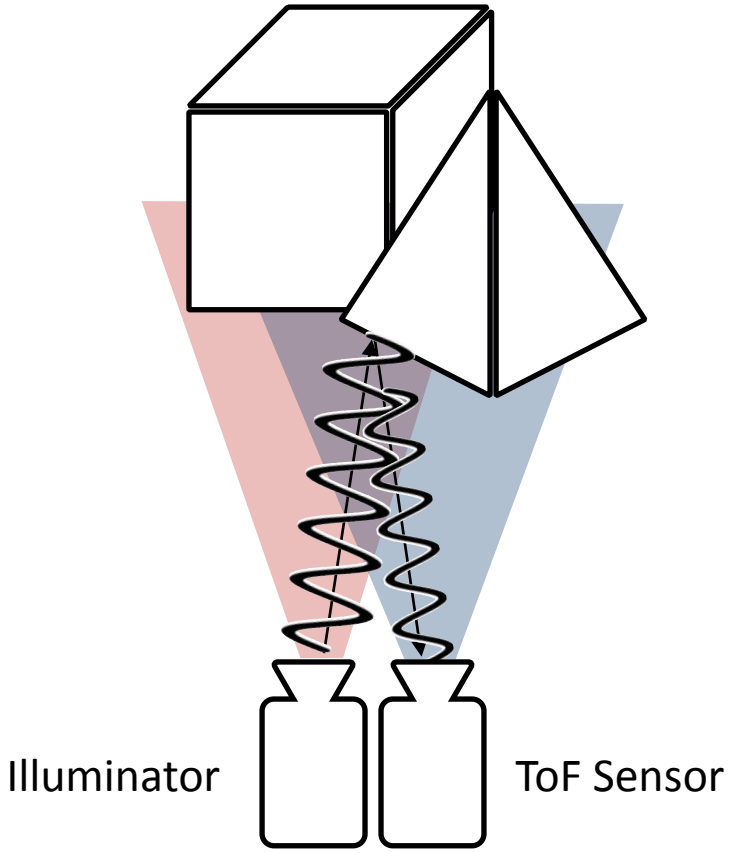
$$d = \frac{c \cdot \phi}{2 \cdot 2\pi f}$$

Distance to target: d
Speed of signal: c
Phase shift (in radians): ϕ
Round trip = 2 legs: 2
Modulation frequency: f

Indirect Time-of-Flight (ToF)

- Ambiguity
  - Usually a problem (e.g. object with another distance creating same phase measurement)





$$d = \frac{c \cdot \phi}{2 \cdot 2\pi f}$$

Distance to target; Speed of signal; Phase shift (in radians); Round trip = 2 legs; Modulation frequency

Active range scanner / depth cameras work in infrared spectrum

We cannot perceive the infrared spectrum…

Active range scanner / depth cameras work in infrared spectrum

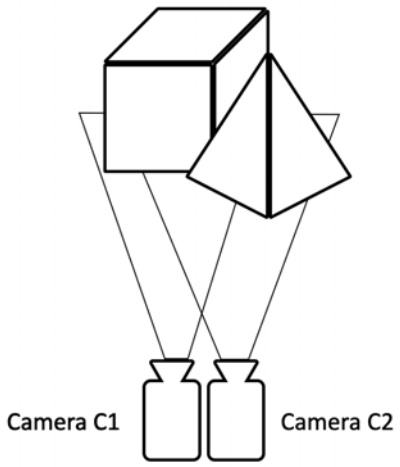
We cannot perceive the infrared spectrum…

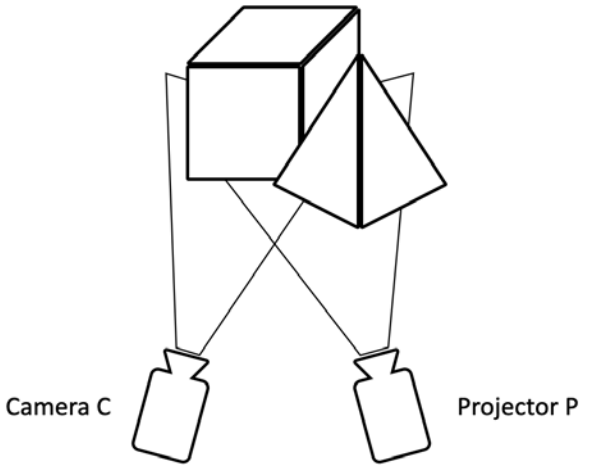


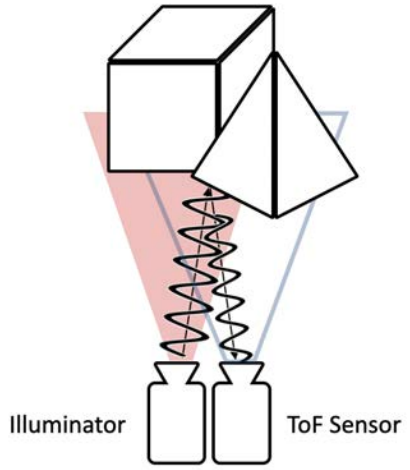
But they can…

Summary Range Scanning

Considerations	Stereo Vision (Passive)	Structured Light (Active)	Time of Flight (Active)
Response time / Speed			
Depth Accuracy			
Low light performance			
Bright light performance			

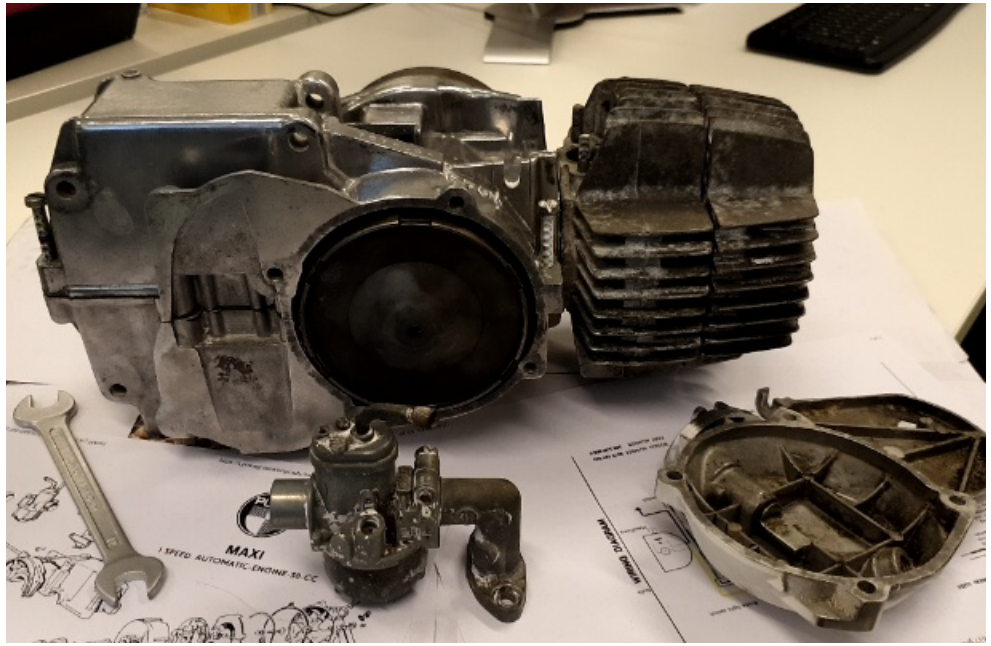


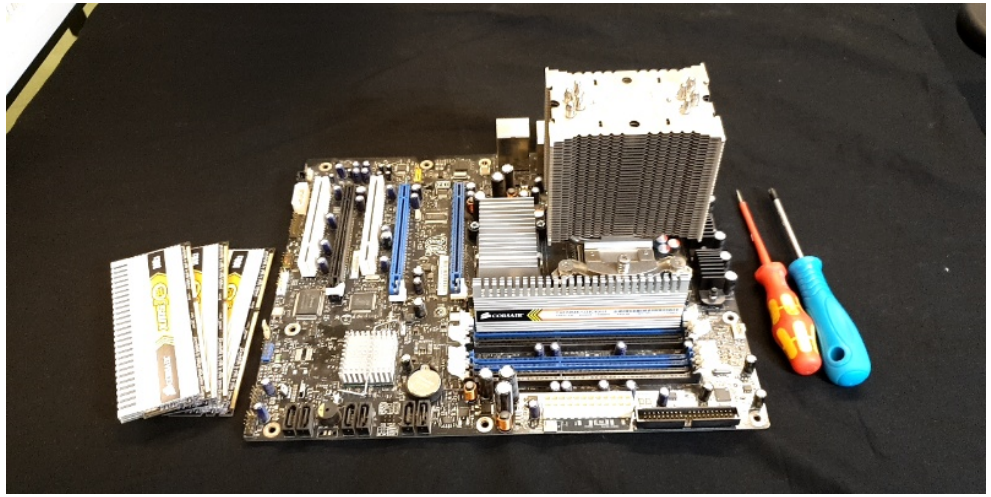


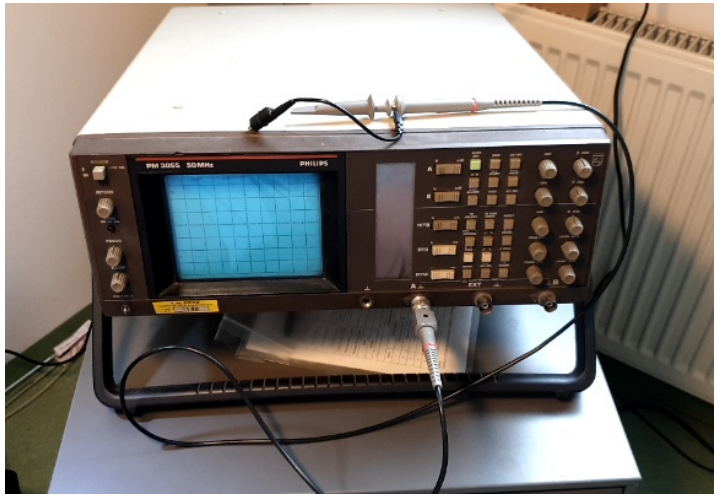


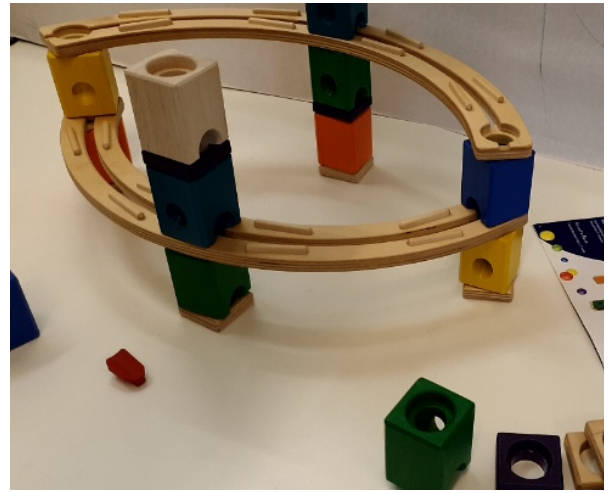
Summary Range Scanning

Any problems when scanning these objects?









No Depth Cam?

Depth from Mono / Depth Estimation using ML

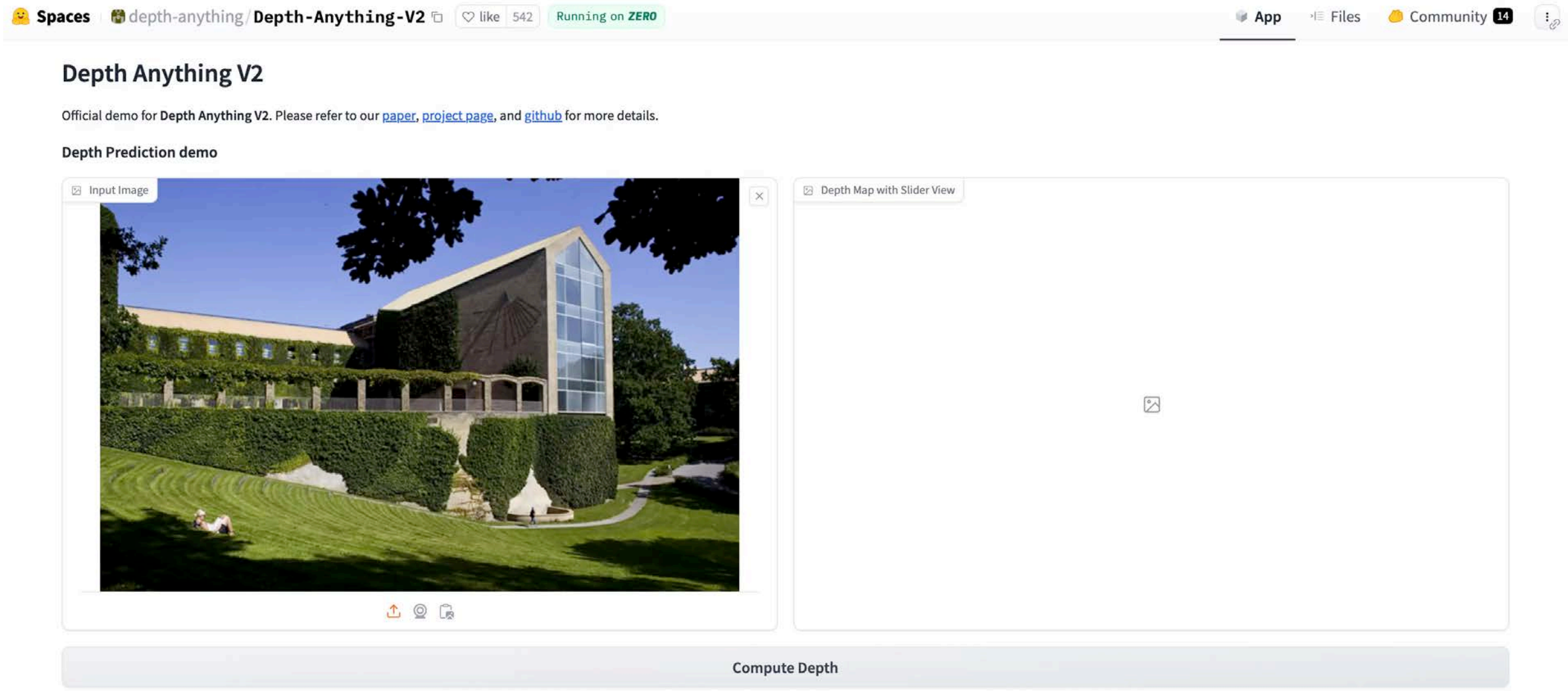
- Convolutional neural networks (CNNs) have been applied to monocular depth estimation
- Common architectures include U-Net, ResNet, and DenseNet
- During training:
  - Network is given pairs of images and their corresponding ground-truth depth maps
  - Minimize the difference between predicted depth and ground truth depth

https://github.com/yuhsuanyeh/BiFuse

Depth from Mono: Depth Anything V2

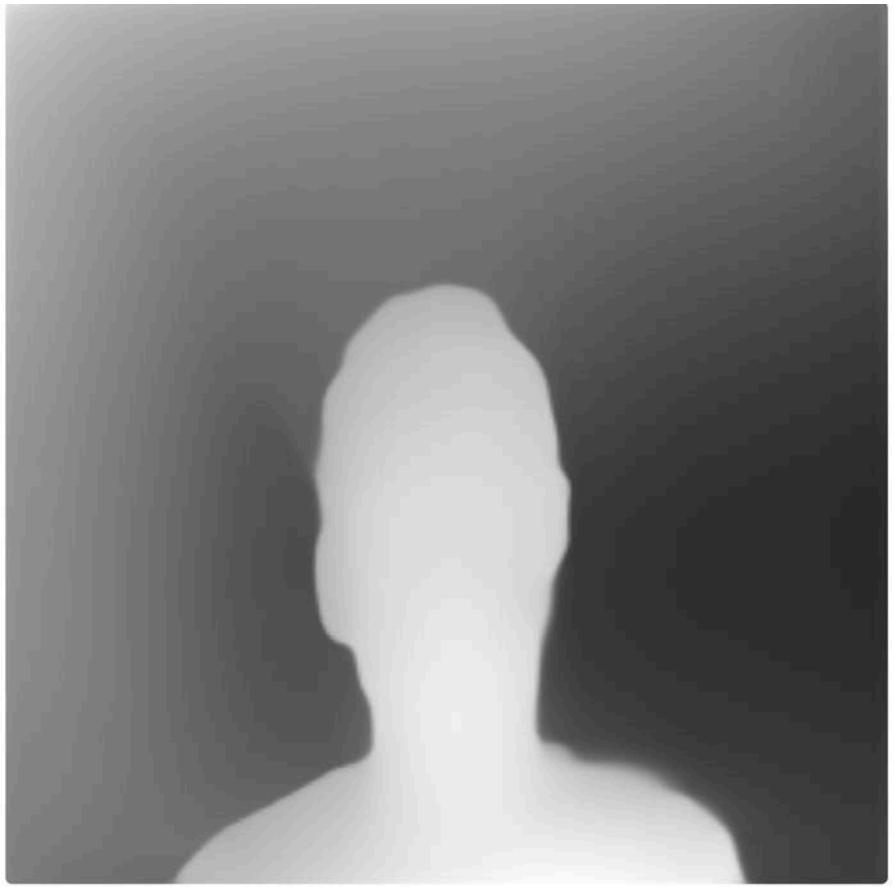
https://huggingface.co/spaces/depth-anything/Depth-Anything-V2

Depth from Mono: Depth Anything V2



https://huggingface.co/spaces/depth-anything/Depth-Anything-V2

Depth from Mono: Tensorflow JS



Average Depth: 126.56

https://kuromadara.github.io/live-depth-prediction-using-tfjs/

Reminder: Depth from Mono / Depth Estimation using ML

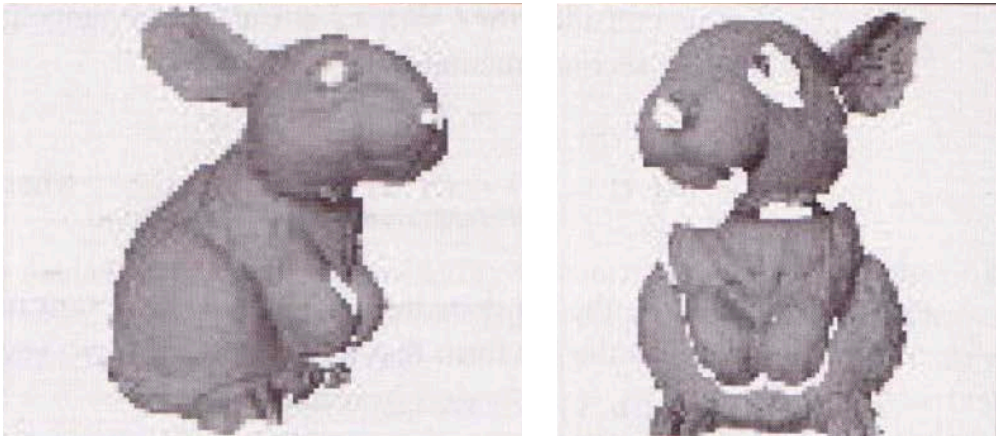
- Convolutional neural networks (CNNs) have been applied to monocular depth estimation
- Common architectures include U-Net, ResNet, and DenseNet
- During training:
  - Network is given pairs of images and their corresponding ground-truth depth maps
  - Minimize the difference between predicted depth and ground truth depth

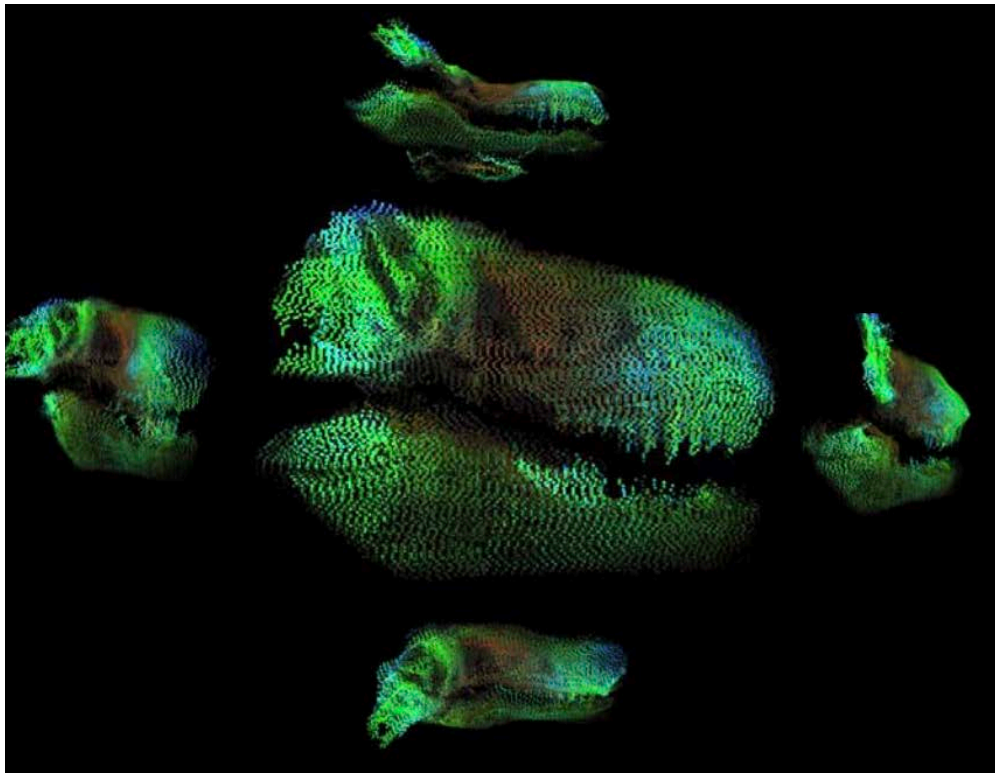
https://github.com/yuhsuanyeh/BiFuse

Registration of Depth Maps

Registration of Depth Maps

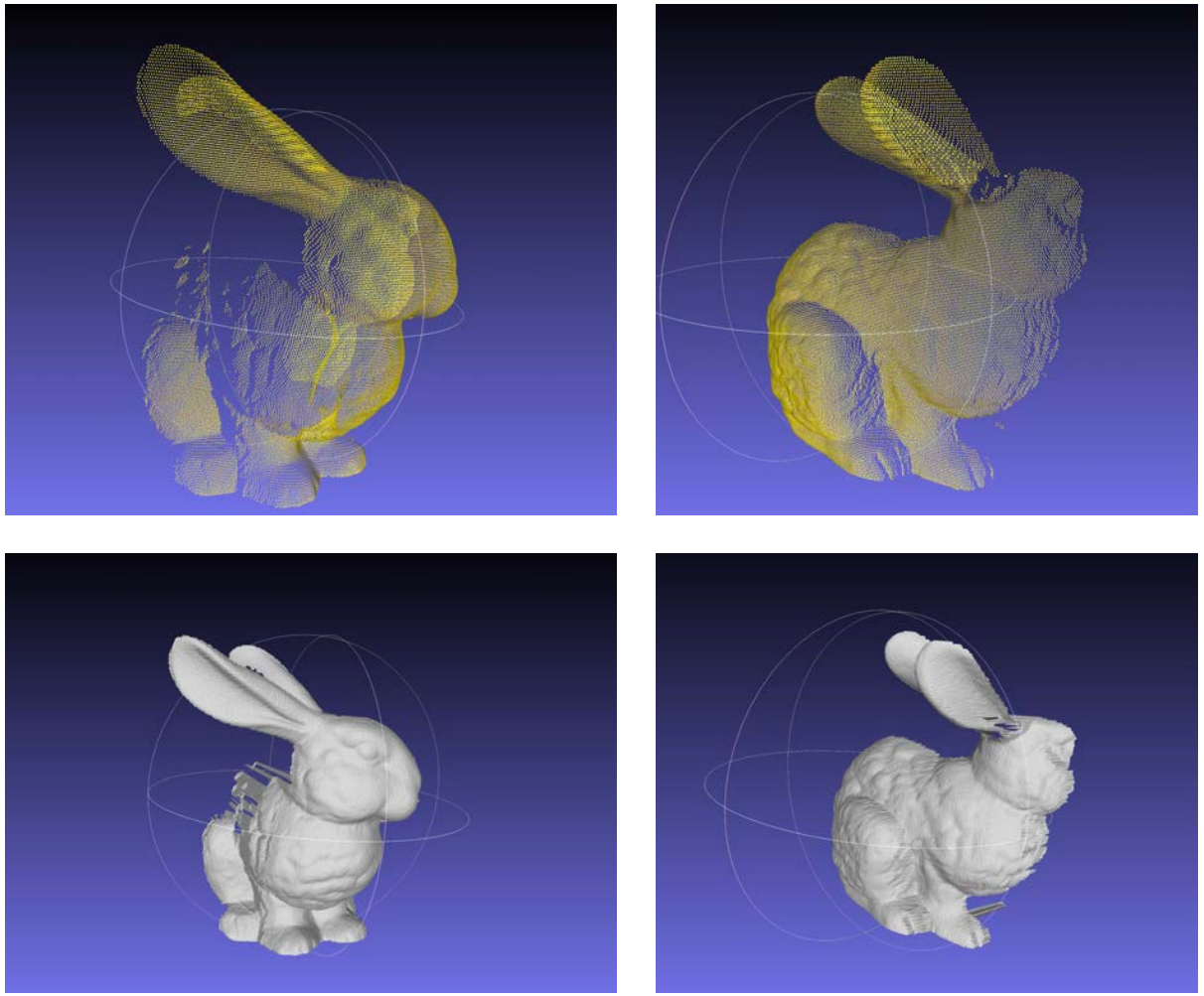
- If correspondences are known, the depth (distance from camera / projector to scene point) can be estimated for every pixel (depth map)
- Having multiple depth maps of the same scene, how can they be merged?
- This process is known as registration





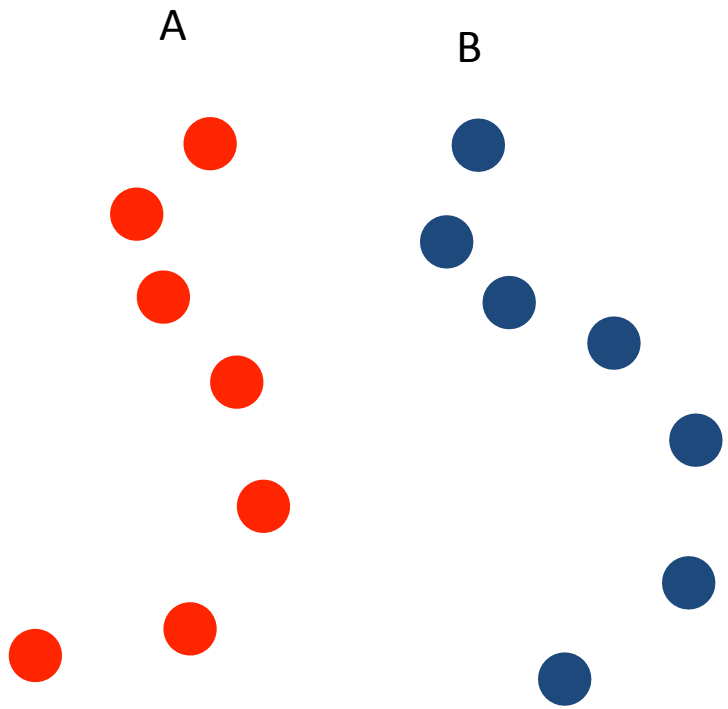
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B’)
  - Compute average distance E between A and B’
  - Repeat until E is minimal



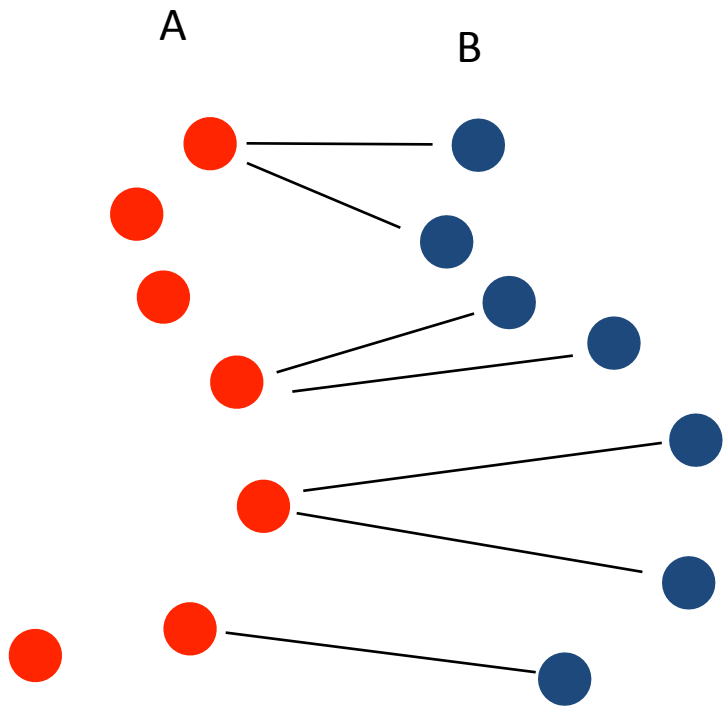
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B’)
  - Compute average distance E between A and B’
  - Repeat until E is minimal



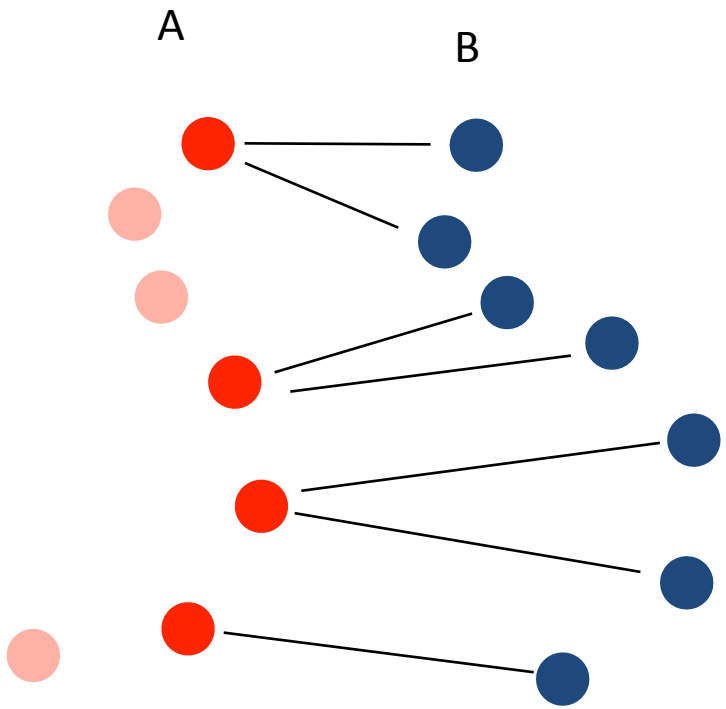
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B')
  - Compute average distance E between A and B'
  - Repeat until E is minimal



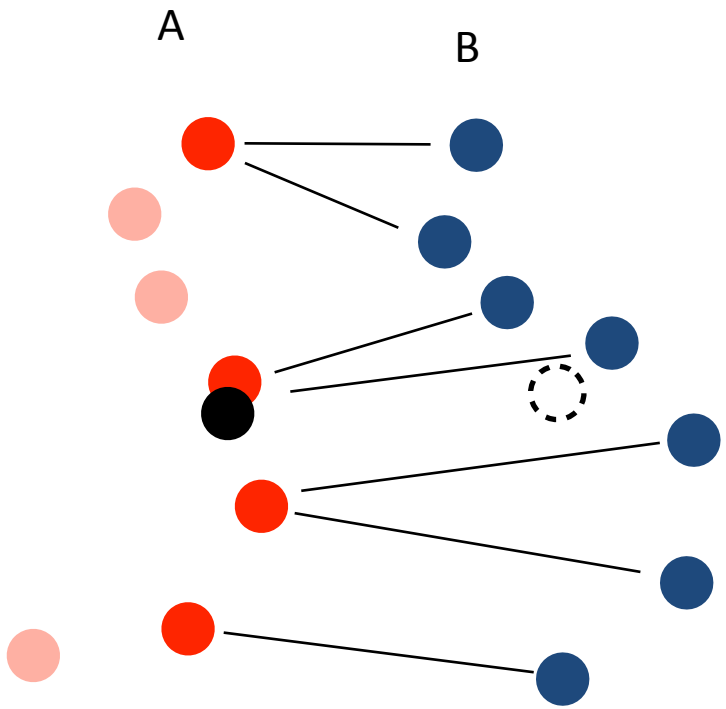
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B’)
  - Compute average distance E between A and B’
  - Repeat until E is minimal



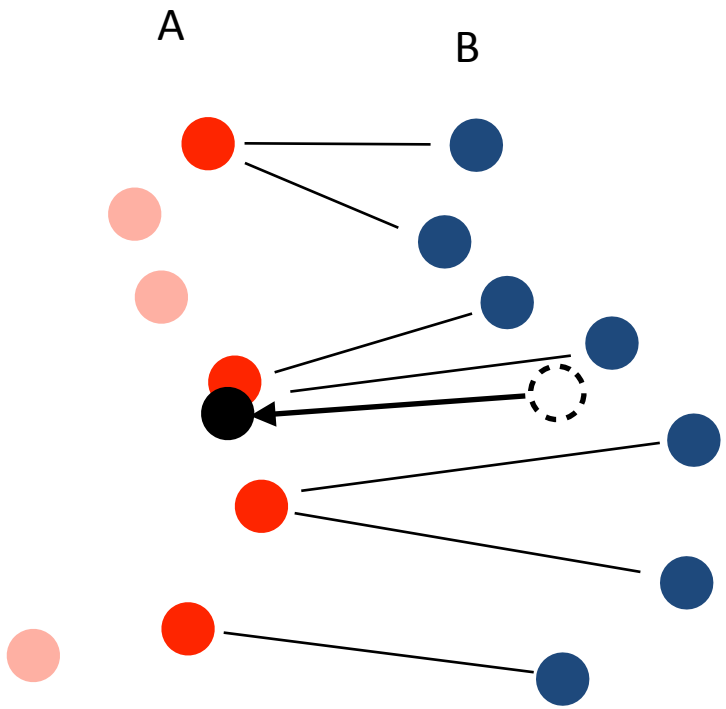
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B’)
  - Compute average distance E between A and B’
  - Repeat until E is minimal



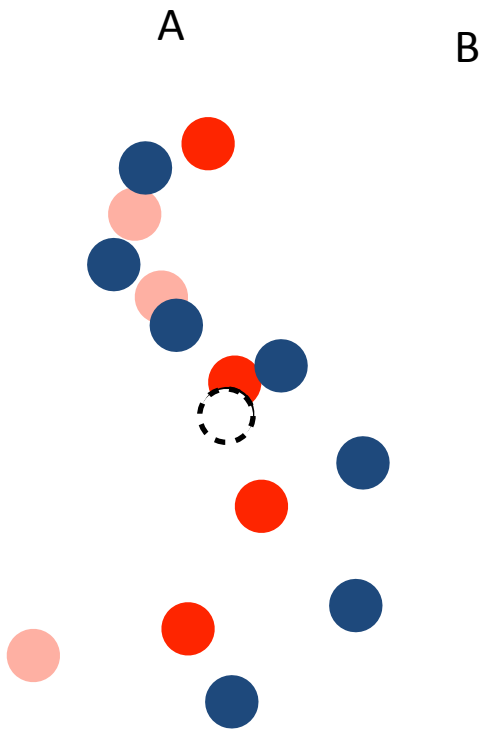
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B')
  - Compute average distance E between A and B'
  - Repeat until E is minimal



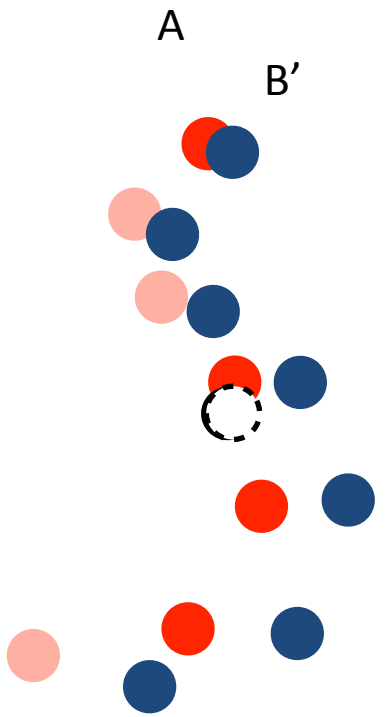
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B’)
  - Compute average distance E between A and B’
  - Repeat until E is minimal



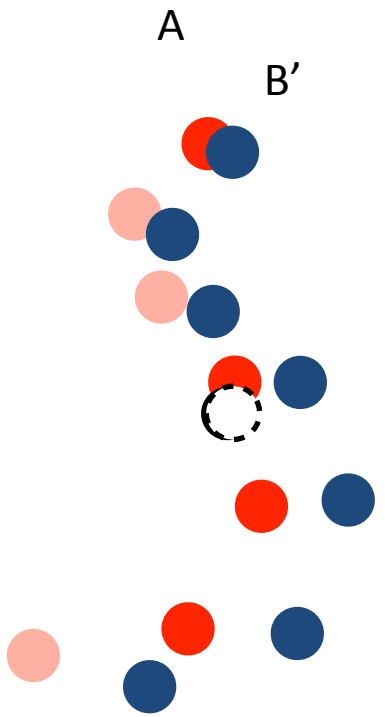
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B’)
  - Compute average distance E between A and B’
  - Repeat until E is minimal



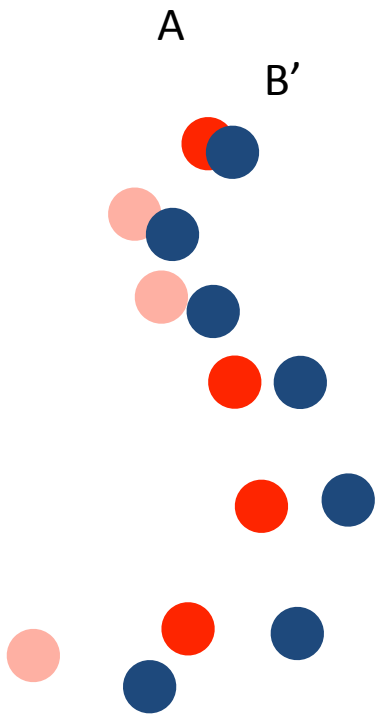
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B’)
  - Compute average distance E between A and B’
  - Repeat until E is minimal



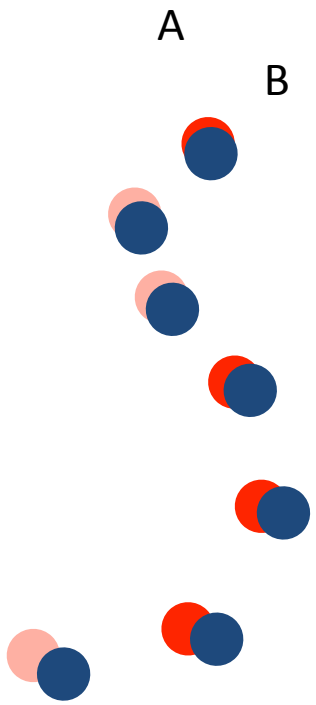
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B’)
  - Compute average distance E between A and B’
  - Repeat until E is minimal



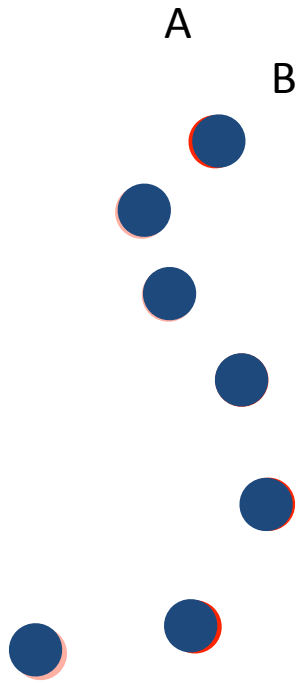
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B’)
  - Compute average distance E between A and B’
  - Repeat until E is minimal



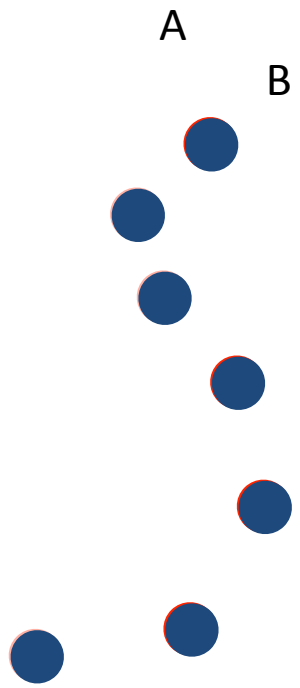
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B’)
  - Compute average distance E between A and B’
  - Repeat until E is minimal



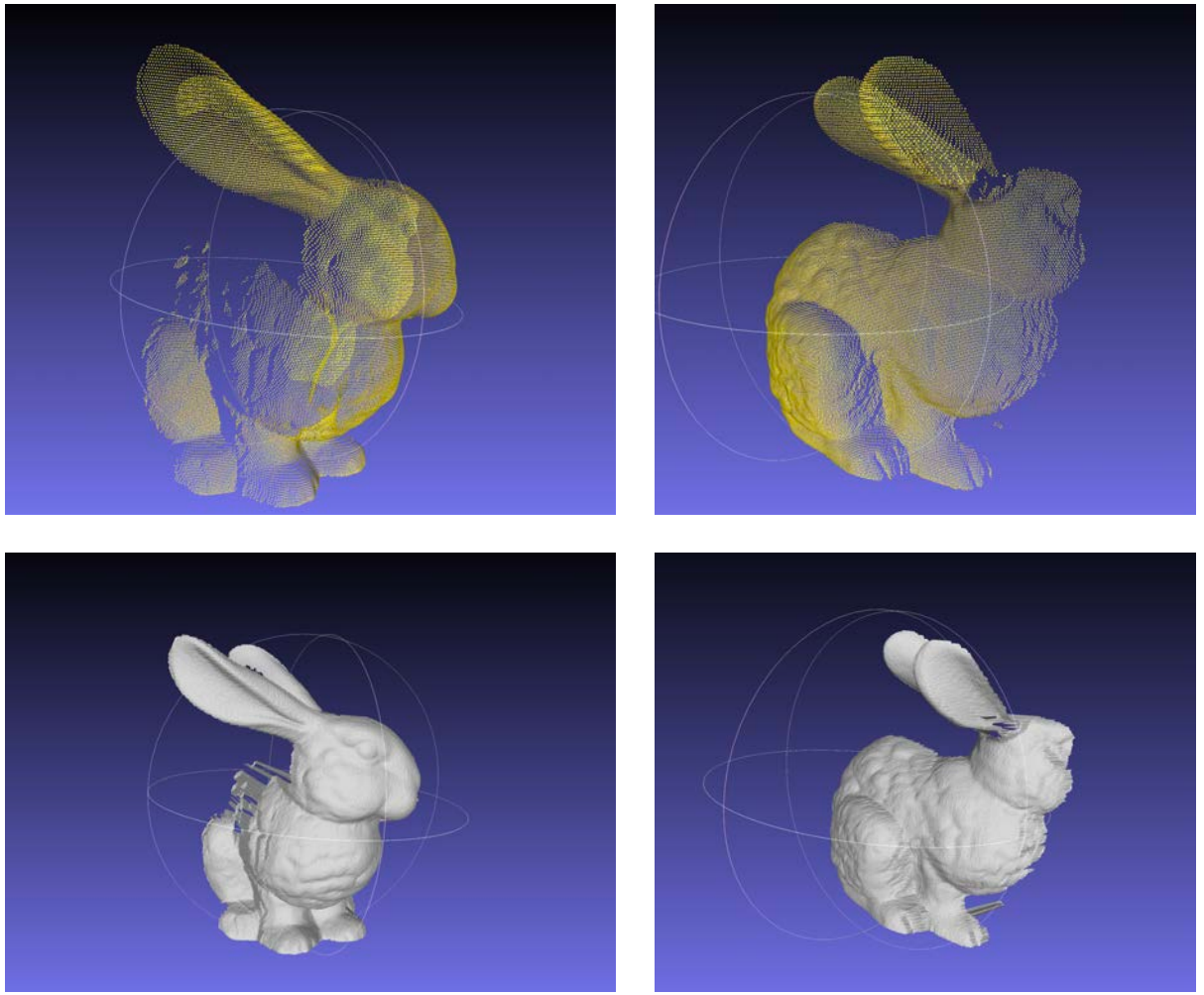
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B’)
  - Compute average distance E between A and B’
  - Repeat until E is minimal



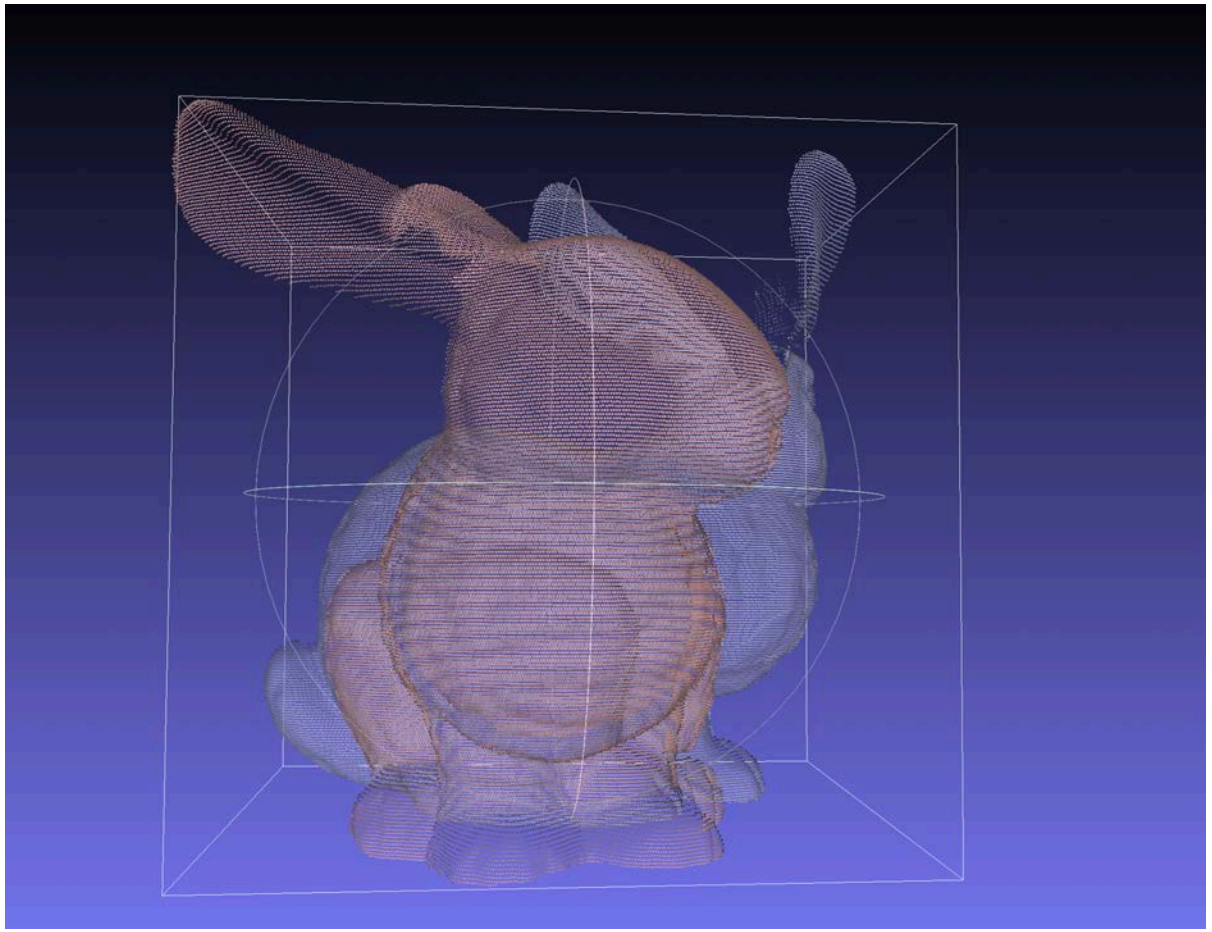
Iterative Closest-Point Method

- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B’)
  - Compute average distance E between A and B’
  - Repeat until E is minimal



Iterative Closest-Point Method

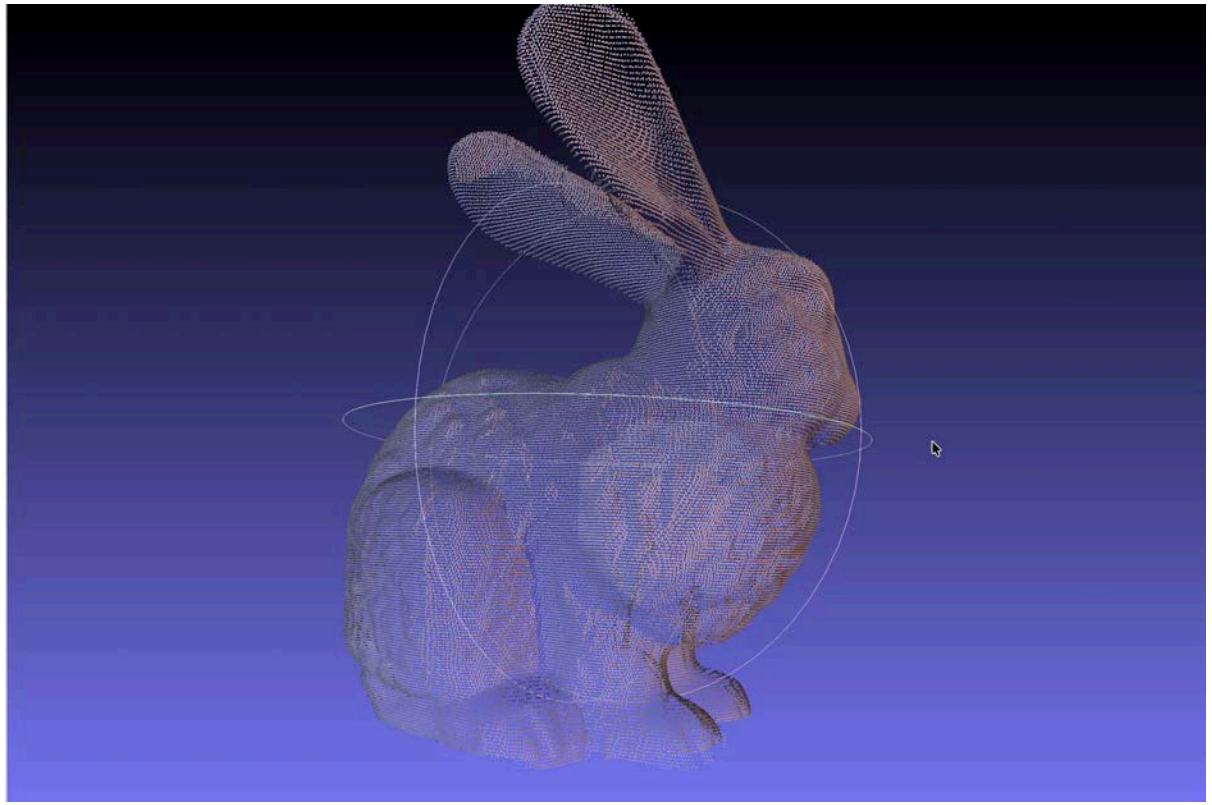
- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B')
  - Compute average distance E between A and B'
  - Repeat until E is minimal



ICP in Meshlab

Iterative Closest-Point Method

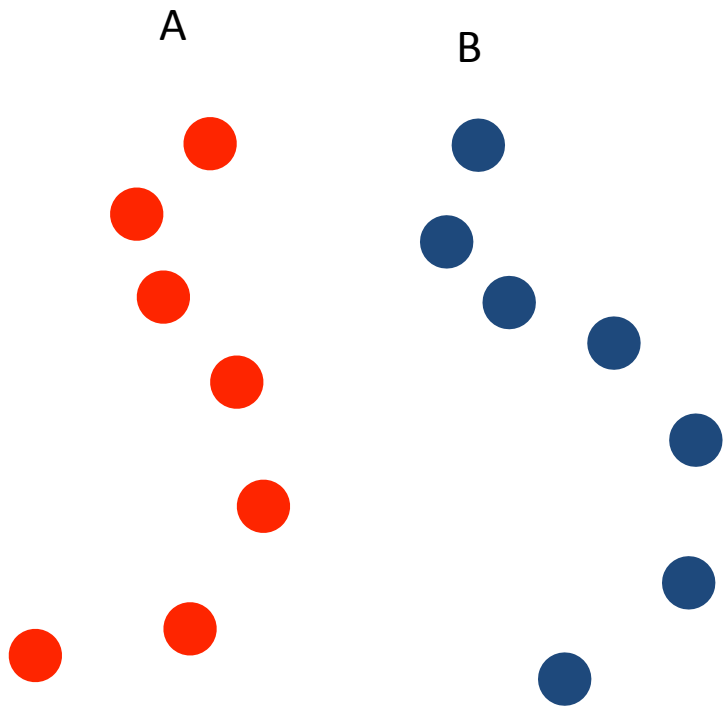
- Registration of two sets of three-dimensional points requires the computation of a rigid transformation that maps the first point set (B) to the second one (A)
- Iteratively minimise the average distance between the two point sets:
  - Find correspondences between the sets by matching every point in B with the point closest to it in A
  - Compute rigid transformation that maps B to A and apply it to B (B')
  - Compute average distance E between A and B'
  - Repeat until E is minimal



ICP in Meshlab

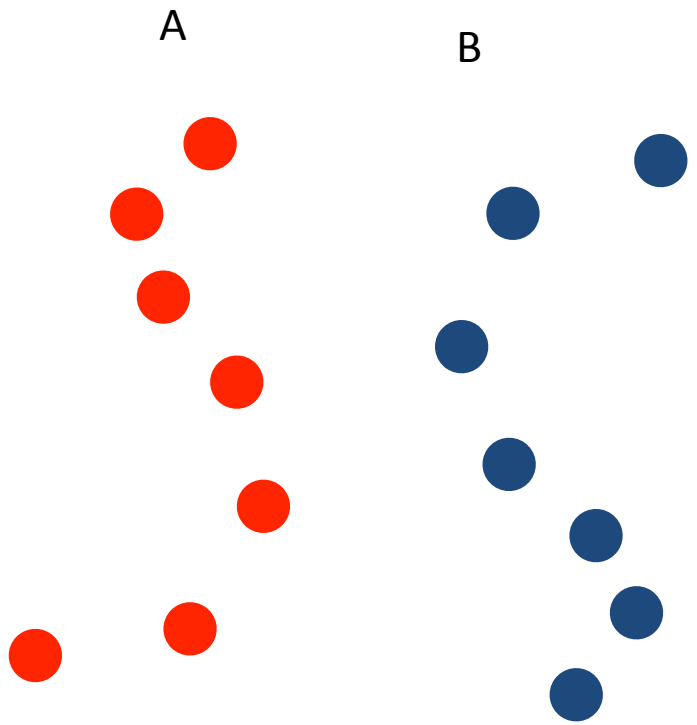
Iterative Closest-Point Method

- How to find a reasonable initial guess?



Iterative Closest-Point Method

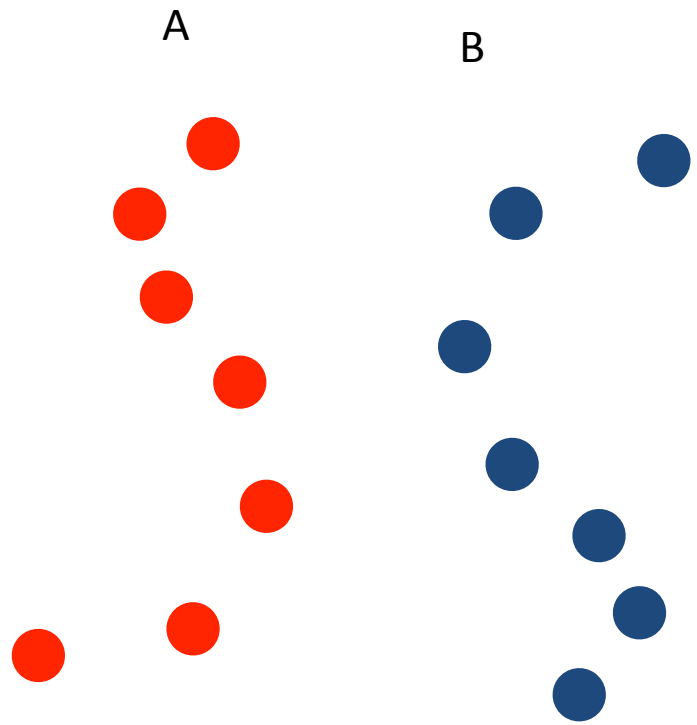
- How to find a reasonable initial guess?



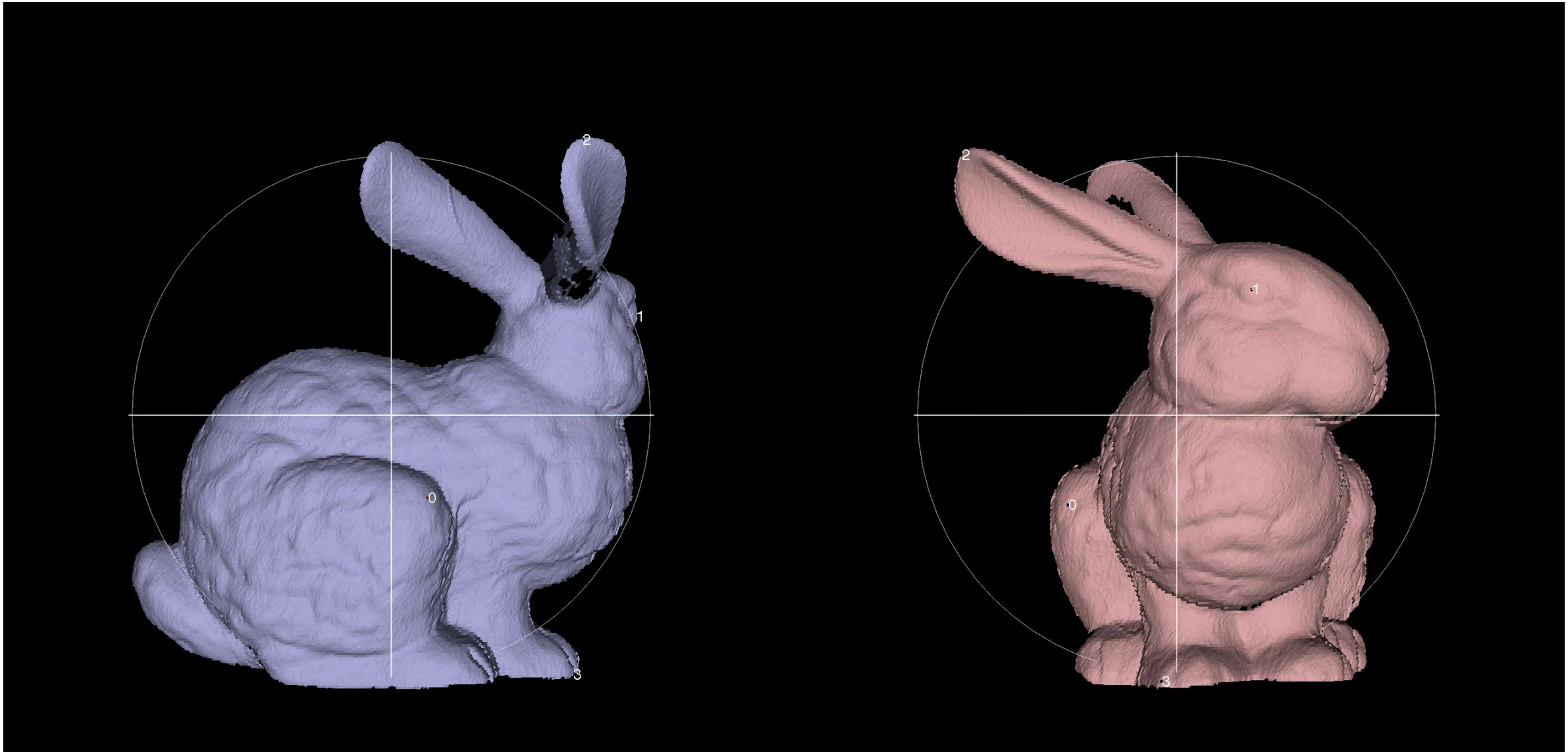
Iterative Closest-Point Method

- How to find a reasonable initial guess?
  - Manually
  - Matching distinctive features between clouds
  - Using sensor data
  - Global registration algorithms (algorithms not needing initial guess)

- At every iteration finding the closest point M in A to a given point S in B takes (for n points) how long?
  - O(n2) brute-force
- Can this be done faster?
  - O(log n) for nearest-neighbour matches with k-d trees



Iterative Closest-Point Method

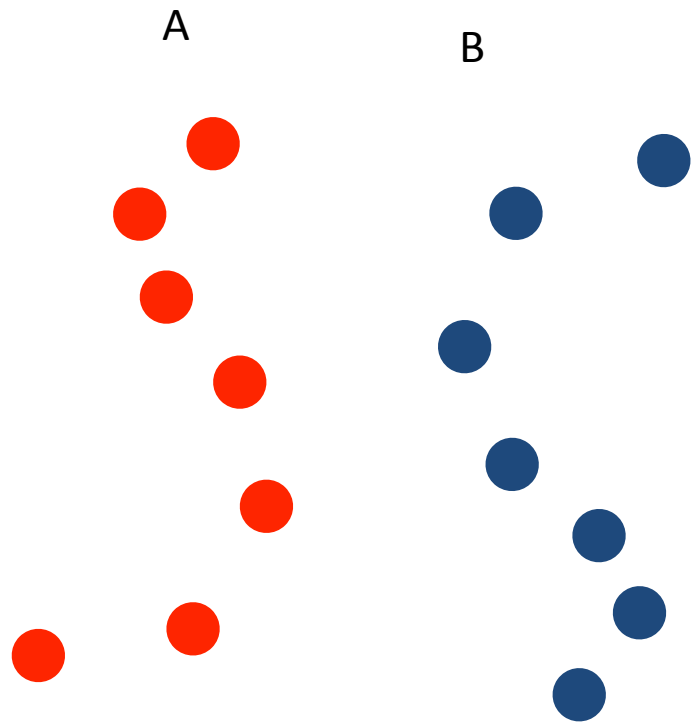


Manual alignment in Meshlab to initialise ICP

Iterative Closest-Point Method

- How to find a reasonable initial guess?
  - Manually
  - Matching distinctive features between clouds
  - Using sensor data
  - Global registration algorithms (algorithms not needing initial guess)

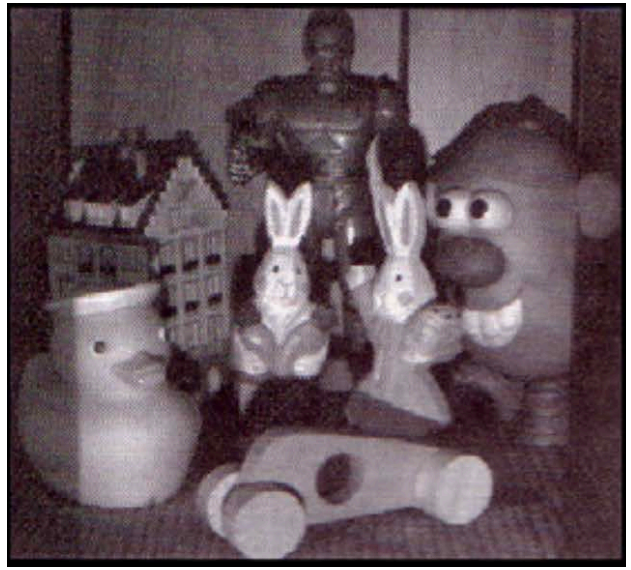
- At every iteration finding the closest point M in A to a given point S in B takes (for n points) how long?
  - O(n2) brute-force
- Can this be done faster?
  - O(log n) for nearest-neighbour matches with k-d trees

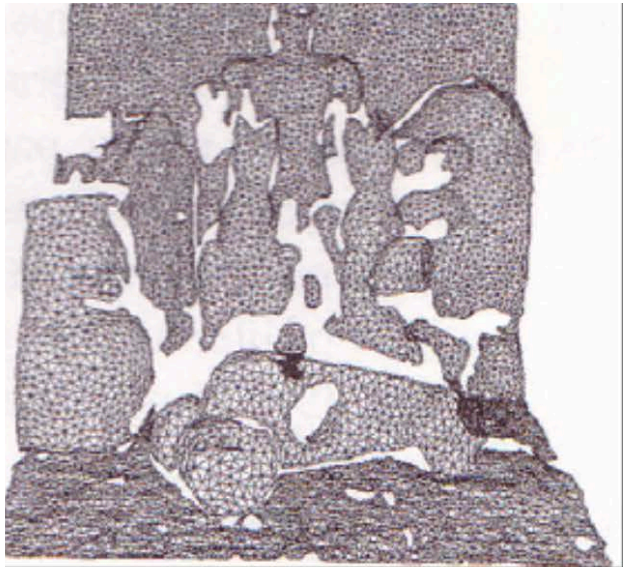


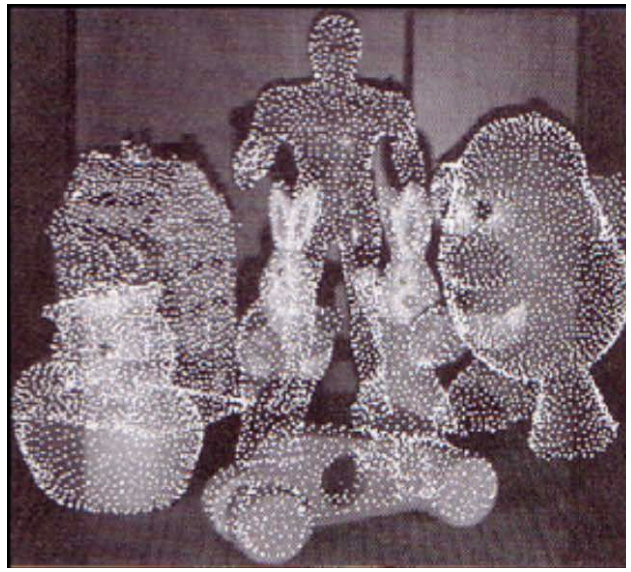
Processing Depth Maps

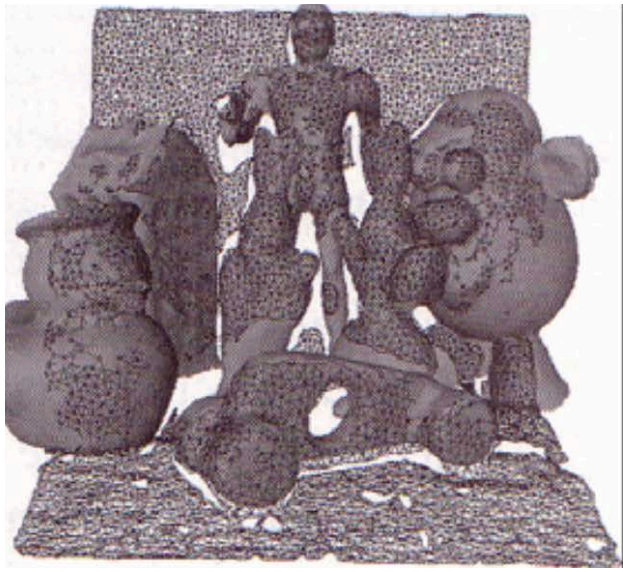
Processing Depth Maps

- Depth maps can be processed like images
- Image operators, such as filters, can be applied to:
  - Handle discontinuities and overlaps
  - Fill missing portions
  - Smoothen geometric noise
  - Analyse depth images
    - Find features such as edges
    - Segmentation of objects
    - Etc.









The end!