INFO0948
Fitting and Shape Matching

Renaud Detry

University of Liège, Belgium

Updated March 31, 2015

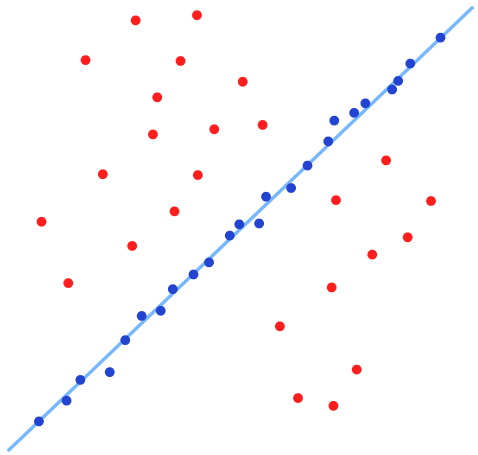
These slides are based on the following book:

> *D. Forsyth and J. Ponce. Computer vision: a modern approach. Prentice Hall Professional Technical Reference, 2002.*

and on the course "Shape Matching & Correspondence" by Maks Ovsjanikov (Standford University).

Fitting and Shape Matching

Fitting: usually refers to fitting a model to a set of points, i.e., finding instances of the model in the set of points.



Fitting a parametric model

Fitting a nonparametric model

Plan

Linear Regression

From n datapoints $\{(x_i, y_i)\}_{i\in[1,n]}$, find the line

$$y = \alpha + \beta x$$

that "best" fits the data.
What the best line is, is part of the problem definition. Typically, the best line is the one that minimizes the sum-of-square error

$$\min_{\alpha,\beta} Q(\alpha,\beta), \text{ where } Q(\alpha,\beta) = \sum_{i=1}^{n} \hat{\varepsilon}_i^2 = \sum_{i=1}^{n} (y_i - \alpha - \beta x_i)^2.$$

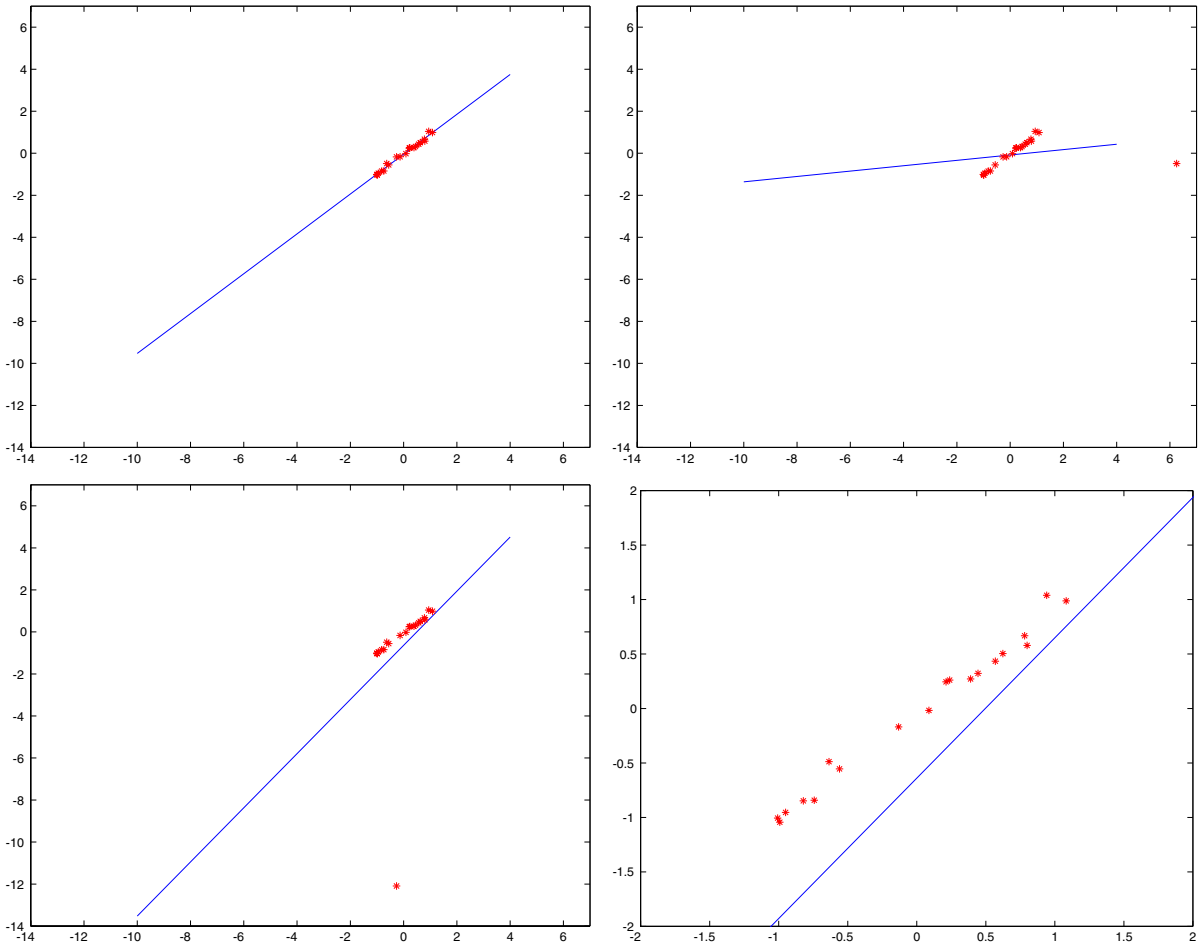
It can be shown that the line that minimizes Q is given by

$$\hat{\beta} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \qquad \hat{\alpha} = \bar{y} - \hat{\beta}\,\bar{x}$$

$$\bar{x} = \frac{1}{n}\sum x_i \qquad \bar{y} = \frac{1}{n}\sum y_i$$

Linear Regression

Least-squares linear regression is extremely sensitive to noise



The (least-squares) line goes through $(\bar{x}, \bar{y})$. What happens if there are multiple lines?

The Hough Transform

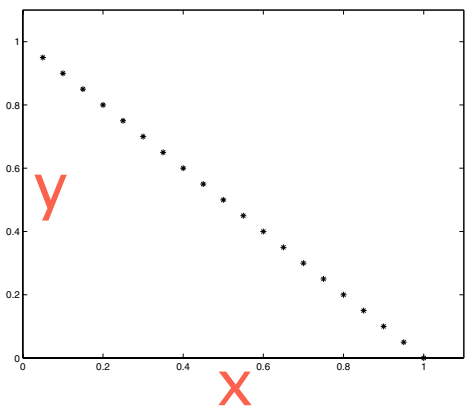
Principle: make each datapoint vote for all the model instances that could pass through it, and select the instance that collects the most votes.

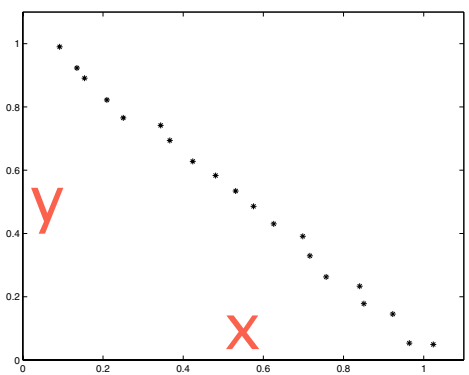
A line can be written as the set of points (x, y) that satisfy

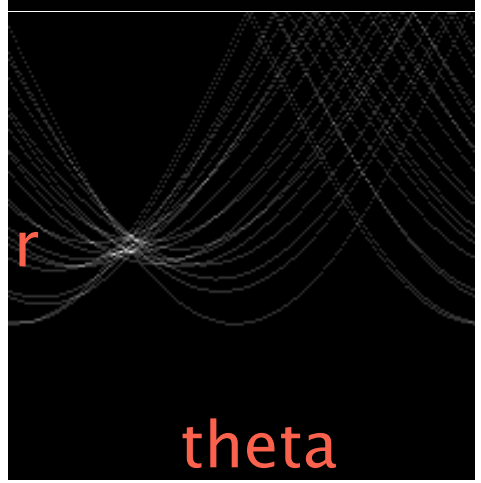
$$x \cos\theta + y \sin\theta + r = 0$$

where (r, θ) are the parameters of the line.
All the lines passing through (x_0, y_0) are in the locus

$$r = -x_0 \cos\theta - y_0 \sin\theta$$







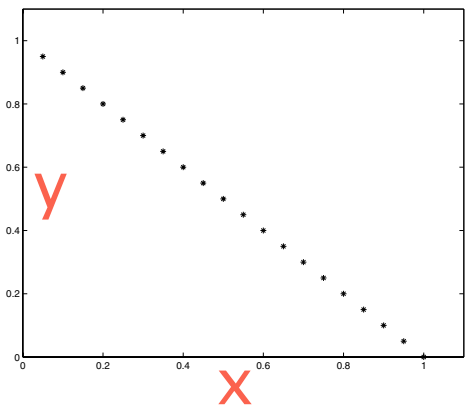
The Hough Transform

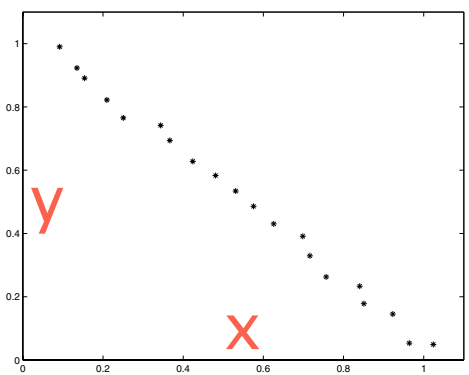
Principle: make each datapoint vote for all the model instances that could pass through it, and select the instance that collects the most votes.

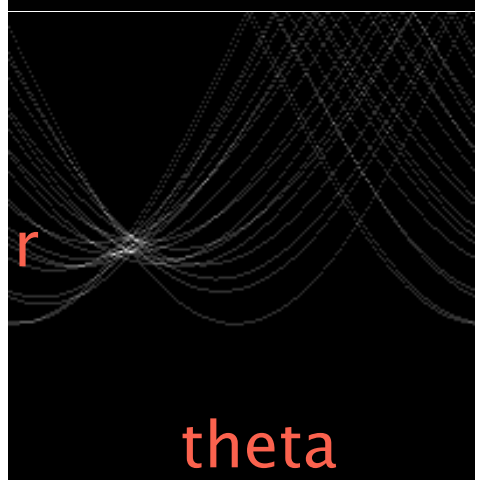
We're only interested in $0 \leq \theta < 2\pi$ and $0 \leq r < R$.

We can discretize this space, yielding a 2D grid.

Each datapoint votes for a cell of the grid (increasing the cell's value by 1).







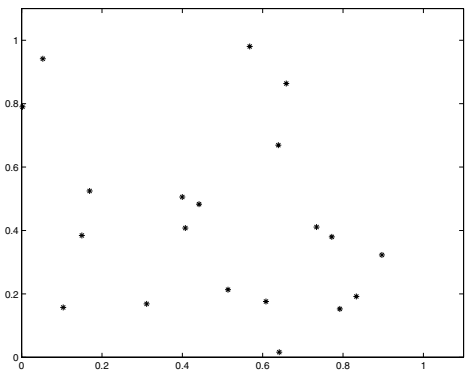
Problems with the Hourgh Transform

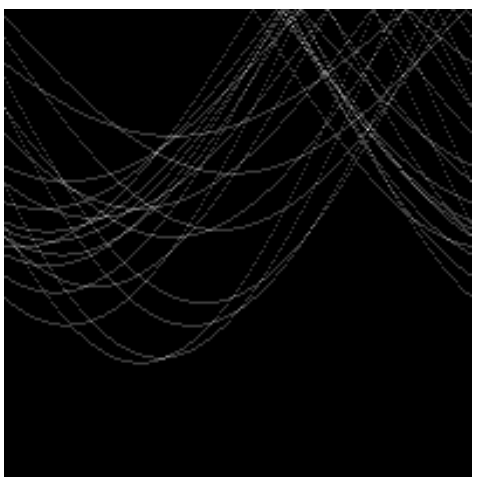
Quantization Errors

- Appropriate grid size difficult to pick. Too coarse, and each cell will represent quite different lines. Too fine, and the data noise will prevent the line points from voting for the same cell, resulting in no cell with a large vote count. → Choose the grid carefully

Difficulties with Noise

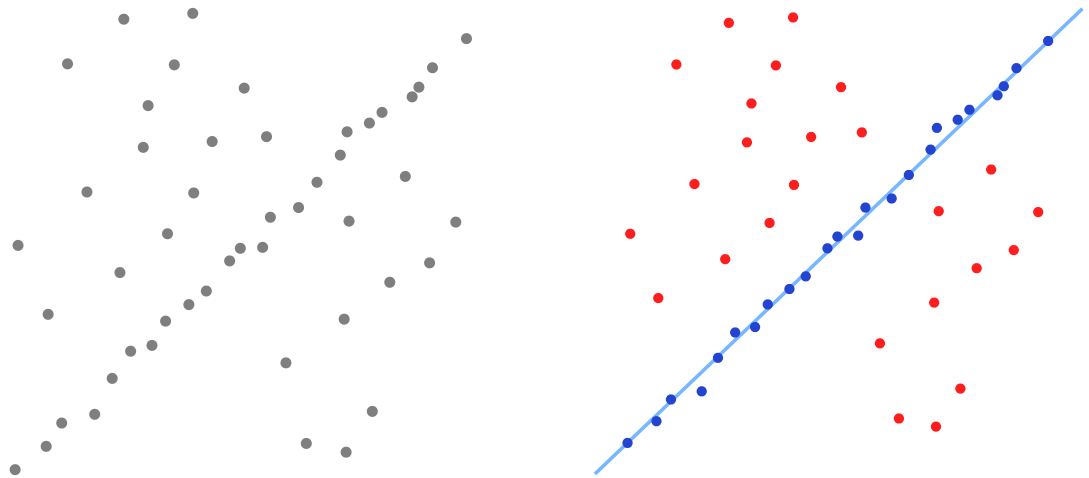
- For instance, "phantom" lines can appear in large sets of randomly distributed datapoints. → Attempt to remove outliers before fitting





The RANSAC (Random Sample Consensus) Algorithm

RANSAC is a stochastic algorithm for fitting a model to a dataset that contains outliers, i.e., points that cannot be explained by a model instance.



Repeat until k iterations or until we found a good fit:

- Find a small subset of points, and fit the model to that subset
- Compute how many points can be explained by the fitted model

If we know that 50% of points are outliers, and we fit the model to random pairs of points, 25% of these pairs will yield a satisfactory model instance.

RANSAC: How many points are necessary?

Method 1

At each iteration, we draw n points at random, where n is the minimum number of points required to fit the model.

- For lines, $n = 2$. For circles, $n = 3$.

We assume that we can get an estimate of the fraction w of inliers within the set of datapoints.

Let us denote by k the number of iterations, and by $E[k]$ estimated value of k so that in one of the k iterations we drawn n inliers. We have

$$\begin{aligned} E[k] &= 1P(\text{one good sample in one draw}) + 2P(\text{one good sample in two draws}) + \ldots \\ &= w^n + 2(1 - w^n)w^n + 3(1 - w^n)^2 w^n + \ldots \\ &= w^{-n} \end{aligned}$$

To increase our confidence in getting at least one good draw, we add a few standard deviations to w^{-n}

$$SD(k) = \frac{\sqrt{1 - w^n}}{w^n}$$

RANSAC: How many points are necessary?

Method 2

Let us denote by z the probability of having at least one outlier in each of the k iterations. (We set z to how much we are willing to risk failing to fit our model.) Then,

$$(1 - w^n)^k = z$$

and

$$k = \frac{\log(z)}{\log(1 - w^n)}.$$

- Typical values for z are in the range $[0.01, 0.05]$, but the choice of z is application-dependent.
- Again, we add a few standard deviations to the estimate of k.

RANSAC: Who is an inlier?

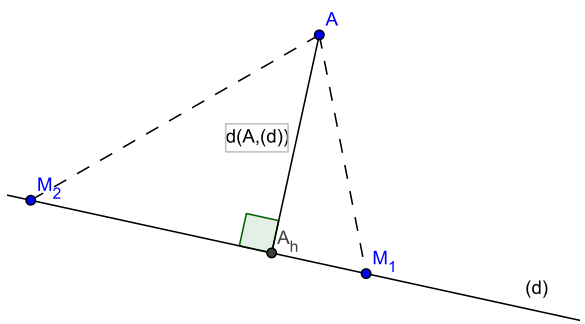
RANSAC Algorithm

Repeat until k iterations or until we found a good fit:

- Find a small subset of points, and fit the model to that subset
- Compute how many points can be explained by the fitted model

We need to define a criterion for deciding which points explain a given model instance.

Typically: a point is explained by a model instance if it lies within a distance d from the model instance.



RANSAC: How many inliers do we need?

RANSAC Algorithm

Repeat until k iterations or until we found a good fit:

- Find a small subset of points, and fit the model to that subset
- Compute how many points can be explained by the fitted model

What is a "good fit"?
Rule of thumb: stop when a model fits to as many points as the number of inliers we expect in the dataset. Denoting the number of points in the dataset by N, stop when the number of inliers is larger than $t = wN$.

RANSAC Algorithm

Determine n, t, d and k

Repeat, until there is a good fit or k iterations have occurred:

- Draw a sample of n points from the data uniformly and at random
- Fit to that set of n points
- For each data point outside the sample:
  - Test the distance from the point to the line against d
  - If the distance from the point to the line is less than d:
    - the point is an inlier
- If there are t or more points inliers
  - Found a good fit! Refit the line using all these points, and terminate

RANSAC: Multiple Instances

RANSAC is designed to fit a model to a dataset that contains one instance of the model.

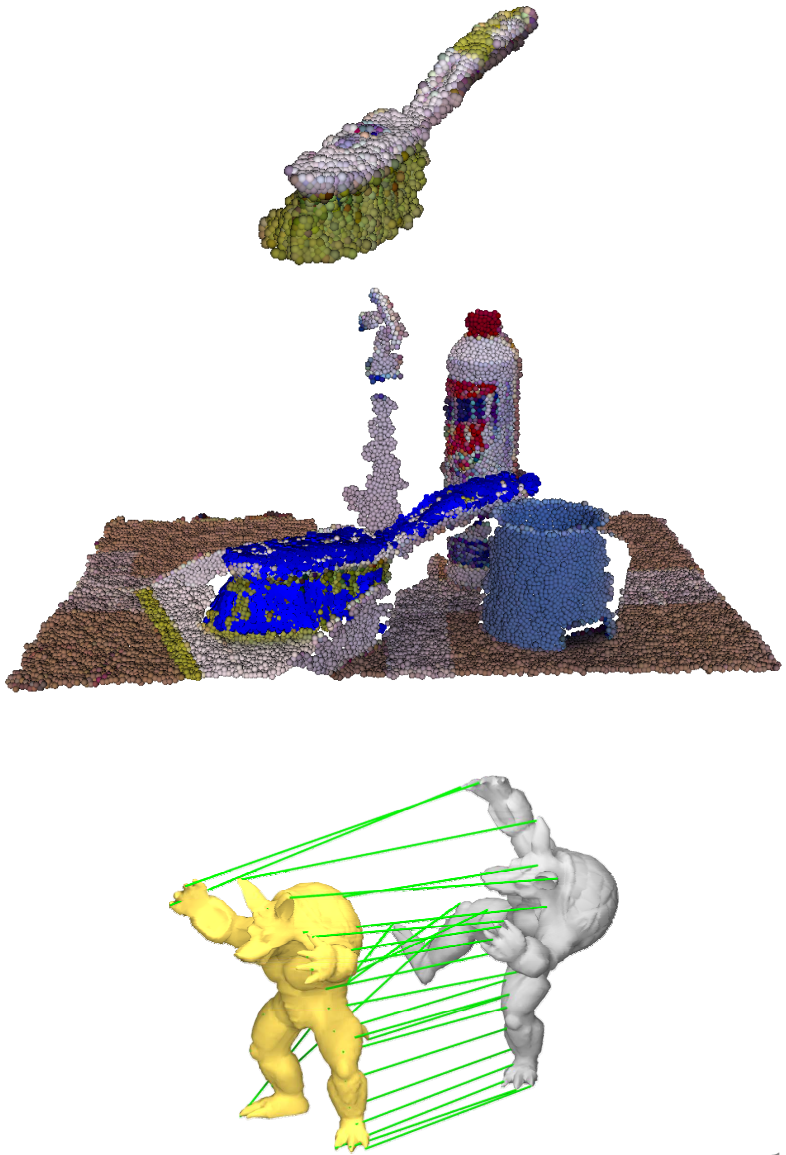
Variants of RANSAC that explicitly address the problem of finding multiple instances exist.

Plan

Shape Matching

Applications:

- Robotics: grasping, object recognition.
- Medicine: Fitting MRI scans.
- Manufacturing: quality control.



Method Taxonomy

Local refinement (ICP)	vs.	Global alignment (search)
Rigid rotation, translation	vs.	Deformable deformation
Pair two shapes	vs.	Collection multiple shapes
Fit to Same Shape without outliers	vs.	Fit to Sub/supershape with outliers

To date, [Local AND Rigid AND Pair AND Same Shape] is solved. All other combinations are actively researched.
Here, we are interested in [Local/Global AND Rigid AND Pair AND Sub/super Shape].

Iterative Closest Point (ICP)

Local, Rigid, Pair, Same Shape

ICP: registration of two 3D point clouds (i.e., finding the 3D transformation that maps the first point cloud to the second one).

Function ICP(M, S): (M: model, S: scene)

$(R, t) =\leftarrow$ *Initialize-Registration*(S, M) *(for instance, RANSAC)*
$E' \leftarrow +\infty$ *(E': matching score of (R, t))*
repeat
    $E \leftarrow E'$
    Transform all points of S with (R, t):
    $S' \leftarrow$ *Transform*(S, R, t)
    For each point of S', compute the closest point in M:
    $P \leftarrow$ *Return-Pairs-Closest-Points*(S', M)
    Estimate the transfo mapping the points of S' onto their matches in M:
    $(R, t, E') \leftarrow$ *Compute-ML-Transformation*(S, M, P, R, t)
until $|E' - E| < \tau$
return (R, t)

ICP: Local Convergence Only

ICP always converges to a local minimum of E.

There is no guarantee that ICP will converge to the global optimum.

A reasonable estimate of the transformation must be provided via *Initialize-Registration()*:

- trying different transformations at random
- Using the moments of the scene and model
- Using RANSAC

ICP: Finding the Closest-Point Pairs

Return-Pairs-Closest-Points()

At most in $O(|S||M|)$.

Using a kd-tree, this function is in $O(|S| \log |M|)$. S and M must be large for the cost of the kd-tree to be amortized.

Estimating the Rigid Transformation

Compute-ML-Transformation()

Let us denote P by $\left\{(x_i^{S'}, x_i^M)\right\}_{i\in[1,n]}$ where $n = |S|$, and let us write x_i^S the counterpart of $x_i^{S'}$ in S.
We seek (R, t) that minimize

$$E = \sum_{i=1}^{n} |x_i^M - Rx_i^S - t|^2$$

Let us note that the value of t that minimizes E must verify

$$0 = \frac{\partial E}{\partial t} = -2\sum_{i=1}^{n}(x_i^M - Rx_i^S - t)$$

Using the quaternion representation of R, the error can be rewritten as

$$E = \mathring{q}\mathcal{B}\mathring{q}^{-1}$$

Minimizing E under the constraint $|q|^2 = 1$ is a linear least-squares problem whose solution is the eigenvector of $\mathcal{B}$ associated with the smallest eigenvalue of this matrix.

ICP Variants

ICP Variants:

- Sub/super Shape: applicable to data with outliers
- Pair weighting
- Other distance metrics

Plan

Global Matching

Approaches:

- Exhaustive search
- Normalization
- Random Sampling
- Invariance

Exhaustive Search

Global, Rigid, Pair, Sub/supershape

- Sample the space of 3D transformations
- Apply ICP to with samples as initial transformations
- Select best-matching result

Efficient if we can strongly constrain the space of possible transformations.

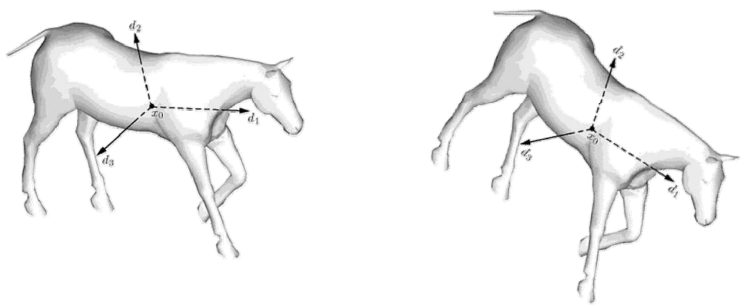
Insufficient for most robotics problems.

Normalization

Global, Rigid, Pair, Sameshape

Algorithm:

- Apply PCA to both point clouds to find R (may be several candidates)
- Compute t from the centers of gravity of both point clouds
- Apply ICP



Properties:

- Works well for noise-free non-isotropic shapes.
- Not applicable to partial views, or scenes with outliers.

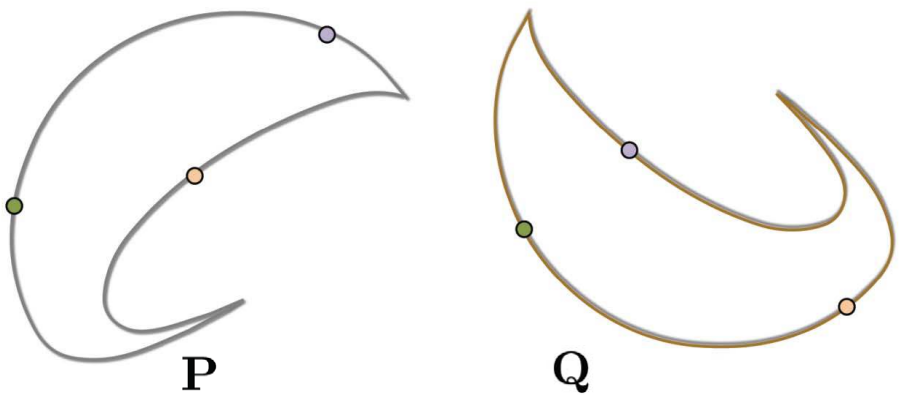
RANSAC

Global, Rigid, Pair, Sub/supershape

- Select 3 pairs of points

$$\{(x_i, x_i') : x_i \in \text{model}, \; x_i' \in \text{scene}\}_{i \in \{1,2,3\}}$$

- Estimate (R, t) (either ICP, or linear least-squares)
- Compute error



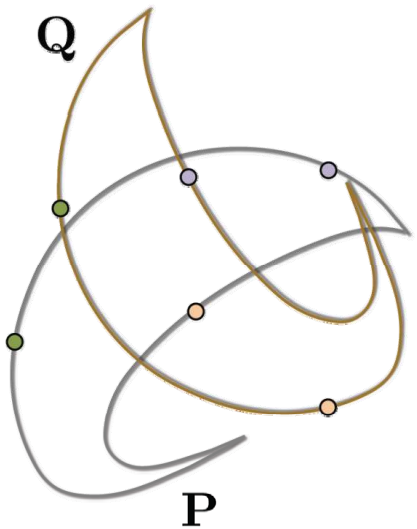
RANSAC

Global, Rigid, Pair, Sub/supershape

- Select 3 pairs of points

$$\{(x_i, x_i') : x_i \in \text{model}, \; x_i' \in \text{scene}\}_{i \in \{1,2,3\}}$$

- Estimate (R, t) (either ICP, or linear least-squares)
- Compute error



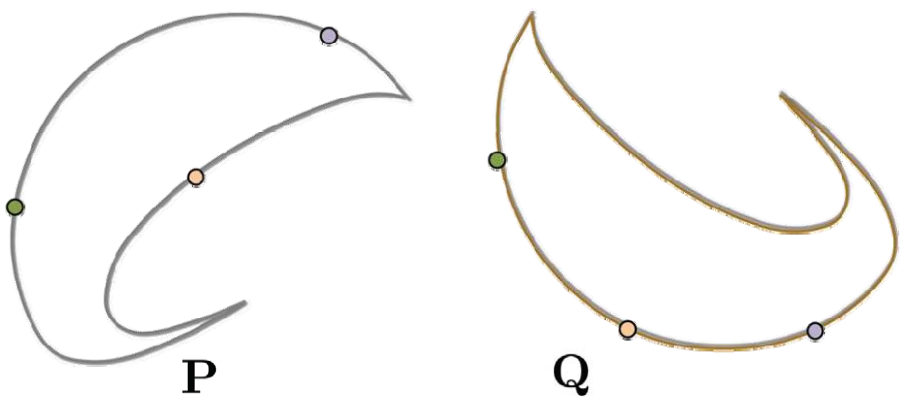
RANSAC

Global, Rigid, Pair, Sub/supershape

- Select 3 pairs of points

$$\{(x_i, x_i') : x_i \in \text{model}, \; x_i' \in \text{scene}\}_{i \in \{1,2,3\}}$$

- Estimate (R, t) (either ICP, or linear least-squares)
- Compute error

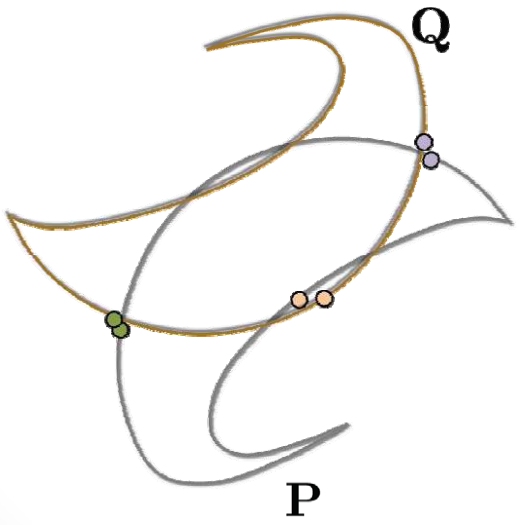


RANSAC

Global, Rigid, Pair, Sub/supershape

- Select 3 pairs of points

$$\{(x_i, x_i') : x_i \in \text{model},\ x_i' \in \text{scene}\}_{i \in \{1,2,3\}}$$



- Estimate (R, t) (either ICP, or linear least-squares)
- Compute error

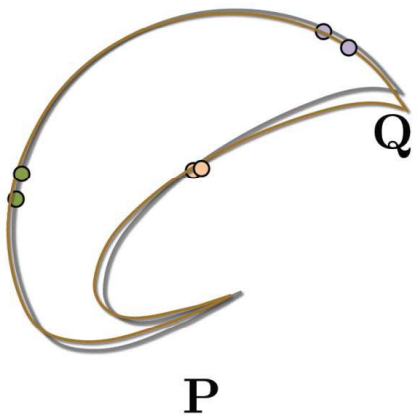
RANSAC

Global, Rigid, Pair, Sub/supershape

- Select 3 pairs of points

$$\{(x_i, x_i') : x_i \in \text{model}, \; x_i' \in \text{scene}\}_{i \in \{1,2,3\}}$$

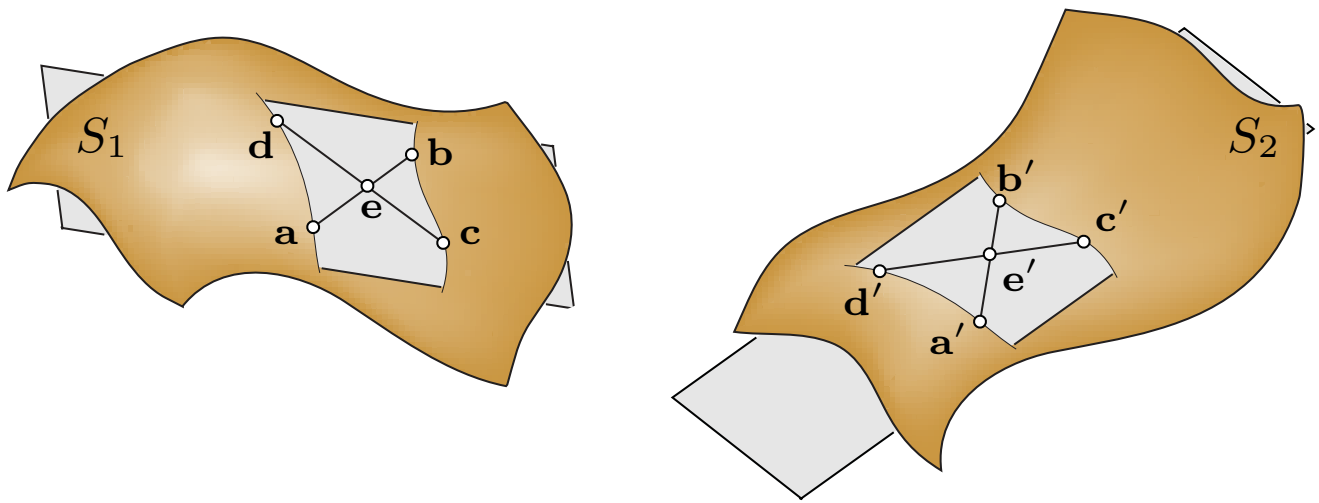
- Estimate (R, t) (either ICP, or linear least-squares)
- Compute error



RANSAC: Going Further

For each triple of points from the model, there are $O(n^3)$ triples in the scene.

By picking quadruples instead of triples, and choosing these quadruples carefully, the same problem becomes $O(2)$



4–points Congruent Sets for Robust Surface Registration, Aiger et al., SIGGRAPH 2008

Invariance

Try to characterize the shape using properties that are invariant under the desired set of transformations.

- (Optional: identify salient points on the object's surface)
- Compute a descriptor at each (salient) point.

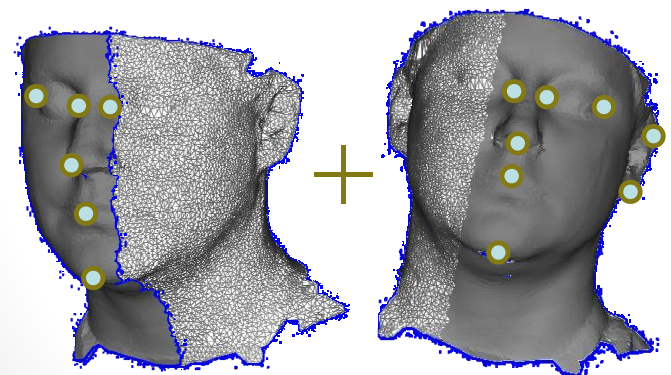
For instance, color is invariant to rigid body transformations

Color is not always an option (varies with lighting, not always available, or we may want to match together objects of similar shapes but different colors).

Invariance:

Global, Rigid, Pair, Sub/supershape

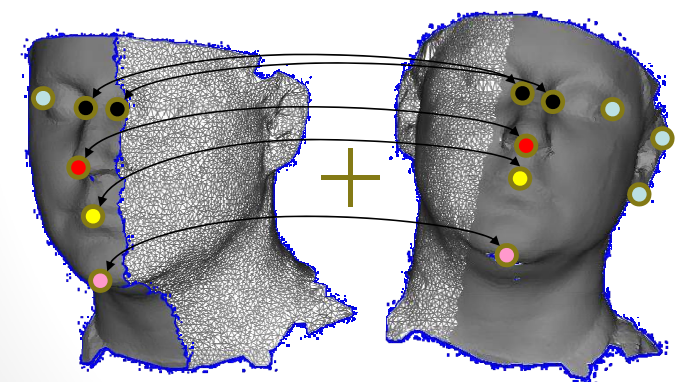
1. Find points of interest (optional, may use all available points instead)



Invariance:

Global, Rigid, Pair, Sub/supershape

1. Find points of interest (optional, may use all available points instead)
2. Compute a transormation-invariant shape descriptor at each of the points selected above

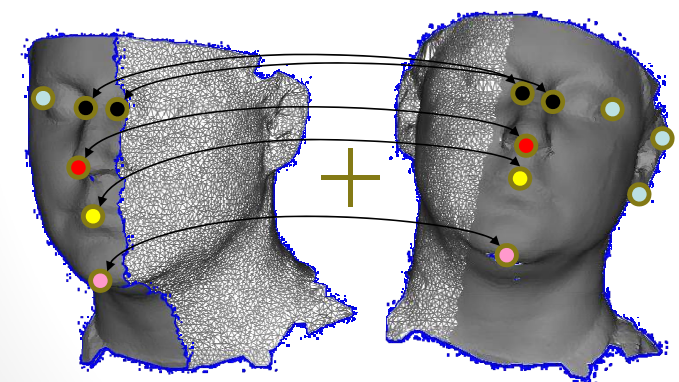


Partially Overlapping Scans

Invariance:

Global, Rigid, Pair, Sub/supershape

1. Find points of interest (optional, may use all available points instead)
2. Compute a transormation-invariant shape descriptor at each of the points selected above
3. Match points using their similarity in the shape descriptor space

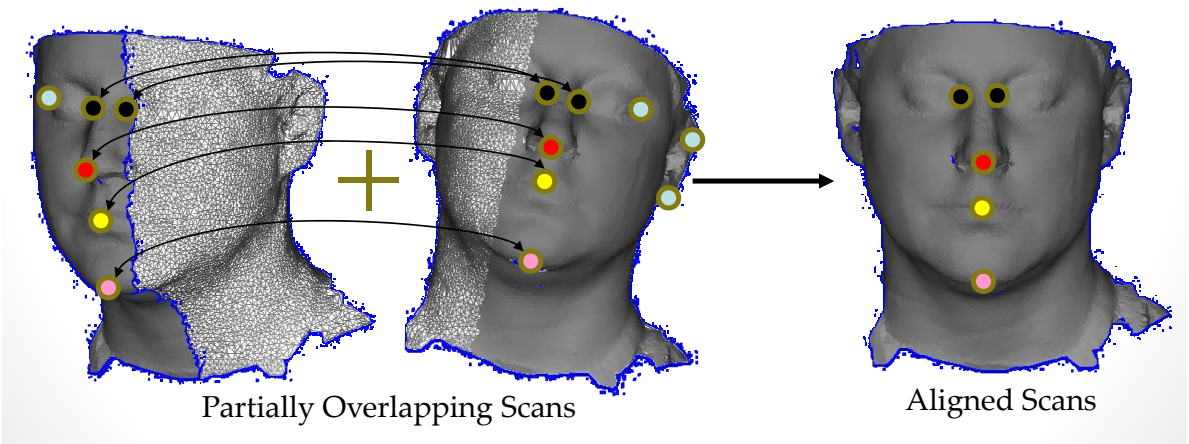


Partially Overlapping Scans

Invariance:

Global, Rigid, Pair, Sub/supershape

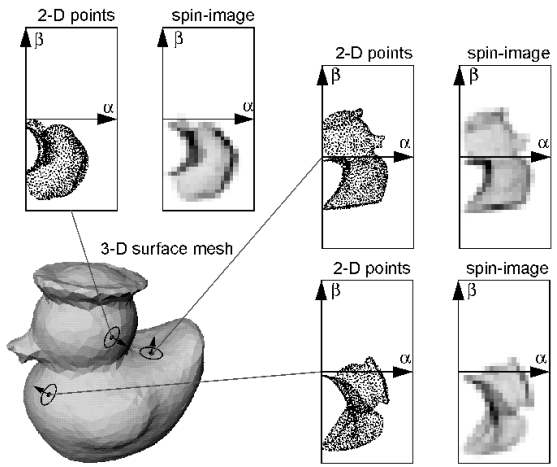
1. Find points of interest (optional, may use all available points instead)
2. Compute a transormation-invariant shape descriptor at each of the points selected above
3. Match points using their similarity in the shape descriptor space
4. Trigger ICP



RBM-invariant Descriptor: Spin Images

At each point x:

- Compute the surface normal at x (PCA), and the tangential plane.
- Project all other points (or a local neighborhood) into a reference frame centered on x, and with Z aligned with the normal
- Transform to a cylindrical coordinate system
- Discard the angular coordinate, keeping only the distances to the tangential plane and the distance to the normal vector



Using Spin Images for Efficient Object Recognition in Cluttered 3D Scenes Johnson et al, PAMI 99