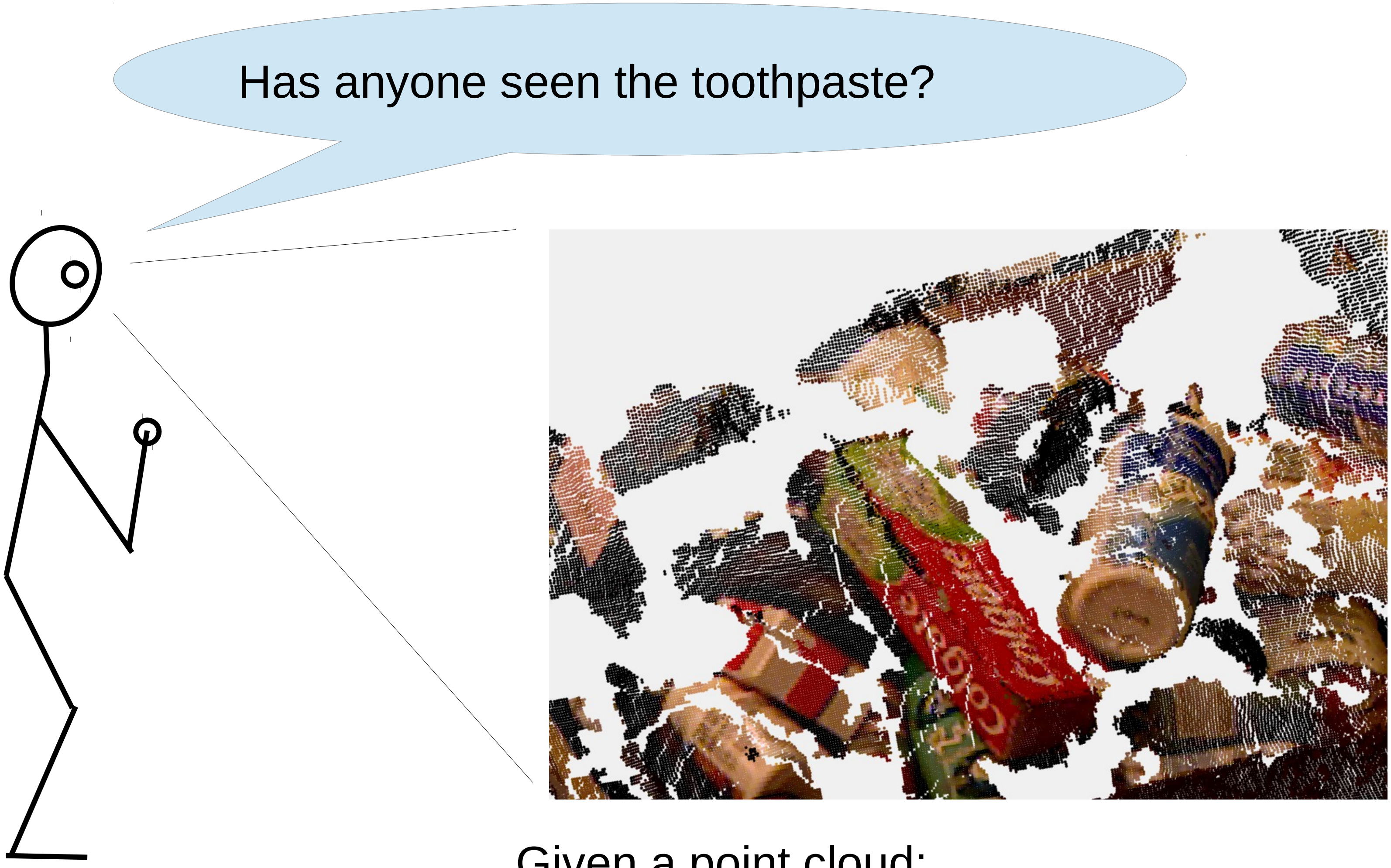
Point Cloud Processing



Given a point cloud:

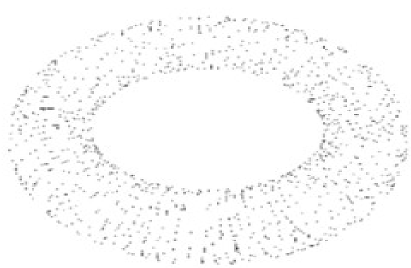
– how do you detect and localize objects?

– how do you map terrain?

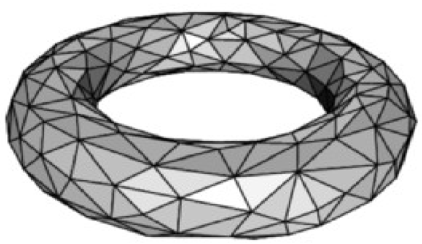
What is a point cloud?

Point cloud: a set of points in 3-D space
– just a set of 3-d points

Mesh: each point is a vertex of a triangulated face
– a set of vertices AND connectivity information



Point cloud



Mesh

Images from Course INF 555 slides, Ecole Polytechnique, Paris

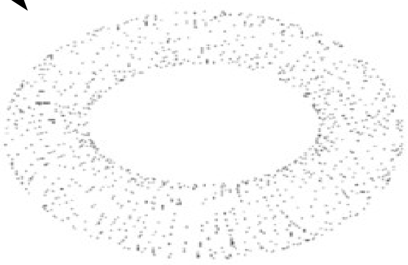
What is a point cloud?

Point cloud: a set of points in 3-D space
– just a set of 3-d points

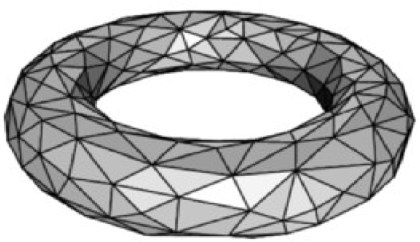
Mesh: each point is a vertex of a triangulated face
– a set of vertices AND connectivity information

Many depth sensors produce point clouds natively

But, a mesh contains a lot more information



Point cloud

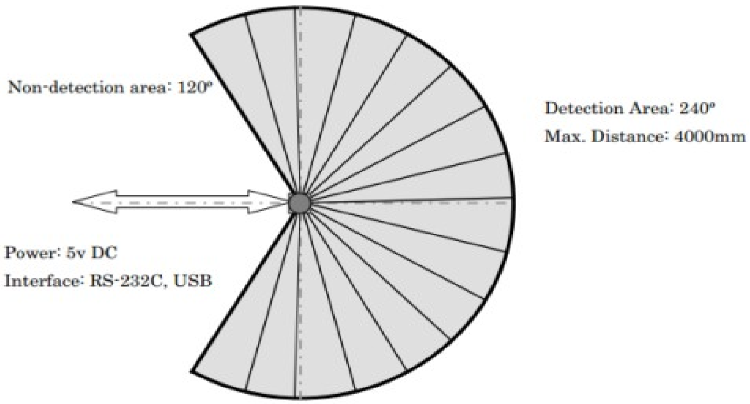


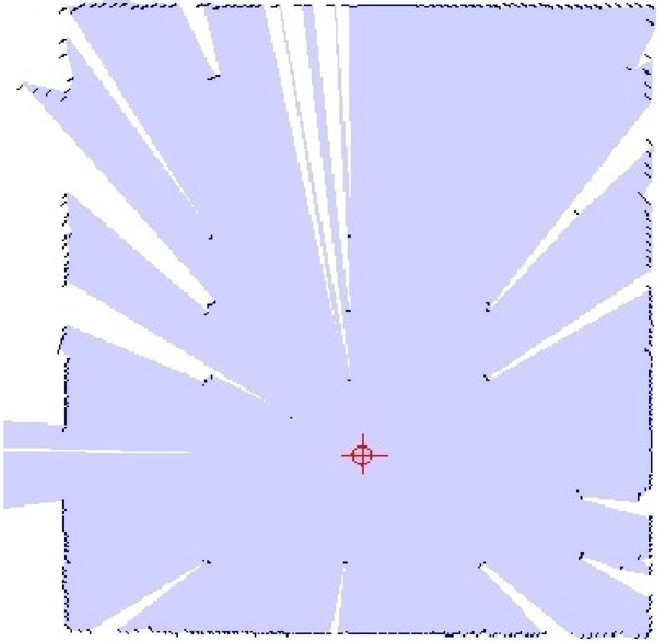
Mesh

Images from Course INF 555 slides, Ecole Polytechnique, Paris

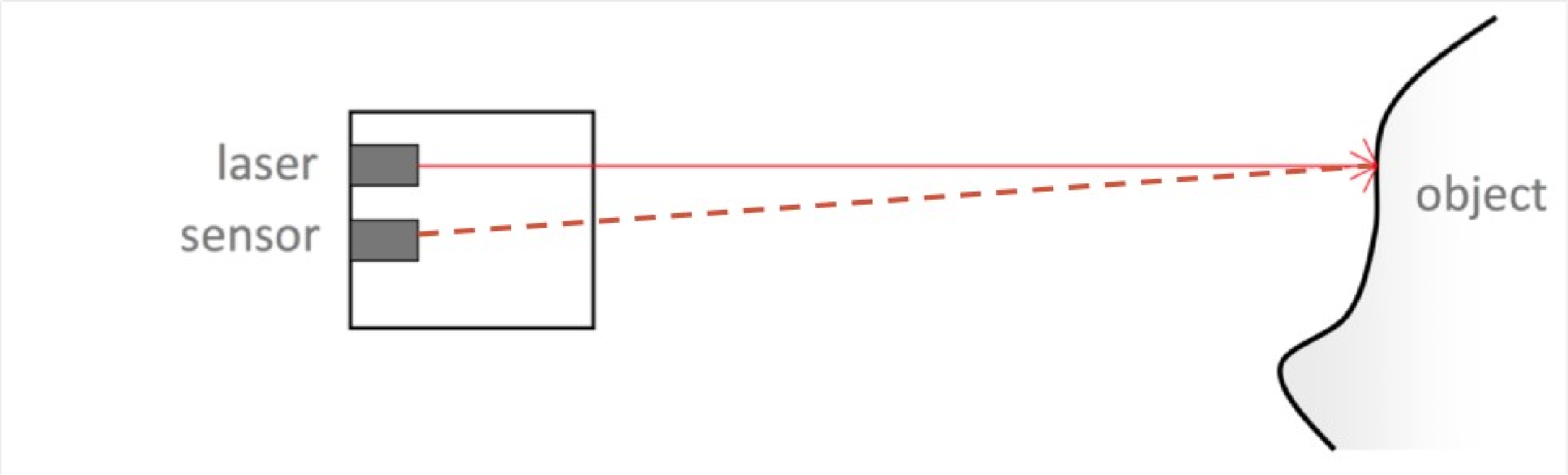
Time of flight sensors

Hokuyo UTM-30LX-EW Scanning Laser Range Finder





Time of flight sensors



1. Emit a short short pulse of laser
2. Capture the reflection.
3. Measure the time it took to come back.
4. Need a very fast clock.
5. Main advantage: can be done over long distances.
6. Used in terrain scanning.

Slide from Course INF 555 slides, Ecole Polytechnique, Paris

Time of flight sensors

source: Michael Wand

Structured light sensors

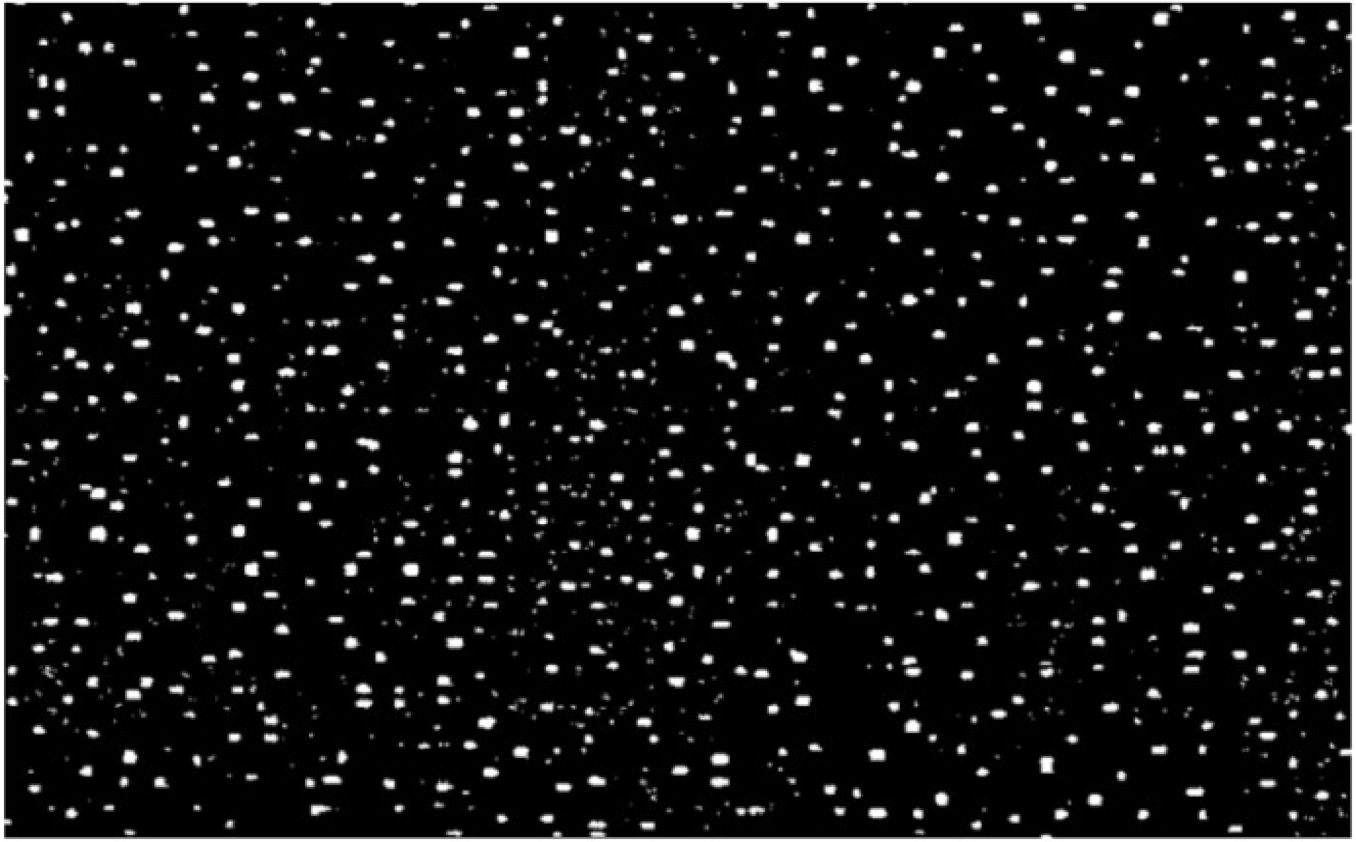
Structured light general principle:

project a known pattern onto the scene and infer depth from the deformation of that pattern

Zhang et al, 3DPVT (2002)

Slide: John MacCormick, Dickinson University

The Kinect uses infrared laser light, with a speckle pattern



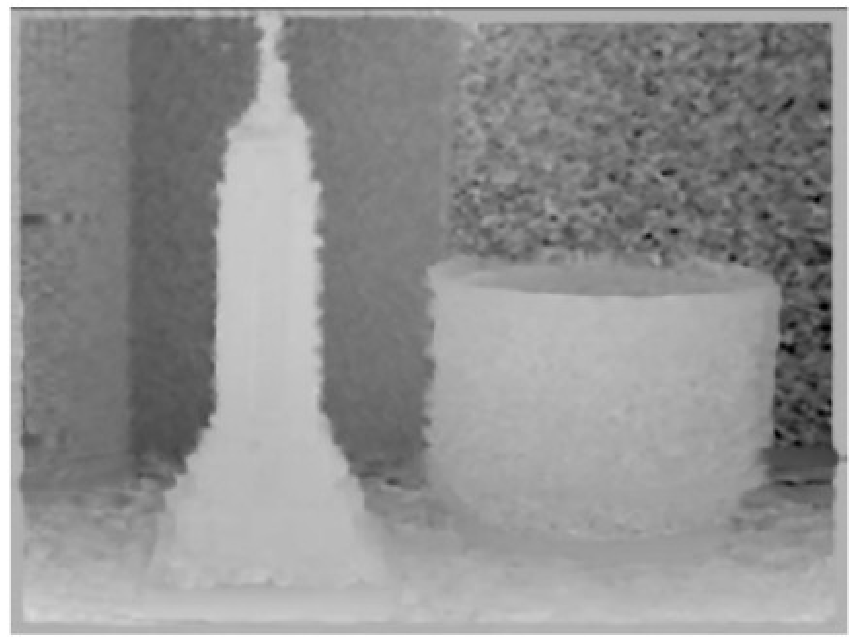
Shpunt et al, PrimeSense patent application
US 2008/0106746

Slide: John MacCormick, Dickinson University

Stage 1: The depth map is constructed by analyzing a speckle pattern of infrared laser light

- The Kinect uses an infrared projector and sensor; it does not use its RGB camera for depth computation
- The technique of analyzing a known pattern is called *structured light*
- The Kinect combines structured light with two classic computer vision techniques: depth from focus, and depth from stereo

Depth from focus uses the principle that stuff that is more blurry is further away



Watanabe and Nayar, IJCV 27(3), 1998

Slide: John MacCormick, Dickinson University

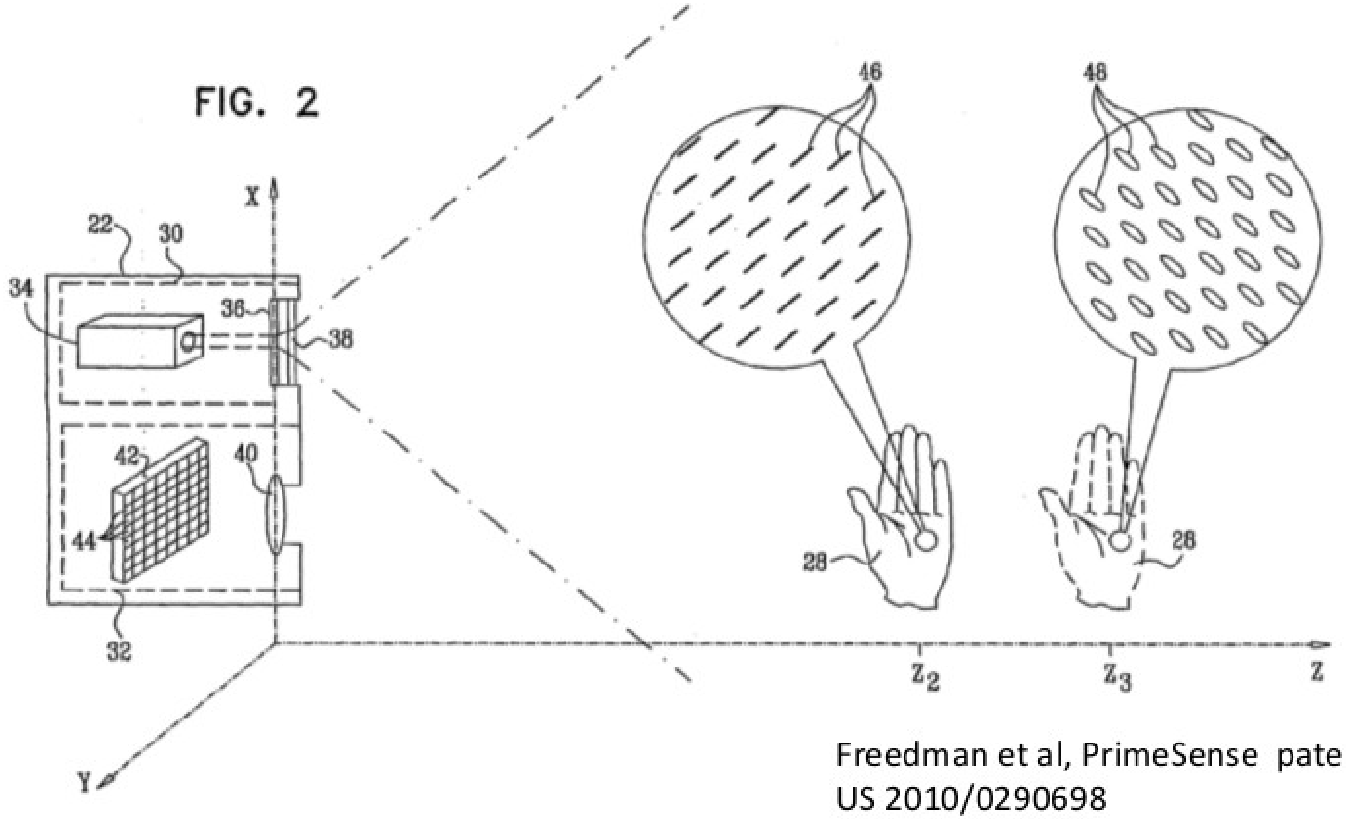
Depth from focus uses the principle that stuff that is more blurry is further away

- The Kinect dramatically improves the accuracy of traditional depth from focus
- The Kinect uses a special (“astigmatic”) lens with different focal length in x- and y-directions
- A projected circle then becomes an ellipse whose orientation depends on depth

Slide: John MacCormick, Dickinson University

The astigmatic lens causes a projected circle to become an ellipse whose orientation depends on depth

FIG. 2

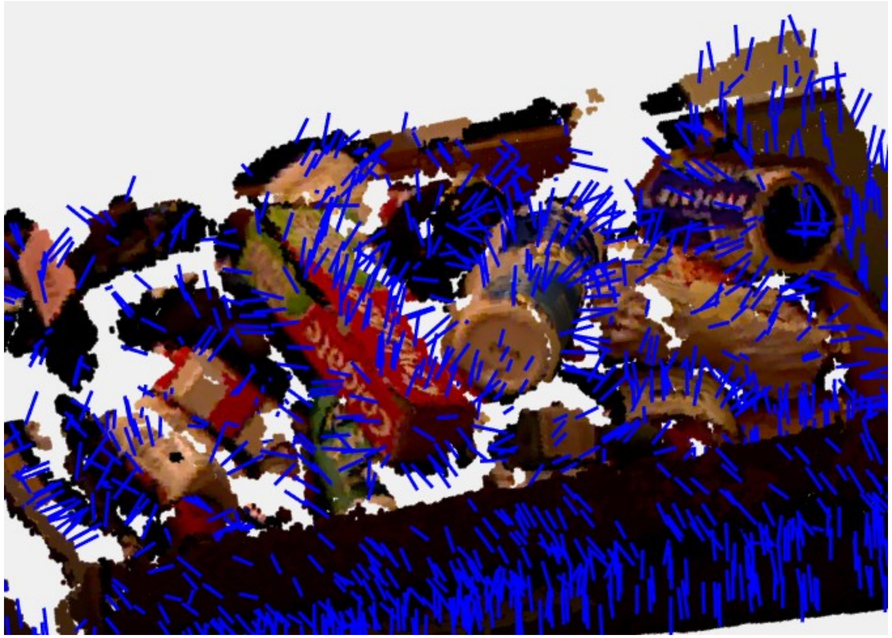


Freedman et al, PrimeSense patent application US 2010/0290698

Slide: John MacCormick, Dickinson University

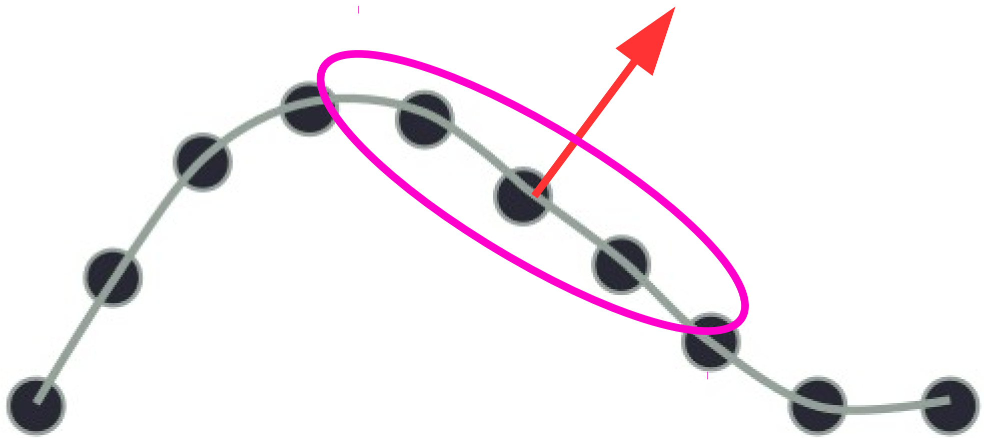
Calculating surface normals

Point cloud



Point cloud w/ surface normals
(normals are subsampled)

Calculating surface normals



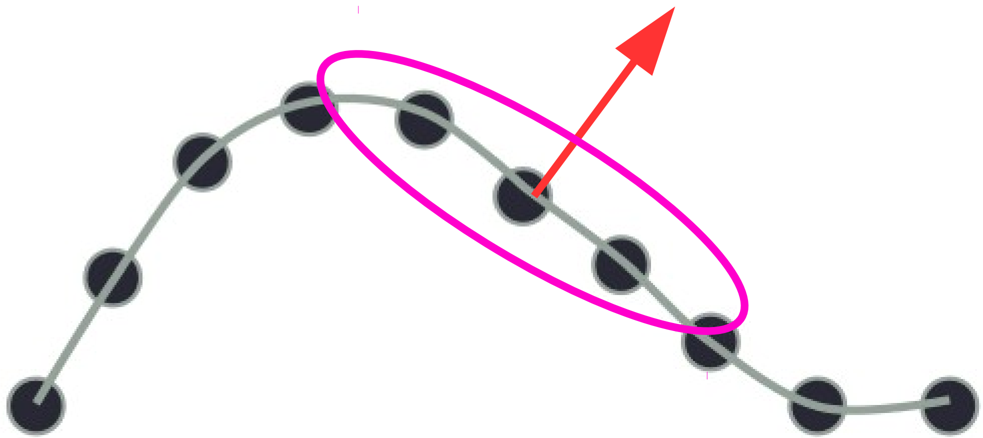
Question: How do we calculate the surface normal given only points?

Answer:

1. Calculate the sample covariance matrix of the points within a local neighborhood of the surface normal

2. Take Eigenvalues/Eigenvectors of that covariance matrix

Calculating surface normals



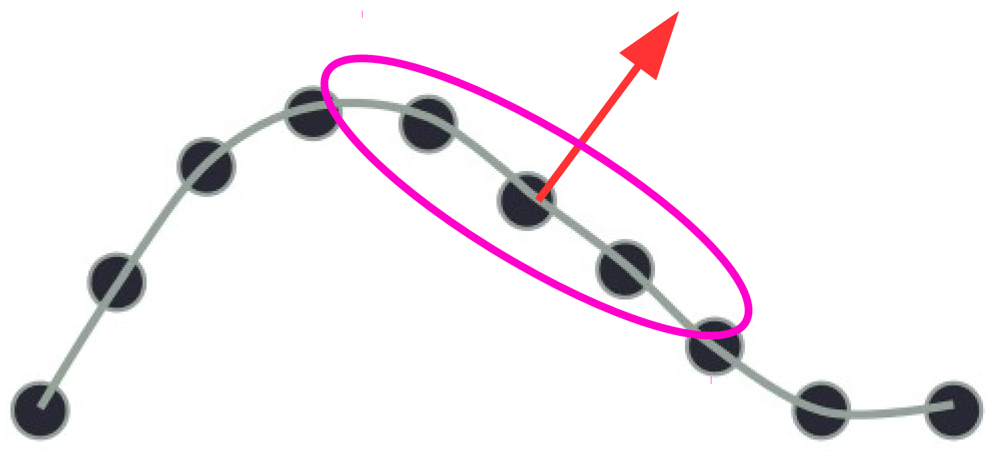
Let *C* denote the set of points in the point cloud

Suppose we want to calculate the surface normal at $x \in C$

Let $B_r(x) \subseteq \mathbb{R}^3$ denote the r-ball about *x*

$N_r(x) = B_r(x) \cap C$ is the set of points in the cloud within *r* of *x*

Calculating surface normals

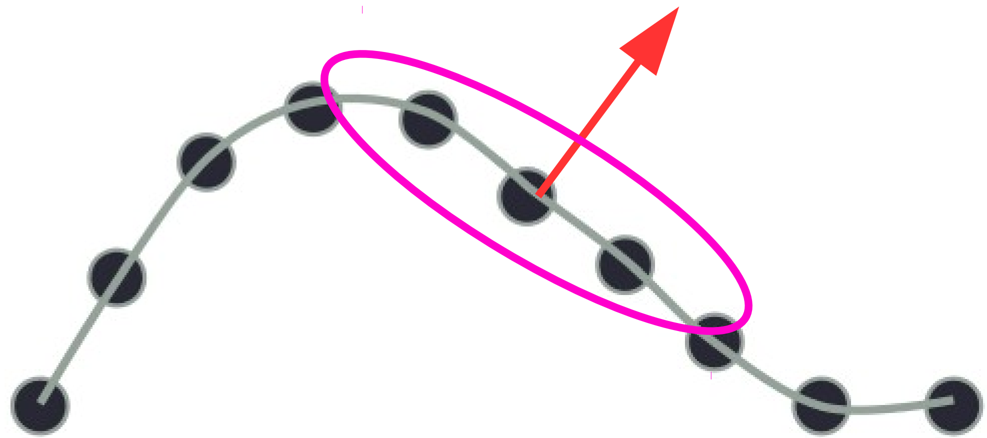


Calculate the sample covariance matrix of the points in $N_r(x)$

$$\Sigma = \sum_{p \in N_r(x)} (p - \bar{p})(p - \bar{p})^T$$

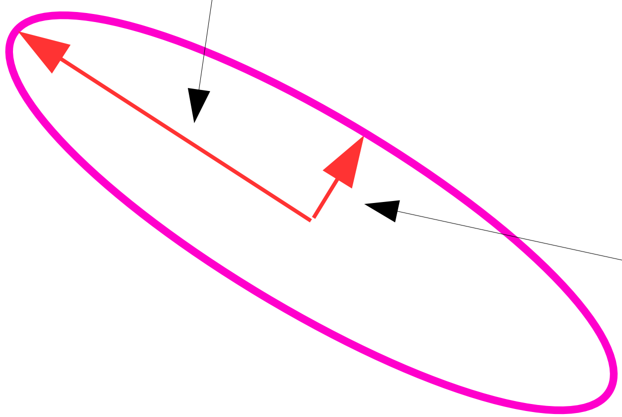
$$\bar{p} = \frac{1}{|N_r(x)|} \sum_{p \in N_r(x)} p$$

Calculating surface normals



Length = $\sqrt{\lambda_{max}}$

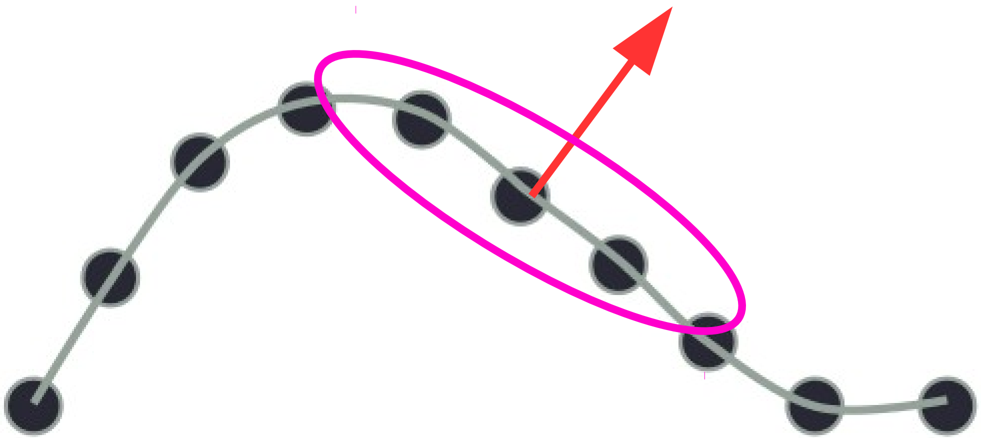
Eigenvalues of Σ



Length = $\sqrt{\lambda_{min}}$

Principle axes of ellipse point in directions of corresponding eigenvectors

Calculating surface normals



So: surface normal is in the direction of the Eigenvector corresponding to the smallest Eigenvalue of $\sum$

Calculating surface normals: Summary

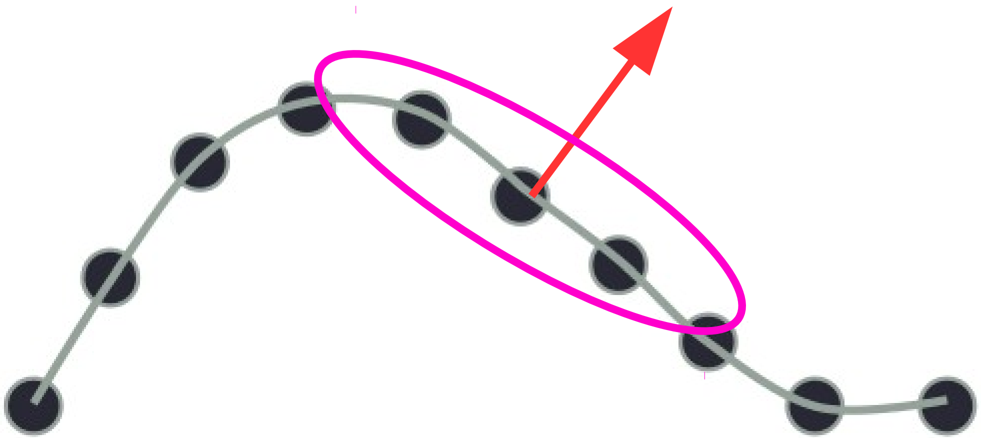
1. calculate points within r-ball about x: $N_r(x) = B_r(x) \cap C$

2. calculate covariance matrix: $\Sigma = \sum_{p \in N_r(x)} (p - \bar{p})(p - \bar{p})^T$

3. calculate Eigenvectors: v_1, v_2, v_3
   and Eigenvalues: $\lambda_1, \lambda_2, \lambda_3$ (\lambda_3 is smallest)

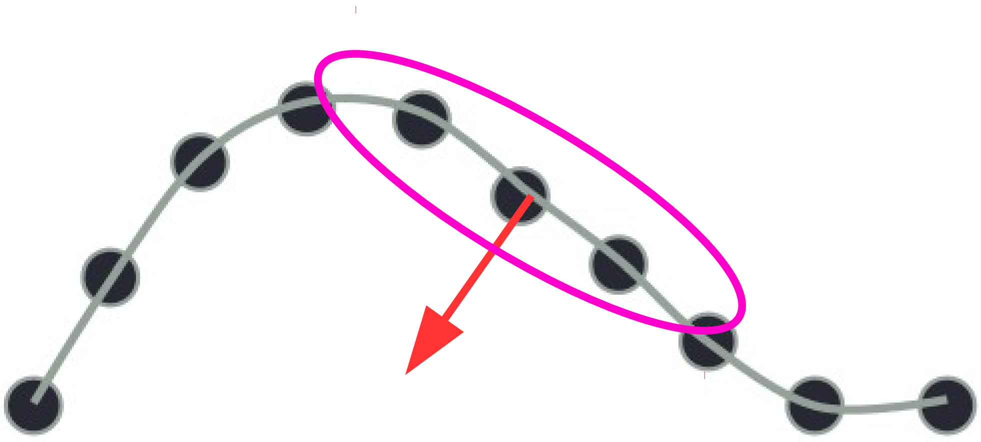
4. v_3 is parallel or antiparallel to surface normal

Calculating surface normals



Important note: the points alone do not tell us the *sign* of the surface normal

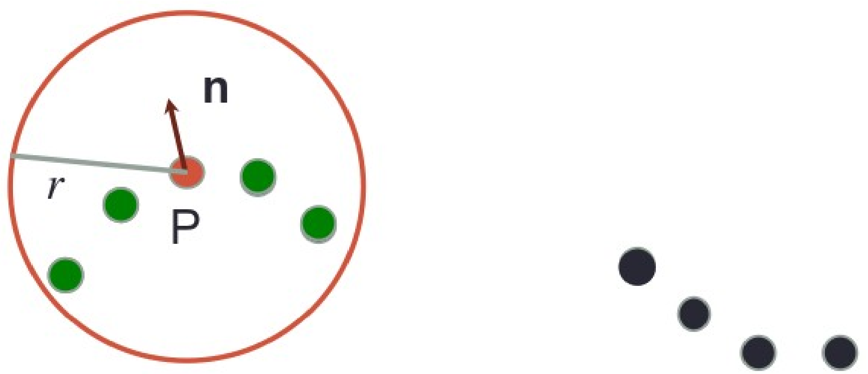
Calculating surface normals



Important note: the points alone do not tell us the *sign* of the surface normal

Calculating surface normals

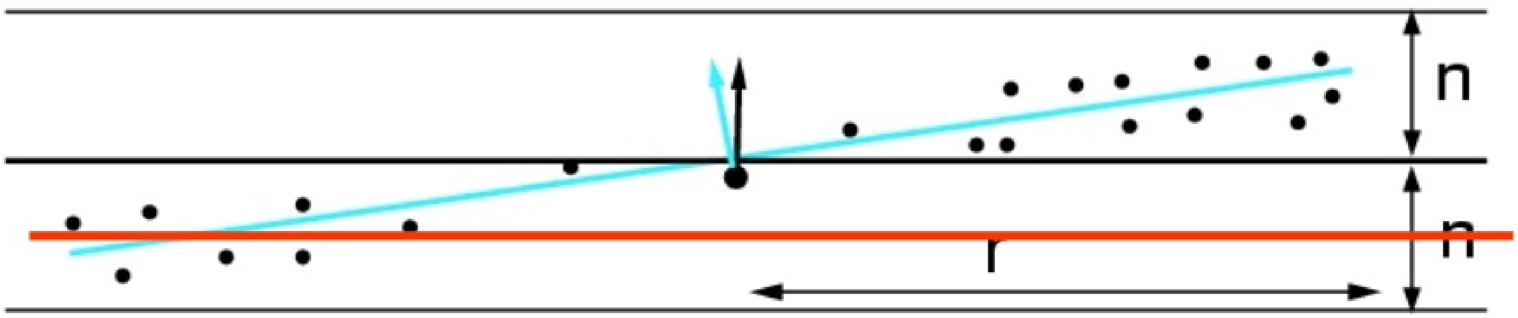
How large a point neighborhood to use when calculating $\sum$?



Because points can be uneven, don't use k-nearest neighbor.
– it's important to select a radius *r* and stick w/ it.
– which what value of *r* to use?

Calculating surface normals

Collusive noise

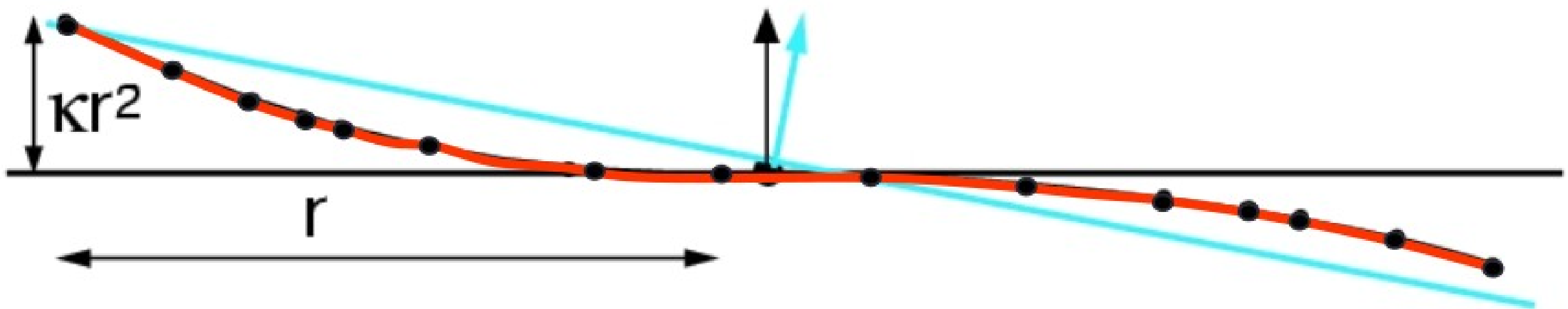


Because of noise in the data, small *r* may lead to underfitting.

Images from Course INF 555 slides, Ecole Polytechnique, Paris

Calculating surface normals

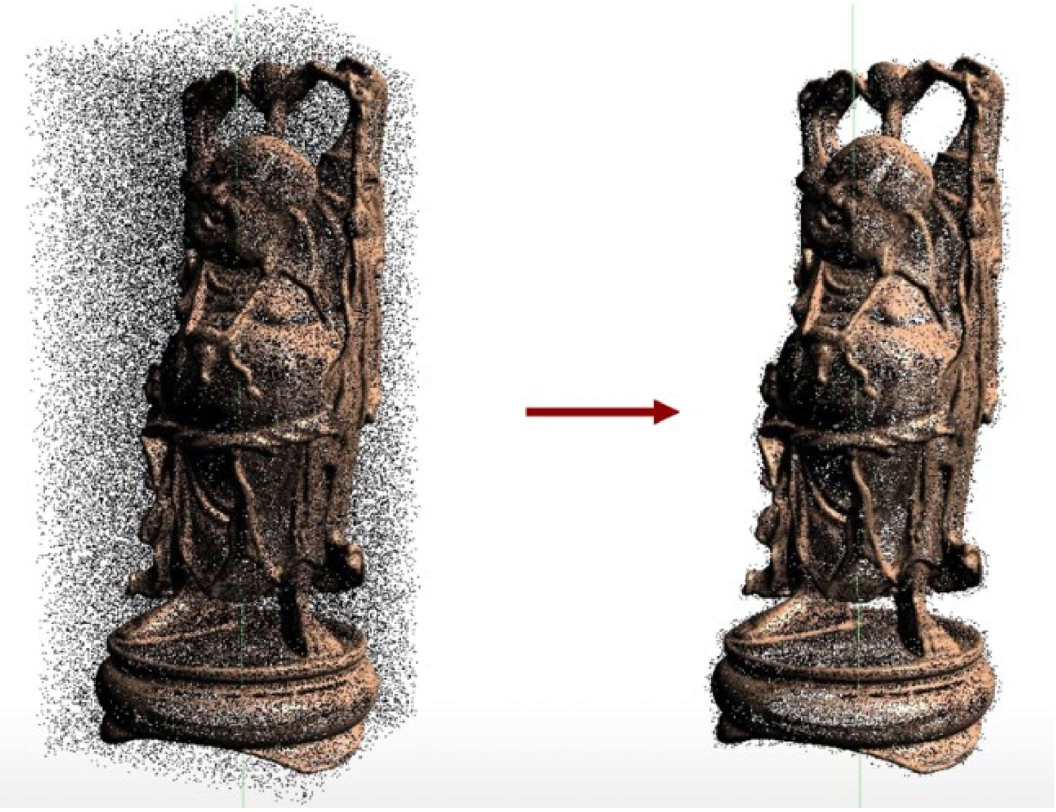
Curvature effect



Due to curvature, large *r* can lead to estimation bias.

Images from Course INF 555 slides, Ecole Polytechnique, Paris

Outlier removal

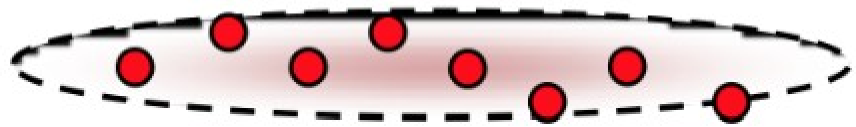


Similar approach as in normal estimation:
1. calculate local covariance matrix
2. estimate Eigenvectors/Eigenvalues
3. use that information somehow...

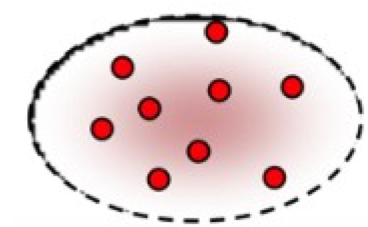
Images from Course INF 555 slides, Ecole Polytechnique, Paris

Outlier removal

If points lie on a line, then $\frac{\lambda_{min}(\Sigma)}{\lambda_{max}(\Sigma)}$ is small



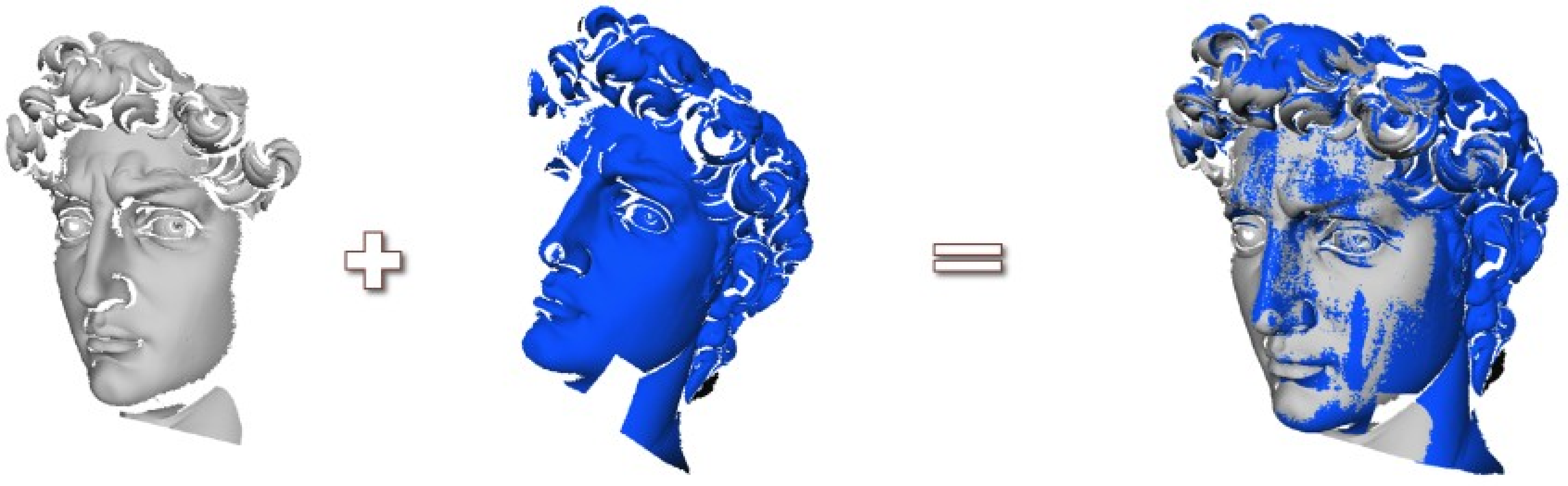
If points are uniformly random, then $\frac{\lambda_{min}(\Sigma)}{\lambda_{max}(\Sigma)}$ is close to 1



Outlier removal: delete all points for which $\frac{\lambda_{min}(\Sigma)}{\lambda_{max}(\Sigma)}$ is above a threshold

Images from Course INF 555 slides, Ecole Polytechnique, Paris

Point cloud registration: ICP



Find an affine transformation that aligns two partially overlapping point clouds

Images from Course INF 555 slides, Ecole Polytechnique, Paris

ICP Problem Statement

- Given: two corresponding point sets:

$$X = \{x_1, ..., x_n\}$$

$$P = \{p_1, ..., p_n\}$$

- Wanted: translation t and rotation R that minimizes the sum of the squared error:

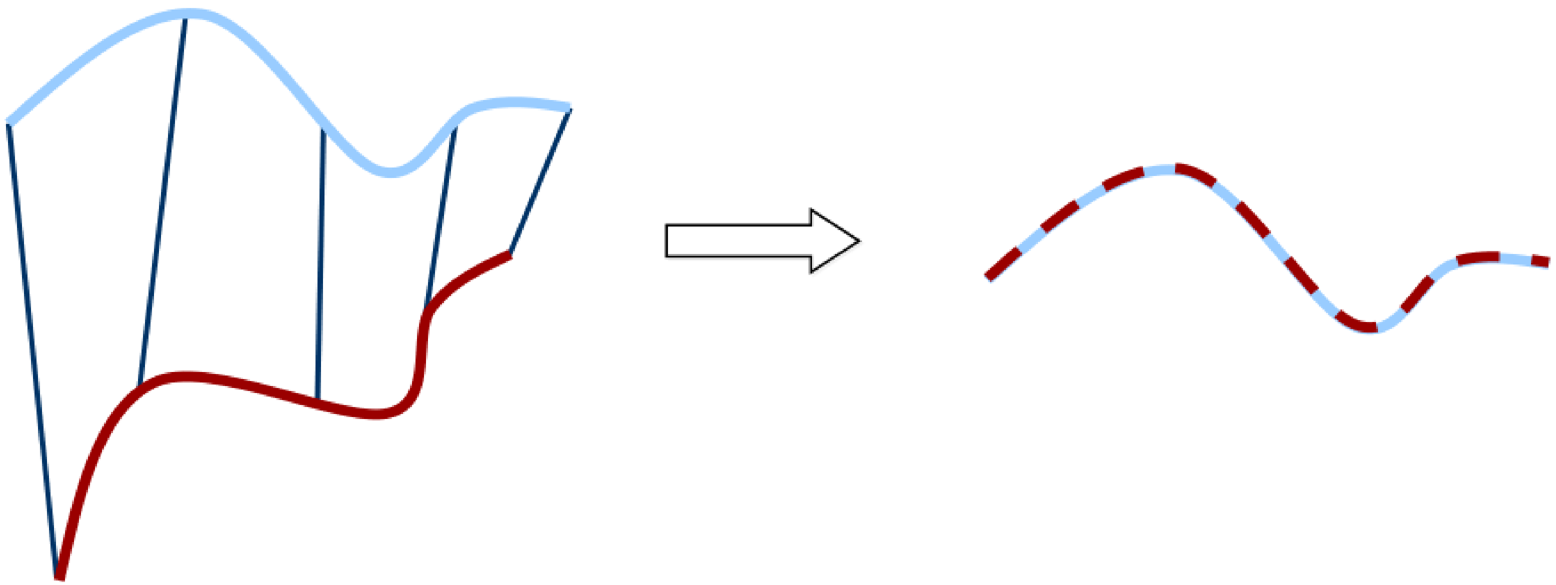
$$E(R,t) = \frac{1}{N_p} \sum_{i=1}^{N_p} ||x_i - Rp_i - t||^2$$

Where x_i and p_i are corresponding points.

This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

ICP: key idea

- If the correct correspondences are known, the correct relative rotation/translation can be calculated in closed form.



This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

Step 1: center the two point clouds

$$\mu_x = \frac{1}{N_x} \sum_{i=1}^{N_x} x_i \quad \text{and} \quad \mu_p = \frac{1}{N_p} \sum_{i=1}^{N_p} p_i$$

are the centers of mass of the two point sets.

Idea:

- Subtract the corresponding center of mass from every point in the two point sets before calculating the transformation.
- The resulting point sets are:

$$X' = \{x_i - \mu_x\} = \{x'_i\}$$

and

$$P' = \{p_i - \mu_p\} = \{p'_i\}$$

This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

Step 2: use SVD to get min t and R

Let $W = \sum_{i=1}^{N_p} x'_i p'^T_i$

denote the singular value decomposition (SVD) of W by:

$$W = U \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} V^T$$

where $U, V \in \mathbb{R}^{3\times 3}$ are unitary, and

$\sigma_1 \geq \sigma_2 \geq \sigma_3$ are the singular values of W.

This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

Step 2: use SVD to get min t and R

Theorem (without proof):

If rank(W) = 3, the optimal solution of E(R,t) is unique and is given by:

$$R = UV^T$$

$$t = \mu_x - R\mu_p$$

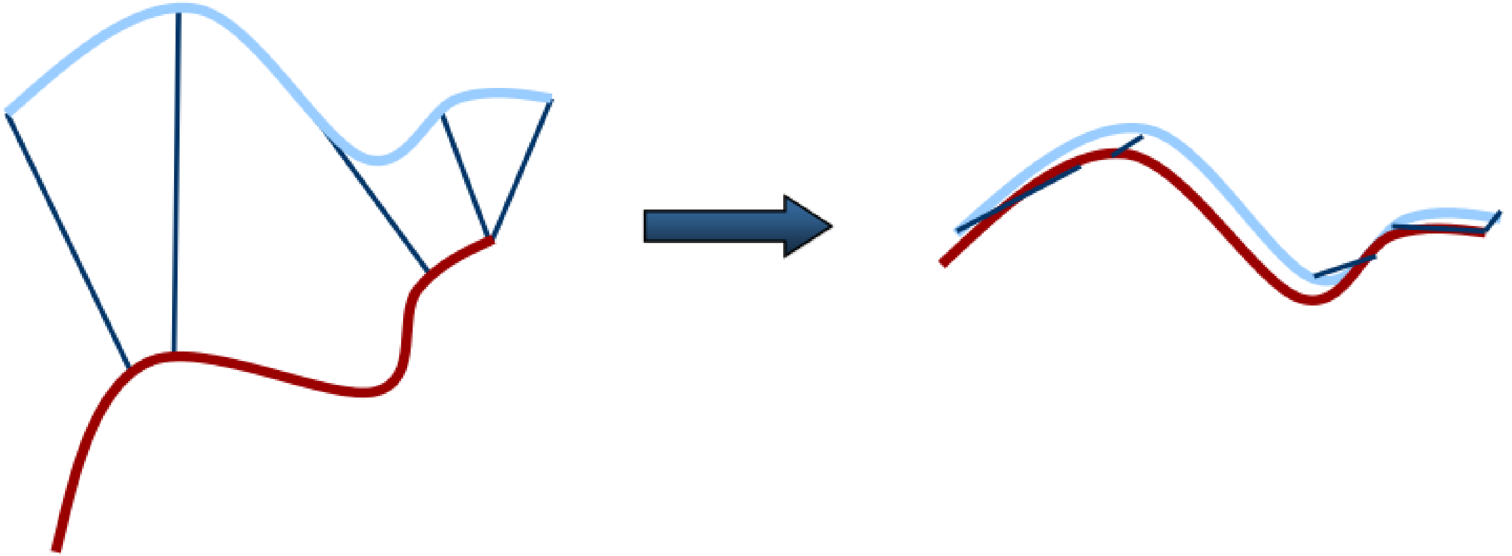
The minimal value of error function at (R,t) is:

$$E(R,t) = \sum_{i=1}^{N_p} (||x_i'||^2 + ||y_i'||^2) - 2(\sigma_1 + \sigma_2 + \sigma_3)$$

This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

ICP data association problem

- If correct correspondences are not known, it is generally impossible to determine the optimal relative rotation/translation in one step



This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

ICP Algorithm

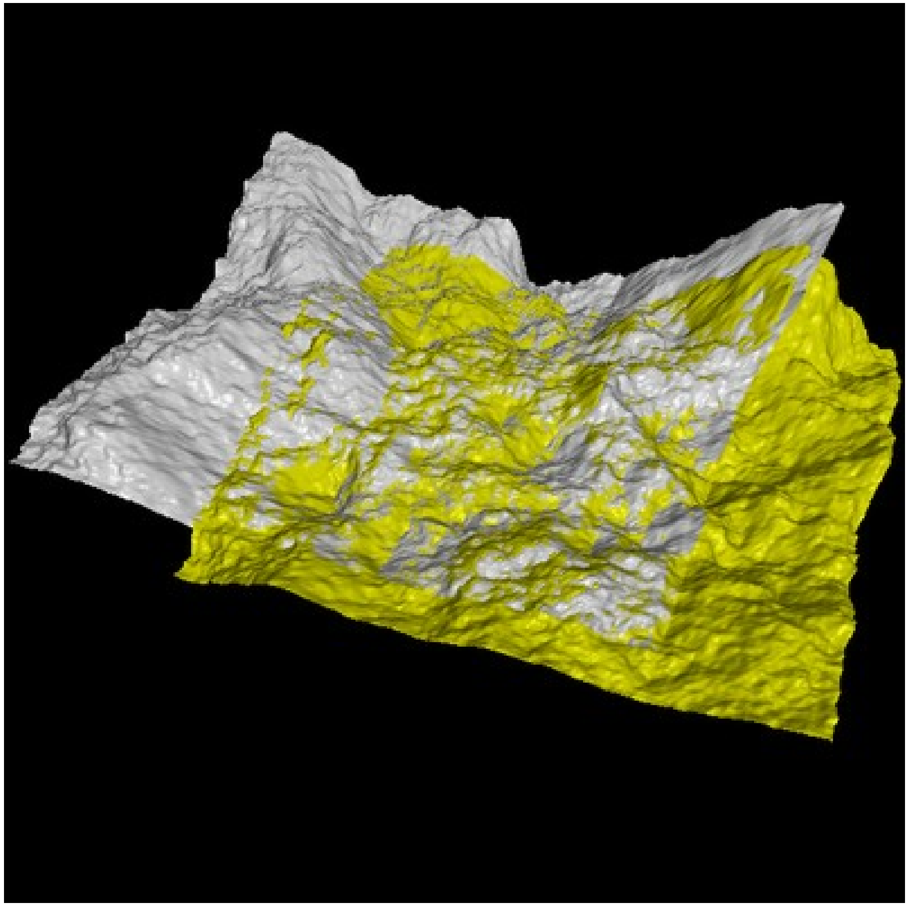
<ins>Input</ins>: two point sets, *X* and *P*

<ins>Output</ins>: translation t and rotation R that best aligns pt sets

1. Start with a “good” alignment
2. Repeat until t and R are small:
3.   for every point in *X*, find its closest neighbor in *P*
4.   find min *t* and *R* for that correspondence assignment
5.   translate and rotate *P* by *t* and *R*
6. Figure out net translation and rotation, *t* and *R*

– Converges if the point sets are initially well aligned
– Besl and McKay, 1992

ICP example



This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

ICP Variants

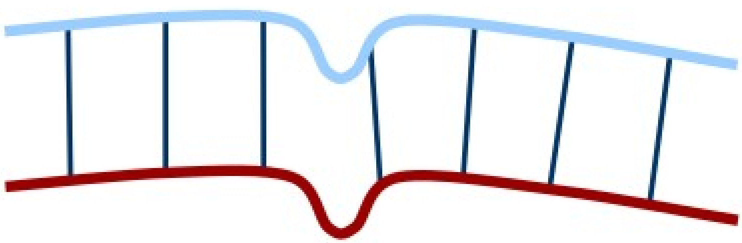
1. Point subsets (from one or both point sets)
2. Weighting the correspondences
3. Data association
4. Rejecting certain (outlier) point pairs

This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

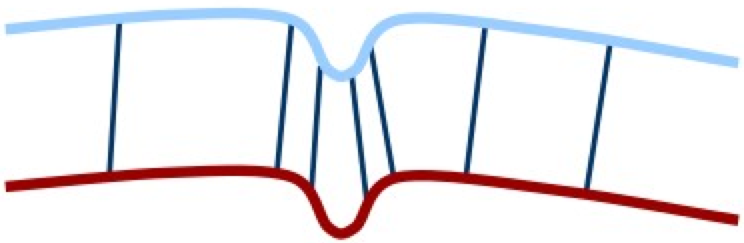
Selecting points to align

- Use all points
- Uniform sub-sampling
- Random sampling
- Feature based Sampling
- Normal-space sampling
  - Ensure that samples have normals distributed as uniformly as possible

Normal-space sampling



uniform sampling

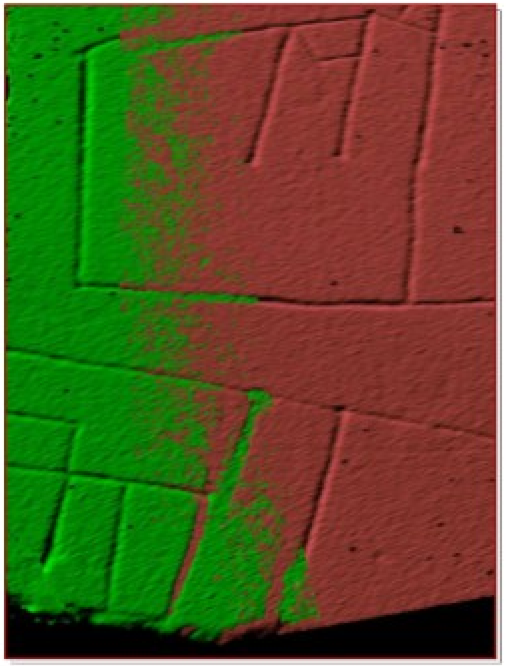


normal-space sampling

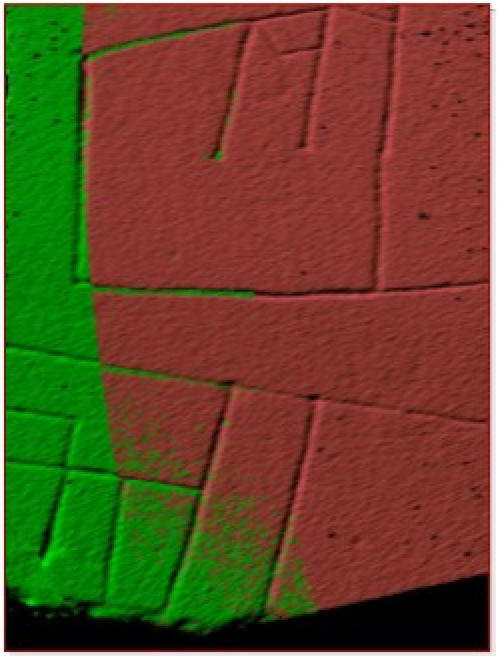
This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

Comparison: normal space sampling vs random

- Normal-space sampling better for mostly-smooth areas with sparse features [Rusinkiewicz et al.]



Random sampling

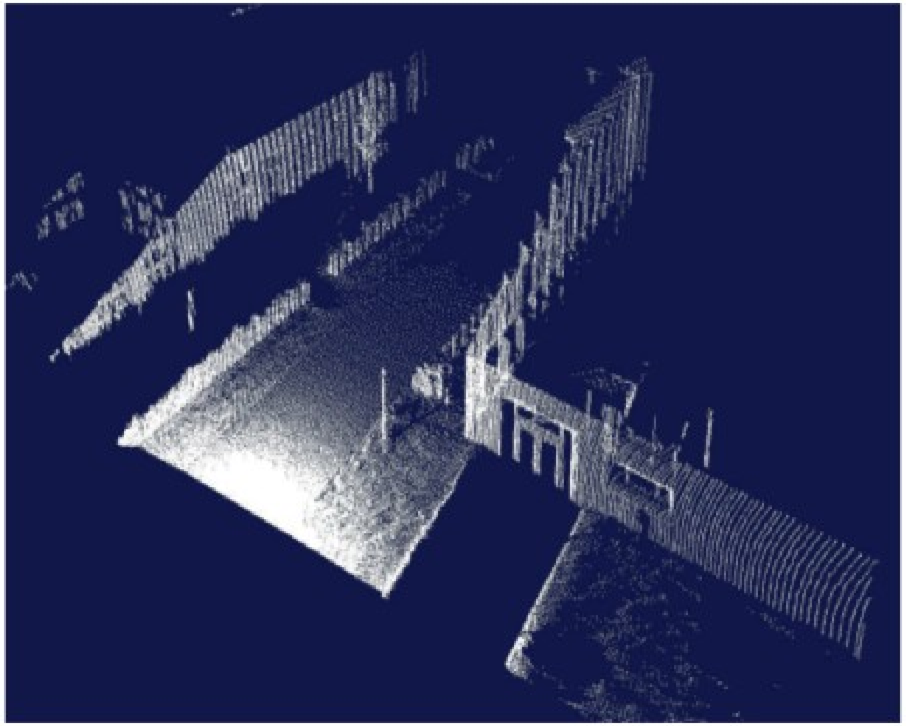


Normal-space sampling

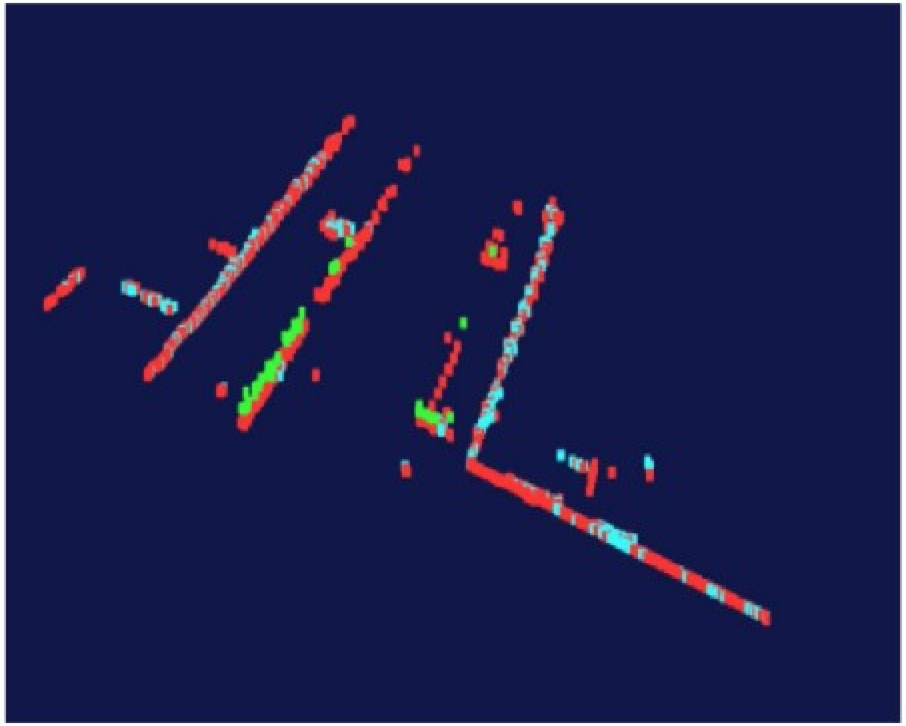
This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

Feature based sampling

- try to find "important" points
- decrease the number of correspondences
- higher efficiency and higher accuracy
- requires preprocessing



3D Scan (~200.000 Points)



Extracted Features (~5.000 Points)

This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

ICP: data association

1. Point subsets (from one or both point sets)
2. Weighting the correspondences
3. Data association
4. Rejecting certain (outlier) point pairs

This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

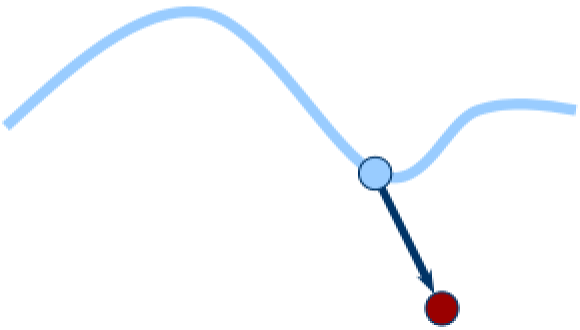
ICP: data association

- has greatest effect on convergence and speed
- Closest point
- Normal shooting
- Closest compatible point
- Projection
- Using kd-trees or oc-trees

This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

Closest point matching

- Find closest point in other the point set

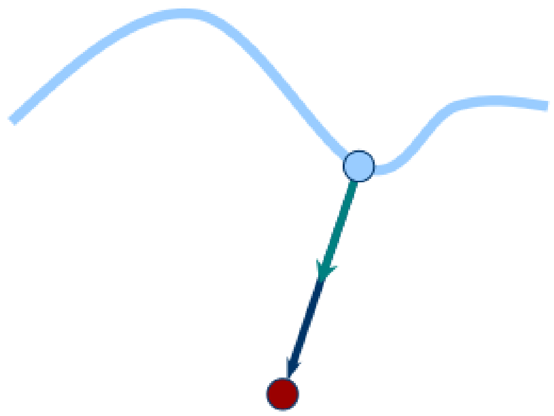


Closest-point matching generally stable, but slow and requires preprocessing

This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

Normal shooting

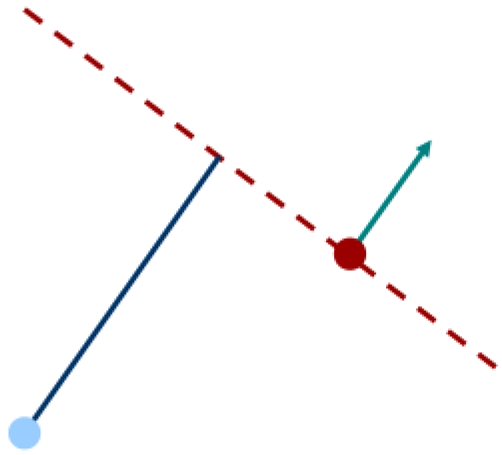
- Project along normal, intersect other point set

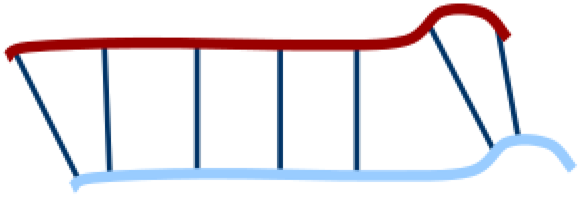


Slightly better than closest point for smooth structures, worse for noisy or complex structures

Point-to-plane distances

- Using point-to-plane distance instead of point-to-point lets flat regions slide along each other [Chen & Medioni 91]





This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

Closest compatible point

- Improves the previous two variants by considering the **compatibility** of the points
- Compatibility can be based on normals, colors, etc.
- In the limit, degenerates to feature matching

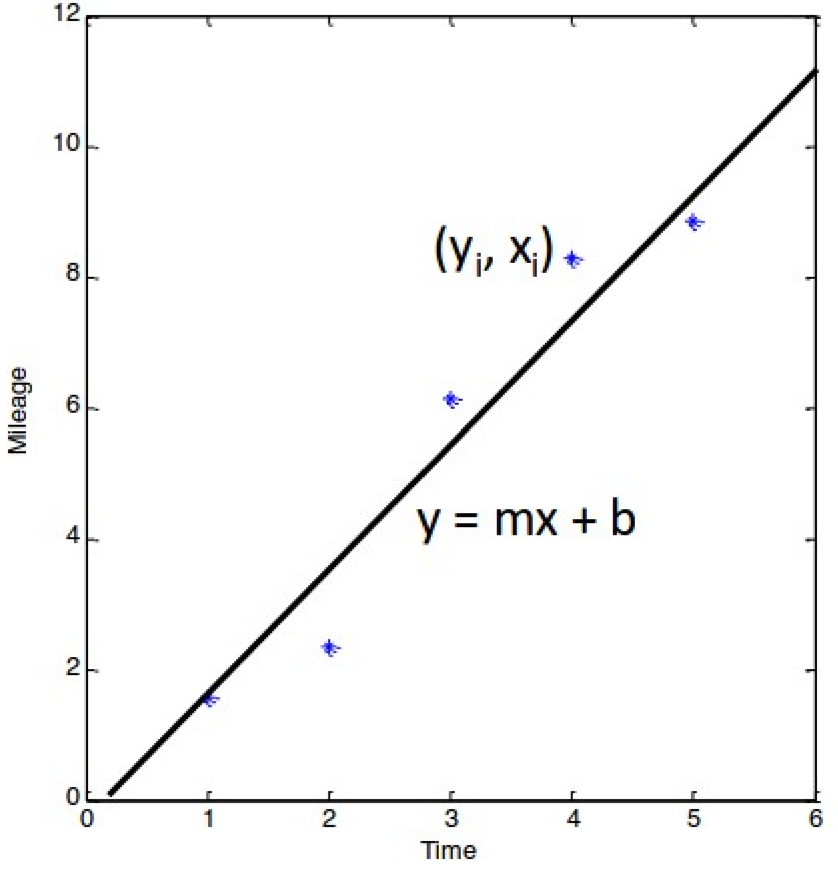
This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

ICP: summary

- ICP is a powerful algorithm for calculating the displacement between scans.
- The major problem is to determine the correct data associations.
- Given the correct data associations, the transformation can be computed efficiently using SVD.

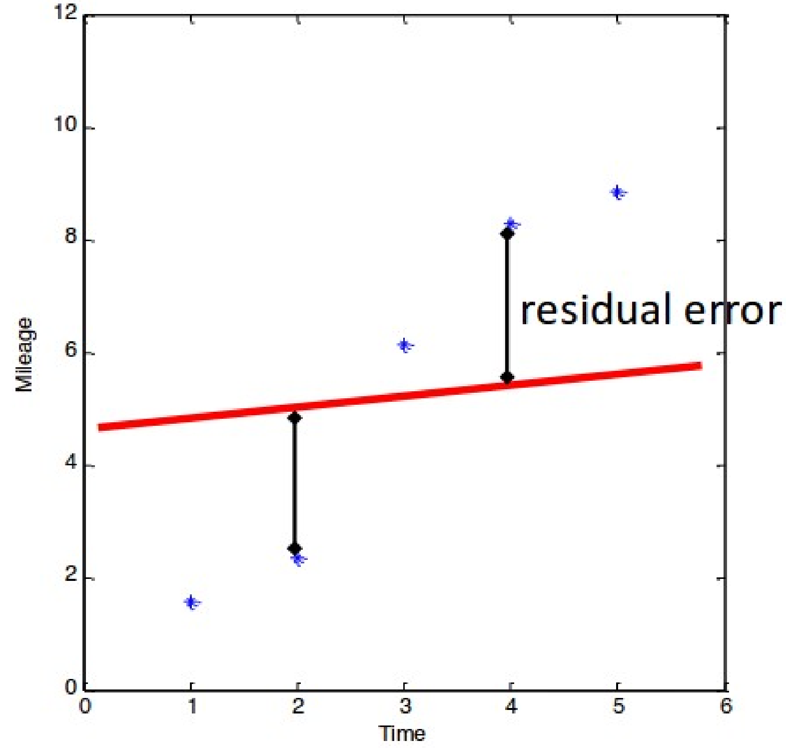
This slide from: Burgard, Stachniss, Bennewitz, Arras, U. Freiburg

Another approach to alignment: RANSAC



This slide from: Kavita Bala, Cornell U.

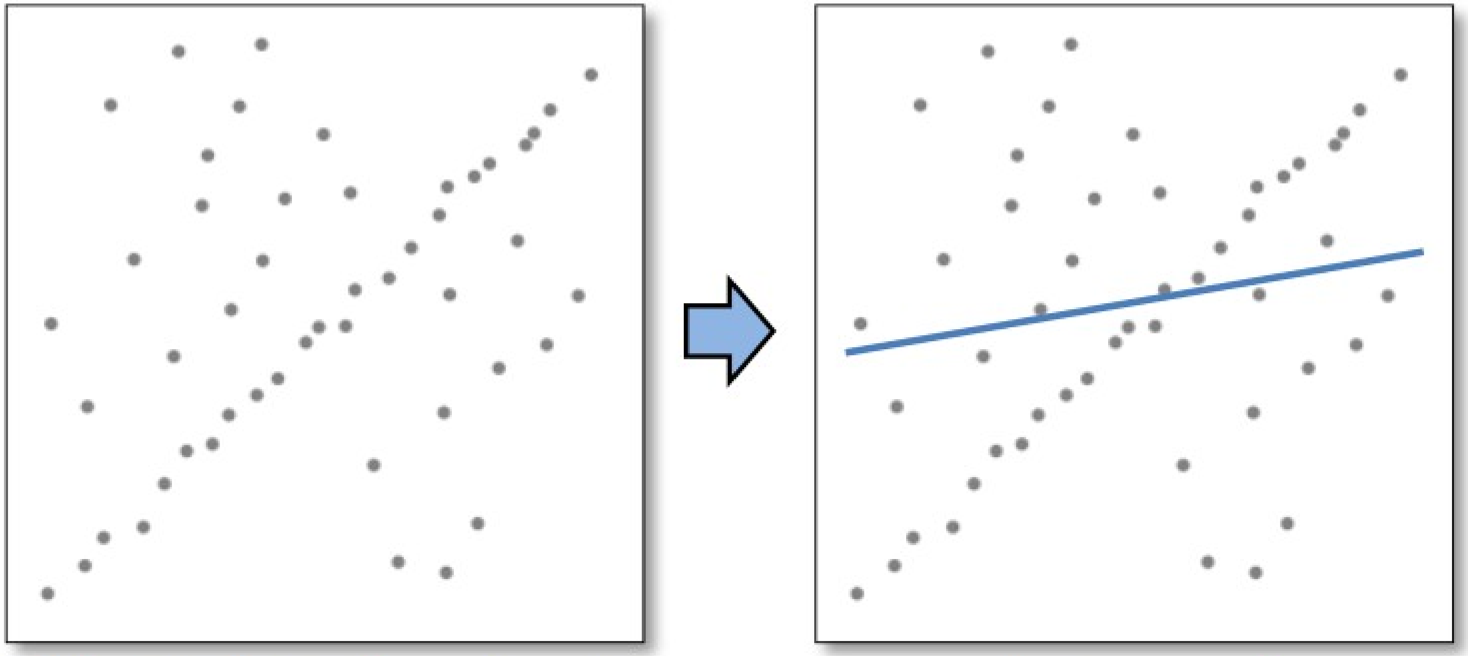
RANSAC



$$\text{Cost}(m, b) = \sum_{i=1}^{n} |y_i - (mx_i + b)|^2$$

This slide from: Kavita Bala, Cornell U.

How does regression work here?



Problem: Fit a line to these datapoints

Least squares fit

Image alignment problem

Given images A and B

1. Compute image features for A and B
2. Match features between A and B
3. Compute homography between A and B using least squares on set of matches

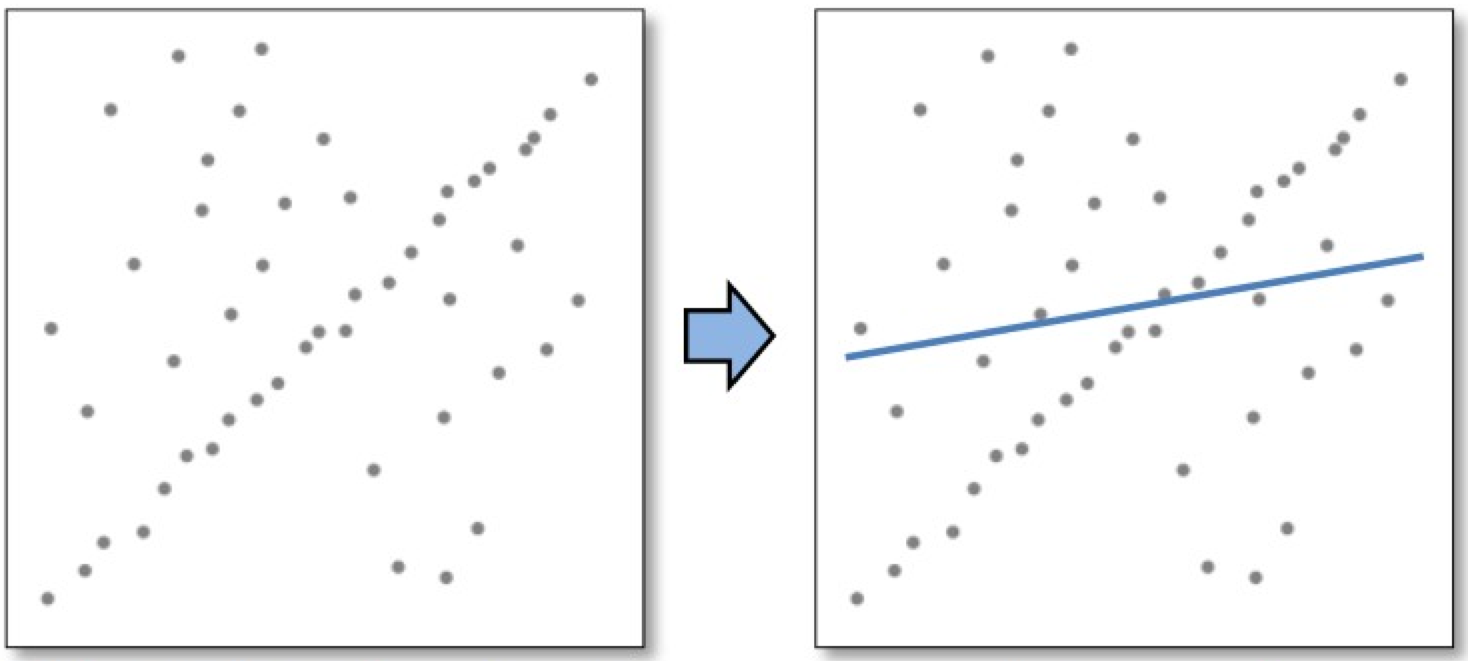
What could go wrong?

This slide from: Kavita Bala, Cornell U.

Outliers

This slide from: Kavita Bala, Cornell U.

RANSAC



Problem: Fit a line to these datapoints

Least squares fit

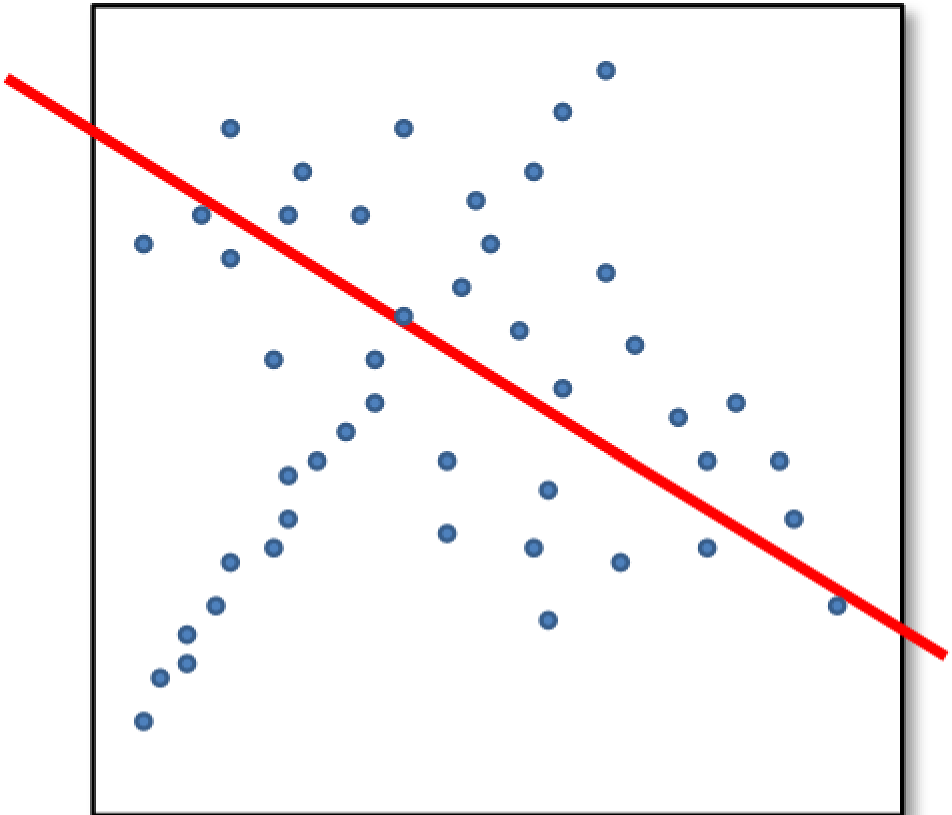
This slide from: Kavita Bala, Cornell U.

RANSAC key idea

- Given a hypothesized line
- Count the number of points that “agree” with the line
  – “Agree” = within a small distance of the line
  – I.e., the **inliers** to that line
- For all possible lines, select the one with the largest number of inliers

This slide from: Kavita Bala, Cornell U.

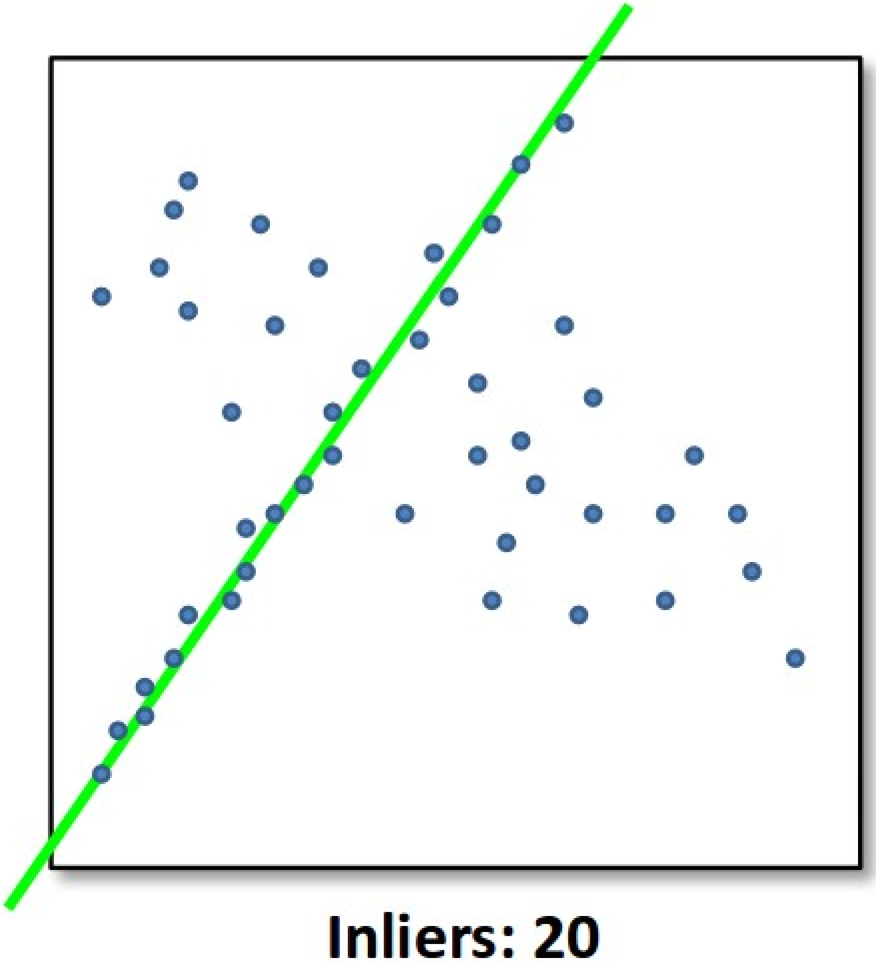
Counting inliers



Inliers: 3

This slide from: Kavita Bala, Cornell U.

Counting inliers



This slide from: Kavita Bala, Cornell U.

How do we find the best line?

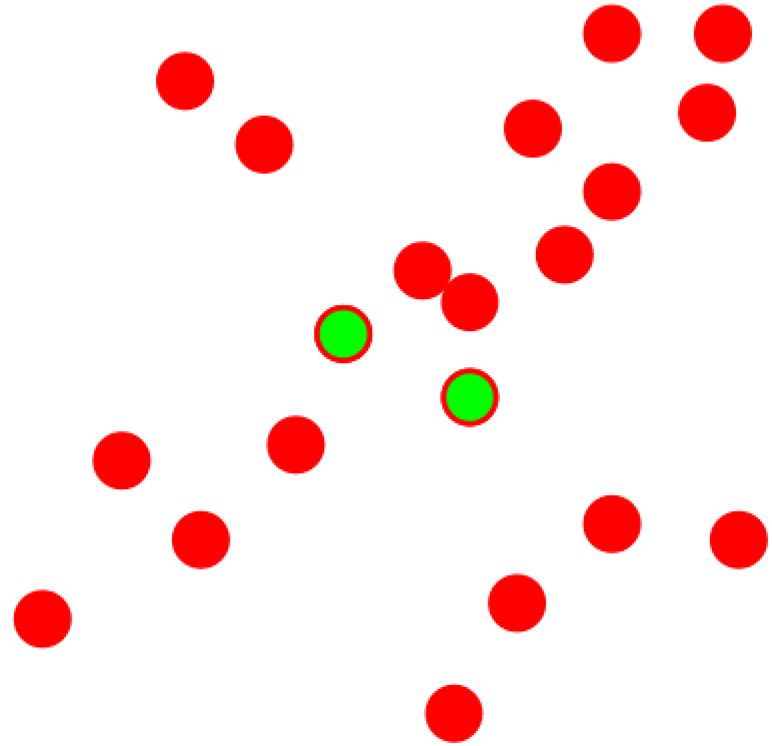
- Unlike least-squares, no simple closed-form solution

- Hypothesize-and-test
  – Try out many lines, keep the best one
  – Which lines?

This slide from: Kavita Bala, Cornell U.

RANSAC

Line fitting example



Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

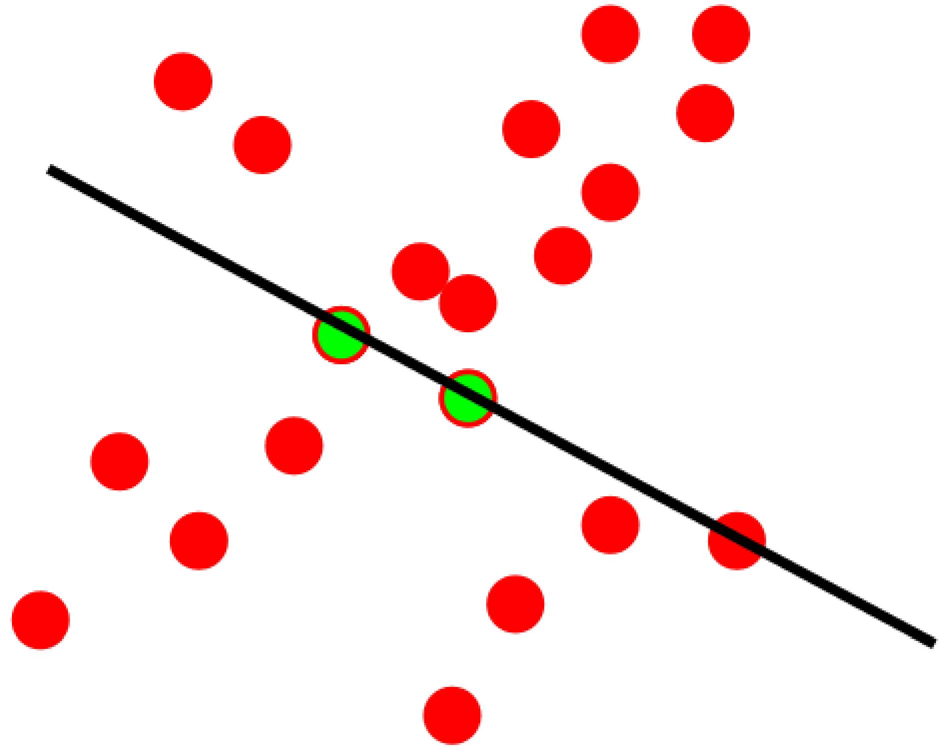
Repeat 1-3 until the best model is found with high confidence

This slide from: Kavita Bala, Cornell U.

Illustration by Savarese

RANSAC

Line fitting example



Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

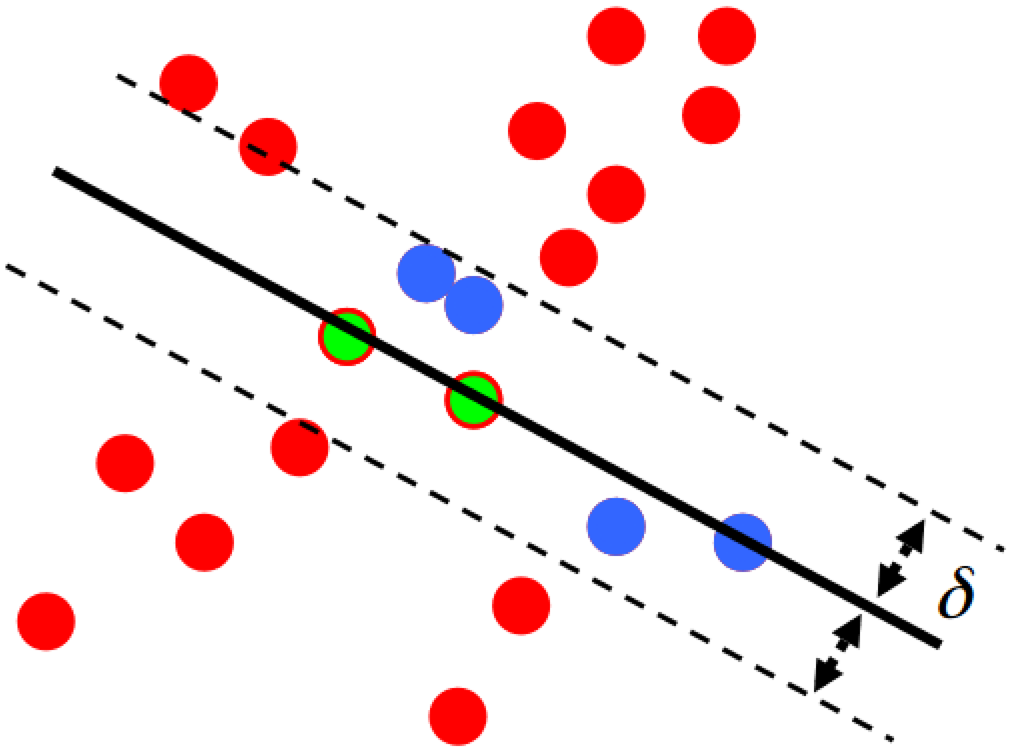
Repeat 1-3 until the best model is found with high confidence

This slide from: Kavita Bala, Cornell U.

RANSAC

Line fitting example

$$N_I = 6$$



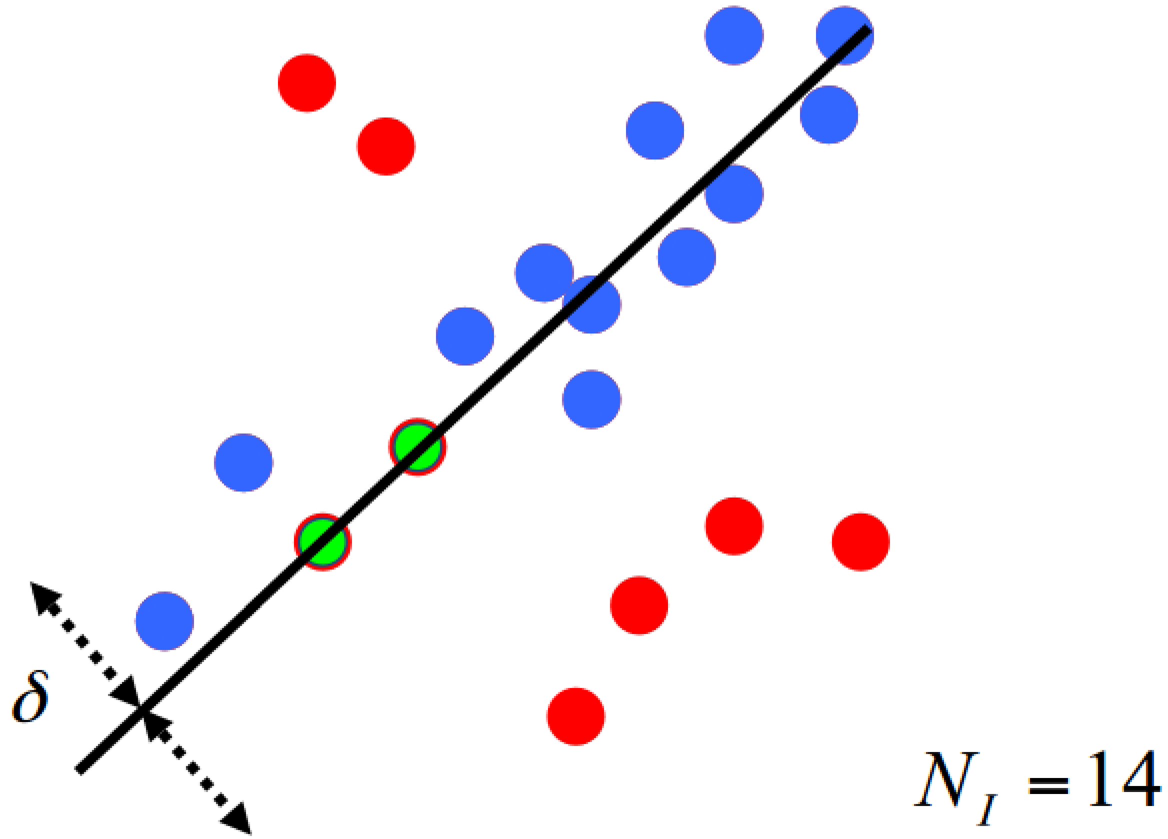
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

This slide from: Kavita Bala, Cornell U.

RANSAC



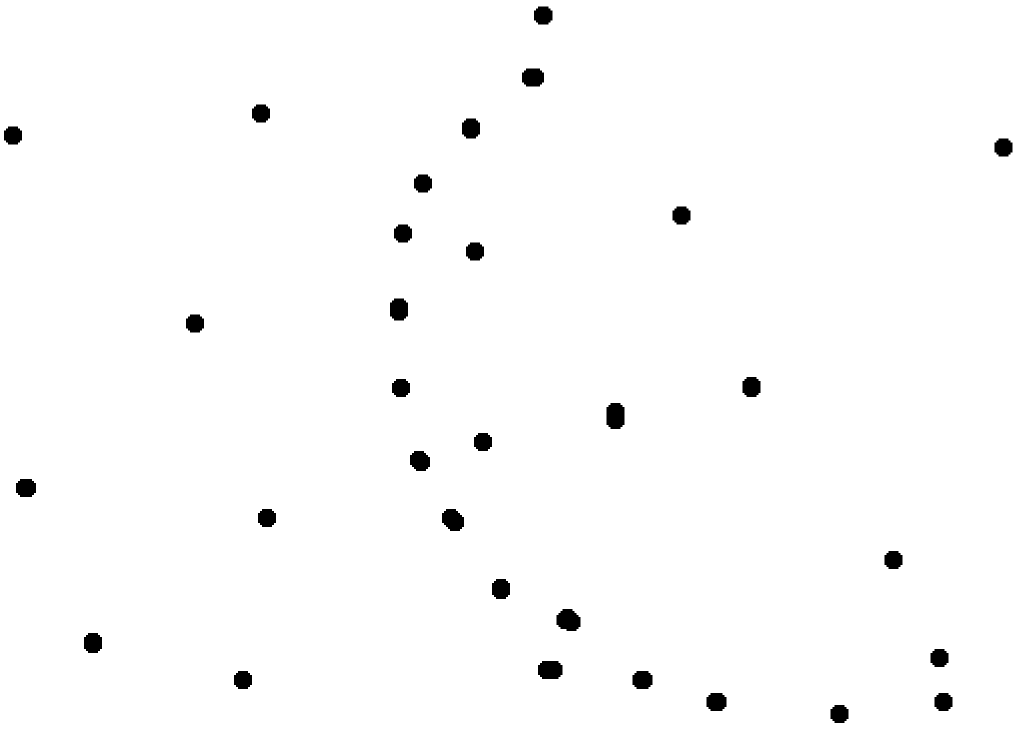
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

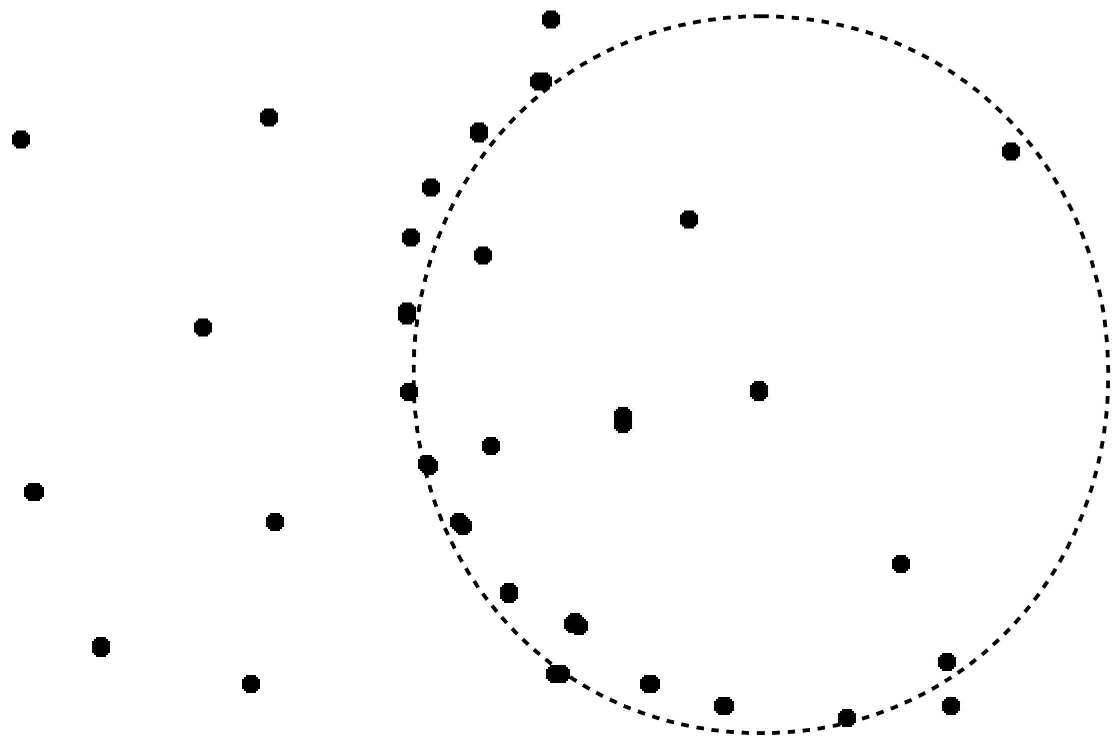
Repeat 1-3 until the best model is found with high confidence

This slide from: Kavita Bala, Cornell U.

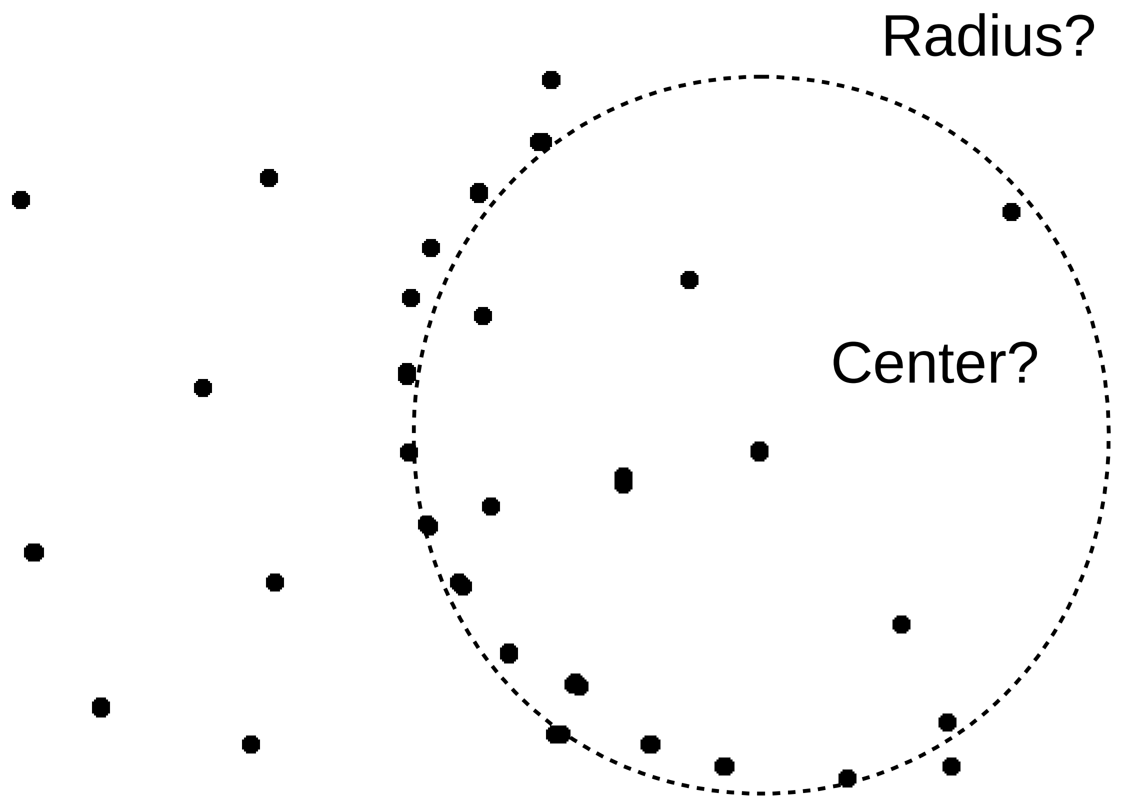
Using RANSAC to Fit a Sphere



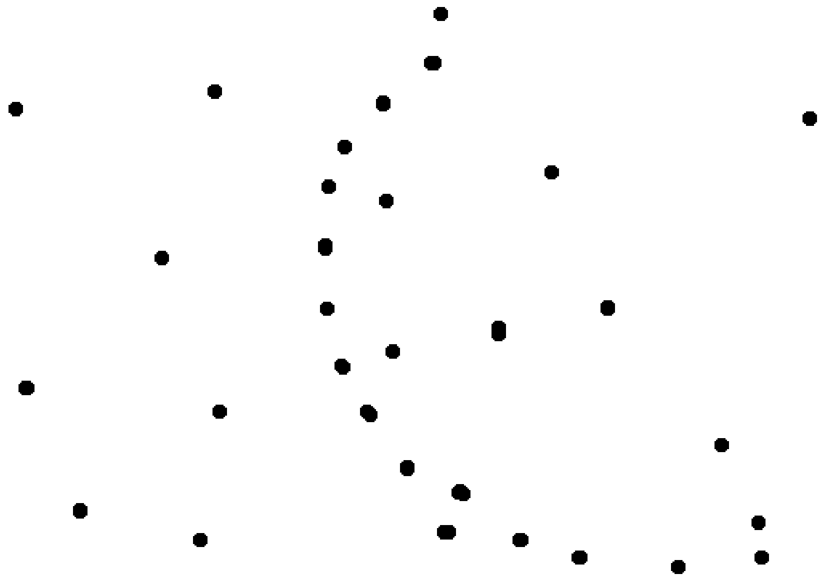
Using RANSAC to Fit a Sphere



Using RANSAC to Fit a Sphere



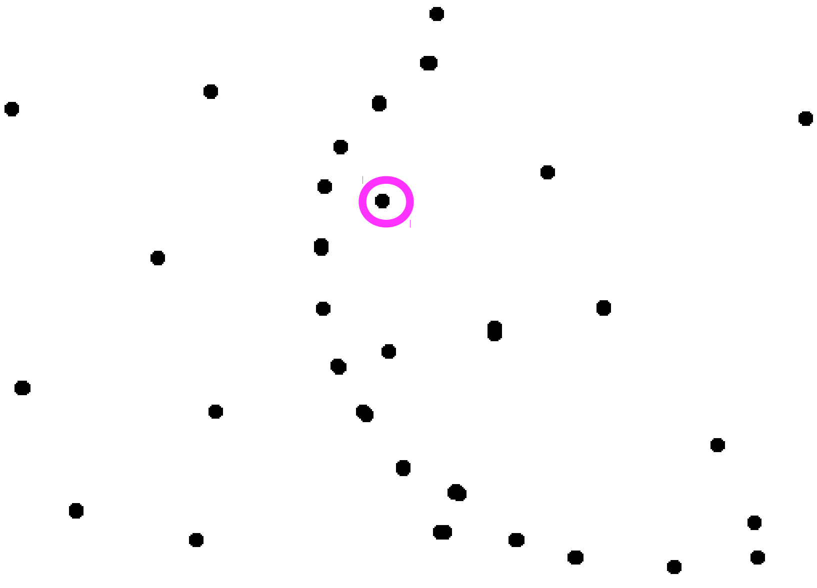
Using RANSAC to Fit a Sphere



How generate candidate spheres?

How score spheres?

Using RANSAC to Fit a Sphere

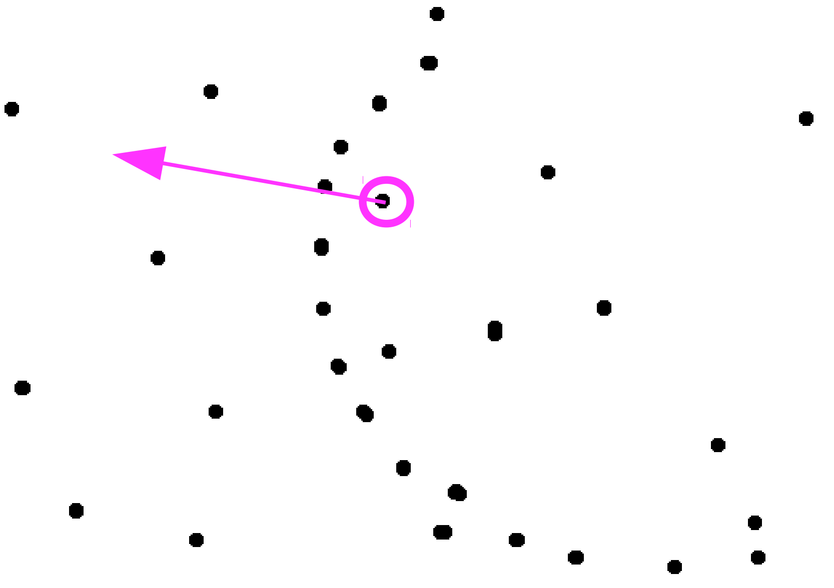


<u>How generate candidate spheres?</u>

1. sample a point

<u>How score spheres?</u>

Using RANSAC to Fit a Sphere

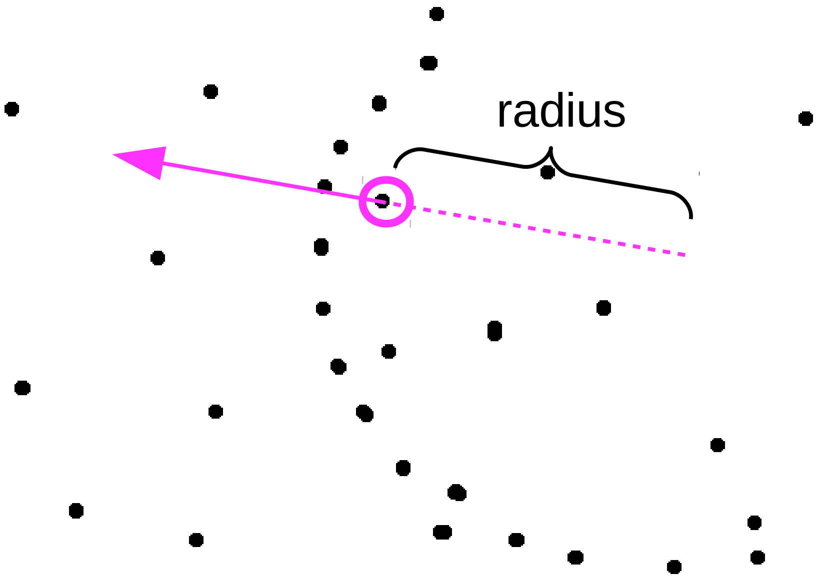


How generate candidate spheres?

1. sample a point
2. estimate surface normal

How score spheres?

Using RANSAC to Fit a Sphere

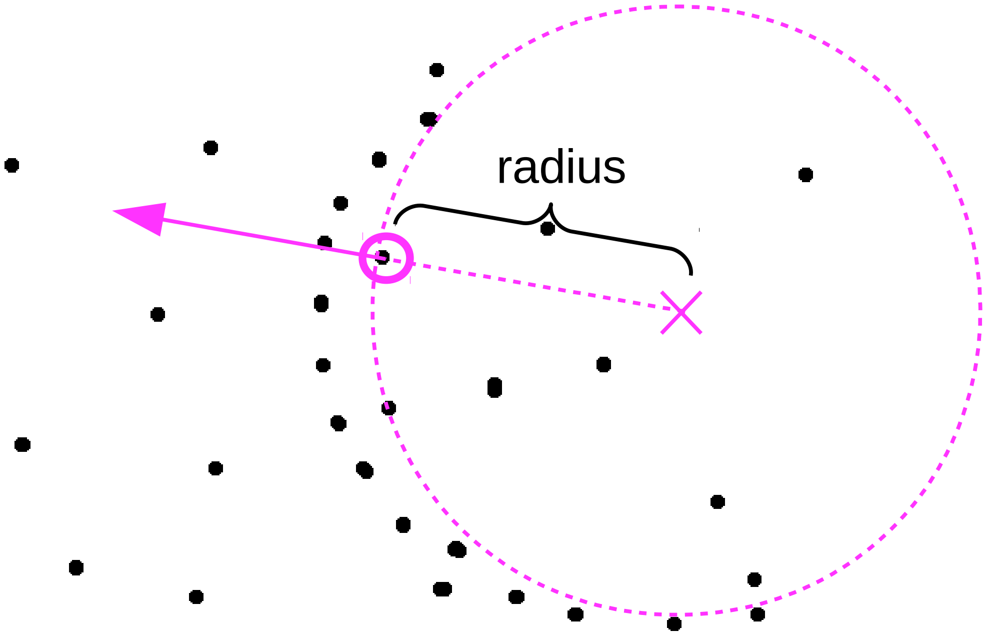


<u>How generate candidate spheres?</u>

1. sample a point
2. estimate surface normal
3. sample radius

<u>How score spheres?</u>

Using RANSAC to Fit a Sphere

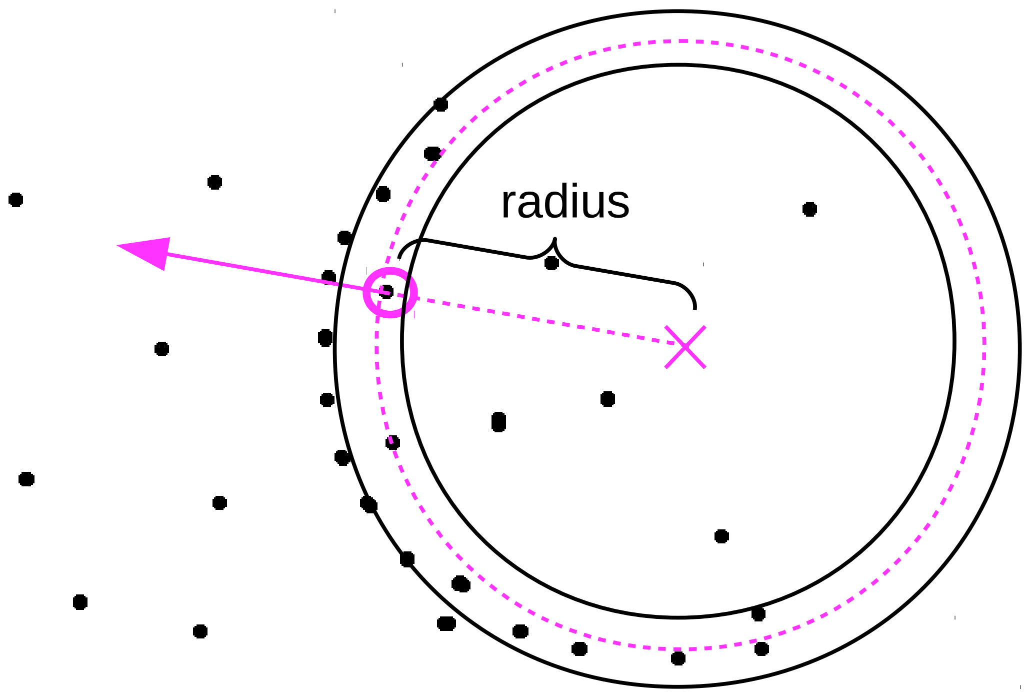


How generate candidate spheres?

1. sample a point
2. estimate surface normal
3. sample radius
4. estimate center to be radius distance
   from sampled point along surface normal

How score spheres?

Using RANSAC to Fit a Sphere



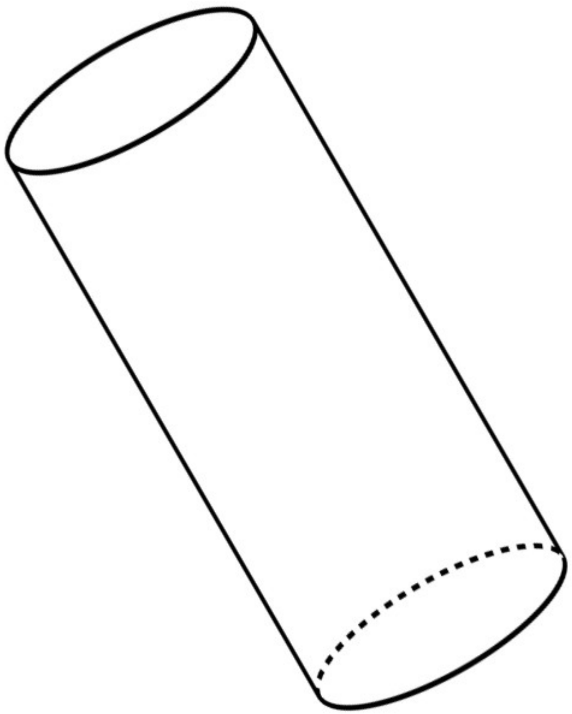
<u>How generate candidate spheres?</u>

1. sample a point
2. estimate surface normal
3. sample radius
4. estimate center to be radius distance from sampled point along surface normal

<u>How score spheres?</u>

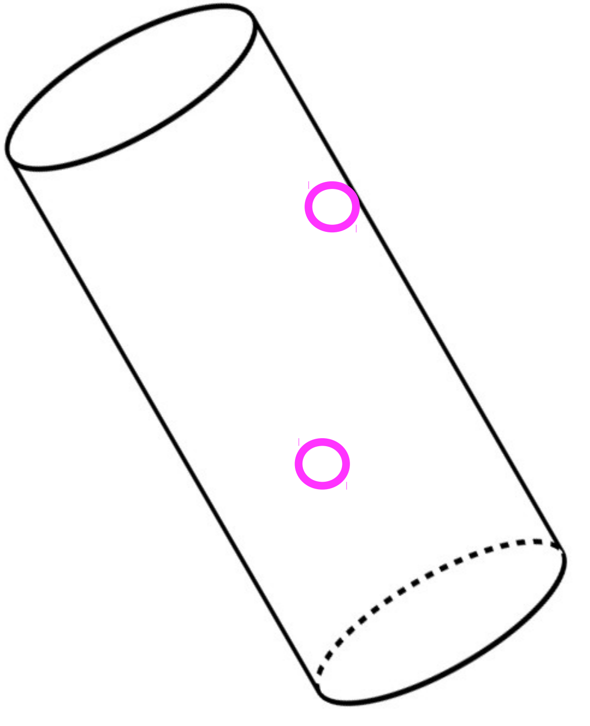
1. count num pts within epsilon of candidate sphere surface

Using RANSAC to Fit a Cylinder



How generate candidate cylinders?

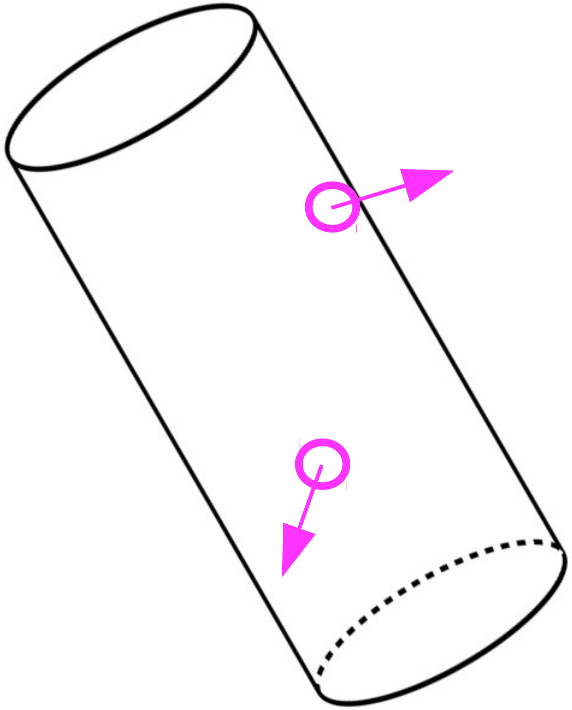
Using RANSAC to Fit a Cylinder



How generate candidate cylinders?

1. sample two pts

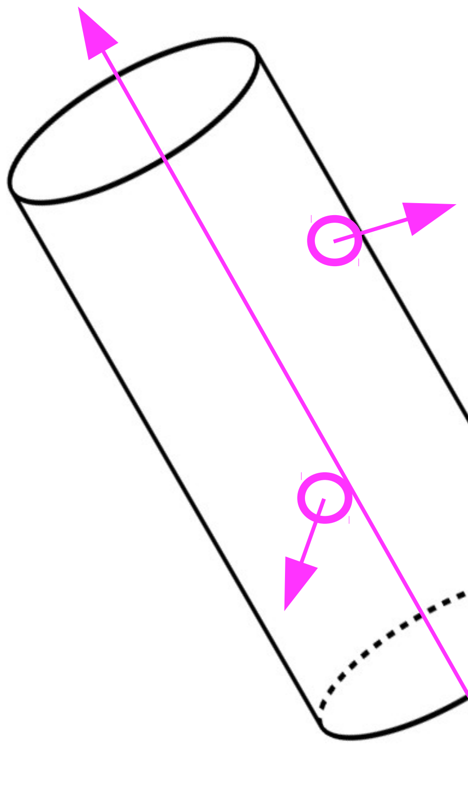
Using RANSAC to Fit a Cylinder



How generate candidate cylinders?

1. sample two pts
2. estimate surface normal at both pts

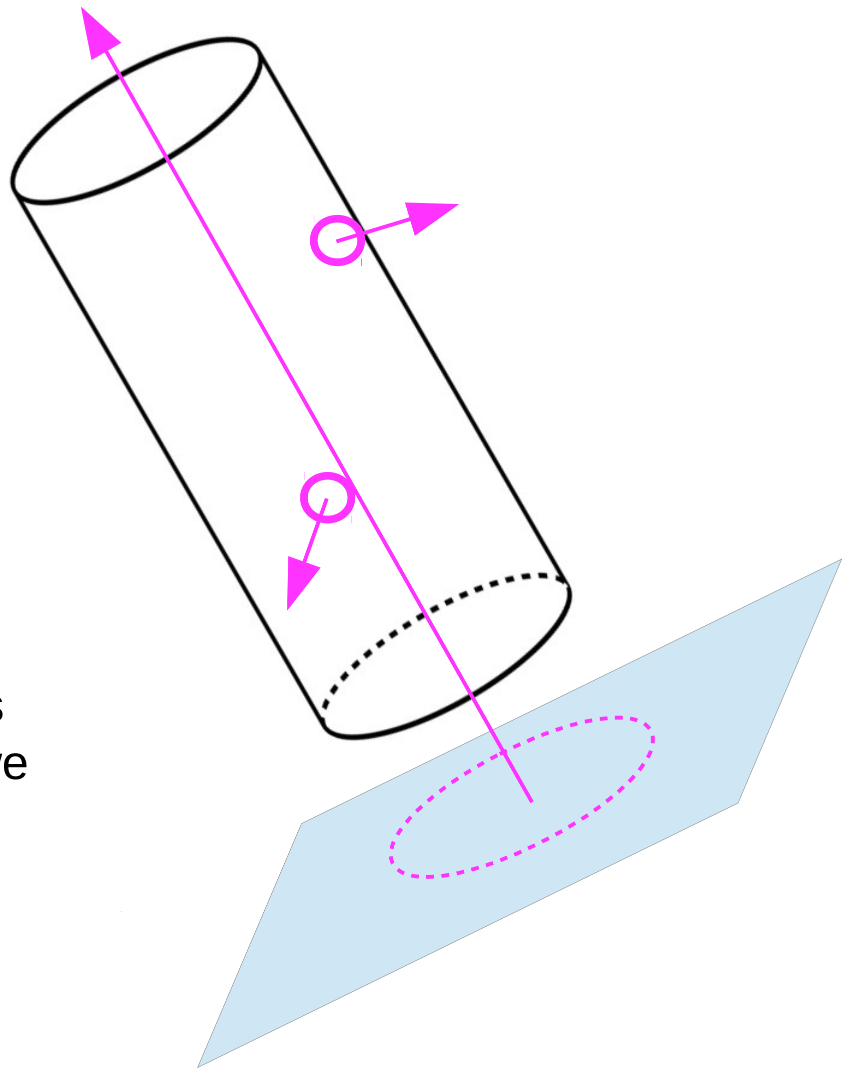
Using RANSAC to Fit a Cylinder



<u>How generate candidate cylinders?</u>

1. sample two pts
2. estimate surface normal at both pts
3. get sample axis by taking cross product between two normals

Using RANSAC to Fit a Cylinder



<u>How generate candidate cylinders?</u>
1. sample two pts
2. estimate surface normal at both pts
3. get sample axis by taking cross product between two normals
4. project points onto plane orthogonal to axis
5. fit a circle using a method similar to what we did for the sphere.

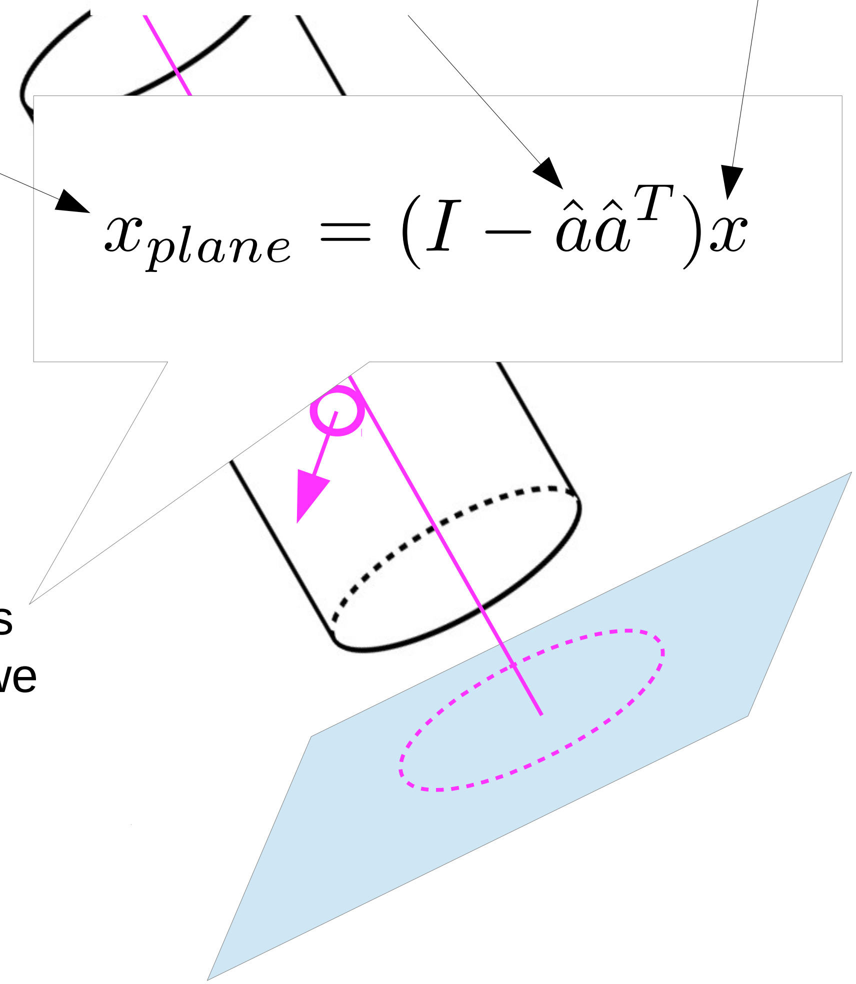
Using RANSAC to Fit a Cylinder

3xn matrix of pts in 3d space

3x1 unit vector in direction of axis

3xn matrix of pts projected onto plane

$$x_{plane} = (I - \hat{a}\hat{a}^T)x$$



<u>How generate candidate cylinders?</u>

1. sample two pts
2. estimate surface normal at both pts
3. get sample axis by taking cross product between two normals
4. project points onto plane orthogonal to axis
5. fit a circle using a method similar to what we did for the sphere.

RANSAC: the parameters

- **Inlier threshold** related to the amount of noise we expect in inliers
  - Often model noise as Gaussian with some standard deviation (e.g., 3 pixels)
- **Number of rounds** related to the percentage of outliers we expect, and the probability of success we'd like to guarantee
  - Suppose there are 20% outliers, and we want to find the correct answer with 99% probability
  - How many rounds do we need?

This slide from: Kavita Bala, Cornell U.

RANSAC: how many rounds?

To ensure that the random sampling has a good chance of finding a true set of inliers, a sufficient number of trials S must be tried. Let p be the probability that any given correspondence is valid and P be the total probability of success after S trials. The likelihood in one trial that all k random samples are inliers is p^k. Therefore, the likelihood that S such trials will all fail is

$$1 - P = (1 - p^k)^S \tag{6.29}$$

and the required minimum number of trials is

$$S = \frac{\log(1 - P)}{\log(1 - p^k)}. \tag{6.30}$$

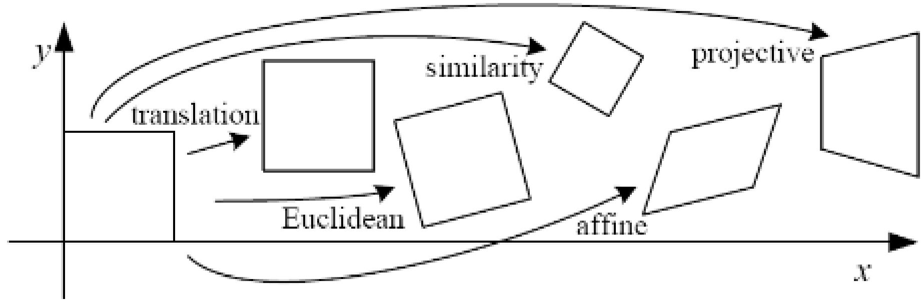
	proportion of inliers p						
k	95%	90%	80%	75%	70%	60%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

$P = 0.99$

This slide from: Kavita Bala, Cornell U.

RANSAC: how many parameters to sample?

- For alignment, depends on the motion model
  - Here, each sample is a correspondence (pair of matching points)



Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\begin{bmatrix} I \mid t \end{bmatrix}_{2\times 3}$	2	orientation $+\cdots$	
rigid (Euclidean)	$\begin{bmatrix} R \mid t \end{bmatrix}_{2\times 3}$	3	lengths $+\cdots$	
similarity	$\begin{bmatrix} sR \mid t \end{bmatrix}_{2\times 3}$	4	angles $+\cdots$	
affine	$\begin{bmatrix} A \end{bmatrix}_{2\times 3}$	6	parallelism $+\cdots$	
projective	$\begin{bmatrix} \tilde{H} \end{bmatrix}_{3\times 3}$	8	straight lines	

This slide from: Kavita Bala, Cornell U.

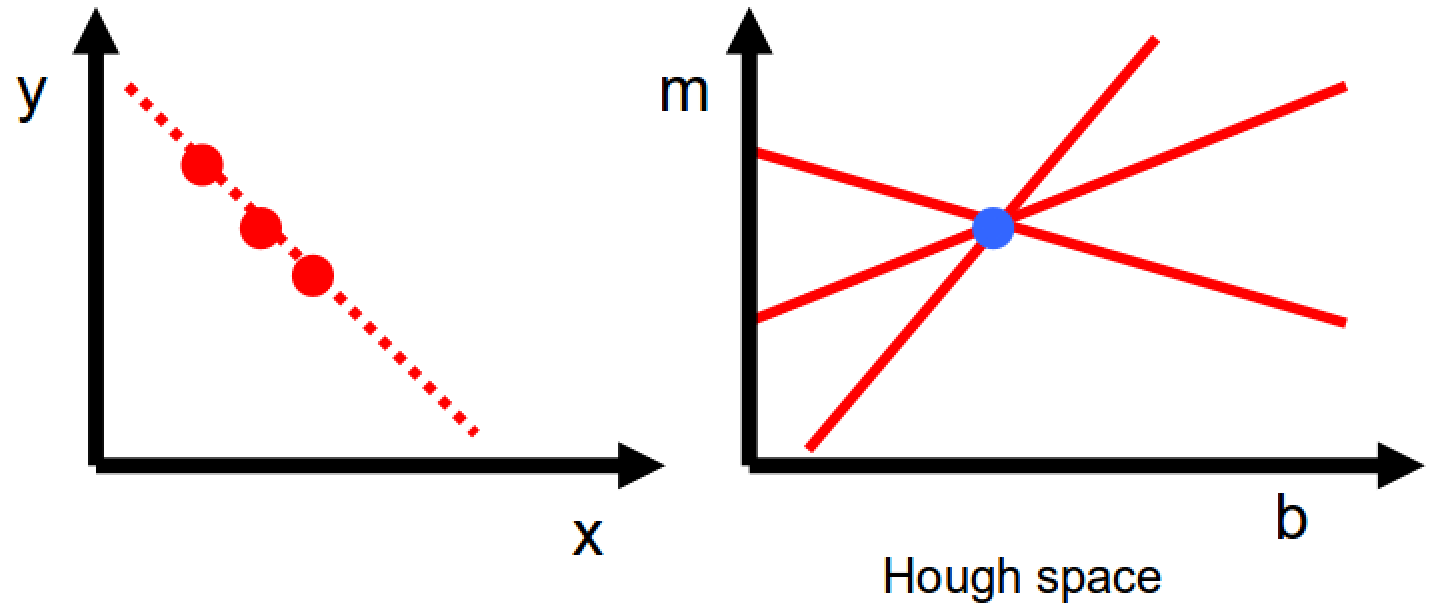
RANSAC Summary

- Pros
  - Simple and general
  - Applicable to many different problems
  - Often works well in practice
- Cons
  - Parameters to tune
  - Sometimes too many iterations are required
  - Can fail for extremely low inlier ratios
  - We can often do better than brute-force sampling

This slide from: Kavita Bala, Cornell U.

Hough transform

Given a set of points, find the curve or line that explains the data points best



$$y = m x + b$$

Slide from S. Savarese

This slide from: Kavita Bala, Cornell U.

Hough transform

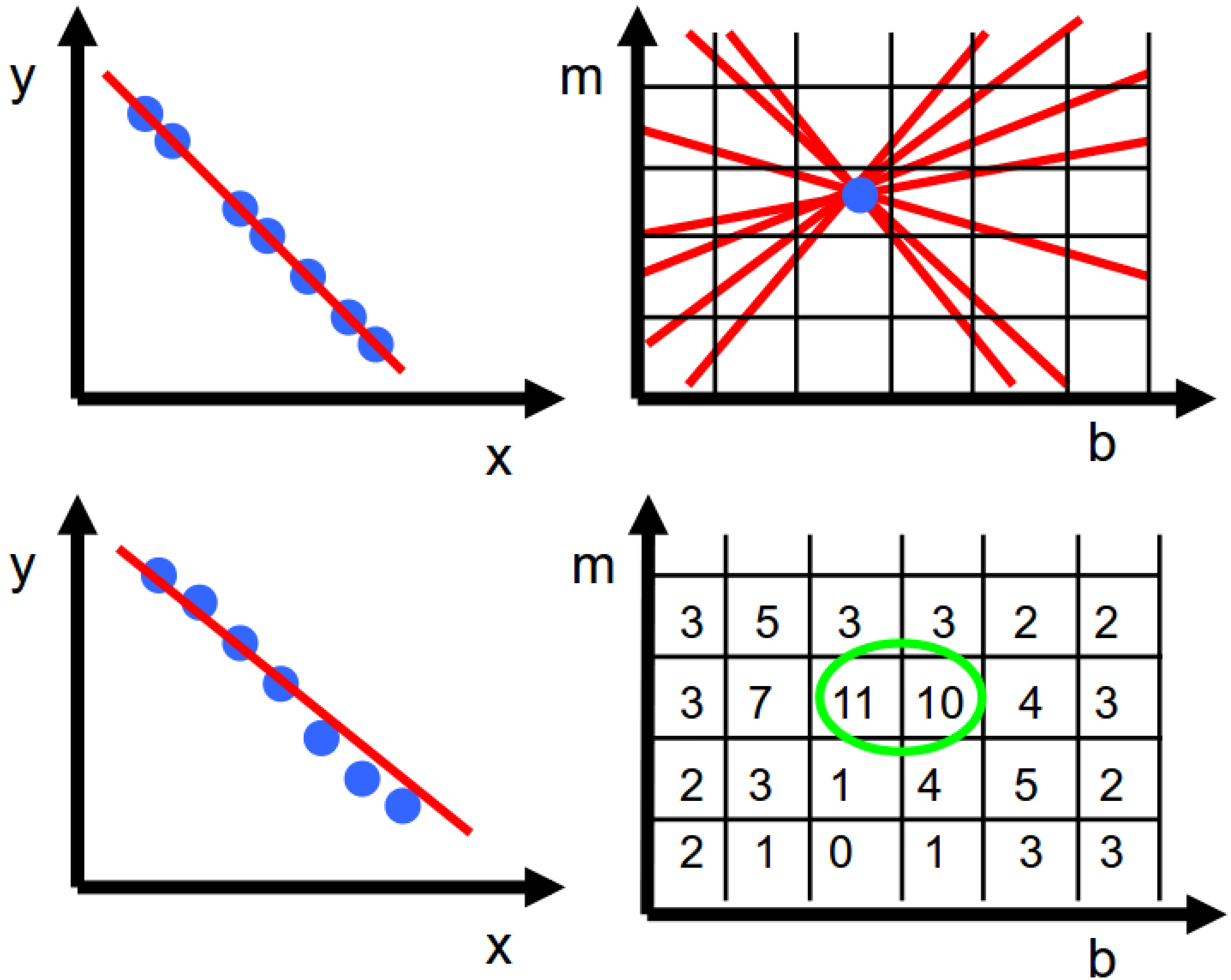
1. Create a grid of parameter values

2. Each point votes for a set of parameters, incrementing those values in grid

3. Find maximum or local maxima in grid

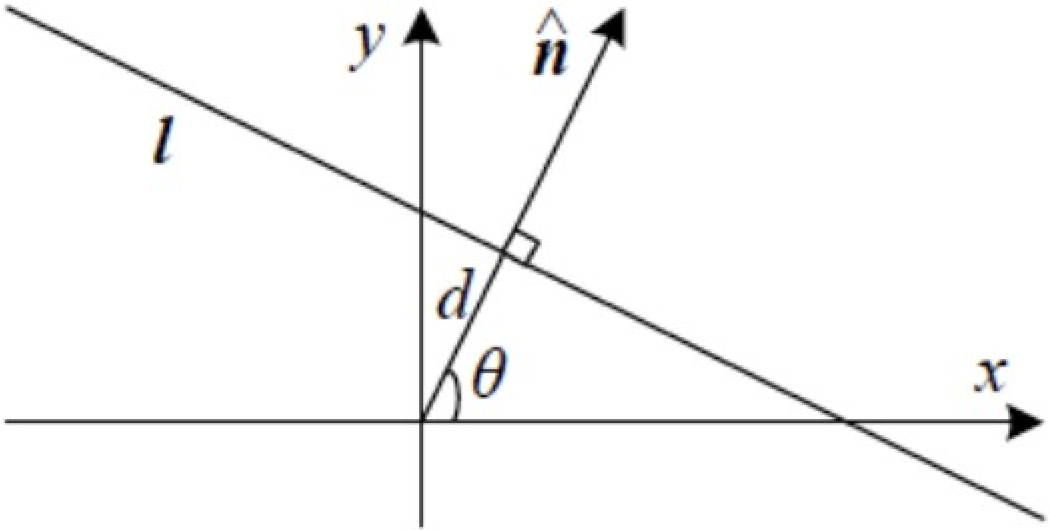
This slide from: Kavita Bala, Cornell U.

Hough transform



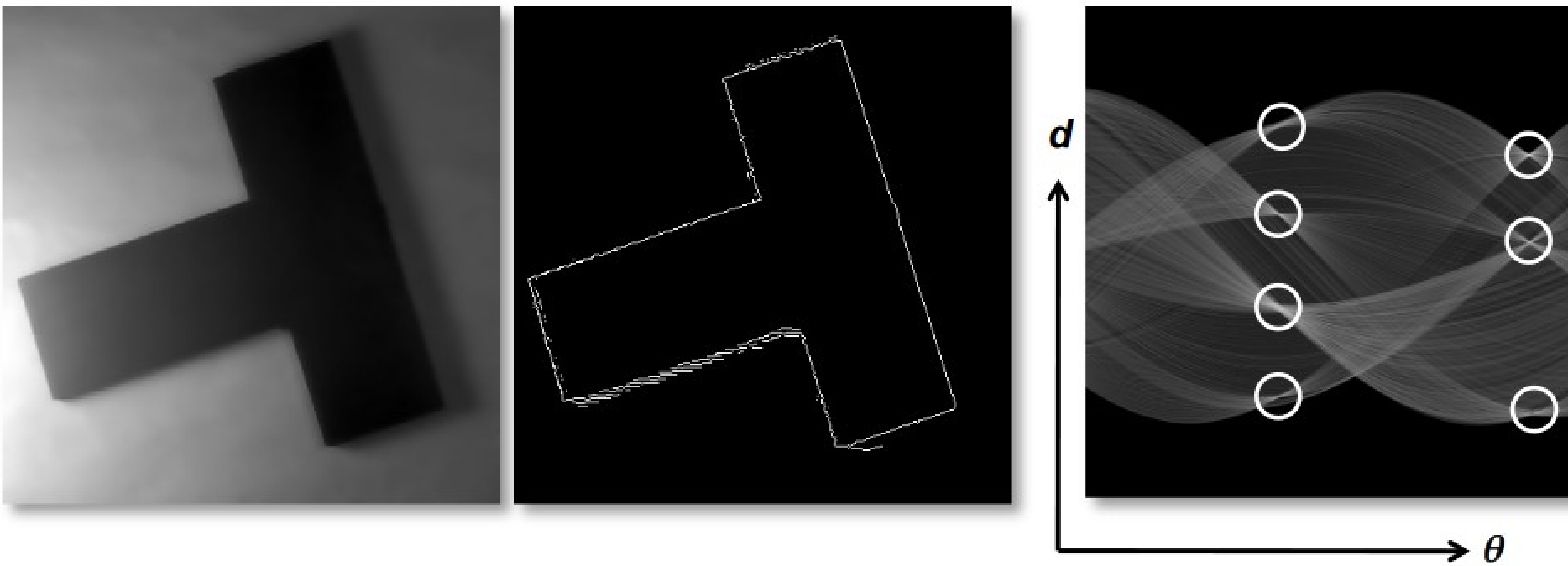
This slide from: Kavita Bala, Cornell U.

Hough transform



This slide from: Kavita Bala, Cornell U.

Hough transorm



This slide from: Kavita Bala, Cornell U.