INFO0948
Model Fitting and Shape Matching

Renaud Detry
Philippe Latour

University of Liège, Belgium

Updated March 23, 2018

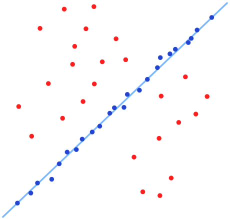
These slides are based on the following book:

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).

Model Fitting and Shape Matching

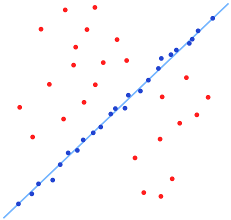
Finding instances of the given model/template in the set of points.
Aligning instances of the known shape/curve with the reference image.
These tasks often involve to search for the best warping of the model/shape during the finding/aligning processes.



Fitting a parametric model

Matching a nonparametric model

Fitting a parametric (geometric) Model



Fitting a parametric model

Fitting (a parametric or geometric model to a set of points) usually refers to finding the values of the model parameters giving rise to the best(?) alignment of the corresponding instances of the model with (a part of) the reference set of points.

Matching a nonparametric model

Matching a nonparametric model

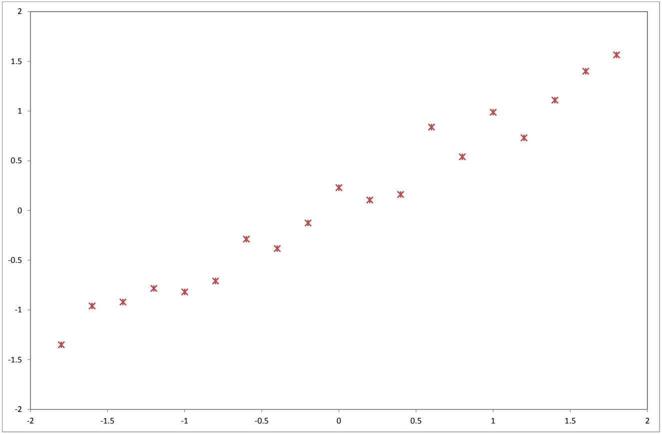
Matching (a nonparametric model to a set of point) usually refers to finding the best correspondence between the points of the model/template/shape and the reference image/set of points. This often implies to search for the transformation (or warping) of the model that will give rise to the best correspondence

Plan

Plan

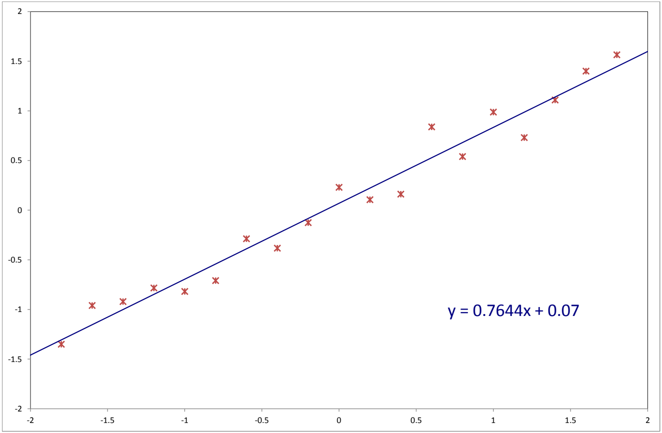
Linear Regression

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



Linear Regression

... find the line $y = \alpha + \beta x$ that "best" fits the data $\{(x_i, y_i)\}_{i \in [1,n]}$



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 is the best line? ... The answer is part of the problem definition!

Typically, the best line is the one that minimizes the sum-of-square error

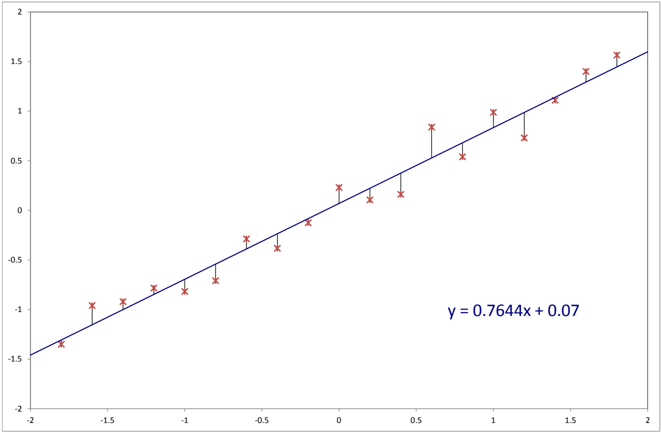
$$\min_{\alpha,\beta} Q(\alpha,\beta),\ \text{where } Q(\alpha,\beta) = \sum_{i=1}^{n} \hat{\varepsilon}_i^2 = \sum_{i=1}^{n} (y_i - \alpha - \beta x_i)^2$$

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

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

Linear Regression

This solution minimizes the sum of the vertical distance between the data points and the line $y = \alpha + \beta x$; it is a geometrical approach!



Linear Regression

Statistically speaking, we can describe the data as a contaminated version of the output of an instance of the model $y = g(x; \Theta) = \alpha + \beta x$.
We consider a particular model $g(x; \tilde{\Theta})$ such that the data are

$$y_i = g(x_i; \tilde{\Theta}) + \epsilon_i = (\tilde{\alpha} + \tilde{\beta} x_i) + \epsilon_i \qquad i = 1, ..., n$$

The errors ϵ_i are i.i.d. and normally distributed random variables with zero mean and, often, common variance σ_i^2. Then, the joint pdf is:

$$f(y_1, ..., y_n) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp[-\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - g(x_i; \tilde{\Theta}))^2]$$

whose Log-likelihood function is

$$\log L(\tilde{\Theta}, \sigma) = -\log(2\pi\sigma^2)^{n/2} - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - g(x_i; \tilde{\Theta}))^2$$

Linear Regression

In the statistical framework, we would like to maximize the probability (or the Log-likelihood) of the observed data.
So, we will have to minimize the same sum of square and we will obtain expressions which are identical to the solution of the geometric approach.

Then $\hat{\alpha}$ and $\hat{\beta}$ can also be viewed as statistical estimators of the true value α and β of the model $y = \alpha + \beta x$.
If x is a control variable which is known perfectly (without error of any kind), it may be proved that the estimators $\hat{\alpha}$ and $\hat{\beta}$ are:

- Normally distributed
- Unbiased $\mathbb{E}(\hat{\alpha}) = \alpha$ and $\mathbb{E}(\hat{\beta}) = \beta$
- Strongly consistent $\hat{\alpha} \to \alpha$ and $\hat{\beta} \to \beta$ when $n \to \infty$ (with probability 1).

Linear Regression

Two Model Fitting Paradigms

- **Geometrical**: Finding the model that passes approximately near (as close as possible) the observed (noisy) data points.
- **Statistical**: Finding the model that passes exactly through the true (but unknown) points that would have been observed in the absence of noise

For linear models, both approaches lead to the same results.
It is not necessarily the case for other models.

Statistical Regression in Computer Vision

Problem Statement - Errors-In-Variables (EIV)

In computer vision, x and y are BOTH contaminated by noise!

- Two variables $\tilde{x}$ and $\tilde{y}$ are assumed to be linked by an exact relationship $\tilde{y} = g(\tilde{x}; \Theta)$.
- We observe a perturbed value of these variables

  $$x_i = \tilde{x}_i + \delta_i \qquad y_i = \tilde{y}_i + \epsilon_i \qquad i = 1, ..., n$$

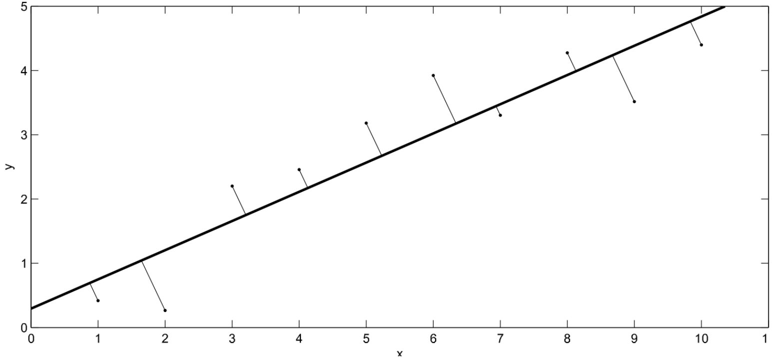
  where δ_i and ϵ_i are $2n$ random variables i.i.d. with zero means.
- We assume that all the δ_i have a common variance σ_x, all the ϵ_i have a common variance σ_y and they are normally distributed

  $$\delta_i \sim N(0, \sigma_x) \qquad \epsilon_i \sim N(0, \sigma_y)$$

- We assume, for isotropy reasons, that $\sigma_x = \sigma_y$

EIV in Computer Vision

The geometric solution to an EIV problem would be to minimize the "Total Least-Square" error representing the sum of the true distance between the data points and the model.



EIV in Computer Vision

The sum of the geometric distances between the data points and a line $y = \alpha + \beta x$ is

$$E_{TLS}(\alpha, \beta) = \frac{1}{1+\beta^2} \sum_{i=1}^{n} (y_i - \alpha - \beta x_i)^2$$

This expression will be minimum when

$$\hat{\beta} = \frac{s_{yy} - s_{xx} + \sqrt{(s_{yy} - s_{xx})^2 + 4 s_{xy}^2}}{2 s_{xy}} \qquad \hat{\alpha} = \bar{y} - \hat{\beta}\bar{x}$$

where

$$s_{xx} = \sum_{i=1}^{n} (x_i - \bar{x})^2 \qquad s_{xy} = \sum_{i=1}^{n} (y_i - \bar{y})^2 \qquad s_{xy} = \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})$$

EIV in Computer Vision

This geometric solution is invariant under orthogonal transformation (translation, rotation) and isotropic scaling and is then independent of the coordinate frame and of the scale of the axis.

This solution is also the Maximum Likelihood solution of the corresponding statistical problem (when $\sigma_x = \sigma_y$).

Unfortunately, in the case of EIV, the estimators $\hat{\alpha}$ and $\hat{\beta}$

- ARE NOT normally distributed.
- HAVE NO finite moments $\mathbb{E}(\hat{\alpha}) \to \infty$ and $\mathbb{E}(\hat{\beta}) \to \infty$
- HAVE NO finite mean squared errors $\mathbb{E}((\hat{\alpha} - \alpha)^2) \to \infty$ and $\mathbb{E}((\hat{\beta} - \beta)^2) \to \infty$

EIV in Computer Vision

Nevertheless, in practice, the previous estimators behave badly only for very high level of noise (large σ) or when $n \to \infty$.

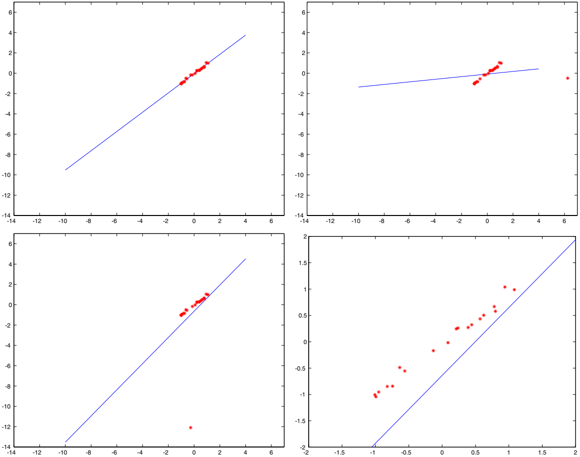
These situations (almost) never happen in Computer Vision applications, where we are interested in the limit case $\sigma \to 0$ (with n more or less constant).

Then, such parametric model fitting works well whenever:

- The level of noise is moderate
- The number of data points is large enough
- There are almost no outliers in the data
- The data comes from only one underlying model

Sensitivity to noise and outliers

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?

Plan

The Hough Transform

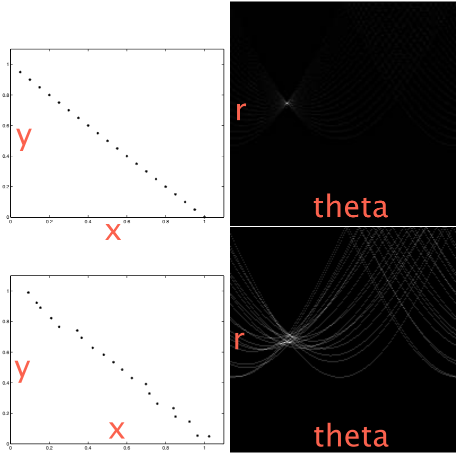
Principle: make each datapoint vote for all the model instances that could pass through it, and select the instance that collects the most votes.

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

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



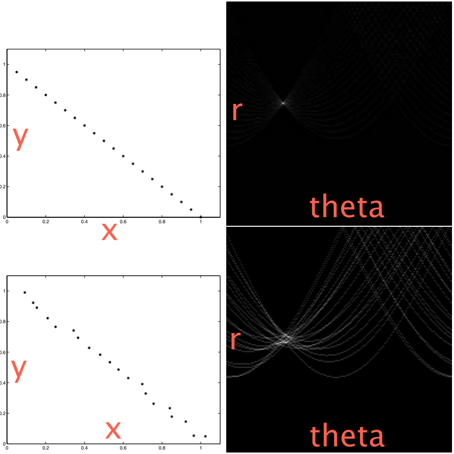
The Hough Transform

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 Hough Transform

Quantization Errors

An appropriate grid size is 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.

$\rightarrow$ Choose the grid carefully

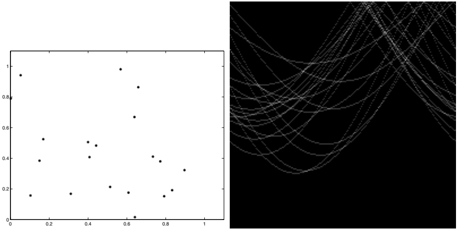
Problems with the Hough Transform

Difficulties with Noise

The Hough Transform is designed to be more robust to outliers and noise than the geometric fitting, but;

- "Phantom" lines can appear in large sets of randomly distributed datapoints.
- When multiple instances of the model are present, any instance is considered as noise for all the others.

$\rightarrow$ Attempt to remove outliers before fitting



Problems with the Hough Transform

Intractable when models have large number of parameters (> 3)

For instance;

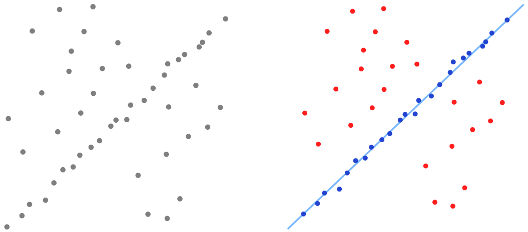
- Line fitting: 2 parameters (r, θ)
- Circle fitting: 3 parameters (x_c, y_c, r)
- Ellipse fitting: 5 parameters (x_c, y_c, θ, a, b)
- Homography estimation: 8 parameters

$\rightarrow$ Reduce the problem in successive steps of lower dimensionality and/or reduce the initial number of hypothesis to test

Plan

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.

If k is the number of iterations needed to finally obtain a set of n samples that are all inliers, and $\mathbb{E}[k]$ is its estimated value. We have

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

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

At each iteration, we draw at random n samples from the data set;

$$Pr(\text{all inliers}) = w^n \quad \Rightarrow \quad Pr(\text{at least one outlier}) = 1 - w^n$$

Let us denote by z the probability of having at least one outlier in each of the k iterations. z is the probability of failure of the random drawing process

$$Pr(\text{at least one outlier in the } k \text{ draw}) = z = (1 - w^n)^k$$

Then, we set z to how much we are willing to risk failing to fit our model

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

- 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?

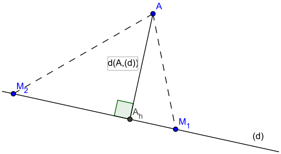
RANSAC Algorithm

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.

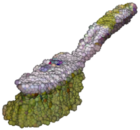
Variants of RANSAC that explicitly address the problem of finding multiple instances exist.

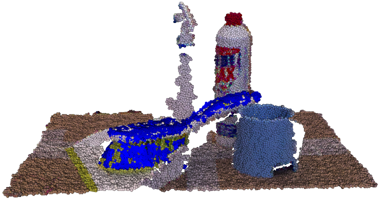
Plan

Shape Matching

Nonparametric model represented by

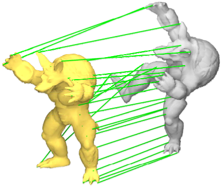
- Image (grey level, colour, depth)
- Set of 3D points
- Global shape descriptors vector
- Vector of local feature points with their descriptors.





Applications:

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

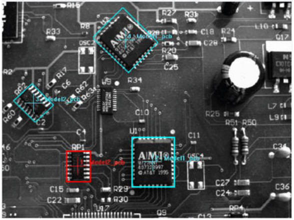


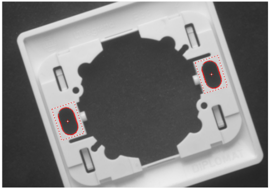
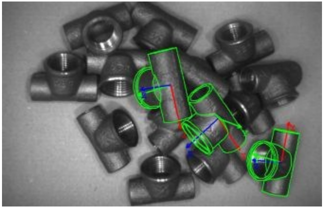
Plan

Template matching (Global, Rigid)

Finding the image transform or warping that would be needed to fit or align a model (a source image for instance) on another one.



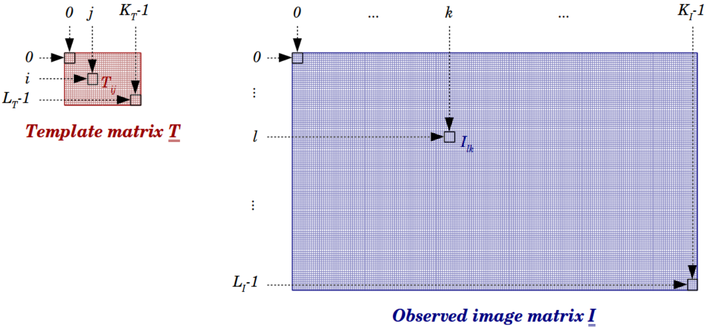


Naive solution I

We consider:

- the image $\boldsymbol{I}$ with L_I lines and K_I columns, represented by the matrix $\underline{\underline{I}} = (I_{lk})$ where $l \in [0, L_I[$, $k \in [0, K_I[$,
- the template $\boldsymbol{T}$ with L_T lines and K_T columns, represented by the matrix $\underline{\underline{T}} = (T_{ij})$ where $i \in [0, L_T[$, $j \in [0, K_T[$.

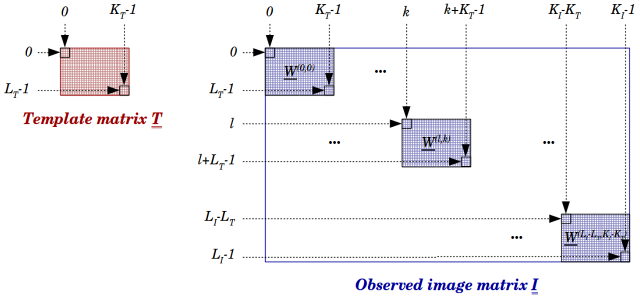


Naive solution II

We define all the admissible (sub-)windows $\boldsymbol{W}^{(\boldsymbol{l},\boldsymbol{k})}$ completely included within the image $\boldsymbol{I}$ and of the same size as the template $\boldsymbol{T}$ by the following sub-matrices $\underline{\underline{W}}^{(l,k)}$:

$$W_{ij}^{(l,k)} = \begin{cases} I_{l+i,k+j} & \text{for } i \in [0, L_T - 1]\,,\ j \in [0, K_T - 1] \\ 0 & \text{otherwise} \end{cases}$$

where $l \in [0, L_I - L_T]$ and $k \in [0, K_I - K_T]$ are the indices, in the image $\boldsymbol{I}$, of the upper left pixel of $\boldsymbol{W}^{(\boldsymbol{l},\boldsymbol{k})}$.



Naive solution III

Compute the Euclidean distance

$$dist\left(\underline{\underline{T}}, \underline{\underline{W}}^{(l,k)}\right) = \sum_{i=0}^{L_T-1}\sum_{j=0}^{K_T-1}\left[T_{ij} - W_{ij}^{(l,k)}\right]^2 \tag{1}$$

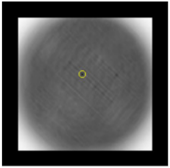
then create the distance map $\boldsymbol{D}$, represented by the matrix $\underline{\underline{D}}$:

$$D_{\left\lfloor \frac{K_T}{2} \right\rfloor + k, \left\lfloor \frac{L_T}{2} \right\rfloor + l} = \begin{cases} dist\left(\underline{\underline{T}}, \underline{\underline{W}}^{(l,k)}\right) & \text{for } l \in [0, L_I - L_T]\,,\ k \in [0, K_I - K_T] \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

and find the location of the minimum in these map.

Template $\boldsymbol{T}$

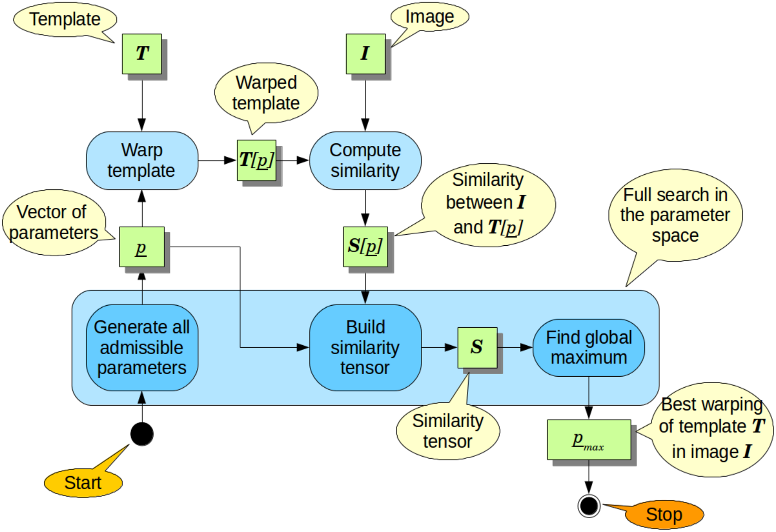
Observed image $\boldsymbol{I}$



Distance map $\boldsymbol{D}$

Block diagram

We may generalize a little bit our solution:



Method Taxonomy

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

Plan

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.

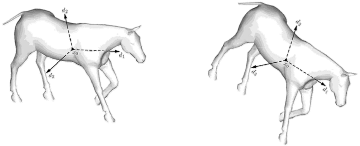
Insufficient for most robotics problems.

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.

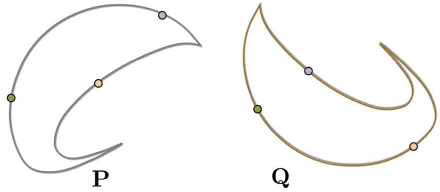
RANSAC

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



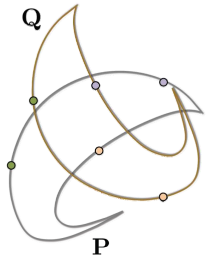
RANSAC

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



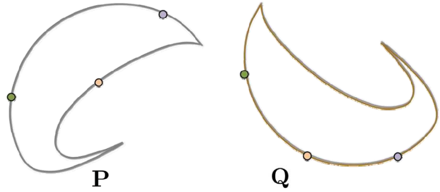
RANSAC

Global, Rigid, Pair, Sub/supershape

- Select 3 pairs of points

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

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



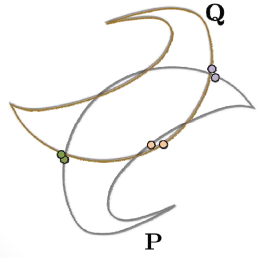
RANSAC

Global, Rigid, Pair, Sub/supershape

- Select 3 pairs of points

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

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



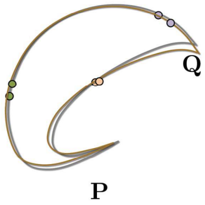
RANSAC

Global, Rigid, Pair, Sub/supershape

- Select 3 pairs of points

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

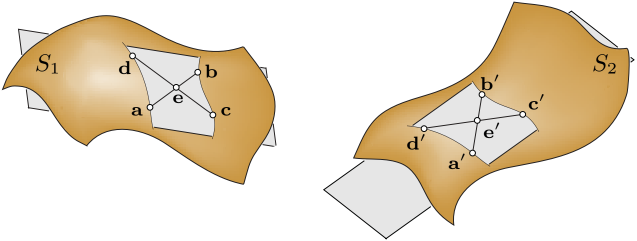
- Estimate (R, t) (either ICP, or linear least-squares)
- Compute error



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 $O(n^2)$



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.

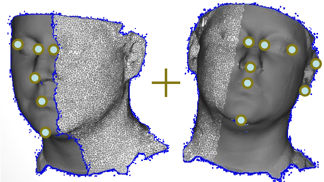
For instance, color is invariant to rigid body transformations

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)

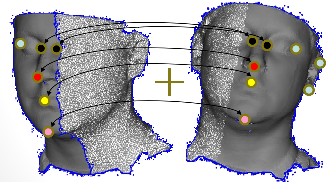


Partially Overlapping Scans

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

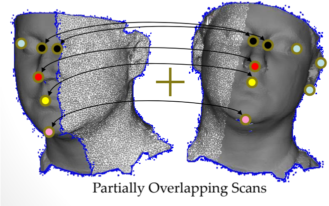


Partially Overlapping Scans

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

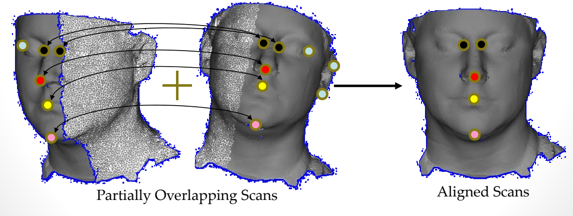


Partially Overlapping Scans

Invariance:

Global, Rigid, Pair, Sub/supershape

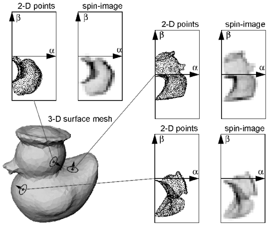
1. Find points of interest (optional, may use all available points instead)
2. Compute a transormation-invariant shape descriptor at each of the points selected above
3. Match points using their similarity in the shape descriptor space
4. Trigger ICP



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