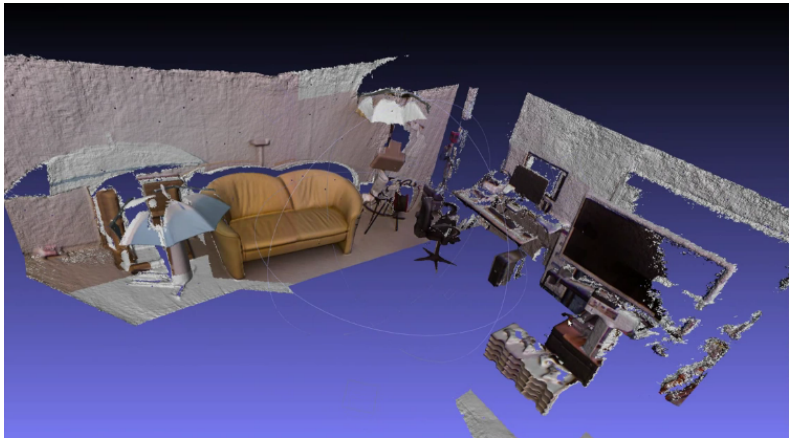
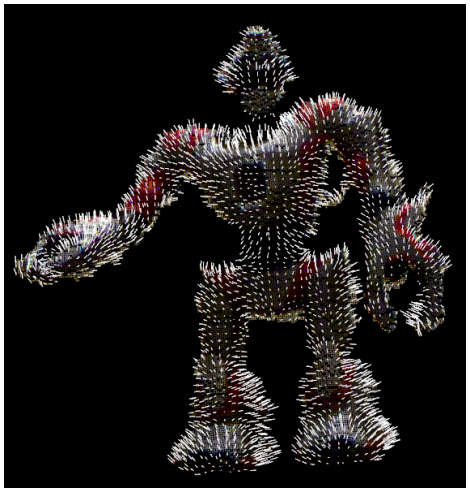
CS 4495 Computer Vision

3D Perception

Kelsey Hawkins
Robotics



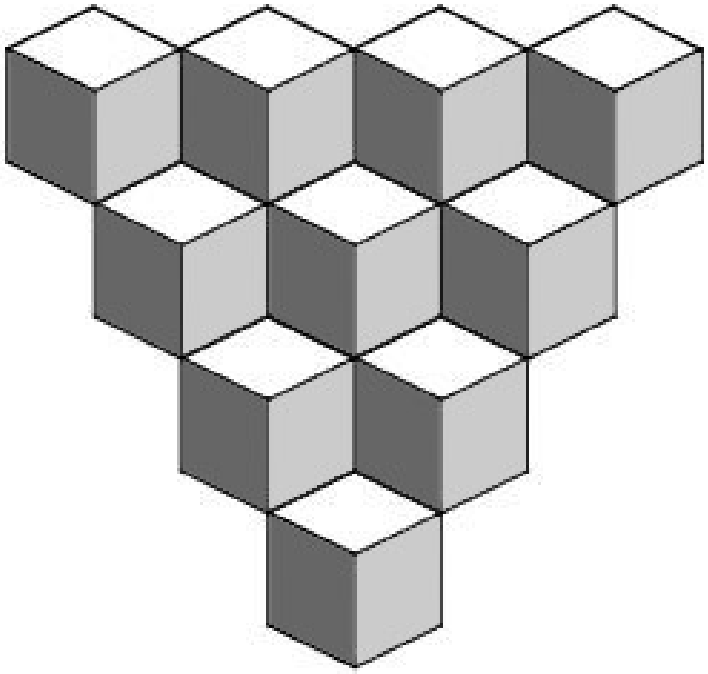


Motivation

- What do animals, people, and robots want to do with vision?
- Detect and recognize objects/landmarks
  - Is that a banana or a snake? A cup or a plate?
- Find location of objects with respect to themselves
  - Want to grasp fruit/tool, where should I put my body/arm?
  - Changes in elevation: steps, rocks, inclined planes
- Determine shape
  - What is the physical 3D structure of this object?
  - Where does an object begin and the background begin?
- Find obstacles and map the environment
  - How do I get my body/arm from A to B without hitting things?
- Others – tracking, dynamics, etc.

Weaknesses of Images

Surface Geometry



Color Inconsistency

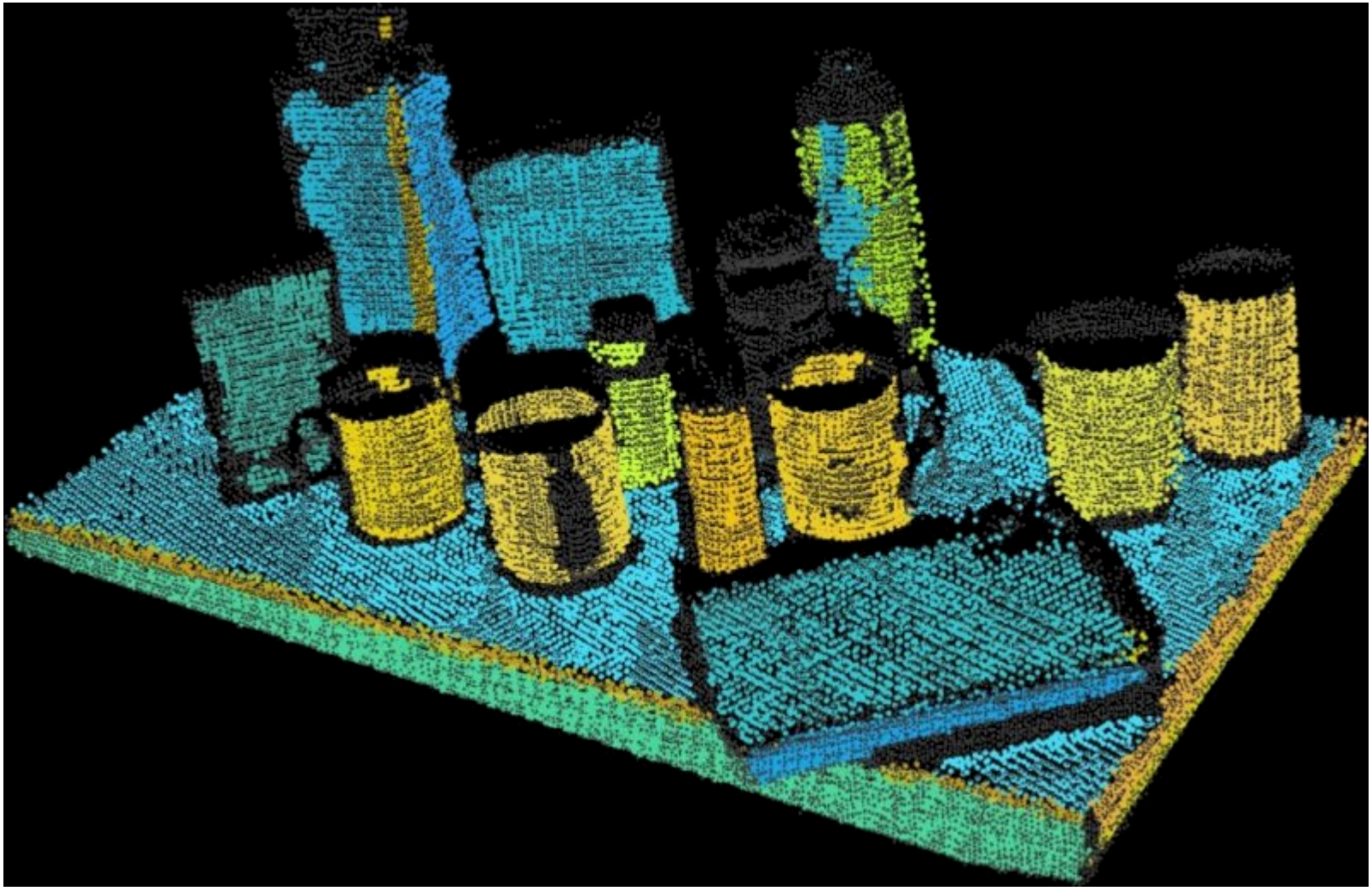
Weaknesses of Monocular Vision

Scale

Lack of texture

Background-foreground similarity

Potential solution: 3D Sensing



pointclouds.org

Types of 3D Sensing

- Passive 3D sensing
  - Work with naturally occurring light
  - Exploit geometry or known properties of scenes
- Active 3D sensing
  - Project light or sound out into the environment and see how it reacts
  - Encode some pattern which can be found in the sensor

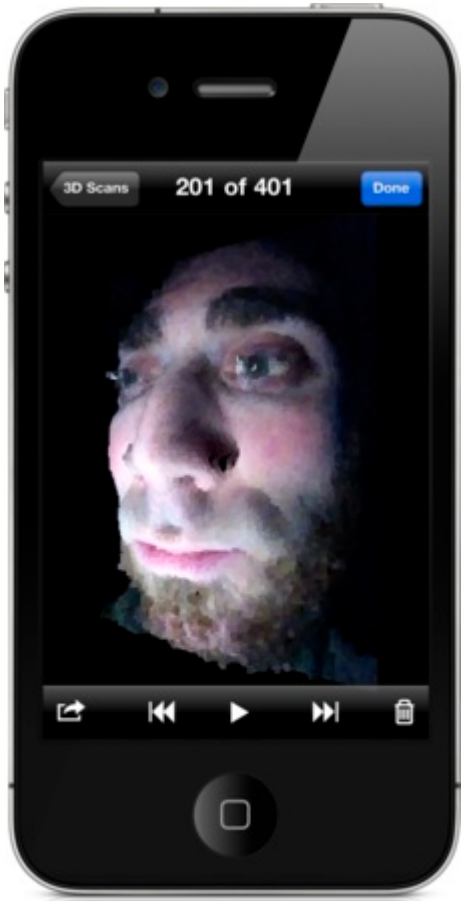
Passive – 3D Sensors

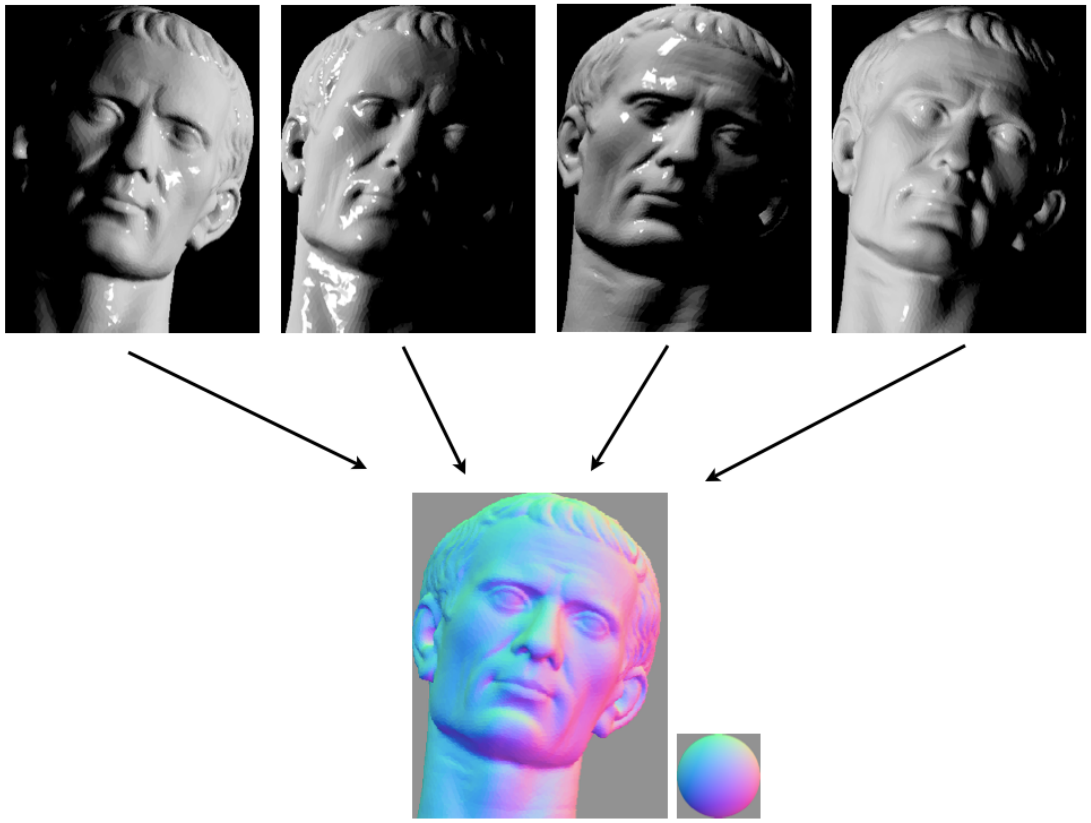
Stereo Rigs

Shape from focus

Nayar, Watanabe, and Noguchi 1996

Active – Photometric Stereo

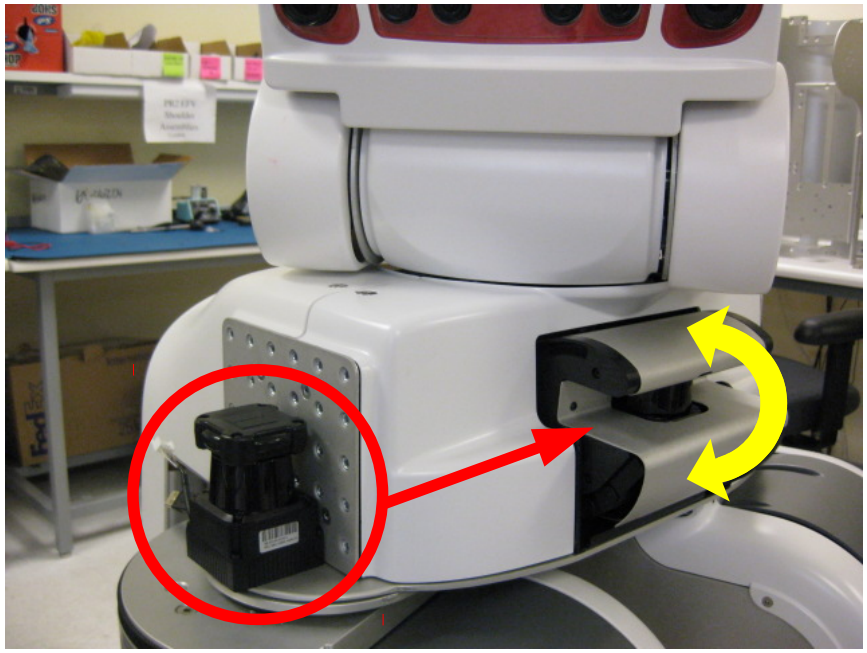




Active – Time of Flight

- Bounce signal off of a surface, record time to come back, X=V*t/2

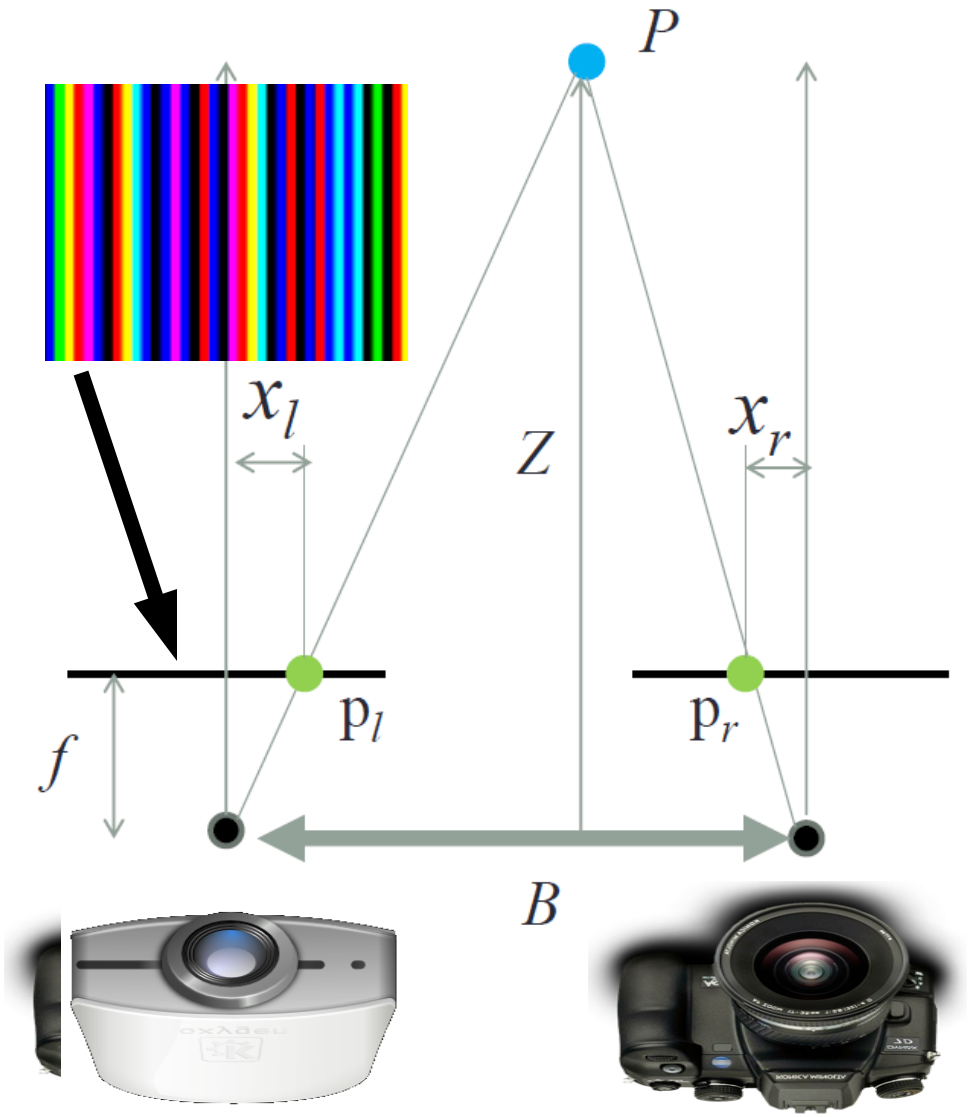
LIDAR / Laser / Range finder



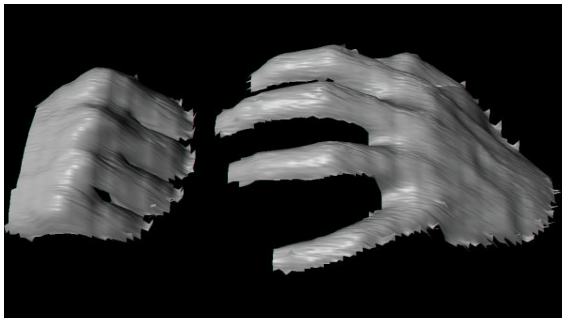
SONAR / Sound / Transceiver

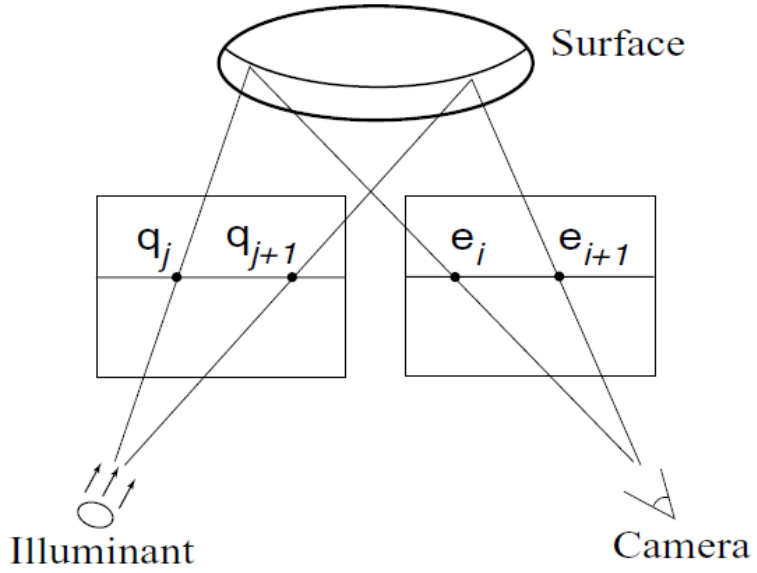
Active - Structured Light

- Remember stereo?
- Let's replace the camera with a projector
- Instead of looking for the same features in both image, we look for a known feature we've projected on the scene

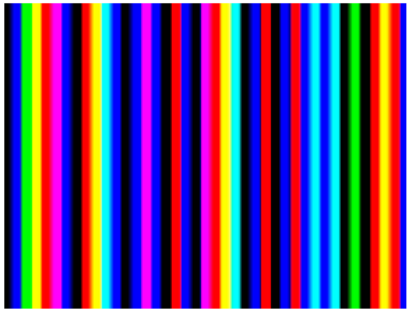


Active – Structured Light

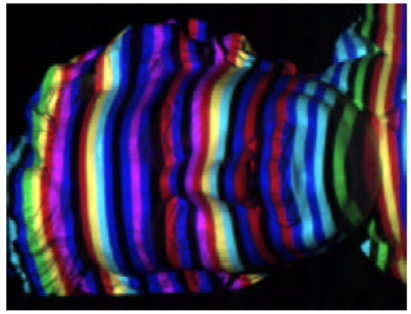




(a)



(b)

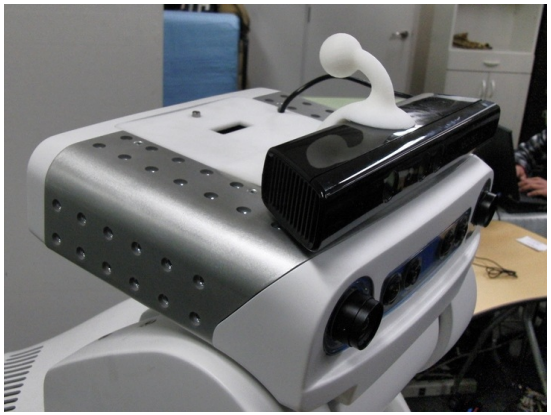


(c)

Zhang, Li et. al. "Rapid shape acquisition..."

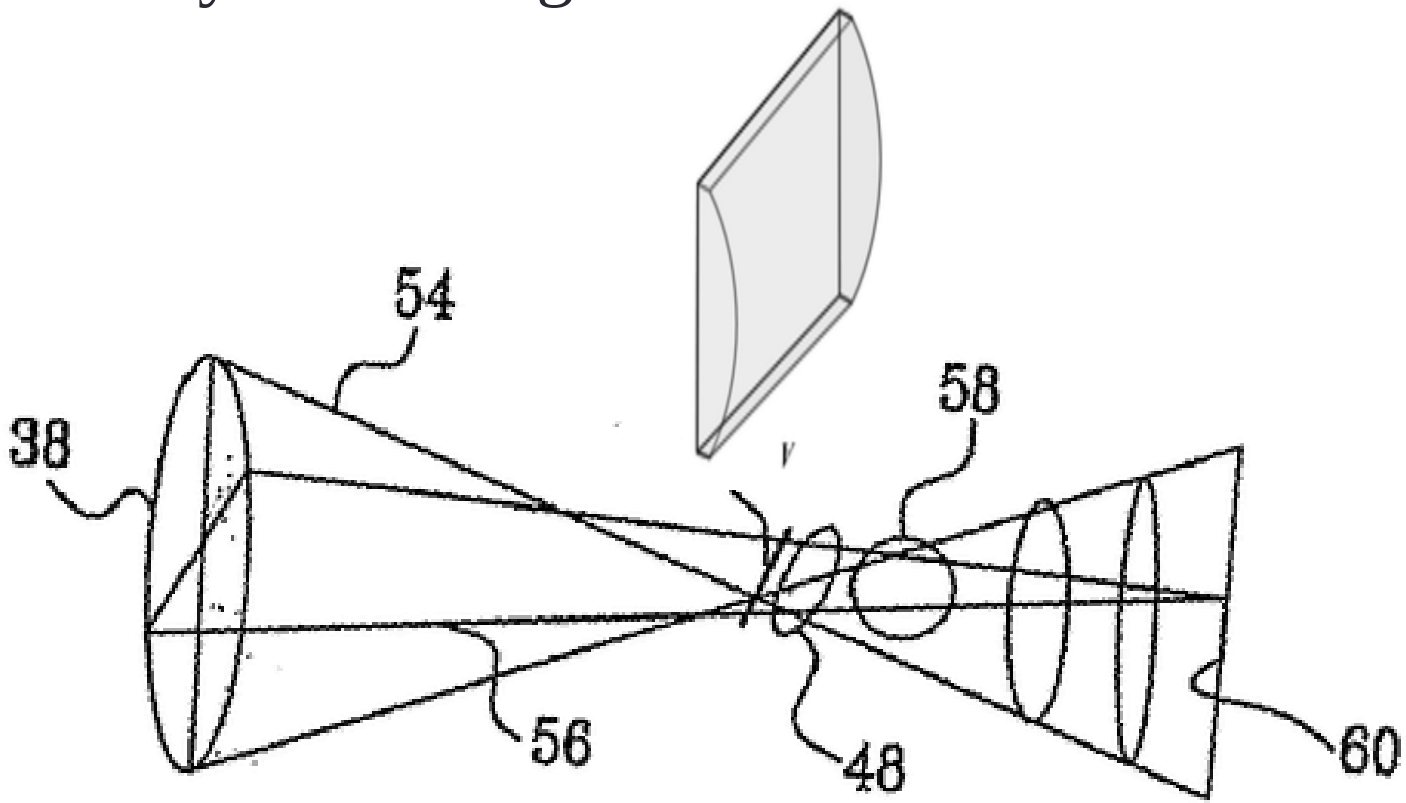
Active – Infrared Structured Light

KINECT
for XBOX 360



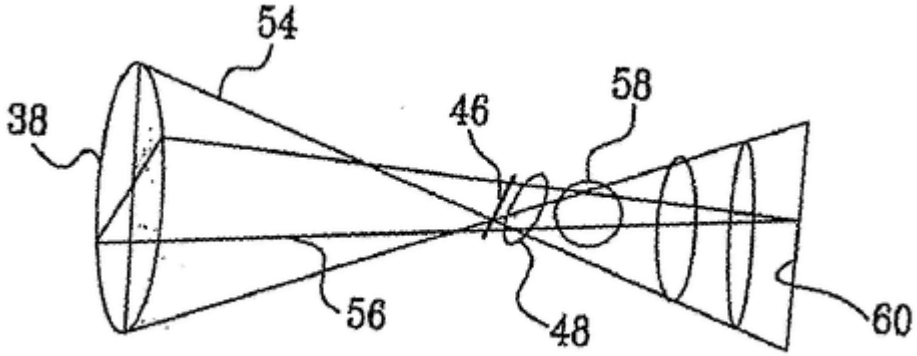
How the Kinect works

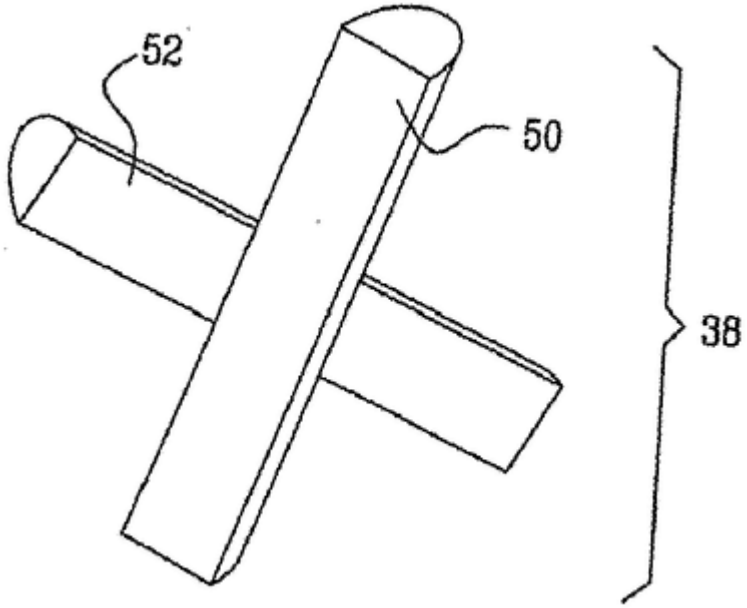
- Cylindrical lens
  - Only focuses light in one direction

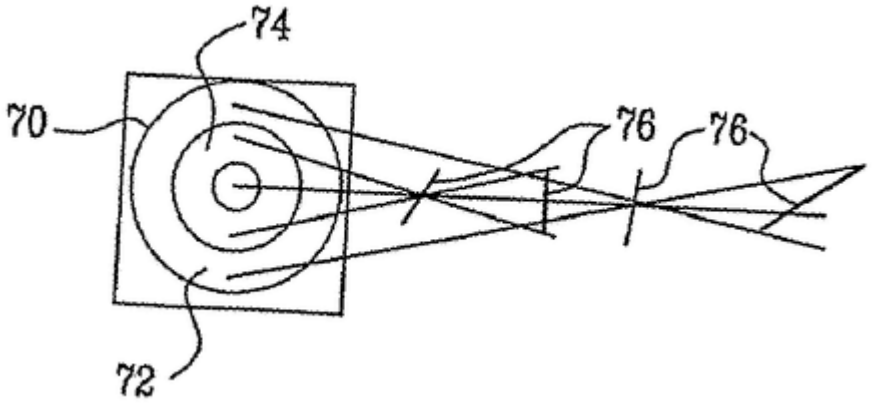


PrimeSense patent 2010/0290698

How the Kinect works

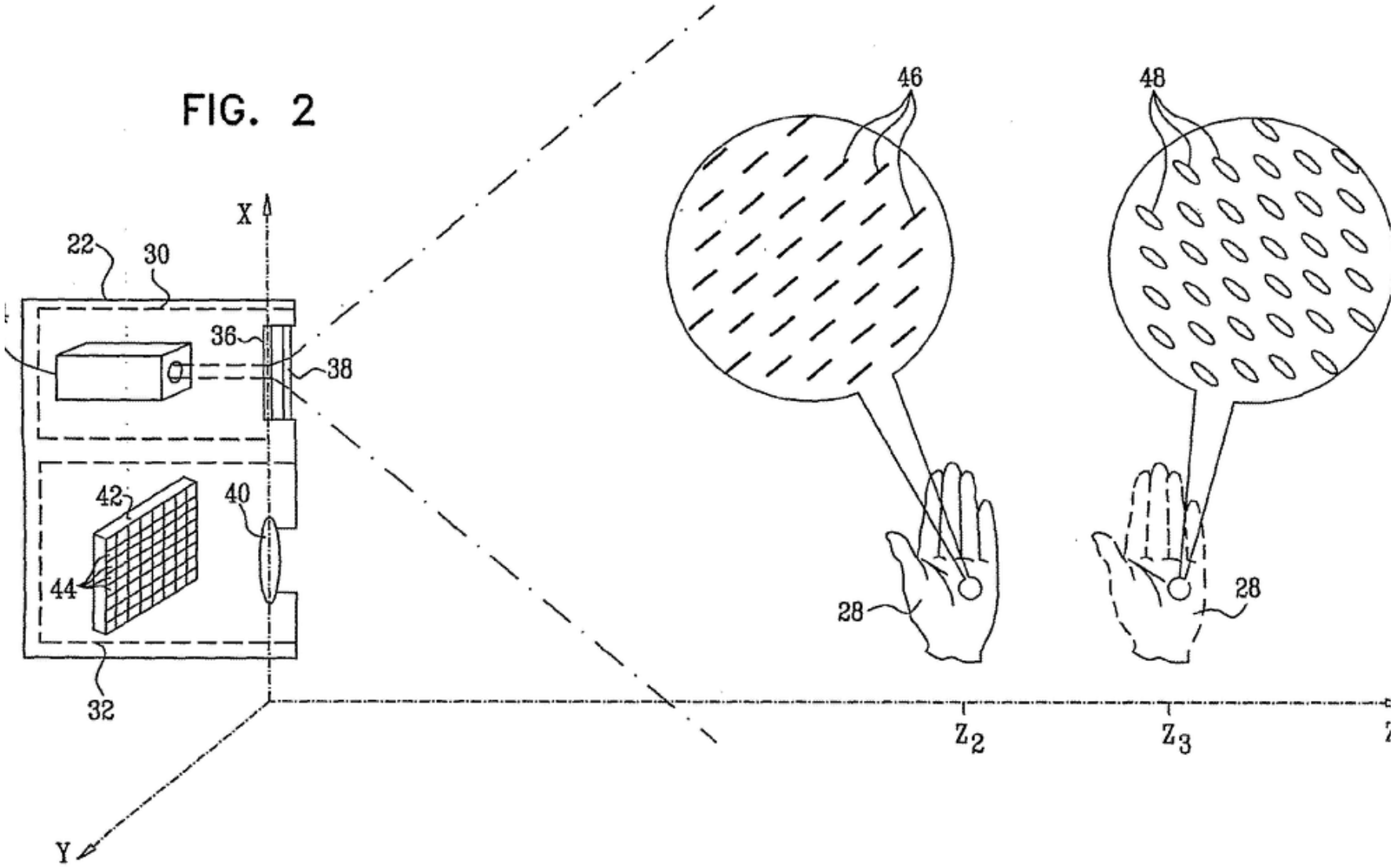






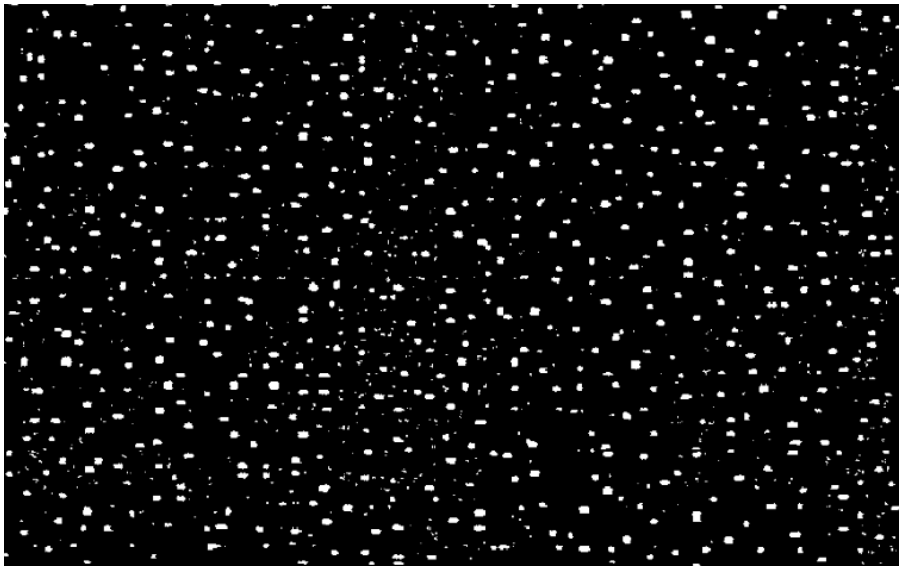
PrimeSense patent No. 20100290698

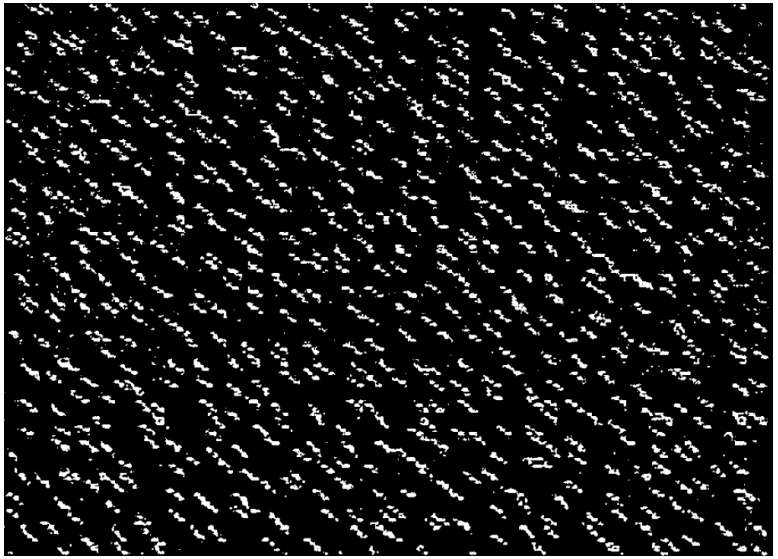
How the Kinect works



PrimeSense patent No. 20100290698

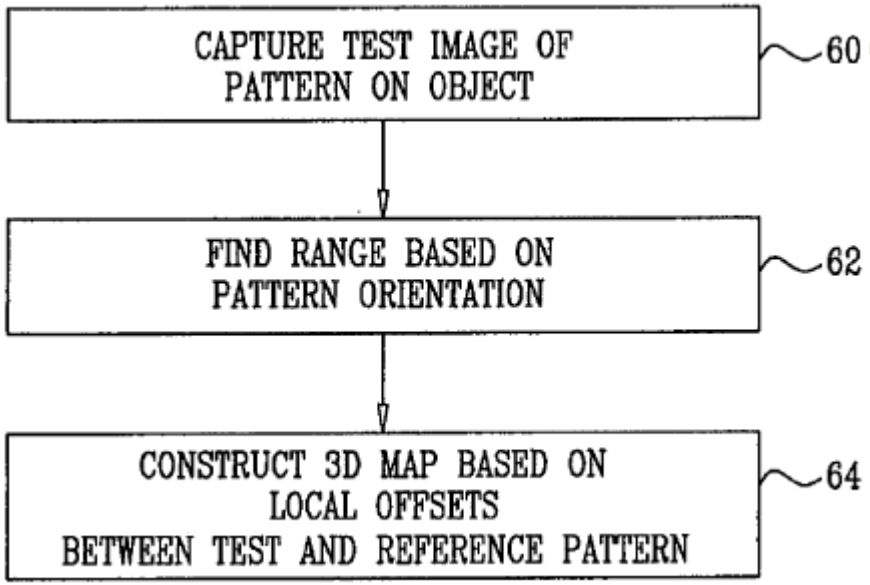
How the Kinect works

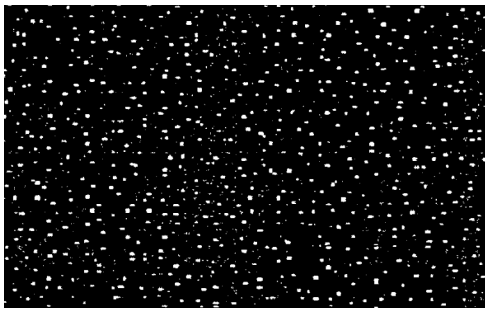


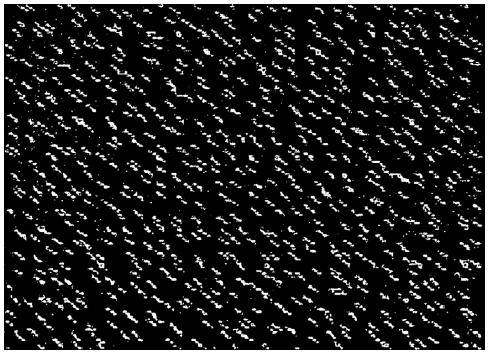


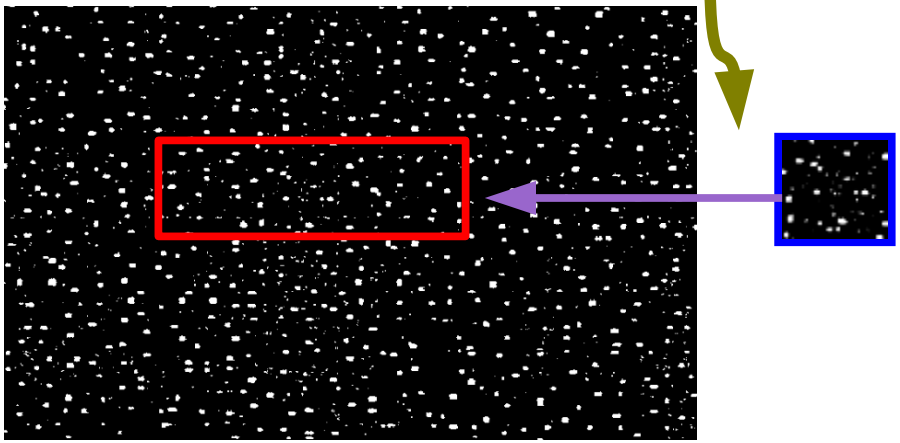
PrimeSense patent 2010/0290698

How the Kinect works









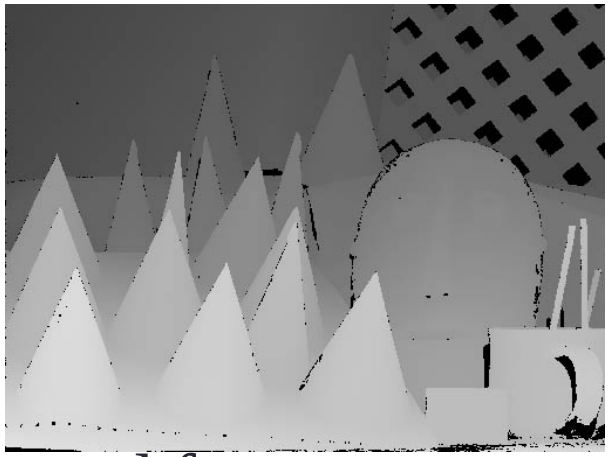
Psuedo-random speckle pattern

PrimeSense patent 2010/0290698

2D vs. 3D Perception

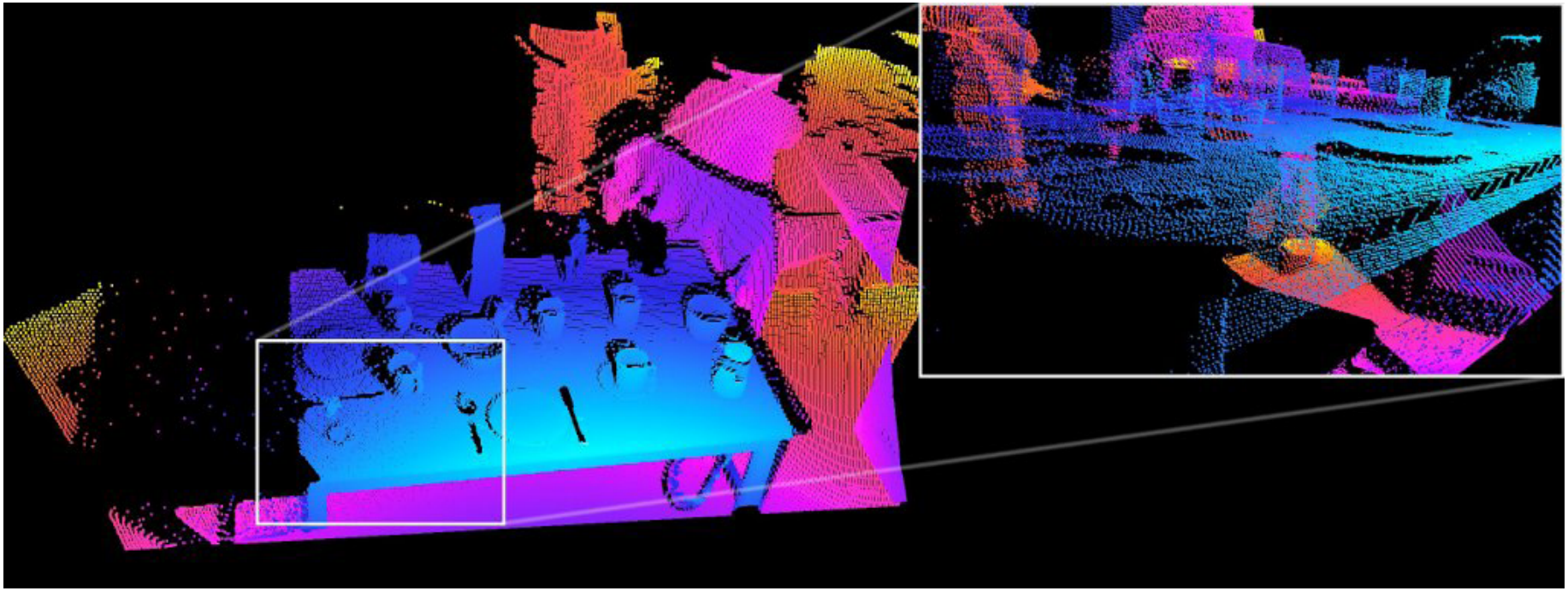
Analysis Tools	2D	3D
Representation	Image (u,v)	• Depth image (u,v,d) • Point cloud (x,y,z)
1st order differential geometry	Image gradients	Surface normals
2nd order differential geometry	Second moment matrix	Principle curvature
Corner detection	Harris image	Surface variation
Feature extraction	HOG	• Point Feature Histograms • Spin Images
Geometric model fitting	Hough transform	Clustering + RANSAC
Alignment	SSD window filter	Iterative Closest Point (ICP)

Depth Images



- Advantages
  - Dense representation
  - Gives intuition about occlusion and free space
  - Depth discontinuities are just edges on the image
- Disadvantages
  - Viewpoint dependent, can't merge
  - Doesn't capture physical geometry
  - Need actual 3D locations

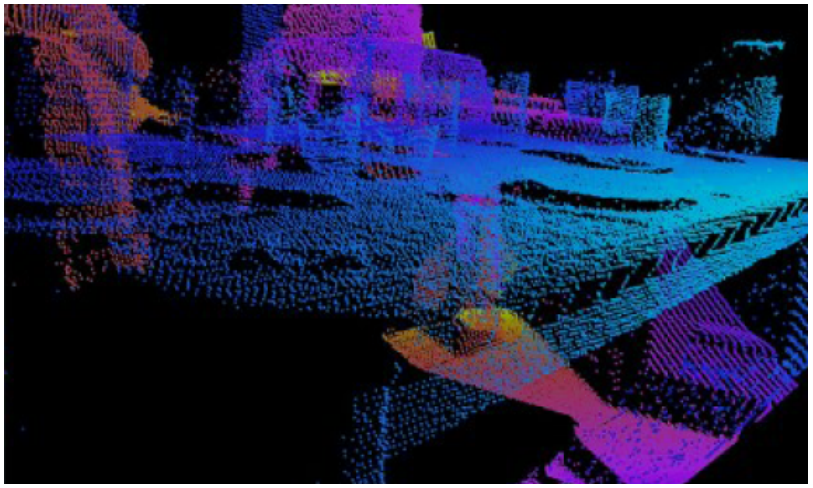
Point Clouds



- Take every depth pixel and put it out in the world
- What can this representation tell us?
- What information do we lose?

R. Rusu's PCL Presentation

Point Clouds

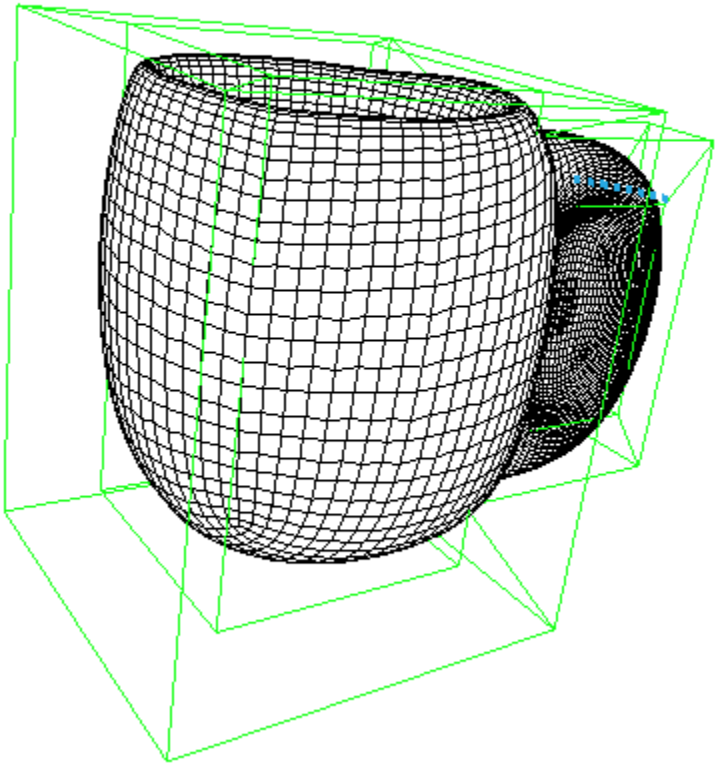


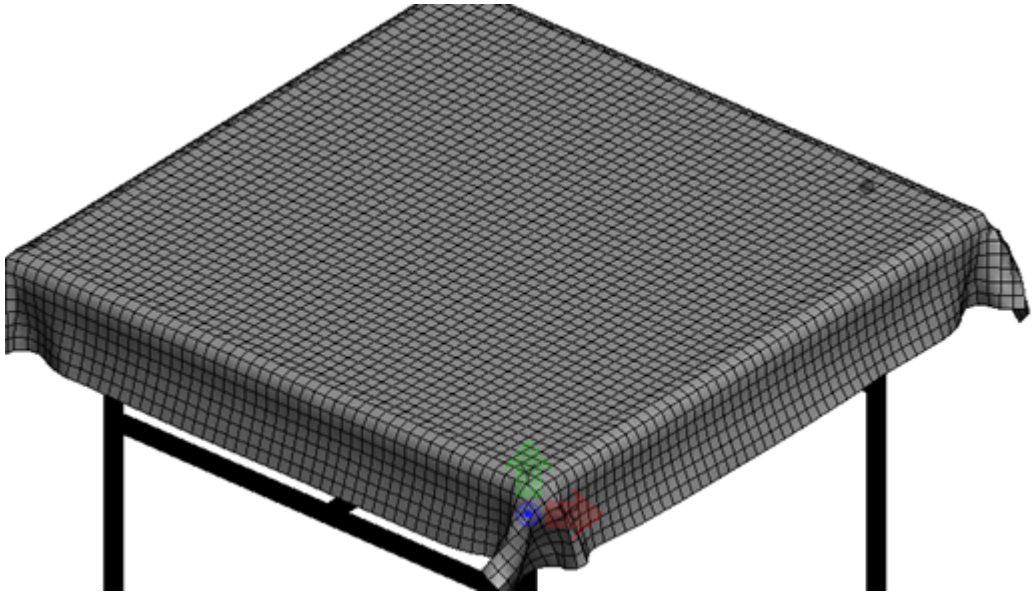
- Advantages
  - Viewpoint independent
  - Captures surface geometry
  - Points represent physical locations
- Disadvantages
  - Sparse representation
  - Lost information about free space and unknown space
  - Variable density based on distance from sensor

R. Rusu's PCL Presentation

Point Clouds and Surfaces

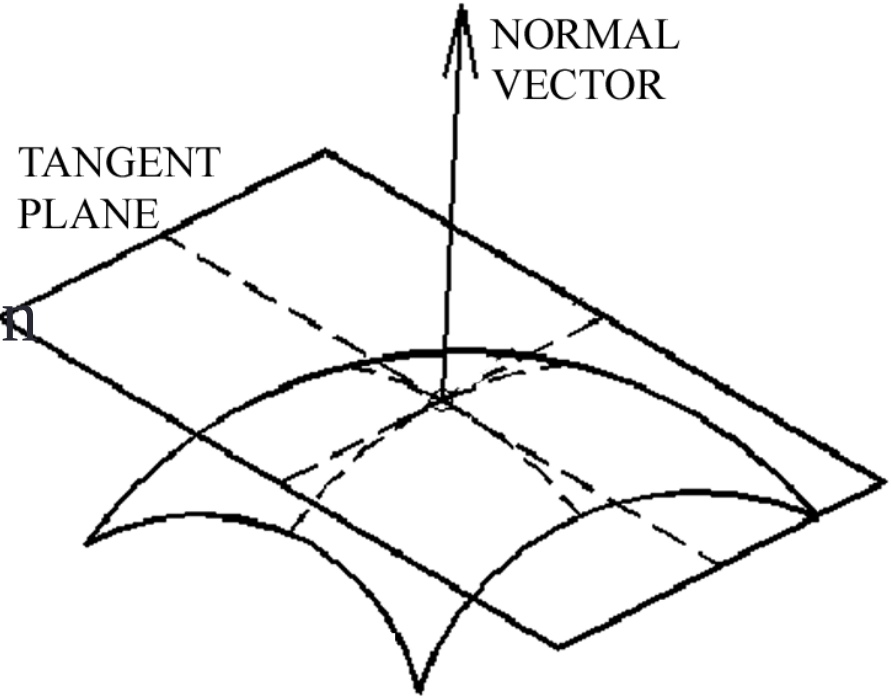
- Point clouds are sampled from the surfaces of the objects perceived
- The concept of volume is inferred, not perceived





Surfaces

- Let's say we'd like to learn the “geometry” around a point in our cloud
- What is the simplest surface representation we could use to approximate the surface about a point?
- Tangent plane
  - Defined by normal
- First-order approximation

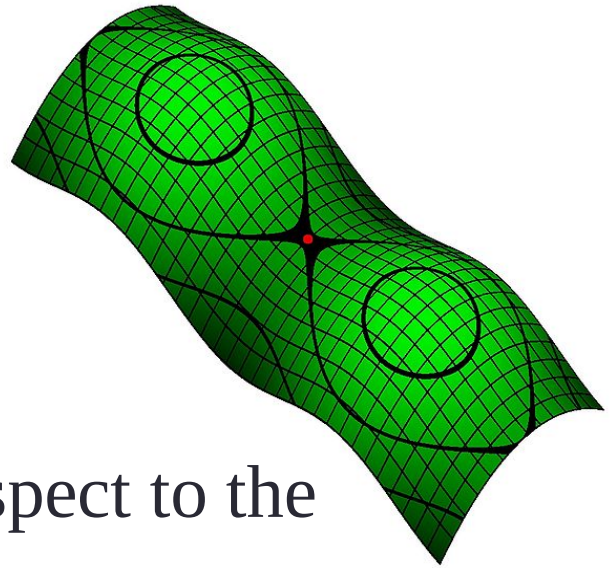


Surfaces

- To understand how we can characterise surfaces, we can look to differential geometry
- A surface is 2D manifold in 3D space

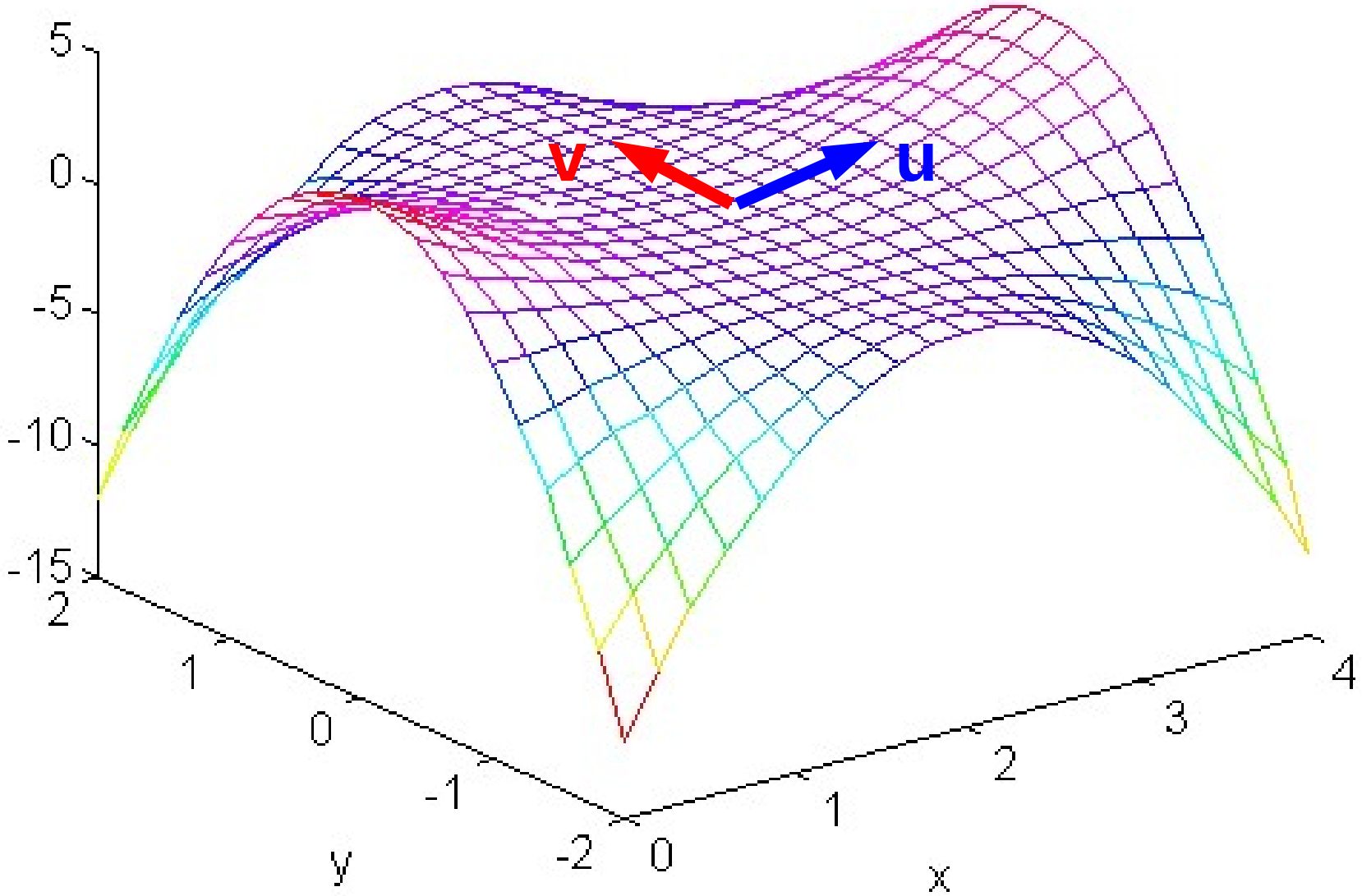
$$f : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$

$$f(u, v) = (x, y, z)$$

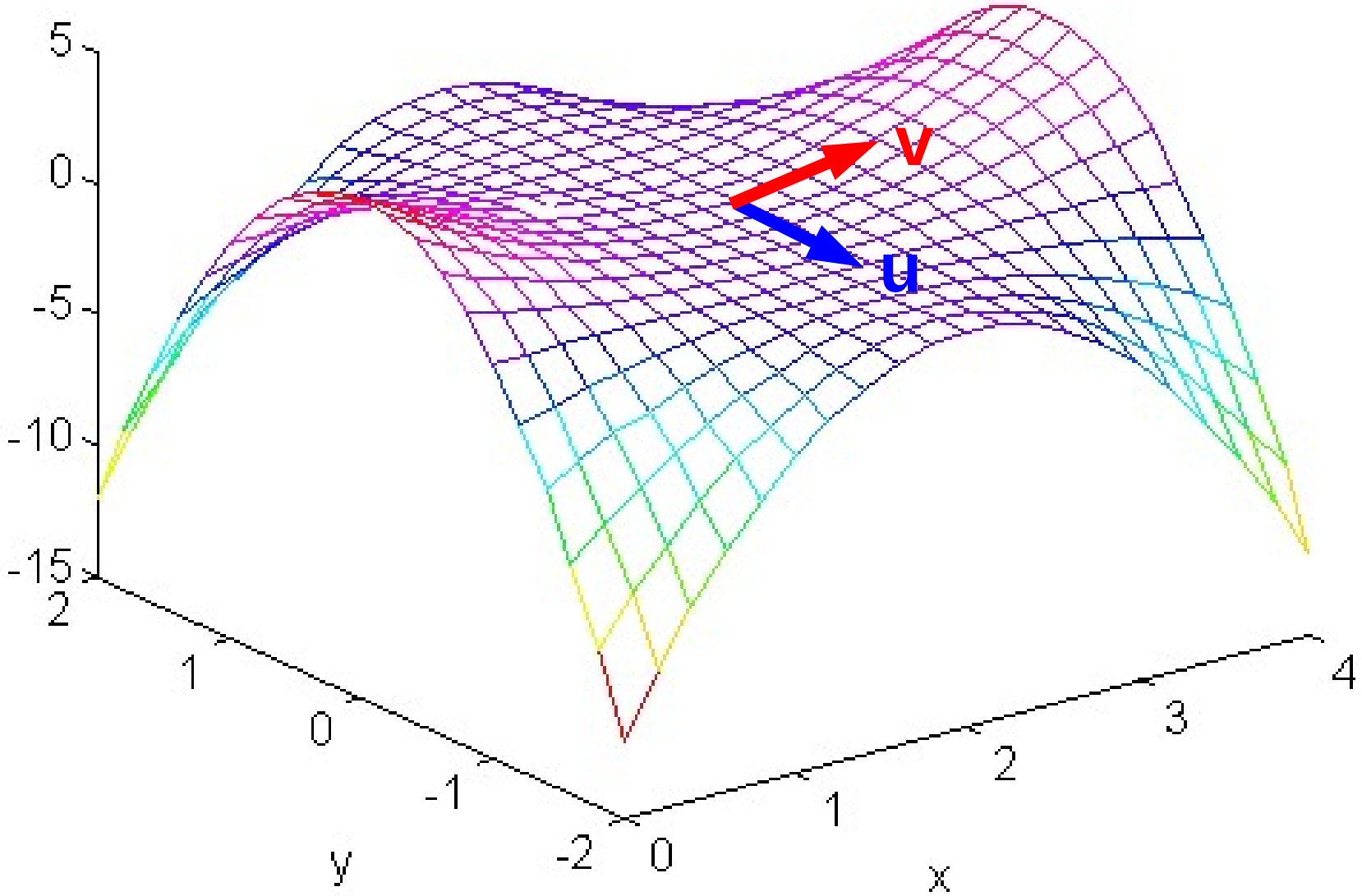


- Parametric representation
- How u and v are “oriented” with respect to the surface is irrelevant

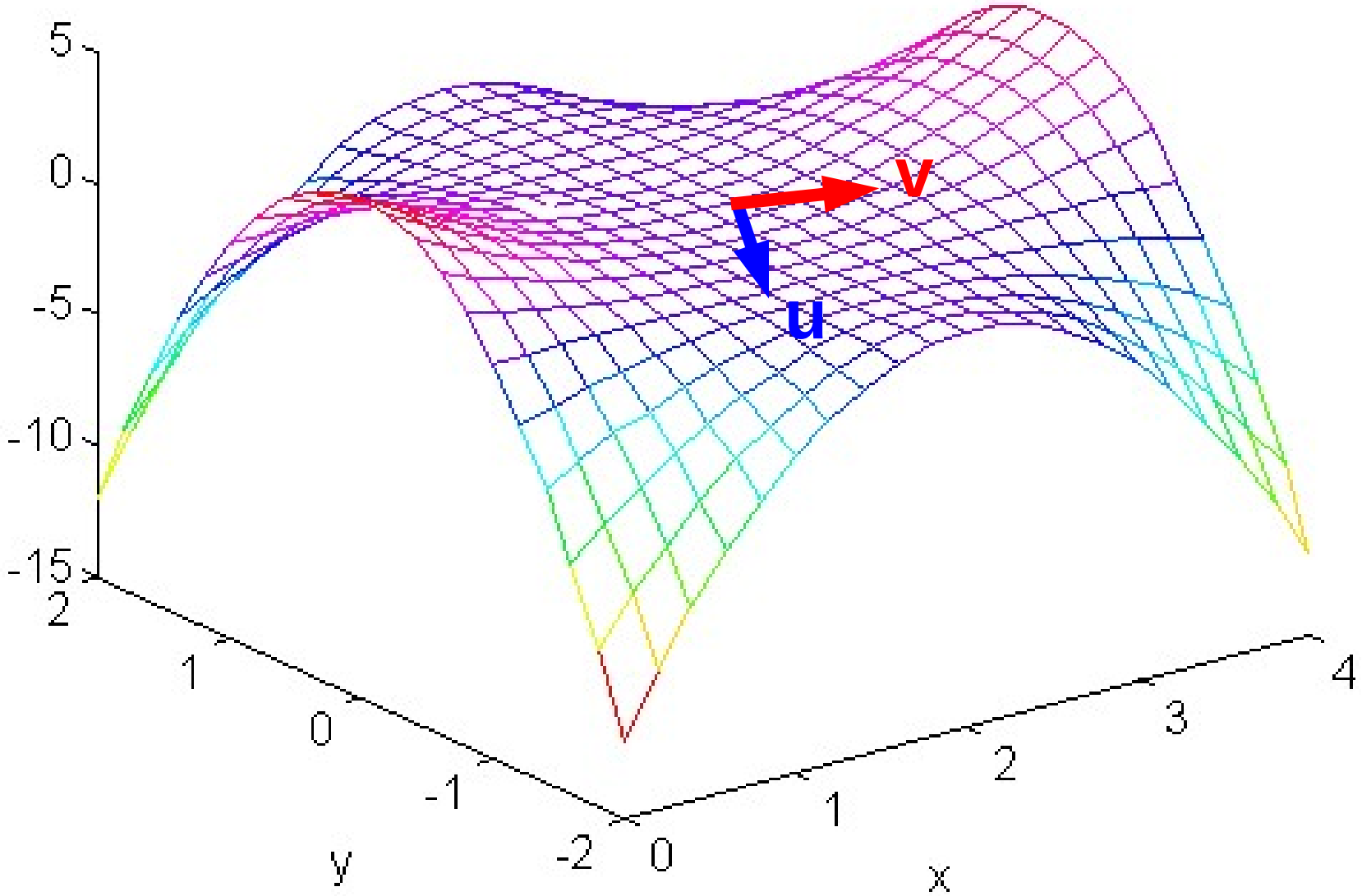
Surfaces



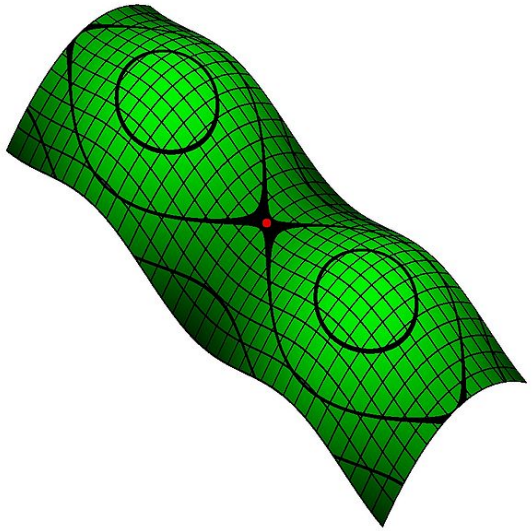
Surfaces



Surfaces



Surface Normals

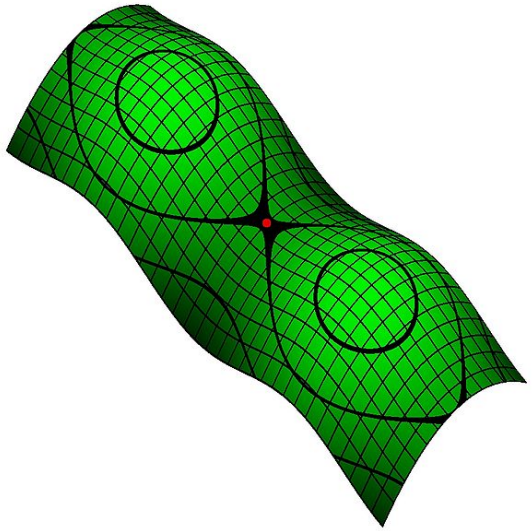


- Want to estimate this function $\mathrm{f}(u,v)$
- What can we do to estimate this function?

Taylor Series 1st order approximation at $(u_{0,}v_0)$

$$\mathrm{f}(u,v) \approx \mathrm{f}(u_{0,}v_0) + [u-u_{0,}\, v-v_0] \begin{bmatrix} \frac{\partial \mathrm{f}}{\partial u}(u_{0,}v_0) \\ \frac{\partial \mathrm{f}}{\partial v}(u_{0,}v_0) \end{bmatrix}$$

Surface Normals



- We have a problem though...
- Don't have (u, v) basis, infinite exist!
- Take a sample of 3D points we believe lie on $f(u, v)$ around (u_0, v_0)

$$A = \begin{bmatrix} f(u_1, v_1) \\ \vdots \\ f(u_n, v_n) \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & z_1 \\ \vdots & & \\ x_n & y_n & z_n \end{bmatrix} = \begin{bmatrix} u_1 - u_0 & v_1 - v_0 \\ \vdots & \\ u_n - u_0 & v_n - v_0 \end{bmatrix} \begin{bmatrix} \frac{\partial f}{\partial u}(u_0, v_0)^T \\ \frac{\partial f}{\partial v}(u_0, v_0)^T \end{bmatrix}$$

- Find n such that

$$An = 0$$

- We've done this before (last eigenvector)

Surface Normals

$$An = \begin{bmatrix} u_1 - u_0 & v_1 - v_0 \\ \vdots & \\ u_n - u_0 & v_n - v_0 \end{bmatrix} \begin{bmatrix} \frac{\partial \mathrm{f}}{\partial u}^T \\ \frac{\partial \mathrm{f}}{\partial v}^T \end{bmatrix} n = 0 \Leftrightarrow \begin{bmatrix} \frac{\partial \mathrm{f}}{\partial u}^T \\ \frac{\partial \mathrm{f}}{\partial v}^T \end{bmatrix} n = 0 \Leftrightarrow \frac{\partial \mathrm{f}}{\partial u} \cdot n = 0, \frac{\partial \mathrm{f}}{\partial v} \cdot n = 0$$

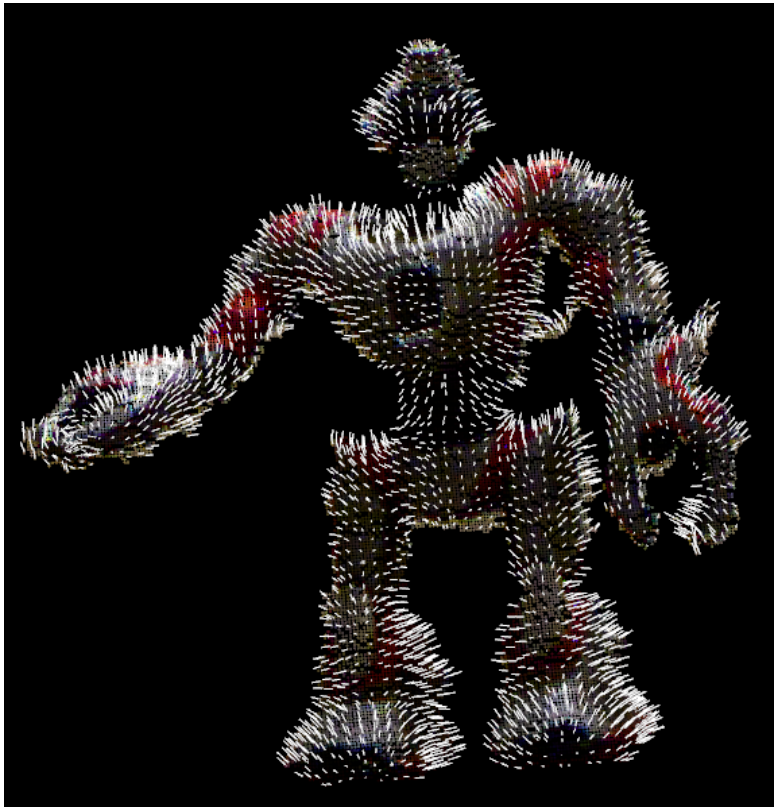
- This n (the normal) is perpendicular to both partials, regardless of basis choice

$$\frac{\partial \mathrm{f}}{\partial u} \perp n, \frac{\partial \mathrm{f}}{\partial v} \perp n$$

- Surface normal is a first order approximation of the surface at the point invariant to basis choice

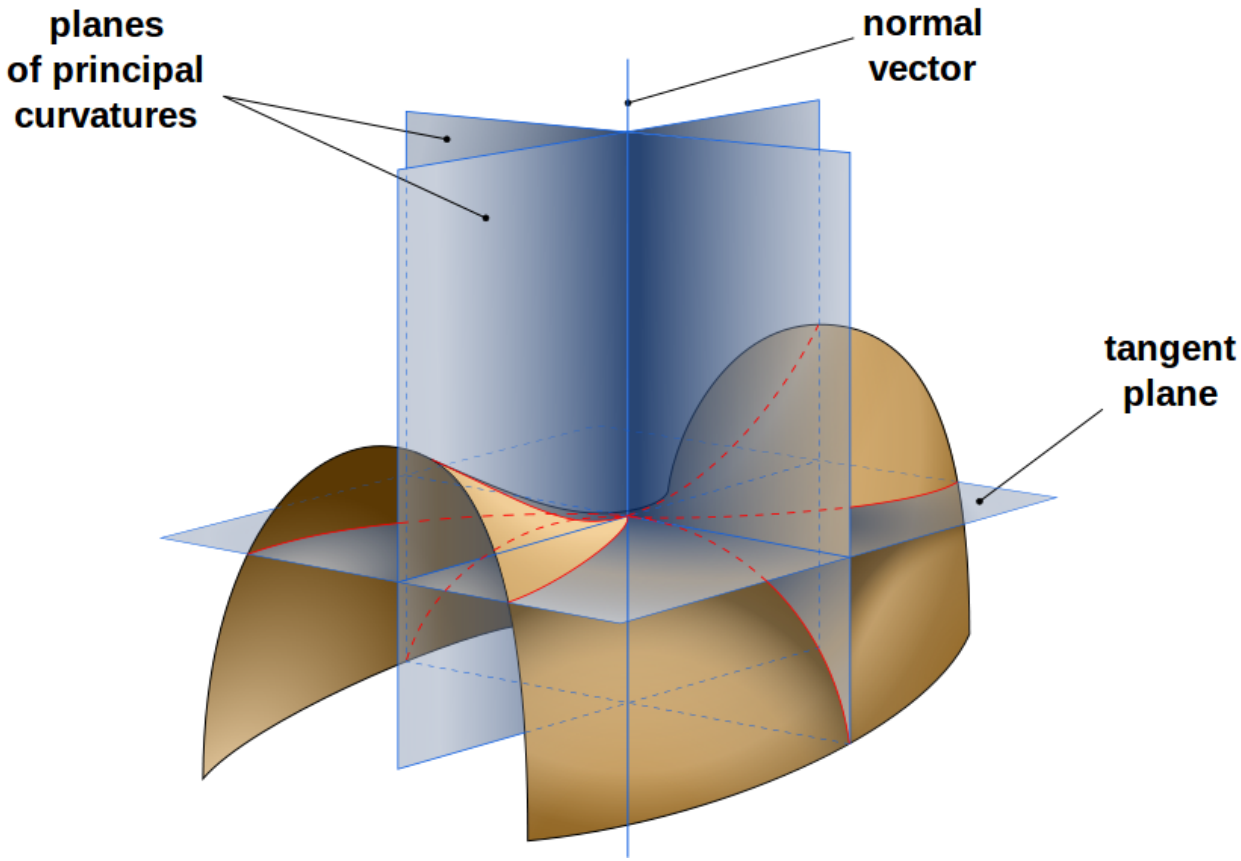
Surface Normals

- Size of patch is like width of Gaussian in image gradient calculation
- We can use them to find planes



Principal Curvature

- Second order approximation



Surface Variation

$$A = \begin{bmatrix} \mathrm{f}(u_1, v_1) \\ \vdots \\ \mathrm{f}(u_n, v_n) \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & z_1 \\ \vdots & & \\ x_n & y_n & z_n \end{bmatrix}$$

$$A = U\,S\,V^T = U \begin{bmatrix} s_2 & 0 & 0 \\ 0 & s_1 & 0 \\ 0 & 0 & s_0 \end{bmatrix} \begin{bmatrix} v_2 & v_1 & v_0 \end{bmatrix}^T$$

Normal

Principal Curvatures

$$\text{surface variation} = \frac{s_0^2}{s_0^2 + s_1^2 + s_2^2}$$

- This is equivalent to finding the eigenvalues/vectors of the covariance matrix $A^T A$

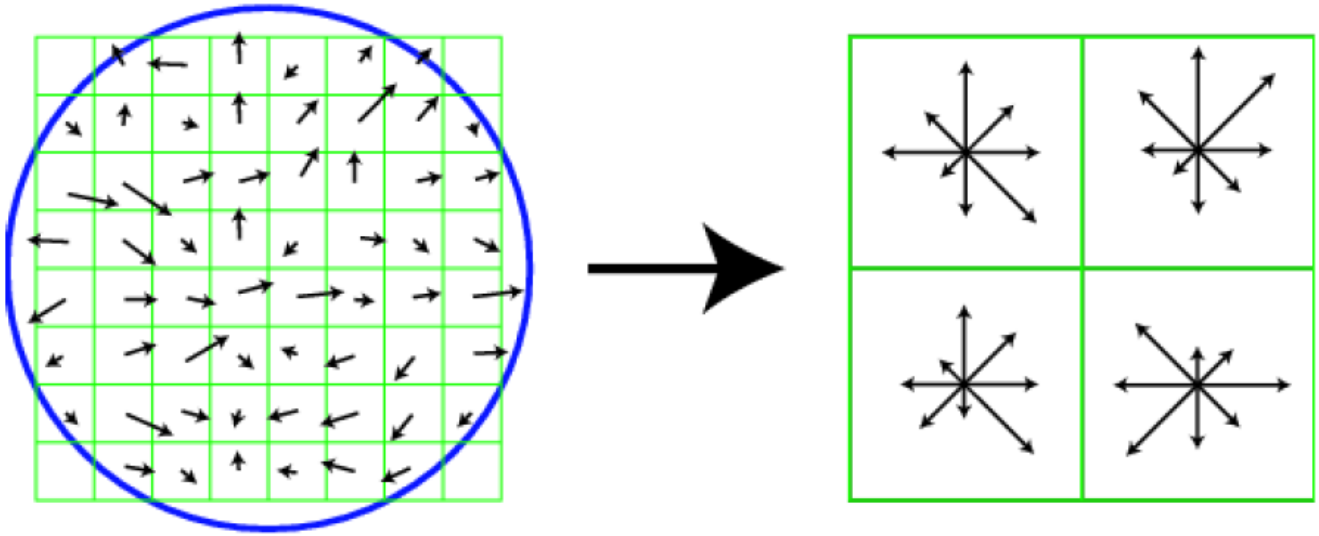
Normals / Surface Variation Demo

Feature Extraction

- Suppose we want a denser description of the local surface function
- Want to find unique patches of surface geometry
- What type of invariance do we need?
- Need viewpoint invariance
  - Translation + orientation
  - Color and texture come automatically!

Point Feature Histograms

- Remember SIFT?

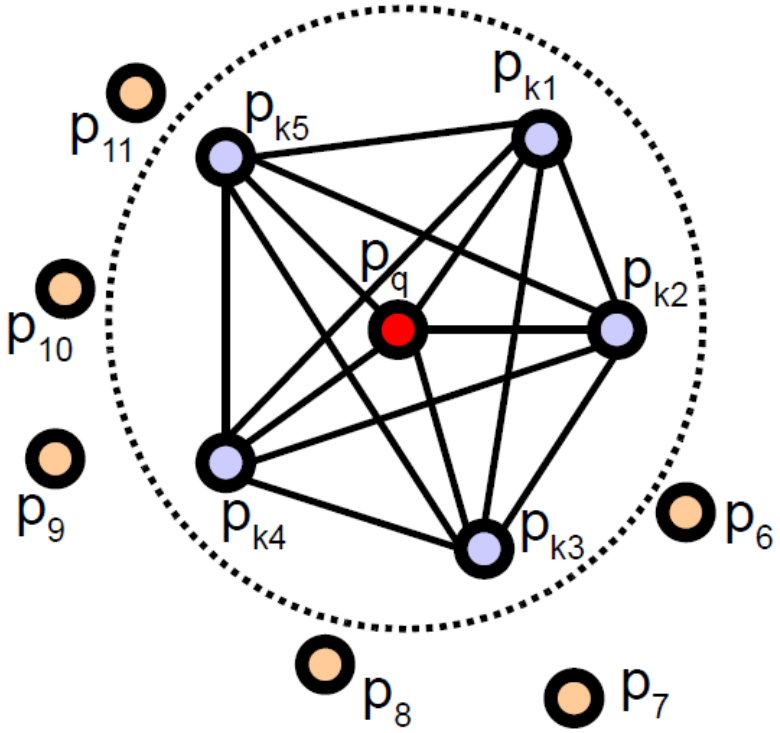


(a) image gradients (b) keypoint descriptor

- We're going to use roughly the same idea
  - Use the normal at the point to establish a dominant orientation
  - Build a histogram of the orientations of normals in the general region with respect to the original

Point Feature Histograms

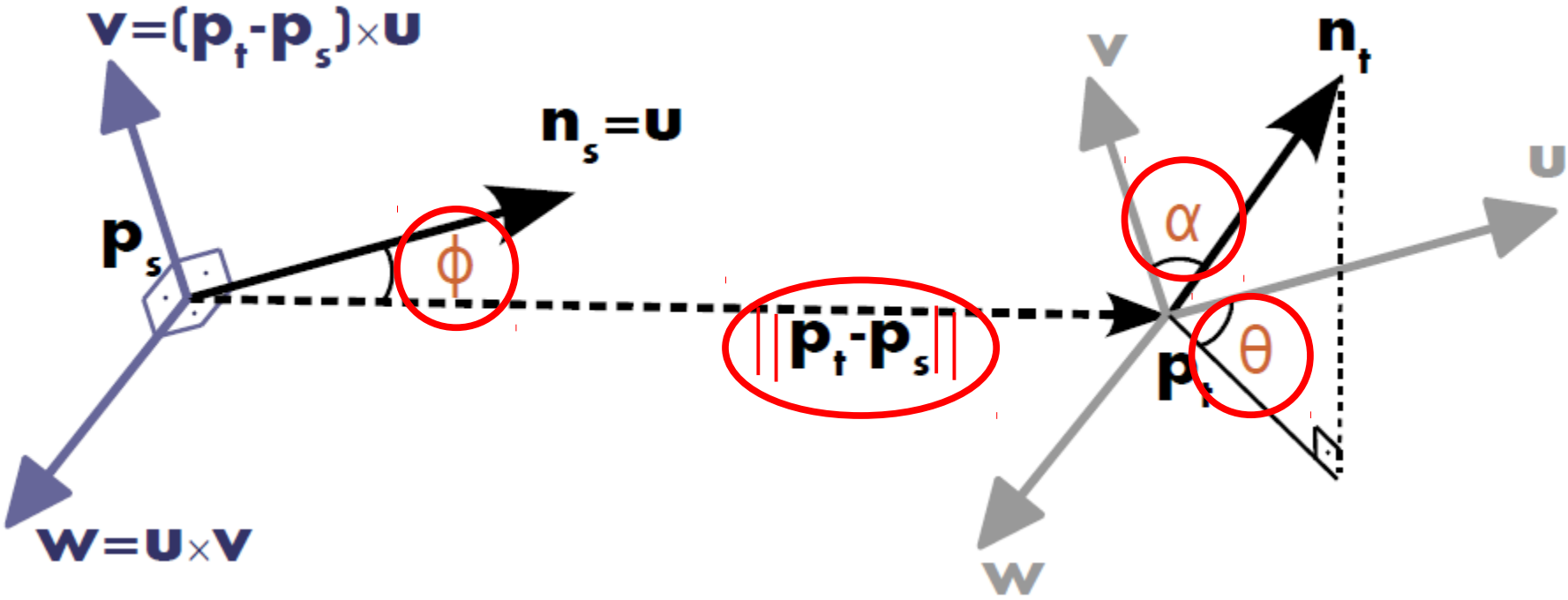
- At a point, take a ball of points around it
- For every pair of points, find the relationship between the two points and their normals
- Must be frame independent



R. Rusu's Thesis

Point Feature Histograms

- Reduce $\begin{pmatrix} x_1, y_1, z_1, n_{x1}, n_{y1}, n_{z1} \\ x_2, y_2, z_2, n_{x2}, n_{y2}, n_{z2} \end{pmatrix}$ to 4 variables



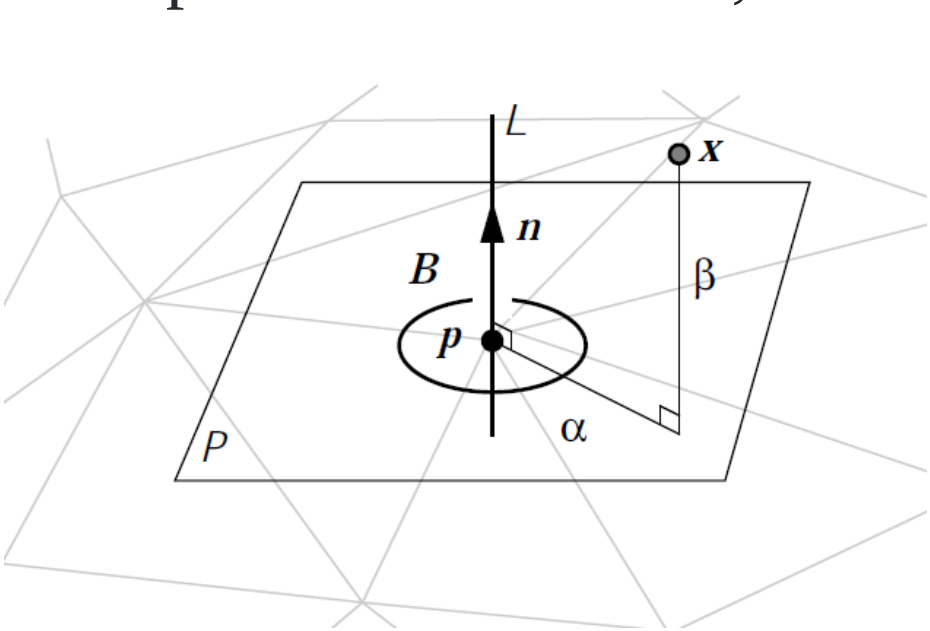
R. Rusu's Thesis

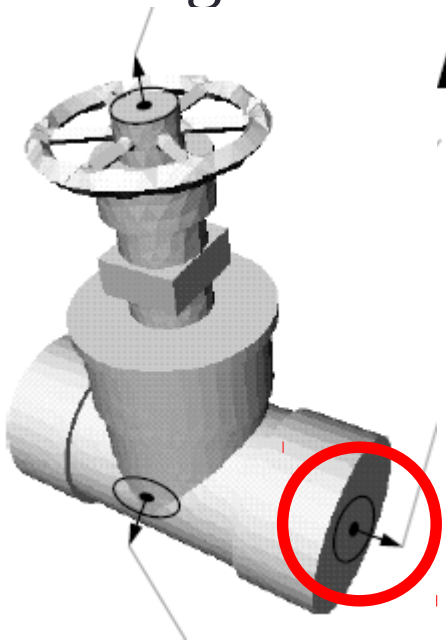
Point Feature Histograms

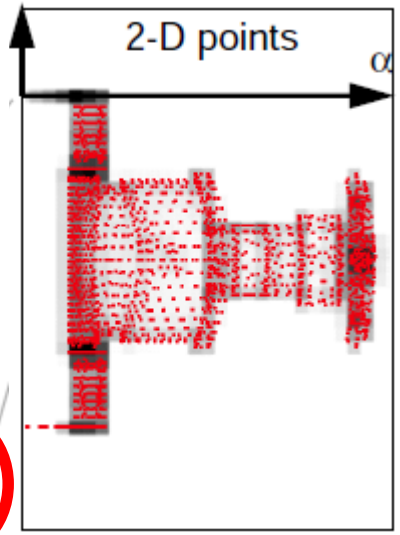
- Find these for variables for every pair in the ball
- Build a 5x5x5x5 histogram of the variables
  - Often the distance variable is excluded
  - In this case, we have a 125-long feature vector
- Use this just like a SIFT feature descriptor
- Usually, a sped-up version called Fast Point Feature Histograms is used for real-time applications

Spin Images

- Rotate plane about normal of a point, project all points onto surface, build a histogram

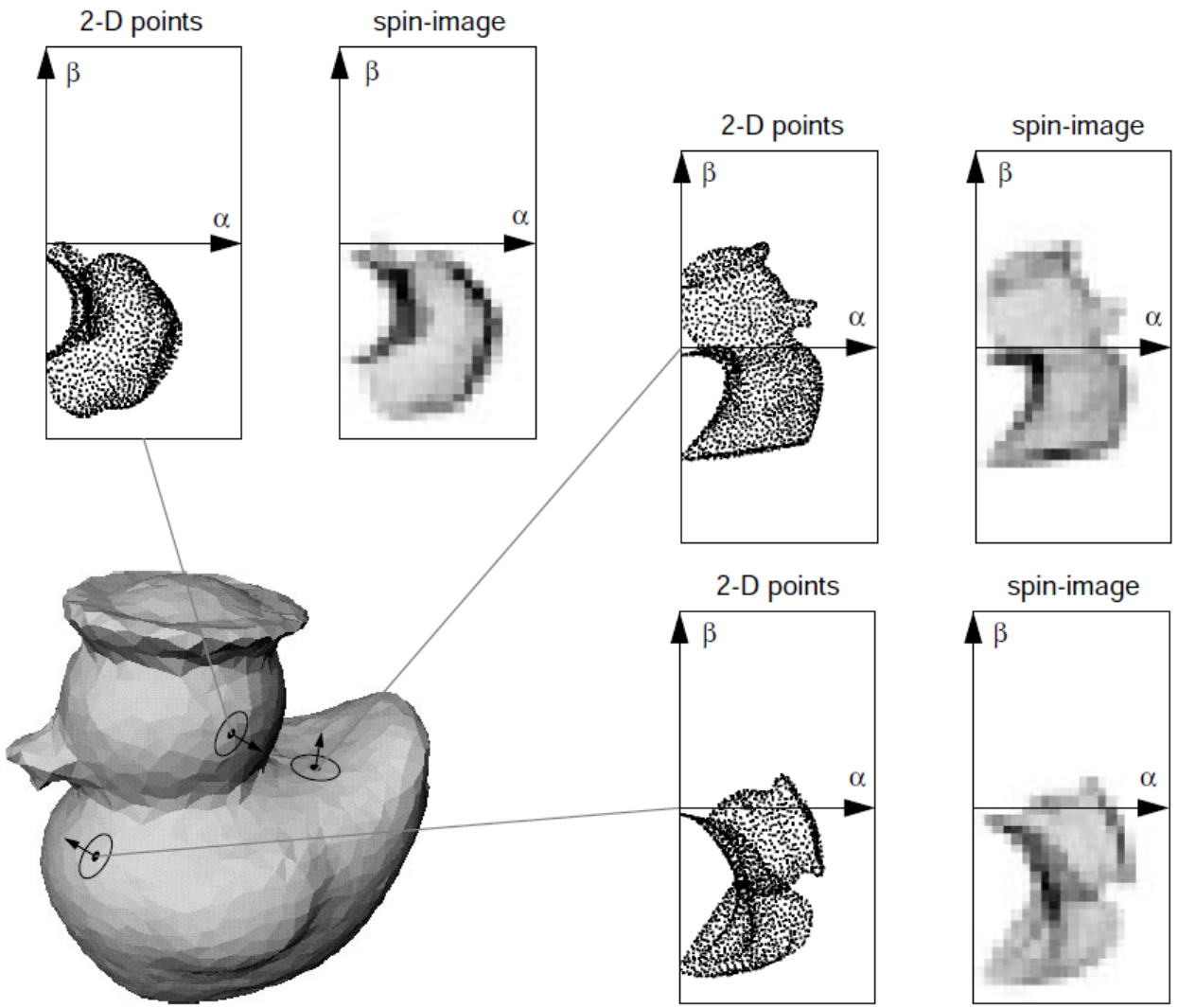






A. Johnson's 1997 Thesis

Spin Images



A. Johnson's 1997 Thesis

Comparison of 3D Descriptors

TABLE I

IMPORTANT FEATURES OF THE DESCRIPTORS EVALUATED IN THIS PAPER. V=VARIABLE; Y=YES; N=NO; O=OMP; G=GPU; n=NUMBER OF POINTS IN INPUT CLOUD; m=NUMBER OF STABLE REGIONS; p=NUMBER OF AZIMUTH BINS. SEE TEXT FOR DETAILS

Descriptor	N.points	Point size	Parallel	Normals
3DSC	$n * p$	V	N	Y
CVFH	$m \leq n$	308	N	Y
ESF	1	640	N	N
FPFH	n	33	O+G	Y
PFH	n	125	G	Y
PFHRGB	n	250	G	Y
PCE	n	5	G	Y
PPF	n	5	O+G	Y
RIFT	n	32	N	N
SHOT	n	9+352	O	Y
SHOTCOLOR	n	9+1344	O	Y
USC	n	V	N	N
VHF	1	308	G	Y

Alexandre, L., 3D Descriptors for Object and Category Recognition: a Comparative Evaluation

Comparison of 3D Descriptors

TABLE II

CATEGORY AND OBJECT RECOGNITION ACCURACY USING THE HARRIS3D KEYPOINT EXTRACTOR AND CLOUD SUB-SAMPLING (WITH 1CM AND 2CM LEAF SIZE)

Descriptor	Harris3D Category	Harris3D Object	Sub-sampl. 1cm Category	Sub-sampl. 1cm Object	Sub-sampl. 2cm Category	Sub-sampl. 2cm Object
3DSC	74.05	32.91	87.34	50.63	78.06	40.51
CVFH	55.16	20.63	64.13	35.23	51.05	19.41
ESF	82.91	39.03	83.54	39.66	81.65	37.34
FPFH	81.89	44.63	87.55	49.58	86.08	47.26
PFH	86.95	48.42	89.87	56.75	89.24	55.49
PFHRGB	**93.89**	**77.89**	**94.09**	**79.32**	**94.73**	**79.75**
PCE	37.89	9.26	50.84	17.09	47.26	16.46
PPF	37.47	8.63	56.33	18.14	52.32	17.09
RIFT	34.11	9.68	75.11	35.44	60.34	24.05
SHOT	81.26	42.53	91.77	55.49	88.19	46.84
SHOTCOLOR	88.40	69.20	92.62	75.53	90.72	73.42
USC	73.00	35.44	86.50	51.05	82.07	45.78
VFH	28.84	6.11	52.11	23.63	21.73	5.27
Average	65.83	34.18	77.83	45.20	71.03	39.13

TABLE III

SIZE OF THE FEATURE DATABASE IN MEGABYTES AND TIME TO PROCESS THE TEST SET IN SECONDS USING THE HARRIS3D KEYPOINT EXTRACTOR AND CLOUD SUB-SAMPLING (WITH 1CM AND 2CM LEAF SIZE)

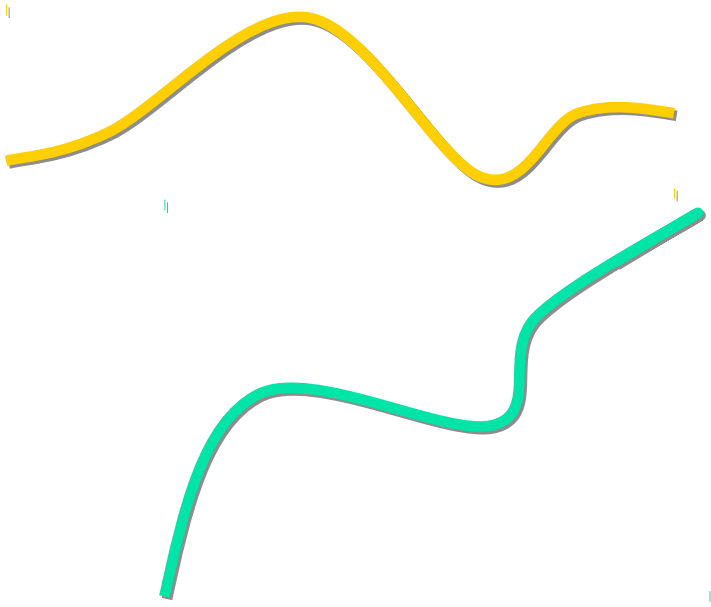
Descriptor	Harris3D Size	Harris3D Time	Sub-sampl. 1cm Size	Sub-sampl. 1cm Time	Sub-sampl. 2cm Size	Sub-sampl. 2cm Time
3DSC	143	475	1011	1206	281	504
CVFH	1.1	15	2.1	13	1.4	13
ESF	5.7	25	5.8	15	5.7	15
FPFH	12	241	69	240	21	228
PFH	21	1054	131	6049	40	1668
PFHRGB	40	1883	249	10753	76	2992
PCE	1.8	77	11	36	3.3	13
PPF	158	92	5400	1550	461	153
RIFT	7.8	15	68	19	17	14
SHOT	50	112	310	175	94	76
SHOTCOLOR	159	121	987	676	297	178
USC	142	184	1014	1195	285	381
VFH	0.8	15	1.8	14	1	13
Average	57.1	331.5	712.3	1687.8	121.8	480.6

Alexandre, L., 3D Descriptors for Object and Category Recognition: a Comparative Evaluation

Alignment

- PFH correspondences + RANSAC can be good at estimating an initial alignment
- Often the alignment is off by a little bit
- Or perhaps we already have a good estimate of the alignment of two point clouds from some other source?
  - Viewpoint is roughly in the same place
  - Use SIFT in 2D
- How can we remove that last bit of error?

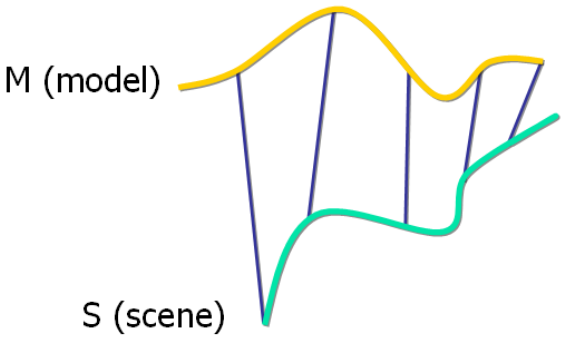
Aligning 3D Data



Slides stolen from Ronen Gvili

Corresponding Point Set Alignment

- Let M be a model point set.
- Let S be a scene point set.



We assume :

1. $N_M = N_S$.
2. Each point S_i correspond to M_i.

Slides stolen from Ronen Gvili

Corresponding Point Set Alignment

The MSE objective function :

$$f(R,T) = \frac{1}{N_S}\sum_{i=1}^{N_S} \left\| m_i - Rot(s_i) - Trans \right\|^2$$

$$f(q) = \frac{1}{N_S}\sum_{i=1}^{N_S} \left\| m_i - R(q_R)s_i - q_T \right\|^2$$

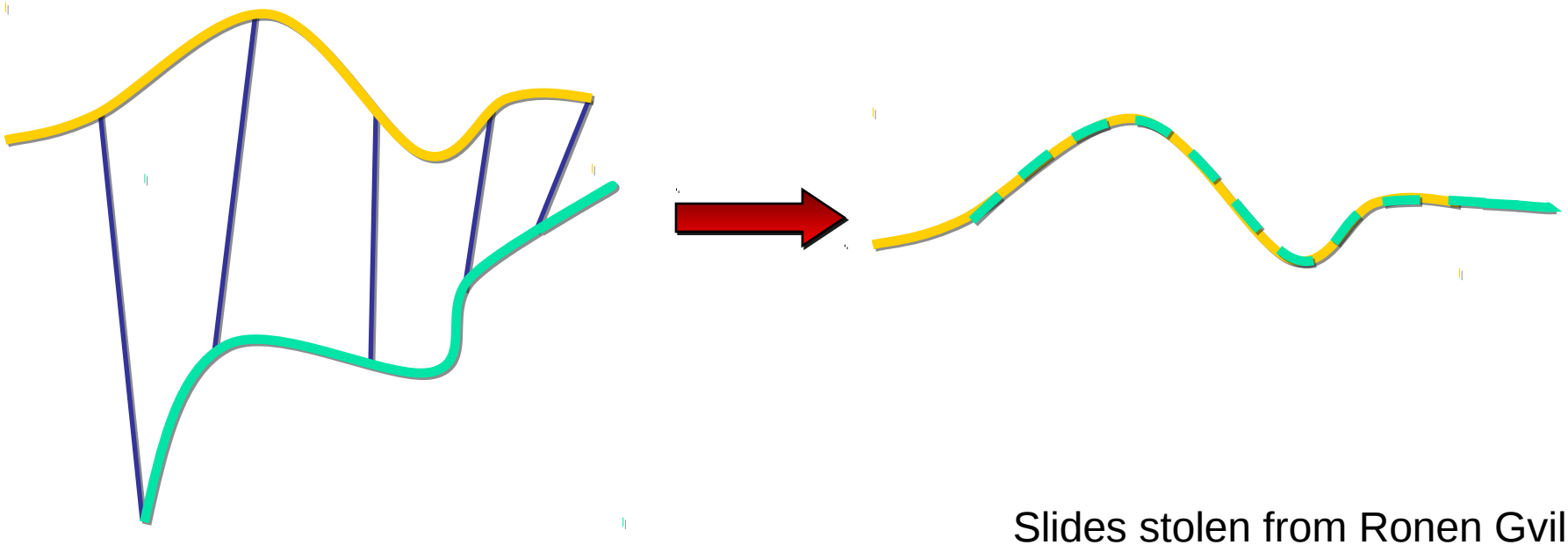
The alignment is :

$$(rot, trans, d_{mse}) = \Phi\,(M, S)$$

Slides stolen from Ronen Gvili

Aligning 3D Data

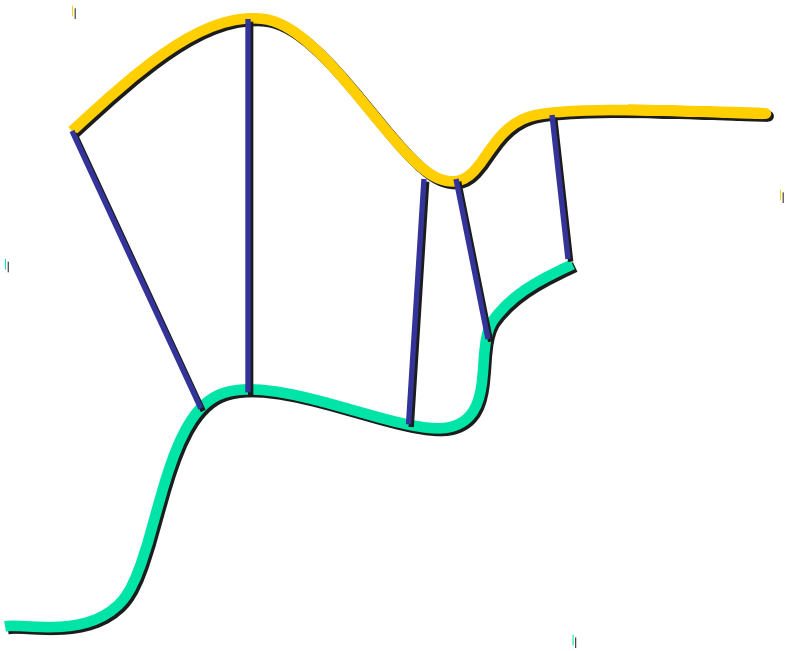
- If correct correspondences are known, can find correct relative rotation/translation



Slides stolen from Ronen Gvili

Aligning 3D Data

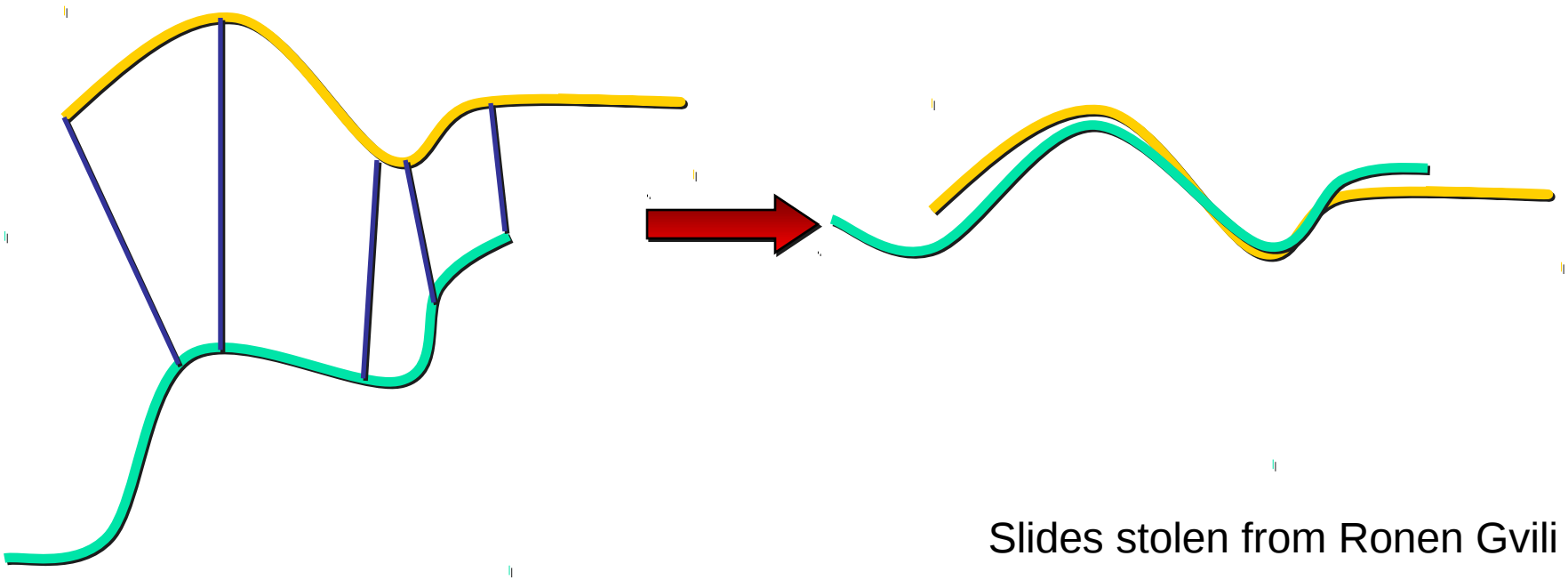
- How to find correspondences: User input? Feature detection? Signatures?
- Alternative: assume closest points correspond



Slides stolen from Ronen Gvili

Aligning 3D Data

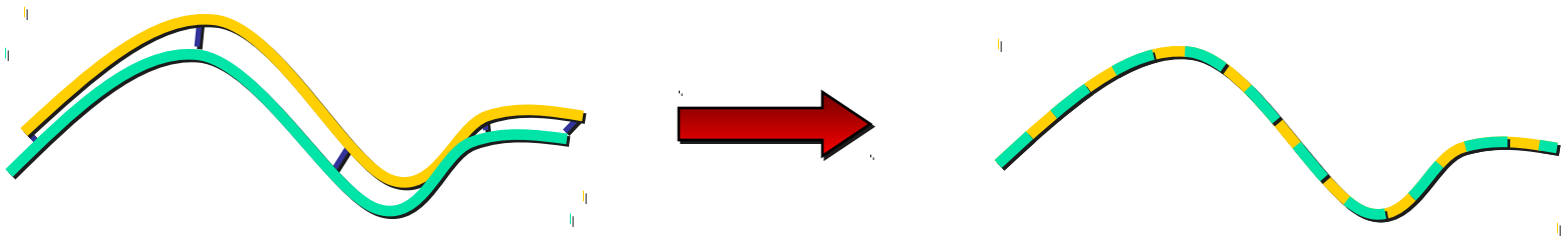
- How to find correspondences: User input? Feature detection? Signatures?
- Alternative: assume closest points correspond



Slides stolen from Ronen Gvili

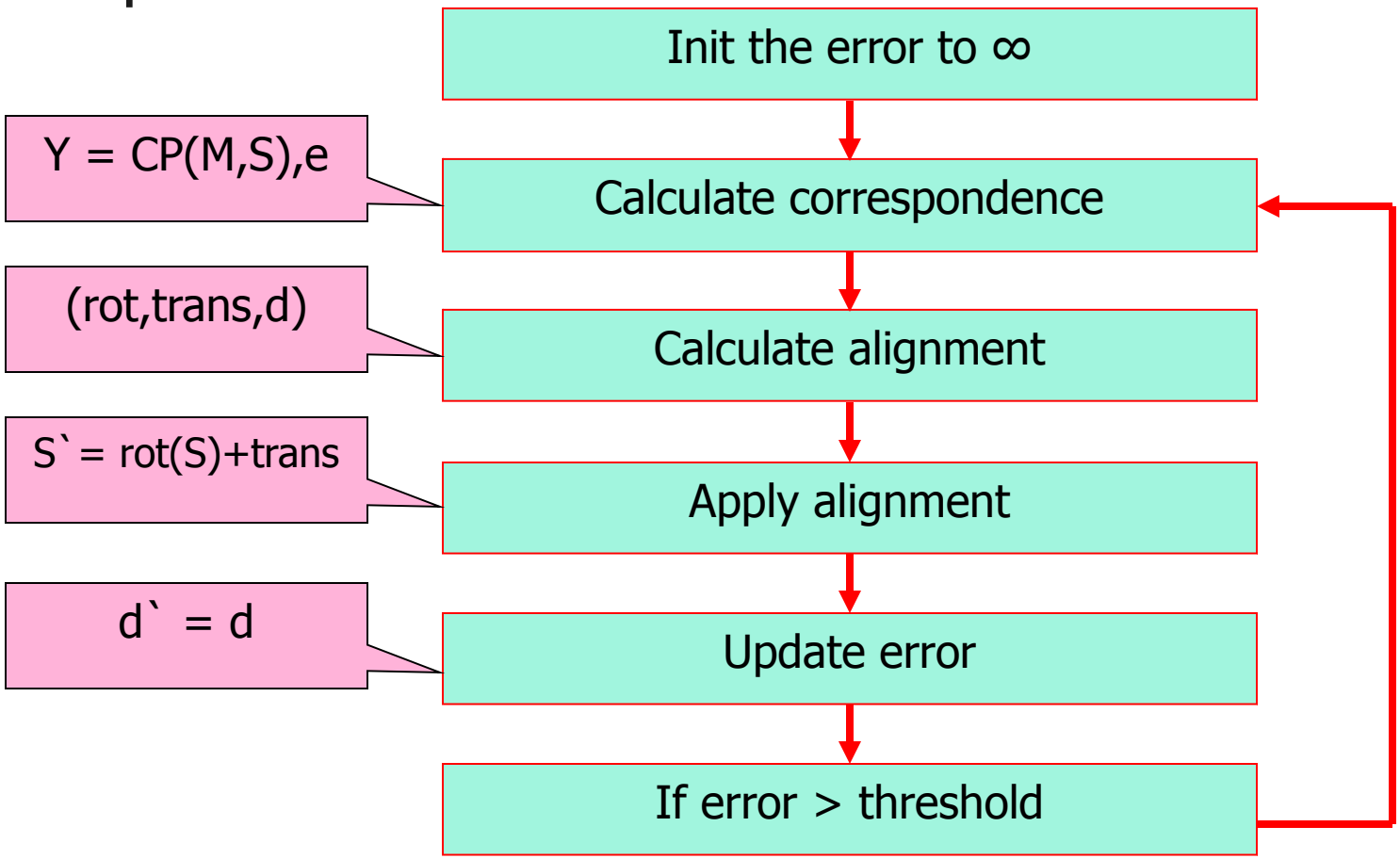
Aligning 3D Data

- Converges if starting position “close enough“

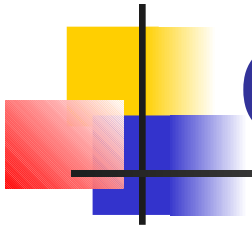


Slides stolen from Ronen Gvili

The Algorithm



Slides stolen from Ronen Gvili



Convergence Theorem

- The ICP algorithm always converges monotonically to a local minimum with respect to the MSE distance objective function.

Slides stolen from Ronen Gvili

RANSAC Segmentation

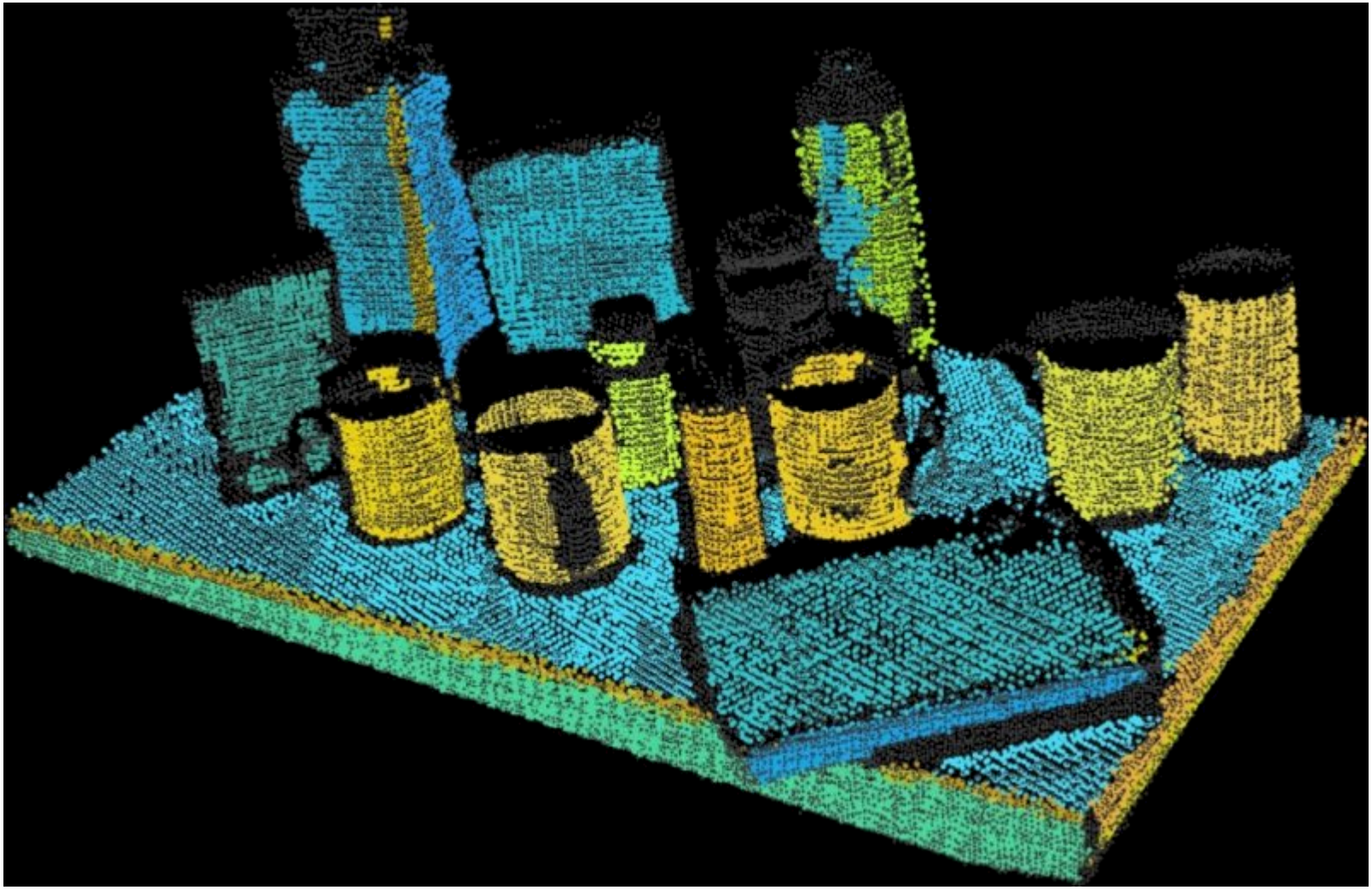
- RANSAC is a very general algorithm
  - Have some model we want to fit
  - Some reasonable percentage of the dataset fits the model
  - Find the best model by subsampling, fitting, reprojecting, and evaluating the model
- Plane model:

$$ax+by+cz+d=0$$

- A limited cylinder model:

$$(x-a)^2+(y-b)^2=r^2$$

RANSAC Cylinder Segmentation



pointclouds.org

Point Cloud Software

- Point Cloud Library (PCL)
  - http://pointclouds.org
- Robot Operating System (ROS)
  - Framework for building systems
  - http://www.ros.org
- Drivers for Kinect and other PrimeSense sensors
  - http://www.ros.org/wiki/openni_launch