“Only one line* passes through any three points”

* if the line is thick enough
Least-Squares Solution of $Ax = b$

CSE 4392-5369 “Vision-based Robot Sensing, Localization and Control”

Dr. Gian Luca Mariottini, Ph.D.

Department of Computer Science and Engineering
University of Texas at Arlington

WEB: http://ranger.uta.edu/~gianluca/
Why do we want to solve $Ax = b$?

This non-homogeneous linear system of equations appears very often!

- in mathematics;
- in Computer Science;
- in Engineering;
- in Robotics;
- in Computer Vision;
- ....

(Robotics) In path-planning the robot must pass through a set of navigation points (curve interpolation)
A simple example

Let us consider two given points \((x_1, y_1)\) and \((x_2, y_2)\).

Then the line passing through these points has parameters:

\[
\begin{align*}
m &= \frac{y_1 - y_2}{x_1 - x_2} \\
q &= \frac{x_1 y_2 - x_2 y_1}{x_1 - x_2}
\end{align*}
\]

Easy to solve!

But how can we solve it when we have many points, affected by noise and with erroneous measurements (outliers)?
Many noisy points

The best-fitting line won't pass through all the points!

\[ mx_1 + q - y_1 = 0 \]
\[ mx_1 + q - y_1 = \delta_1 \]
\[ mx_2 + q - y_2 = \delta_2 \]
\[ \vdots \]
\[ mx_n + q - y_n = \delta_n \]

Our goal can then be re-written as:

Find \((m, q)\) such that \(J = \sum_{i=1}^{n} (\delta_i)^2\) is minimized

that is:

\[ (\hat{m}, \hat{q}) = \arg\min_{(m,q)} \sum_{i=1}^{n} (mx_i + q - y_i)^2 \]

\(\delta_i\) are errors and we will refer to them as \(e_i\)
Writing our problem differently

The best-fitting line won't pass through all the points!

\[ J = \sum_{i=1}^{n} (mx_i + q - y_i)^2 \]

\[ = \left( \begin{bmatrix} x_1 & 1 \\ m & q \\ \end{bmatrix} - y_1 \right)^2 + \left( \begin{bmatrix} x_2 & 1 \\ m & q \\ \end{bmatrix} - y_2 \right)^2 + \ldots + \left( \begin{bmatrix} x_n & 1 \\ m & q \\ \end{bmatrix} - y_n \right)^2 \]

\[ = [e_1, e_2, \ldots, e_n] \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix} = e^T e \]

We can write it in this way:

\[ e = Ax - b \]

“Non-homogeneous Linear System”

where:

\[ A = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \quad x = \begin{bmatrix} m \\ q \end{bmatrix} \quad b = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \]
Least-Squares Minimization of $J$

Our goal is to find $\mathbf{X}$ such that

$$J = (\mathbf{Ax} - \mathbf{b})^T (\mathbf{Ax} - \mathbf{b})$$

is minimized.

$$\frac{\partial J}{\partial \mathbf{x}} = 0 = 2 \left[ \frac{\partial (\mathbf{Ax} - \mathbf{b})}{\partial \mathbf{x}} \right]^T (\mathbf{Ax} - \mathbf{b}) = 2 \mathbf{A}^T (\mathbf{Ax} - \mathbf{b}) = 0$$

$$(\mathbf{A}^T \mathbf{A}) \mathbf{x} = \mathbf{A}^T \mathbf{b} \quad \Rightarrow \quad \mathbf{x} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{b}$$

The inverse of a generic 2x2 matrix

$$\mathbf{B} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \Rightarrow \quad \mathbf{B}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

EXAMPLE (Line through 2 points)
Application: Estimating the intersection of Lines

\[ y = m_1 x + q_1 \]
\[ y = m_2 x + q_2 \]
\[ y = m_3 x + q_3 \]
Application: Estimation of the principal point
Application: Estimation of the principal point

Gian Luca Mariottini, Univ. Texas at Arlington
Application: Estimation of the principal point
Application: Estimation of the principal point

$(u_0, v_0)$
So far we assumed that no outliers were present.

However, outliers can often appear!
- Large measurement errors;
- Unknown correspondences $x_i \leftrightarrow y_i$.

Outliers can severely affect the estimated parameters!

See Ex_LineFitRobust.m.

Our GOAL becomes then to:

<< Find the line that minimizes the sum of squared perpendicular distances and eliminate the outliers from the presented data >>

This is usually referred to as robust estimation.

How would you solve this problem?
The main idea in RANSAC

Given a set of 2-D points, find the line that minimizes the sum of squared perpendicular distances subject to none of the selected points (inlier) deviates more than $t$.

For example, the threshold $t$ can be set according to the measurement noise (e.g., $3\sigma$).

We will describe a successful robust estimator: RANSAC (RANdom Sample Consensus) [Fishler, Bolles 1981].
The RANSAC Algorithm

Idea: Two points are Sampled RANdomly and a line is fitted to them. Support is computed, as the number of points lying within a distance threshold to the obtained line (Consensus). This random selection is repeated a number of times and the line with highest consensus is selected. The points within that threshold are the inliers.

Comparison:
- Least Squares: uses all data possible
- RANSAC: uses as small an initial data set as feasible, and enlarges it with consistent data.

---

Objective

Robust fit of a model to a data set $S$ which contains outliers.

Algorithm

(i) Randomly select a sample of $s$ data points from $S$ and instantiate the model from this subset.
(ii) Determine the set of data points $S_i$ which are within a distance threshold $t$ of the model. The set $S_i$ is the consensus set of the sample and defines the inliers of $S$.
(iii) If the size of $S_i$ (the number of inliers) is greater than some threshold $T$, re-estimate the model using all the points in $S_i$ and terminate.
(iv) If the size of $S_i$ is less than $T$, select a new subset and repeat the above.
(v) After $N$ trials the largest consensus set $S_i$ is selected, and the model is re-estimated using all the points in the subset $S_i$. 
The Hough Transform was introduced to detect complex patterns of points in binary images.

In other words, we want to find instances of an object (with a certain type of parameters) using a voting procedure in their parameter space.
The Hough Transform for Lines (1)

Basic steps:

1) Transform the line detection into a line intersection problem:

- i.e., any **point in the image** can be transformed to a **line in the parameter space**

As \((m, q)\) vary, they represent all the possible lines through \(u_i\) so....

- a line in the image defined by another point \(u_j\) is identified by the **intersection of two lines** in the parameter space.
2) Transform the line intersection in a voting algorithm:

- The parameter space is divided into finite grid cells (bins);
- Each point in the image is transformed in a line in the parameter space;
- Increment all counters on the corresponding line in parameter space;
- Peaks in the parameter space will correspond now to lines in the image voted the most
The Hough Transform for Lines (3)

Keeping the parameter space finite
- m and q take values on an (possibly) infinite range! (time consuming)
- parameterizing with (m,q) does not allow for x=k (with k constant).

The **polar line-representation** can be introduced:

\[ \rho = x \cos \theta + y \sin \theta \]

Notice that now an image point \((x,y)\) is represented by a sinusoid in the parameter space.