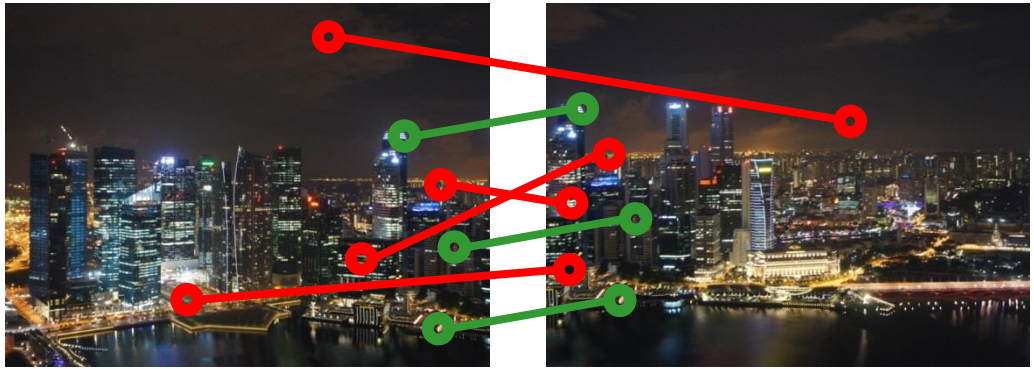
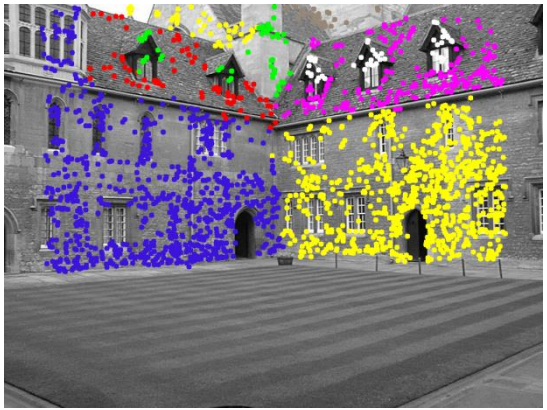
Robust Fitting and Optimization

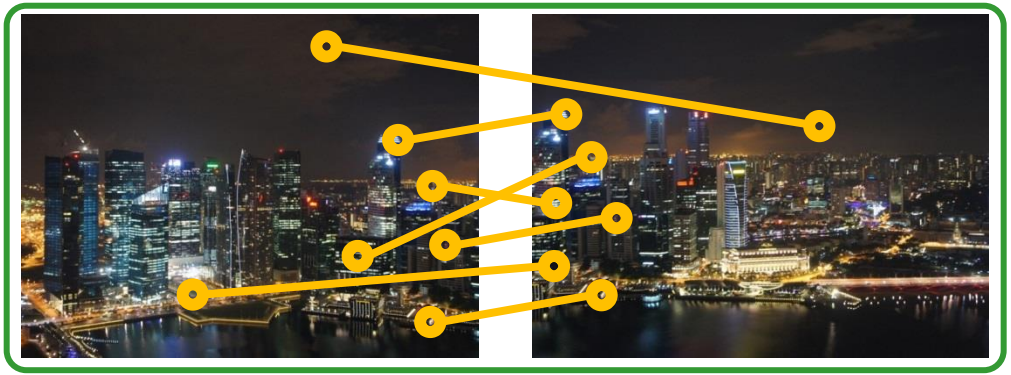


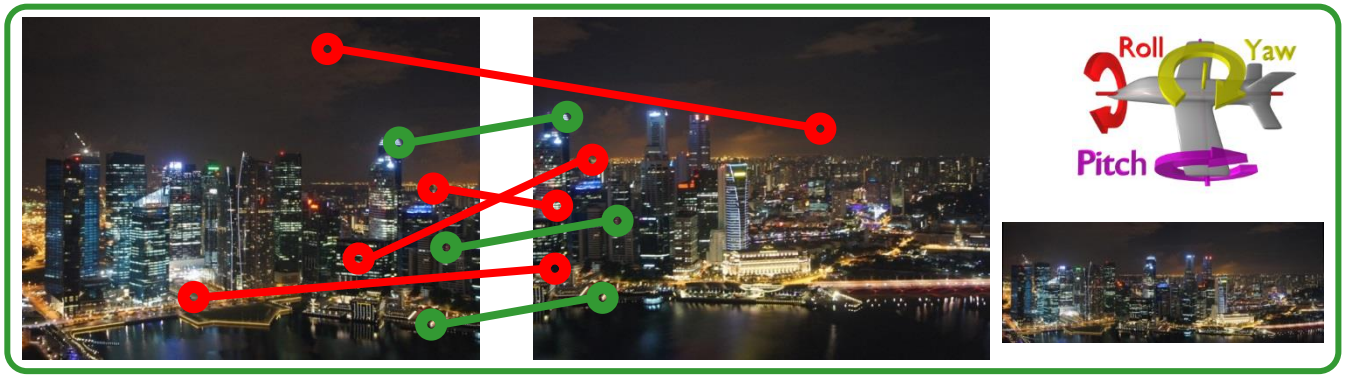


Dr. Jean-Charles Bazin

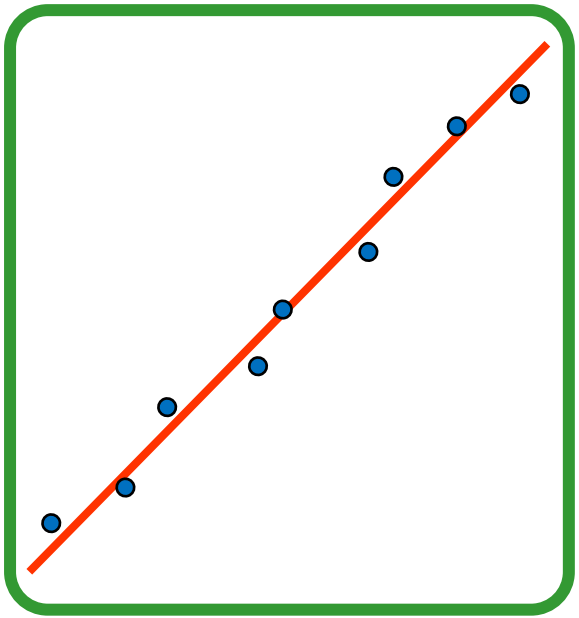
Motivation

Motivation

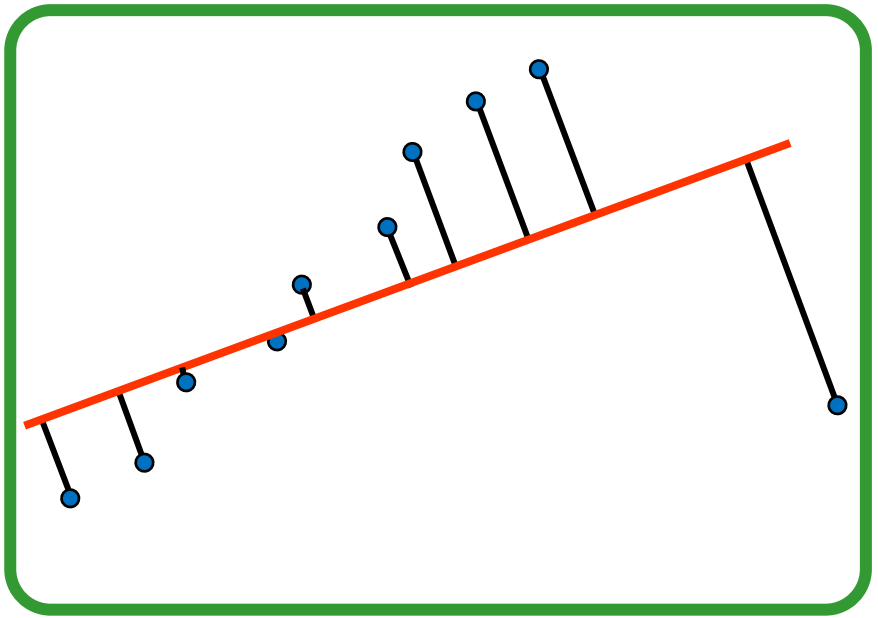




Outlier - a motivation example



Only **inliers**



Effects due to a single **outlier**

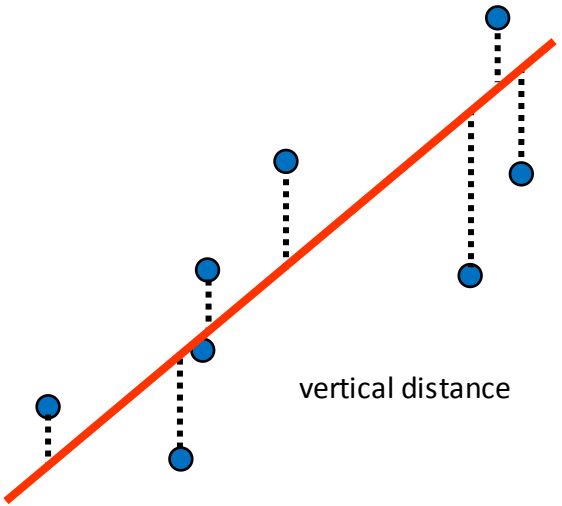
Detecting outliers is very important!

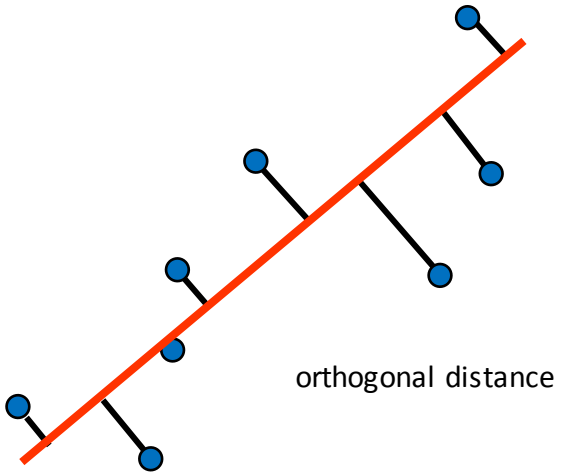
Contents

- Fitting and norms
- RANSAC
- WLS, IRLS, MLS

Fitting and norms

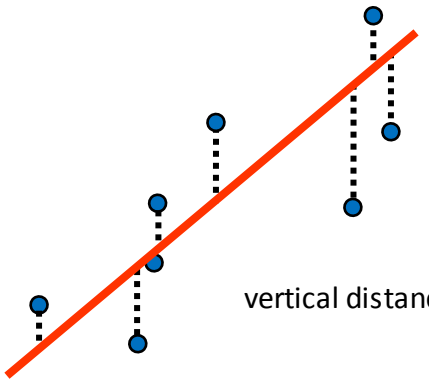
- Line fitting
  - Input: a set of *n* 2D points (x_i, y_i)
  - Output: the line $\Theta = (a, b)$ minimizing the fitting error
- Line equation: $y = ax + b$
- Fitting cost: YOU choose it





Fitting and norms

- In high school
  - **Sum** of **squared vertical** distances



$$\min_{\Theta} \sum_{i=1}^{n} (y_i - (ax_i + b))^2$$

  - Solving

$$\frac{\partial R}{\partial a} = -2\sum_{i=1}^{n} (y_i - (ax_i + b))\, x_i = 0$$

$$\frac{\partial R}{\partial b} = -2\sum_{i=1}^{n} (y_i - (ax_i + b)) = 0$$

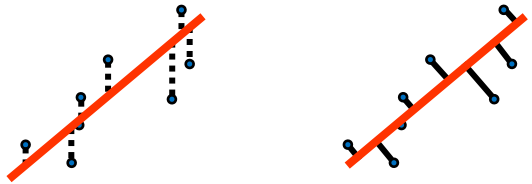
$$b\sum_{i=1}^{n} x_i + a\sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} x_i y_i$$

$$nb + a\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$$

...

Norms

- In grad studies:
  - Why **vertical**? Application-dependent
  - Why **sum**? Why **squared**? Related to norm



- What about sum of absolute distances?

$$\sum_{i=1}^{n} |y_i - (ax_i + b)| = |y_1 - (ax_i + b)| + \ldots + |y_n - (ax_n + b)|$$

- What about the max error?
  - Also called minmax or minimax

$$\max_i |y_i - (ax_i + b)| = \max\left(|y_1 - (ax_1 + b)|, \ldots, |y_n - (ax_n + b)|\right)$$

- How to solve? See next slides

Norms

- Notations $\mathbf{x} \in \mathbb{R}^n$ $\mathbf{x} = (x_1, \ldots, x_n)$
- L_p norm

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

$$\mathbf{x} = (x_1, x_2, x_3) = (-3, 0, 4)$$
$$\|\mathbf{x}\|_2 = \sqrt{(-3)^2 + 0^2 + 4^2} = \sqrt{25} = 5$$
$$\|\mathbf{x}\|_1 = |-3| + |0| + |4| = 7$$

- "Special" norms
  - L_∞ $\|\mathbf{x}\|_\infty = \max_i |x_i| = \max(|x_1|, \ldots, |x_n|)$
    - Widely applied for "worst case"
  - L_0 $\|\mathbf{x}\|_0 = \text{card}(i | x_i \neq 0)$ i.e. the number of non-zero elements
    - Very popular in sparse representation and learning. See also RPCA lecture $\|\mathbf{x}\|_0 = 2$

Formal definition

Let us consider a vector space V over a subfield F of the complex numbers. A norm on V is a function $p : V \to \mathbb{R}$ with the following properties:

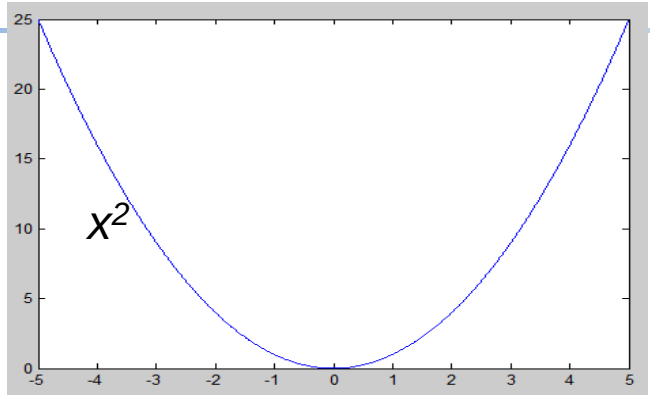
For all $a \in F$ and all $\mathbf{u}, \mathbf{v} \in V$,

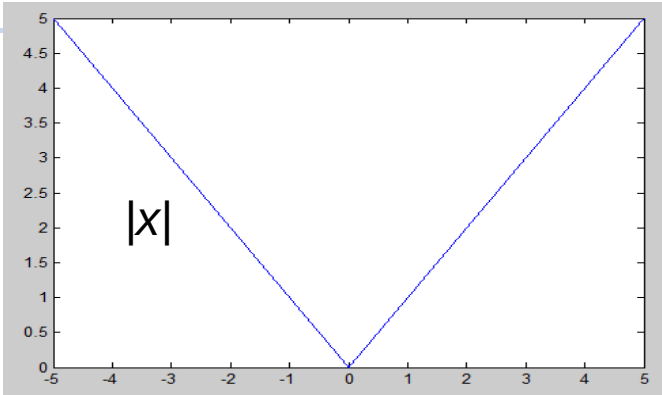
- $p(a\mathbf{v}) = |a|p(\mathbf{v})$
- $p(\mathbf{u}+\mathbf{v}) \leq p(\mathbf{u}) + p(\mathbf{v})$
- If $p(\mathbf{v}) = 0$ then $\mathbf{v}$ is the zero vector

- Will be used for the RPCA (rank minimization) lecture
- Quick question: is $\sum_{i=1}^{n} x_i$ a norm?

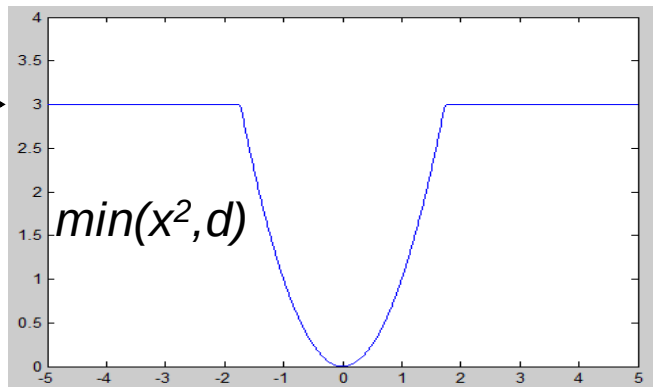
Popular cases (in 1D)

- L_2: $\arg\min_u \sum_{i=1}^{n} |u - x_i|^2$
  - Solution?
    - average $u = \frac{1}{n} \sum_{i=1}^{n} x_i$ To prove
- L_1: $\arg\min_u \sum_{i=1}^{n} |u - x_i|$
  - Solution?
    - median. Remember how to compute it?





d →

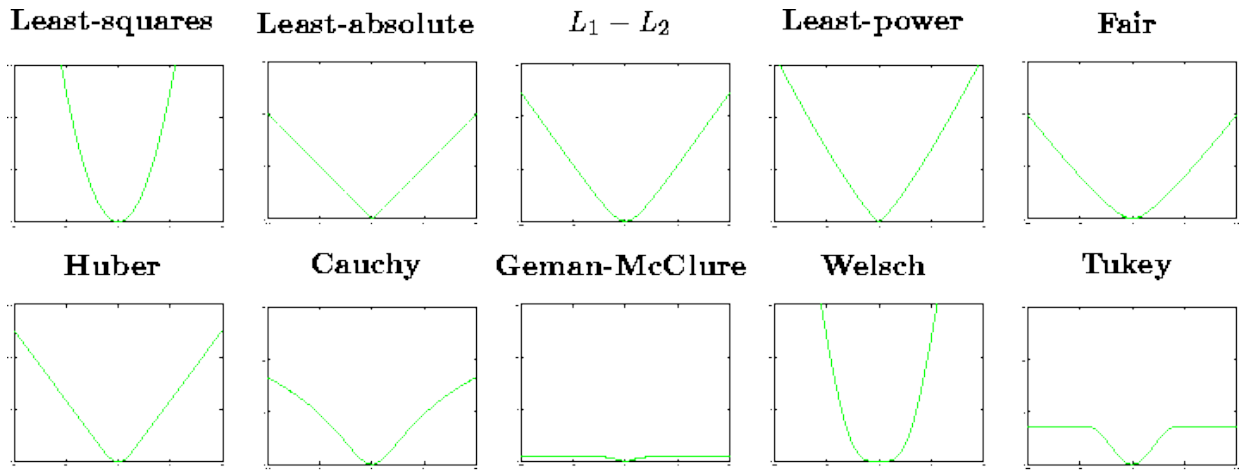


Design/play with your cost function

Which one to choose?

Loss functions

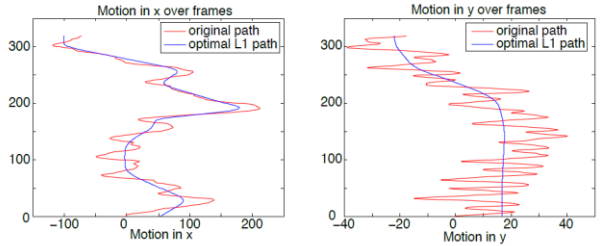
- Beyond just norms
- Also called “cost functions”
- YOU choose/define it



type	$\rho(x)$
L_2	$x^2/2$
L_1	$\lvert x\rvert$
$L_1 - L_2$	$2(\sqrt{1 + x^2/2} - 1)$
L_p	$\frac{\lvert x\rvert^\nu}{\nu}$
“Fair”	$c^2[\frac{\lvert x\rvert}{c} - \log(1 + \frac{\lvert x\rvert}{c})]$
Huber $\begin{cases}\text{if } \lvert x\rvert \le k \\ \text{if } \lvert x\rvert \ge k\end{cases}$	$\begin{cases}x^2/2 \\ k(\lvert x\rvert - k/2)\end{cases}$
Cauchy	$\frac{c^2}{2}\log(1 + (x/c)^2)$
Geman-McClure	$\frac{x^2/2}{1 + x^2}$
Welsch	$\frac{c^2}{2}[1 - \exp(-(x/c)^2)]$
Tukey $\begin{cases}\text{if } \lvert x\rvert \le c \\ \text{if } \lvert x\rvert > c\end{cases}$	$\begin{cases}\frac{c^2}{6}(1 - [1 - (x/c)^2]^3) \\ (c^2/6)\end{cases}$

- L2, L1, Huber are the most popular, depending on the applications
- Huber treats small errors like L2 (efficiency) and large errors like L1 (robustness)

Application – video stabilization



$$\mathcal{O}(P) = w_1|D(P)|_1 + w_2|D^2(P)|_1 + w_3|D^3(P)|_1$$

Figure 4: Optimal camera path obtained via our constrained LP formulation for the video in fig. 10. Shown is the motion in x and y over a period of 320 frames, using the inclusion constraint for a crop window of 75% size of the original frame. Note how the optimal path is composed of constant, linear and parabolic arcs. Our method is able to replace the low-frequency bounce in y (person walking with a camera) with a static camera while guaranteeing that all pixels within the crop window are valid.

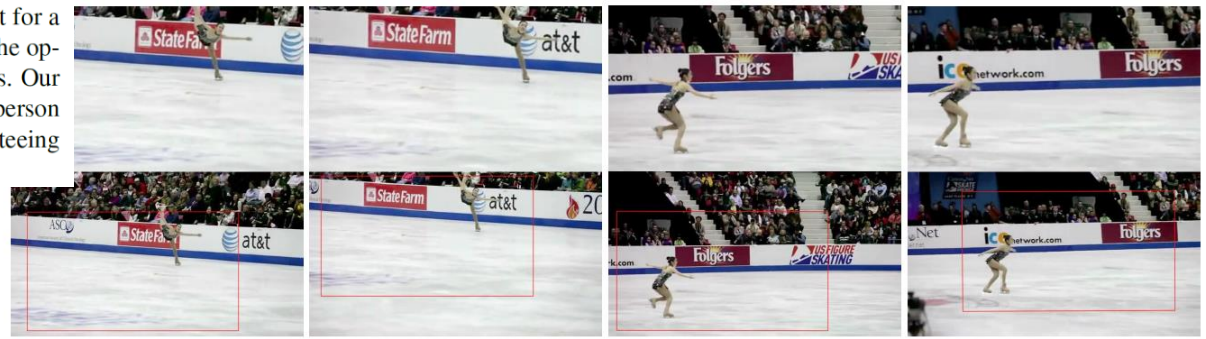


Figure 11: Example from YouTube "Fan-Cam" video. Top row: Stabilized result, bottom row: Original with optimal crop window. Our system is able remove jitter as well as low-frequency bounces. Our L1 optimal camera path conveys a viewing experience that is much closer to a professional broadcast than a casual video. Please see video.

YouTube

See youtube.com/editor

"Auto-Directed Video Stabilization with Robust L1 Optimal Camera Paths", by M. Grundmann and V. Kwatra and I. Essa, CVPR 2011

http://cpl.cc.gatech.edu/projects/videostabilization/

Application – video stabilization

video

"Auto-Directed Video Stabilization with Robust L1 Optimal Camera Paths", by M. Grundmann and V. Kwatra and I. Essa, CVPR 2011
http://cpl.cc.gatech.edu/projects/videostabilization/

Linear system

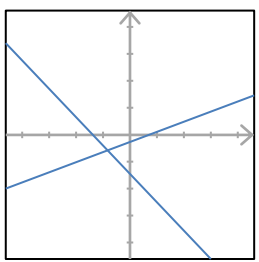
$$Ax = b$$

mxn nx1 mx1

See more in the next lecture (applications and solving)

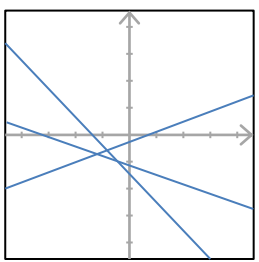
- *m=n: "critical case"*

$$5x + 2y = 9$$
$$3x + 4y = 12$$



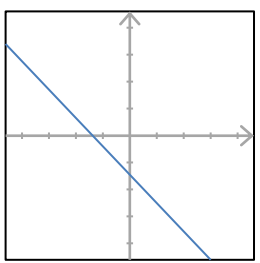
- ***m>n*: over-determined**
  → (linear) least-square

$$5x + 2y = 9$$
$$3x + 4y = 12$$
$$4x + 5y = 14$$



- *m<n*: under-determined
  - Infinite number of solutions
  - "family" of solutions

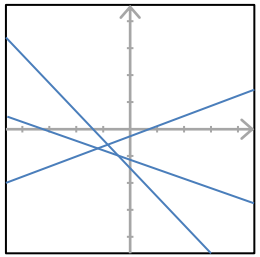
$$5x + 2y = 9$$



Linear system

$$Ax = b$$

$$5x + 2y = 9$$
$$3x + 4y = 12$$
$$4x + 5y = 14$$



- If overdetermined, no solution generally (if noise)
  - i.e. no solution verify all the equations

- "Least squares“: minimization of the sum of the squares of the errors
  - Not L2, but squared L2
  - The **least squares** method approximates the solution of overdetermined system

$$Ax = b \Longrightarrow \min_x \|Ax - b\|$$

$$\arg\min_x \|Ax - b\|_2 = \arg\min_x \|Ax - b\|_2^2$$

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

$$\arg\min_x \|Ax - b\|_2 = \arg\min_x \|(A_1 x - b_1, \ldots, A_m x - b_m)\|_2$$
$$= \arg\min_x \sqrt{\sum_{i=1}^{m} (A_i x - b_i)^2}$$
$$= \arg\min_x \sum_{i=1}^{m} (A_i x - b_i)^2$$

Solution by yourself: compute the gradient, set it to 0, etc..

Linear system

So, solving linear system with L2 is easy:

$$\arg\min_{\mathbf{x}} \|A\mathbf{x}-b\|_2 = \arg\min_{\mathbf{x}} \|A\mathbf{x}-b\|_2^2 = \arg\min_{\mathbf{x}} \sum_{i=1}^{m} (A_i\mathbf{x}-b_i)^2 = \arg\min_{\mathbf{x}} \sum_{i=1}^{m} e_i^2$$

What about L_1?

$$\arg\min_{\mathbf{x}} \|A\mathbf{x}-b\|_1 = \arg\min_{\mathbf{x}} \sum_{i=1}^{m} |A_i\mathbf{x}-b_i| = \arg\min_{x} \sum_{i=1}^{m} |e_i|$$

What about L_∞?

$$\arg\min_{\mathbf{x}} \|A\mathbf{x}-b\|_\infty = \arg\min_{\mathbf{x}} \max_{i=1,\ldots,m} |A_i\mathbf{x}-b_i| = \arg\min_{\mathbf{x}} \max_{i=1,\ldots,m} |e_i|$$

See homework#2

Linear system (L_1)

$$\arg\min_{\mathbf{x}} \|A\mathbf{x} - b\|_1 = \arg\min_{\mathbf{x}} \sum_{i=1}^{m} |A_i\mathbf{x} - b_i|$$

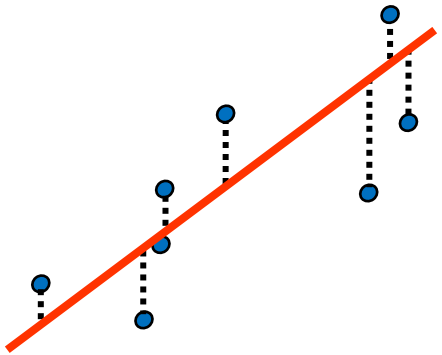
$$\min_{\mathbf{x}, t_1, \ldots, t_m} \sum_{i=1}^{m} t_i$$

$$s.t. |A_i\mathbf{x} - b_i| \leq t_i \ , \ i = 1, \ldots, m$$

$$\min_{\mathbf{x}, t_1, \ldots, t_m} \sum_{i=1}^{m} t_i$$

$$s.t. -t_i \leq A_i\mathbf{x} - b_i \leq t_i \ , \ i = 1, \ldots, m$$

Solved by Linear Programming



$|u| \leq q \Leftrightarrow -q \leq u \leq q$
i.e. $-q \leq u$ and $u \leq q$

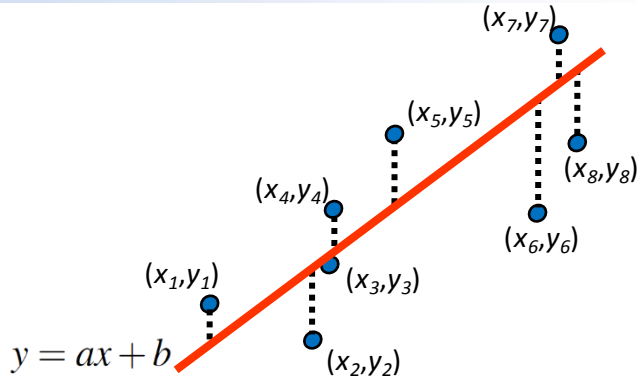
Linear system (L_1)

Don't be confused by the names of the variables

$$\arg\min_{\mathbf{x}} \|A\mathbf{x}-b\|_1 = \arg\min_{\mathbf{x}} \sum_{i=1}^{m} |A_i\mathbf{x}-b_i|$$

$$\min_{\mathbf{x},t_1,\ldots,t_m} \sum_{i=1}^{m} t_i$$

$$s.t. -t_i \le A_i\mathbf{x}-b_i \le t_i \ , \ i=1,\ldots,m$$



Line model: $y = ax + b$

Parameters: $\Theta = (a,b)$

Data points: (x_i,y_i)

L_1 cost: $\min_{\Theta=(a,b)} \sum_{i=1}^{n} |y_i-(ax_i+b)|$

$$\min_{\mathbf{x},t_1,\ldots,t_m} \sum_{i=1}^{m} t_i$$

$$s.t. -t_i \le A_i\mathbf{x}-b_i \le t_i \ , \ i=1,\ldots,m$$

$$A_i = \begin{pmatrix} x_i & 1 \end{pmatrix} \qquad b_i = \begin{pmatrix} y_i \end{pmatrix} \qquad \mathbf{x} = (a,b)$$

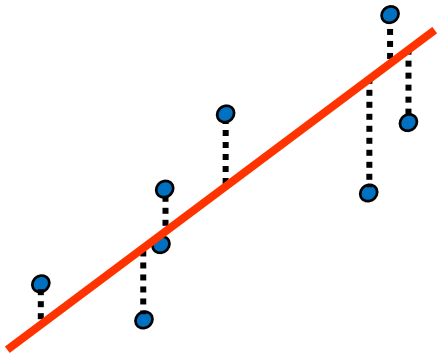
Linear system (L_∞)

$$\arg\min_{\mathbf{x}} \|A\mathbf{x} - b\|_\infty = \arg\min_{\mathbf{x}} \max_{i=1,\ldots,m} |A_i\mathbf{x} - b_i|$$

Not robust, but to minimize the "worst" distance

$$\min_{\mathbf{x},t} t$$
$$s.t. |A_i\mathbf{x} - b_i| \le t \ , \ i = 1, \ldots, m$$

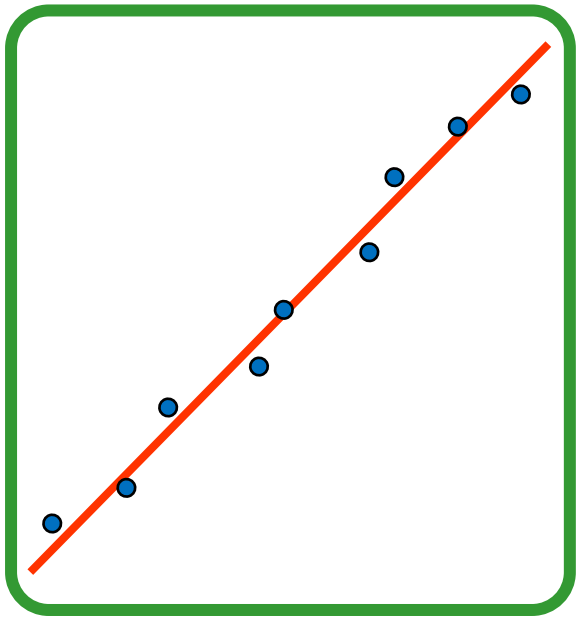
$$\min_{\mathbf{x},t} t$$
$$s.t. -t \le A_i\mathbf{x} - b_i \le t \ , \ i = 1, \ldots, m$$



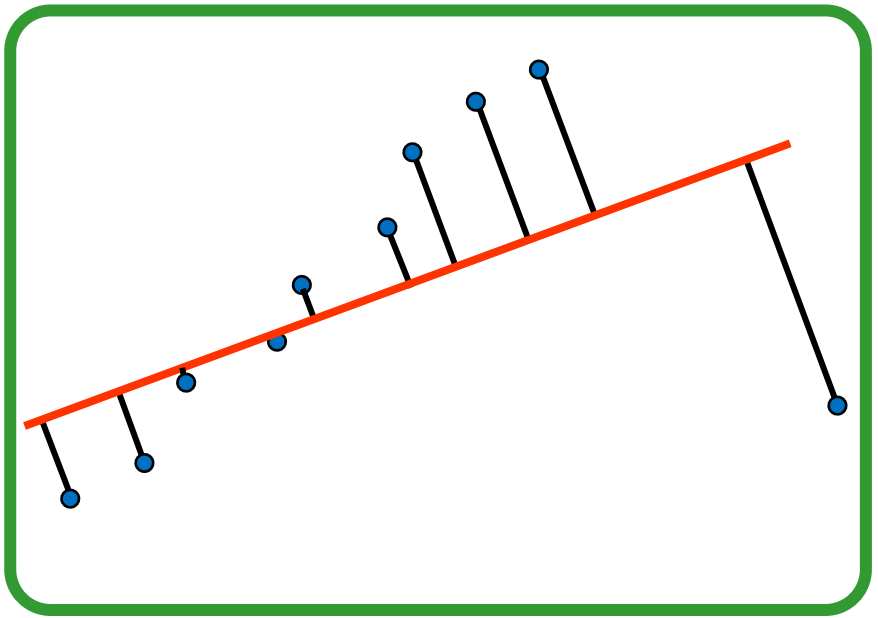
$$|u| \le q \Leftrightarrow -q \le u \le q$$
i.e. $-q \le u$ and $u \le q$

Solved by Linear Programming

Outlier - a motivation example



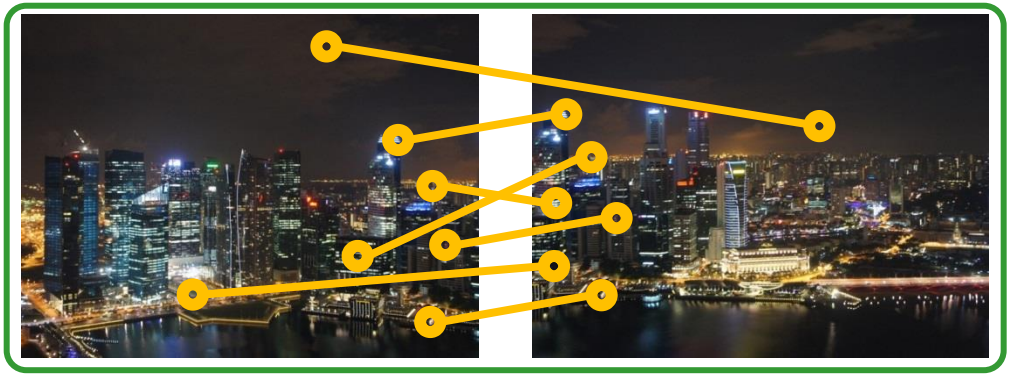
Only **inliers**

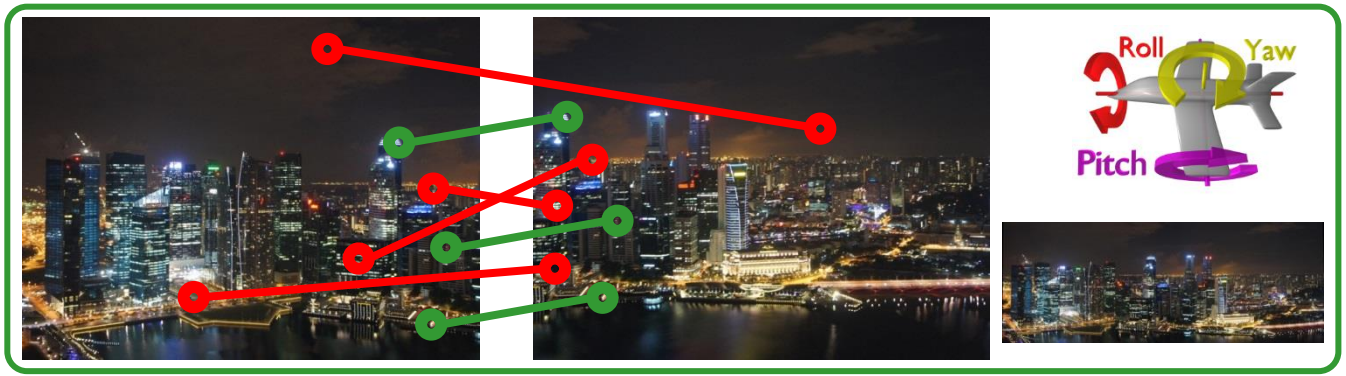


Effects due to a single **outlier**

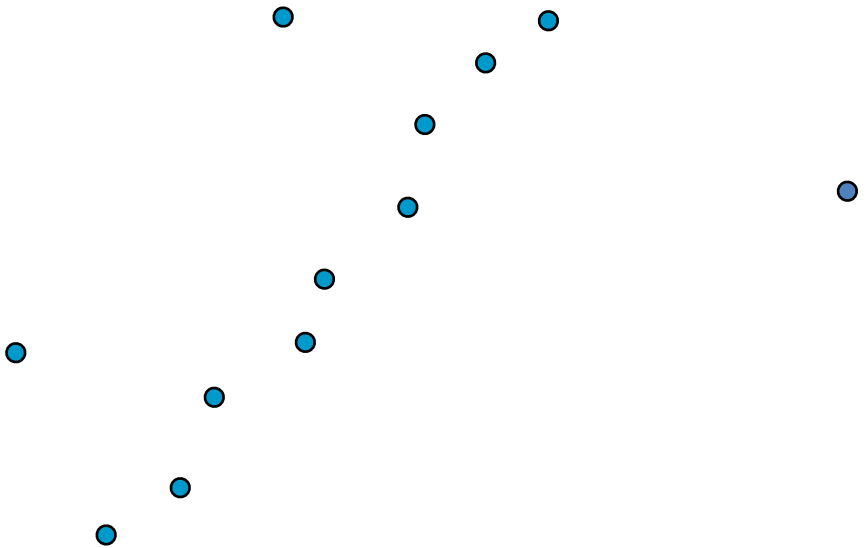
Detecting outliers is very important!
How to detect them? → RANSAC, robust least squares, M-estimator, etc…

Motivation



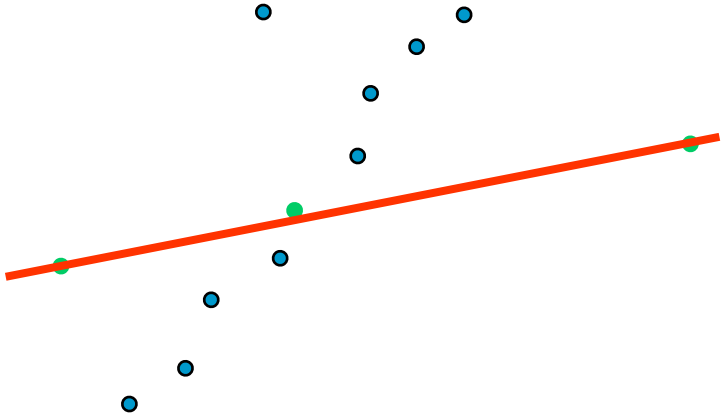


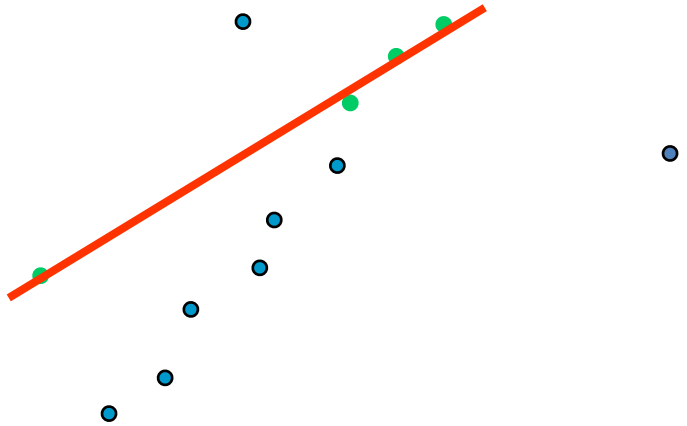
RANSAC



Input: set of points

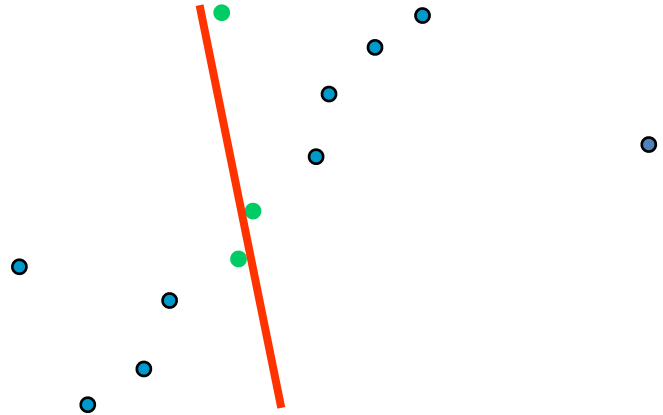
Output: estimate the line equation and determine the inliers/outliers

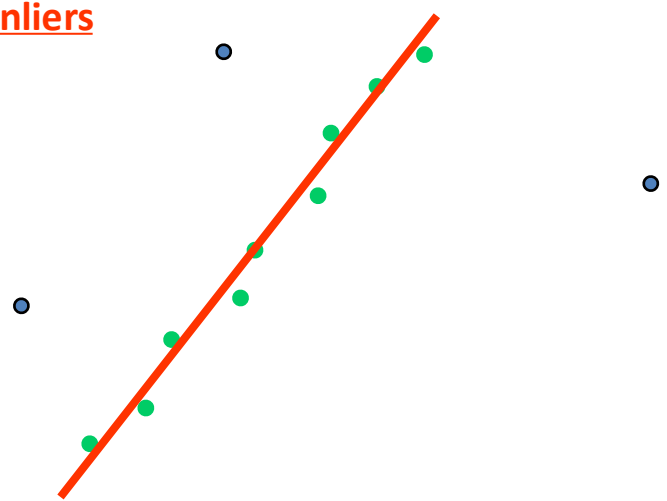




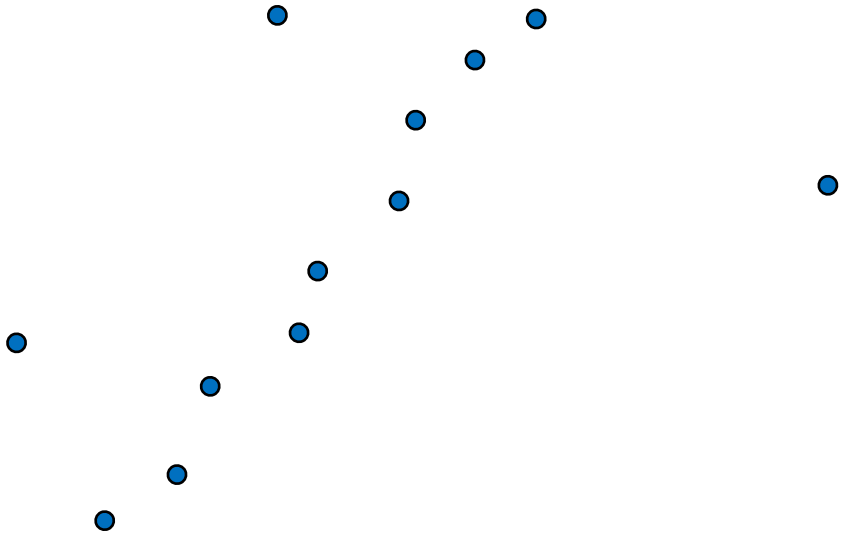
Which one do you like the most?

→ So let's **maximize the nb of inliers**



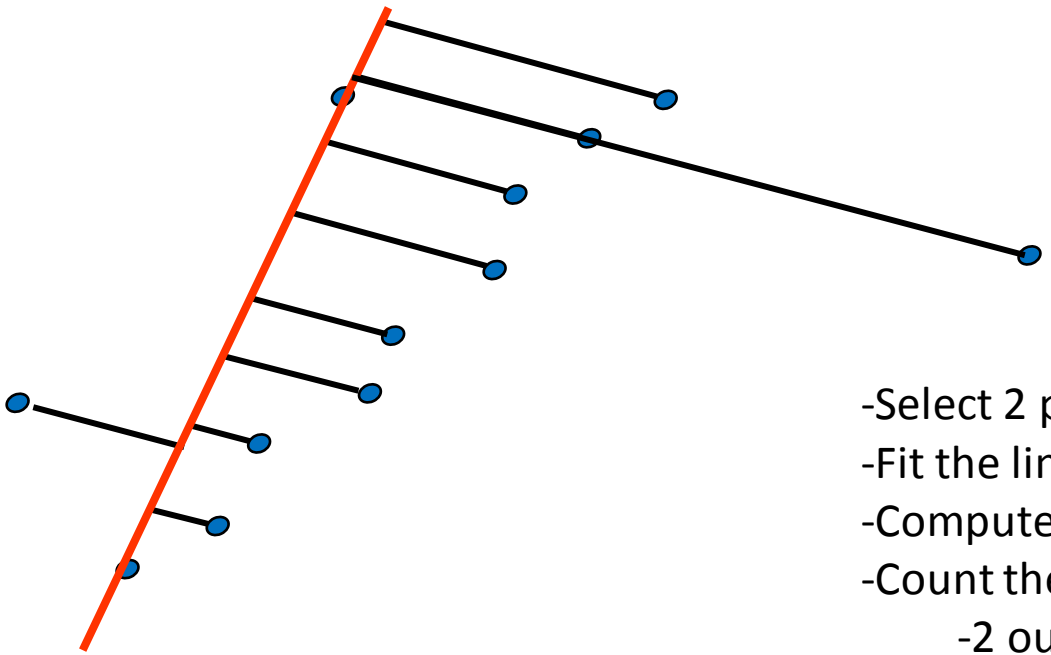


RANSAC



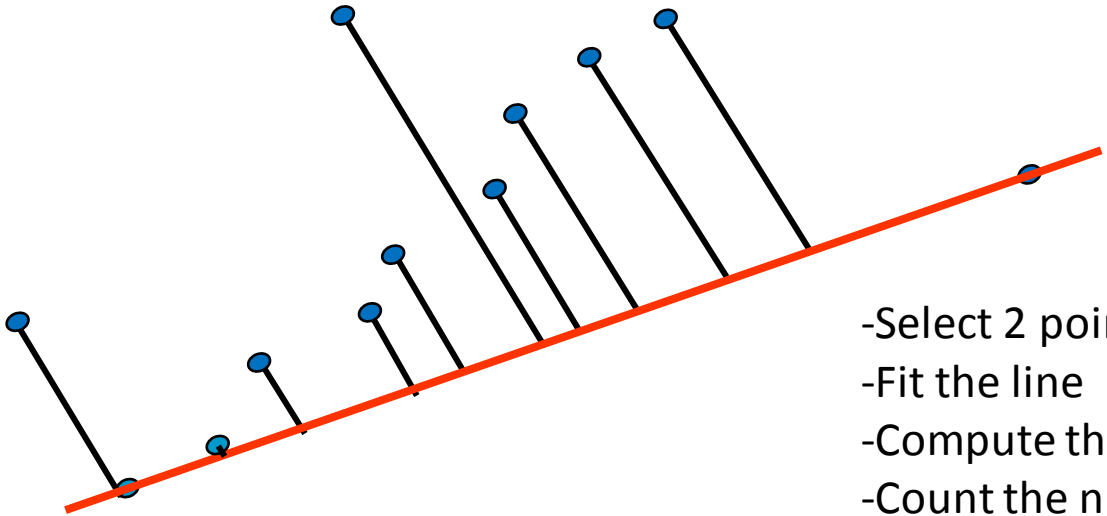
- RANdom SAmple Consensus, by Pischler and Bolles, 1981
- Very easy method and good results
- Aims to **maximize the nb of inliers** (**“consensus set maximization”**) by random sampling

RANSAC



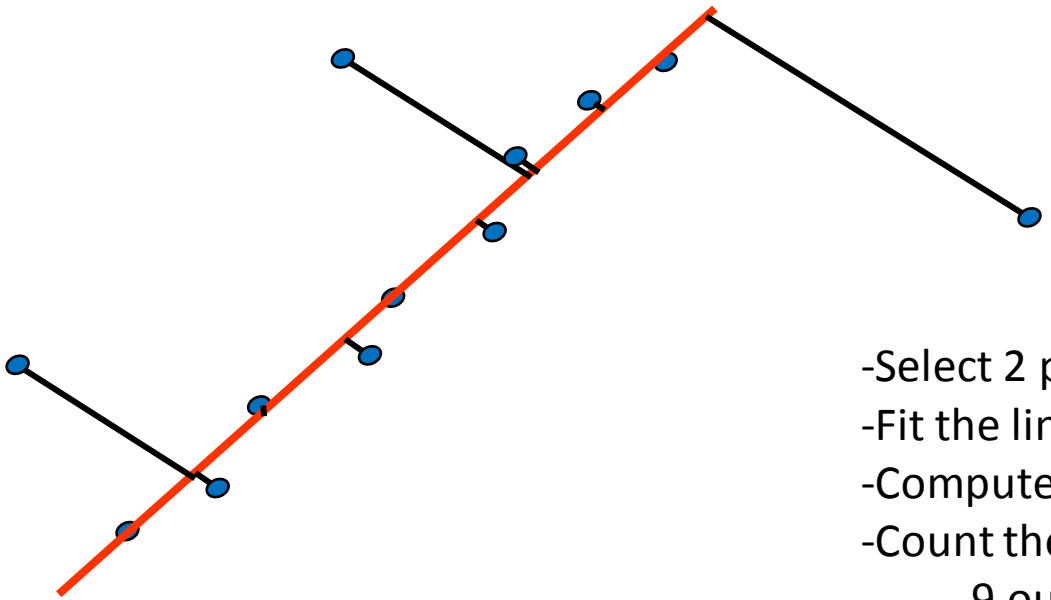
-Select 2 points
-Fit the line
-Compute the distances
-Count the nb of inliers
    -2 out of 13

RANSAC



-Select 2 points
-Fit the line
-Compute the distances
-Count the nb of inliers
-3 out of 13

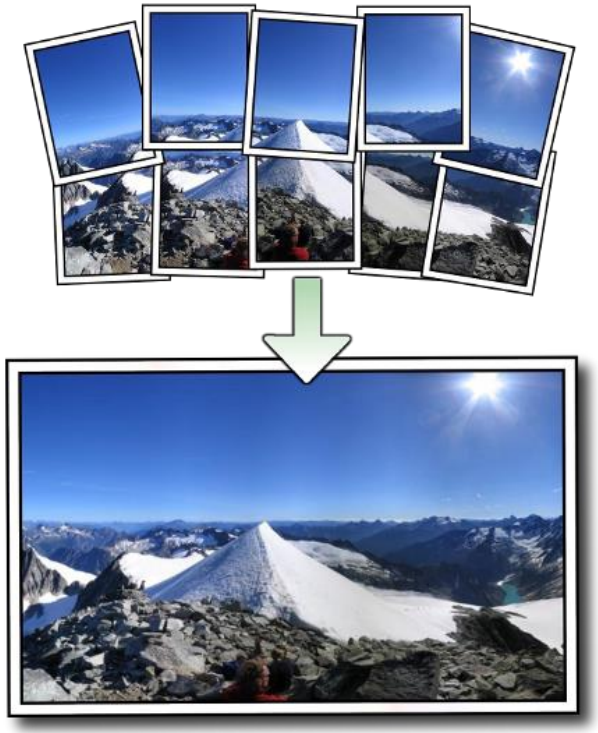
RANSAC



-Select 2 points
-Fit the line
-Compute the distances
-Count the nb of inliers
    -9 out of 13

Panoramic stitching

"The first fully automatic 2D image stitcher"

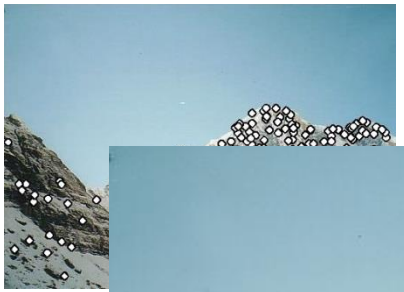
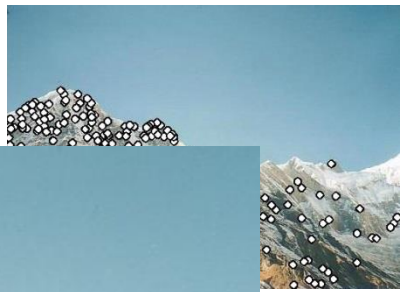


https://www.cs.bath.ac.uk/brown/autostitch/autostitch.html

"Automatic Panoramic Image Stitching using Invariant Features", by M. Brown and D. Lowe, IJCV 2007

"Recognising Panoramas", by M. Brown and D. G. Lowe, ICCV 2003

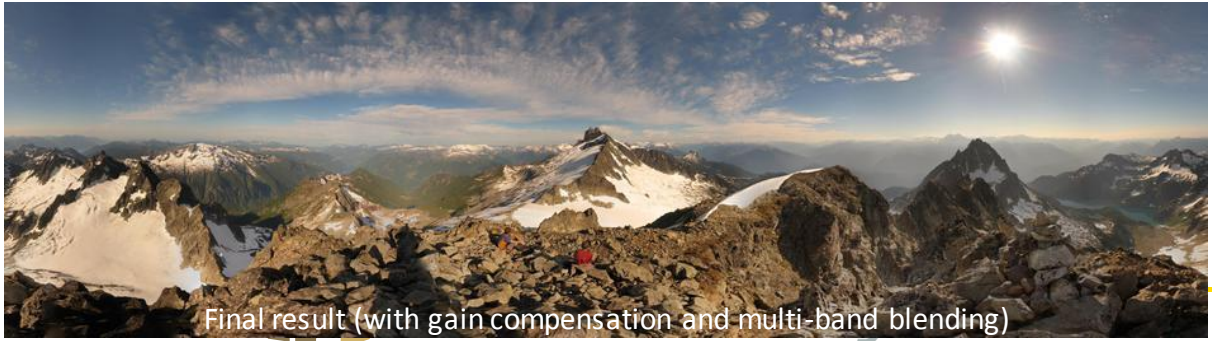


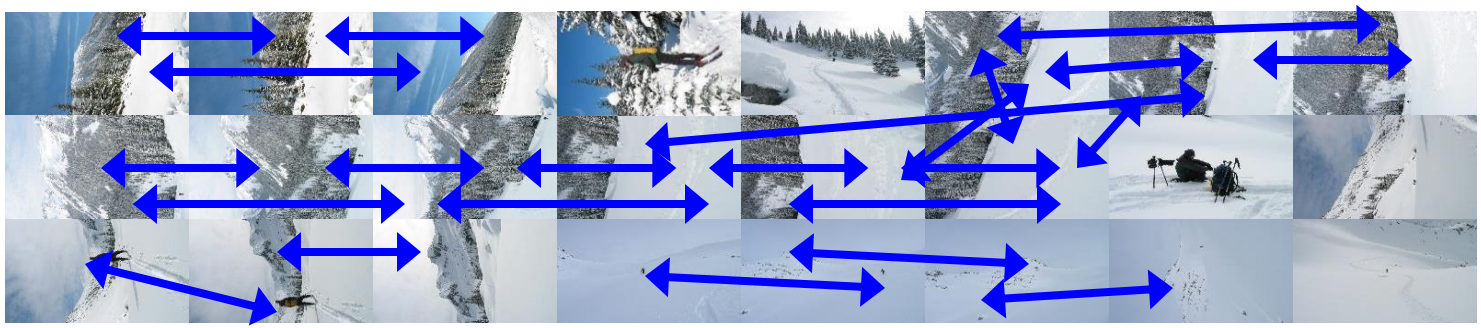
Images aligned according to a homography


25 of 57 images aligned


All 57 images aligned

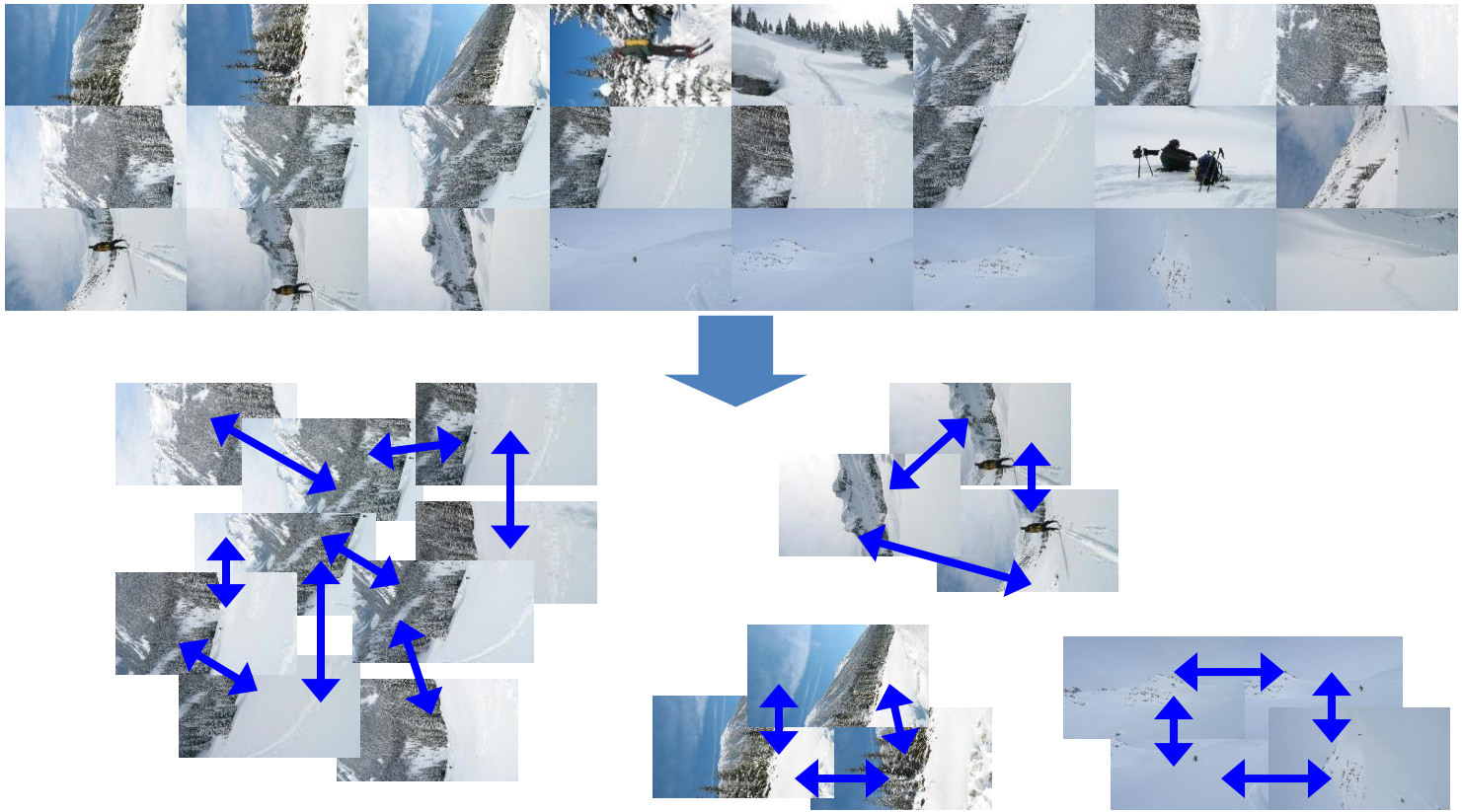

Final result (with gain compensation and multi-band blending)

Finding the panoramas



Not just “building” panoramas, but also “finding” them

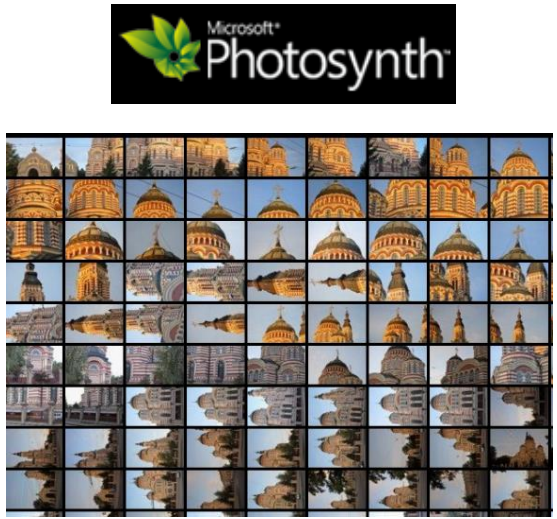
Finding the panoramas

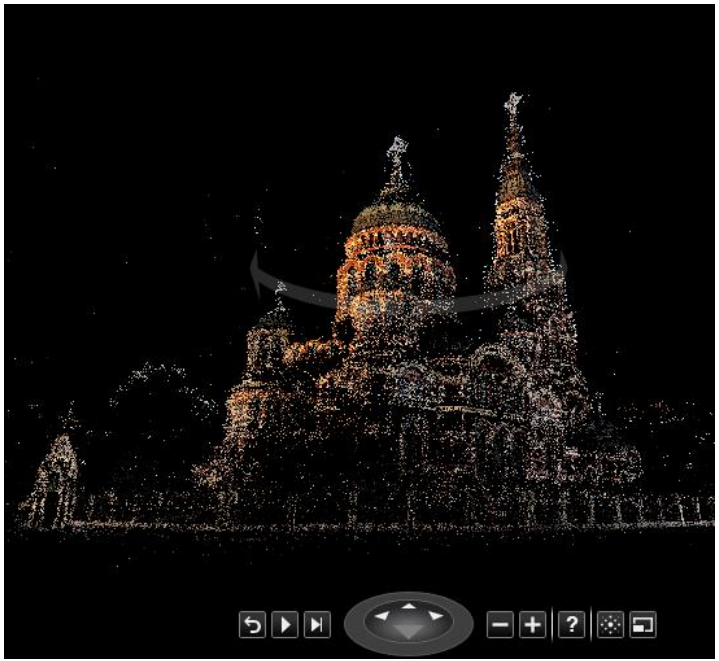


Finding the panoramas

Results

Applications





Photosynth takes a bunch of photos and automatically stitches them all together into one big interactive 3D viewing experience

Check out http://photosynth.net

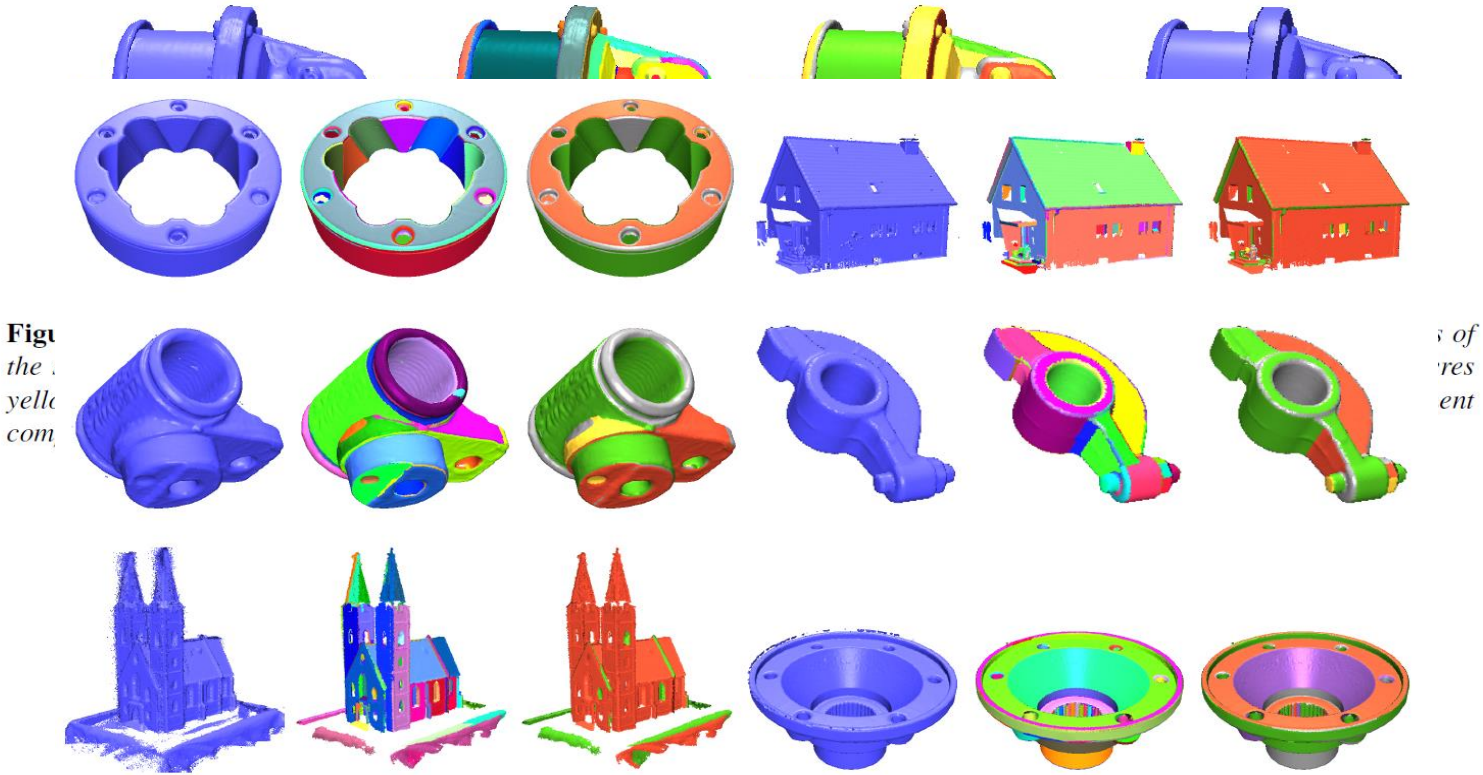
Applications

Microsoft Photosynth

Applications

Snavely et al., “Finding paths through the world’s photos”, SIGGRAPH’08

Applications



Ruwen et al., “Efficient RANSAC for Point-Cloud Shape Detection”, Computer Graphics Forum, 2007

Applications

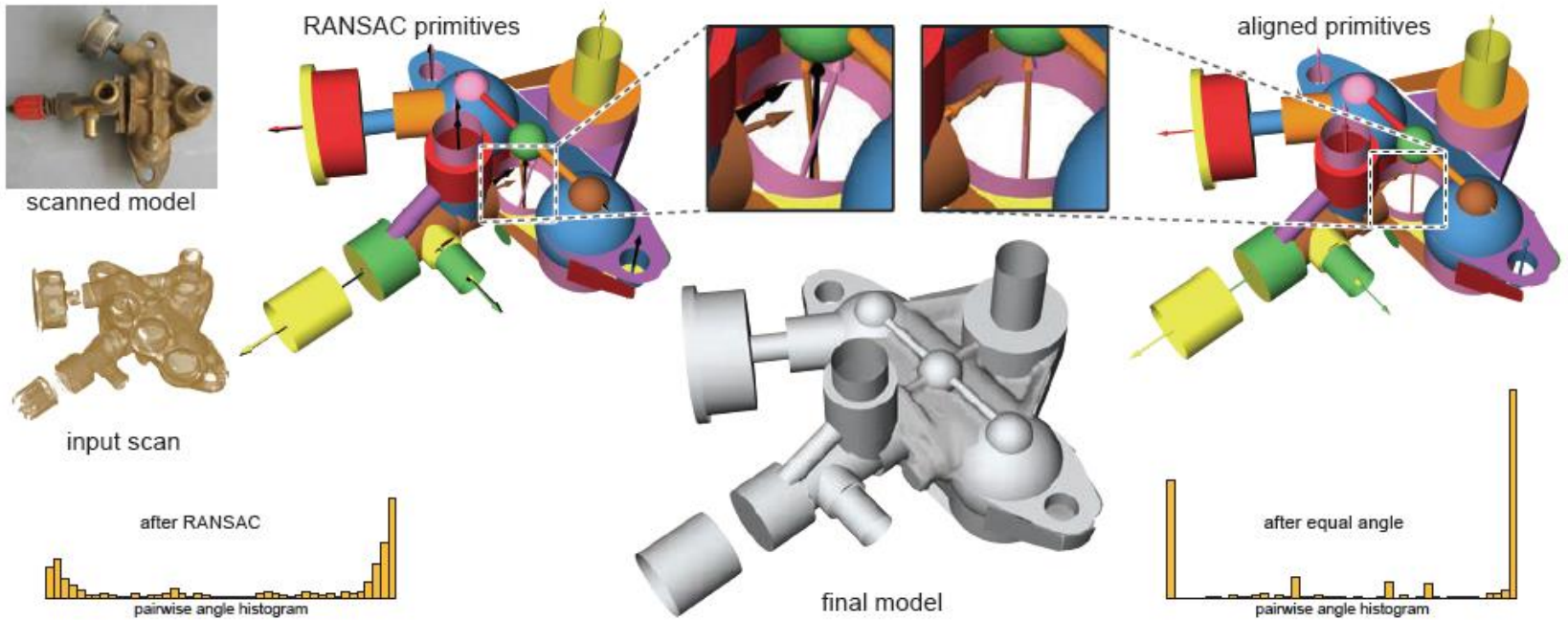
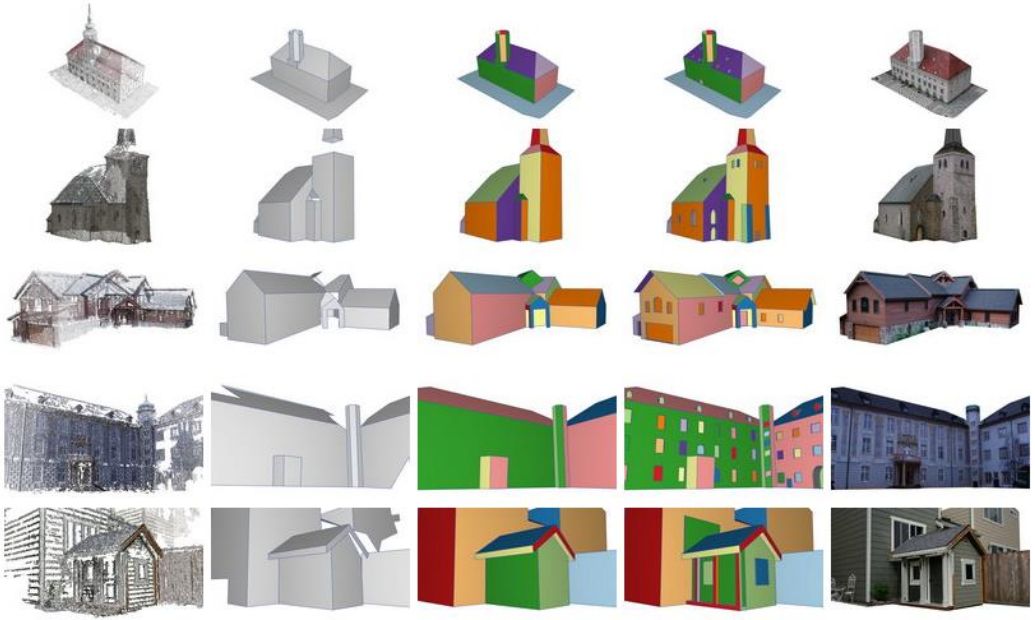


Figure 13: *Cylinder, plane, and sphere primitives are aligned using extracted coaxis, coplanar, parallel/orthogonal axes, equal angle as well as equal length constraints. They converge to the final model after two iterations of RANSAC fitting and constraint optimization. We overlap the final result on the initial RANSAC results for comparison; the histograms demonstrate the effect in the primitive pair angle space.*

Li et al., "GlobFit: Consistently Fitting Primitives by Discovering Global Relations", SIGGRAPH'11

Architecture modeling

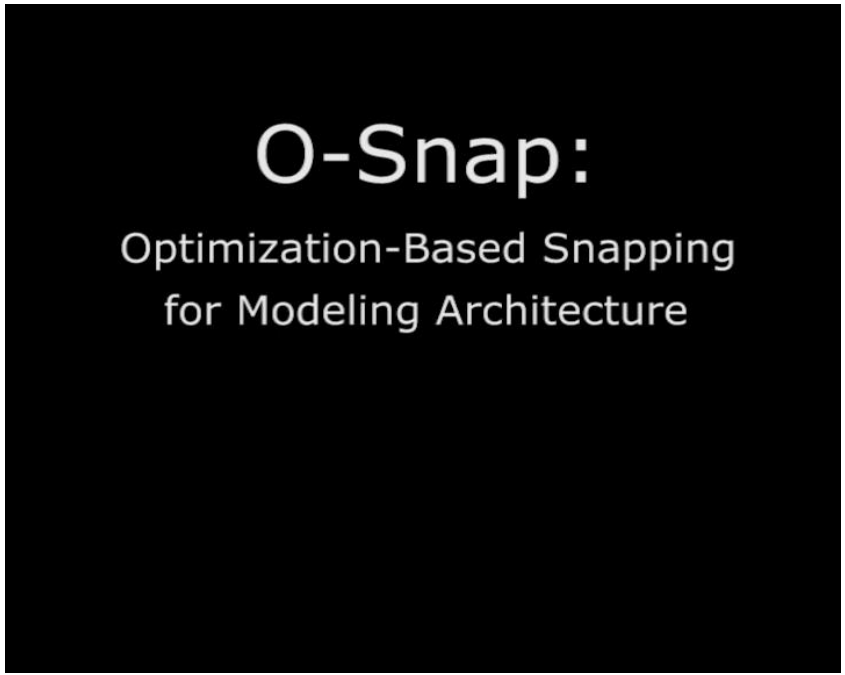


Results from five point cloud data sets (generated from photos) shown from top to bottom: town hall, old church (Photos courtesy of Rainer Brechtken), mountain house (Photos courtesy of Sinha et al. [2008]), castle-P19 (Photos courtesy of Strecha et al. [2008]) and playhouse (Photos courtesy of Sinha et al. [2008]). Left to right: input point cloud, initial automatic reconstruction, refined model, final model with advanced details added, final textured model (with the photos simply back-projected onto the model).

http://osnap.vrvis.at/

"O-Snap: Optimization-Based Snapping for Modeling Architecture", author = "Murat Arikan and Michael Schwarzler and Simon Flory and Michael Wimmer and Stefan Maierhofer", TOG 2013

Architecture modeling



video

http://osnap.vrvis.at/

"O-Snap: Optimization-Based Snapping for Modeling Architecture", author = "Murat Arikan and Michael Schwarzler and Simon Flory and Michael Wimmer and Stefan Maierhofer", TOG 2013

RANSAC - iterations

- Notations:
  - p : probability that at least one sample has no outliers ("success rate")
    - i.e. the selected points are all inliers at least once, usually set to 0.99
  - ε : outlier ratio, so inlier ratio $=1-\varepsilon$
  - s : sample size (minimum data)
    - e.g. ***s***=2 for line fitting, ***s***=3 for circle
- Derivations
  - Probability that the ***s*** selected points at an iteration are all inliers: $(1-\varepsilon)^s$
    - Assuming independent selection
  - Probability that at least one of the ***s*** points is an outlier: $1-(1-\varepsilon)^s$
    - i.e. the generated model is corrupted by at least one outlier
  - Repeating over ***N*** iterations: $(1-(1-\varepsilon)^s)^N$
    - i.e. prob. that a pure inlier set is not selected over ***N*** and thus $=1-p$

$$(1-(1-\varepsilon)^s)^N = 1-p \Longrightarrow N = \frac{\log(1-p)}{\log(1-(1-\varepsilon)^s)}$$

RANSAC - parameters

- How many iterations?

probability that at least one sample has no outliers (usually set to 0.99)

$$N = \frac{\log(1-p)}{\log(1-(1-\varepsilon)^s)}$$

Outlier ratio

Sample size (minimum data)
e.g. s=2 for line fitting, s=3 for circle

- Choice of the inlier threshold
  - Heuristic
  - Trial-and-error (can be tricky)
  - Adaptive threshold

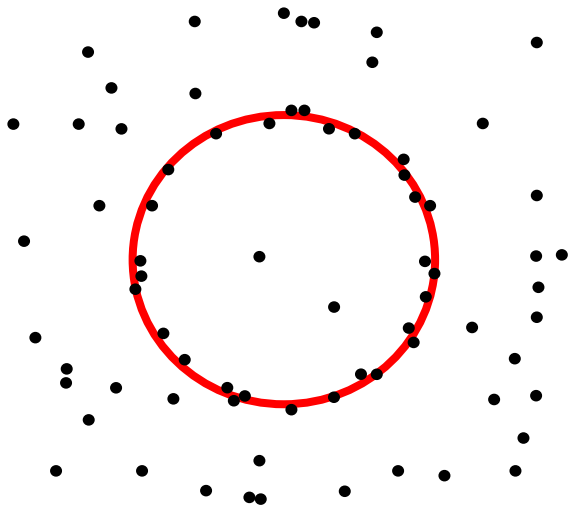
RANSAC

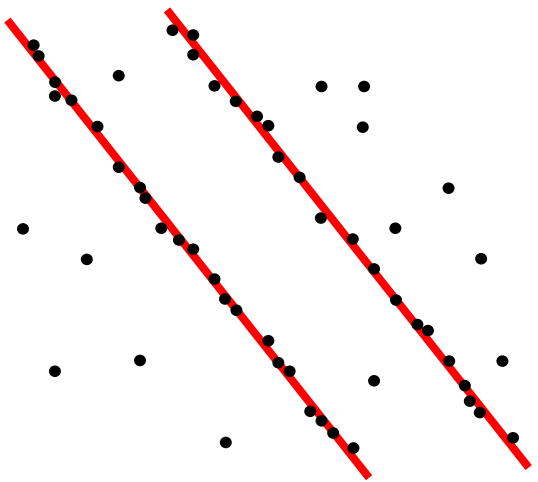
Sample size	proportion of outliers ϵ						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

*The number of RANSAC iterations required to ensure, with a probability p = 0.99, that at least one sample has no outliers for a given number s of minimal points and a proportion of outliers **ε**.*
From [Multiple View Geometry].

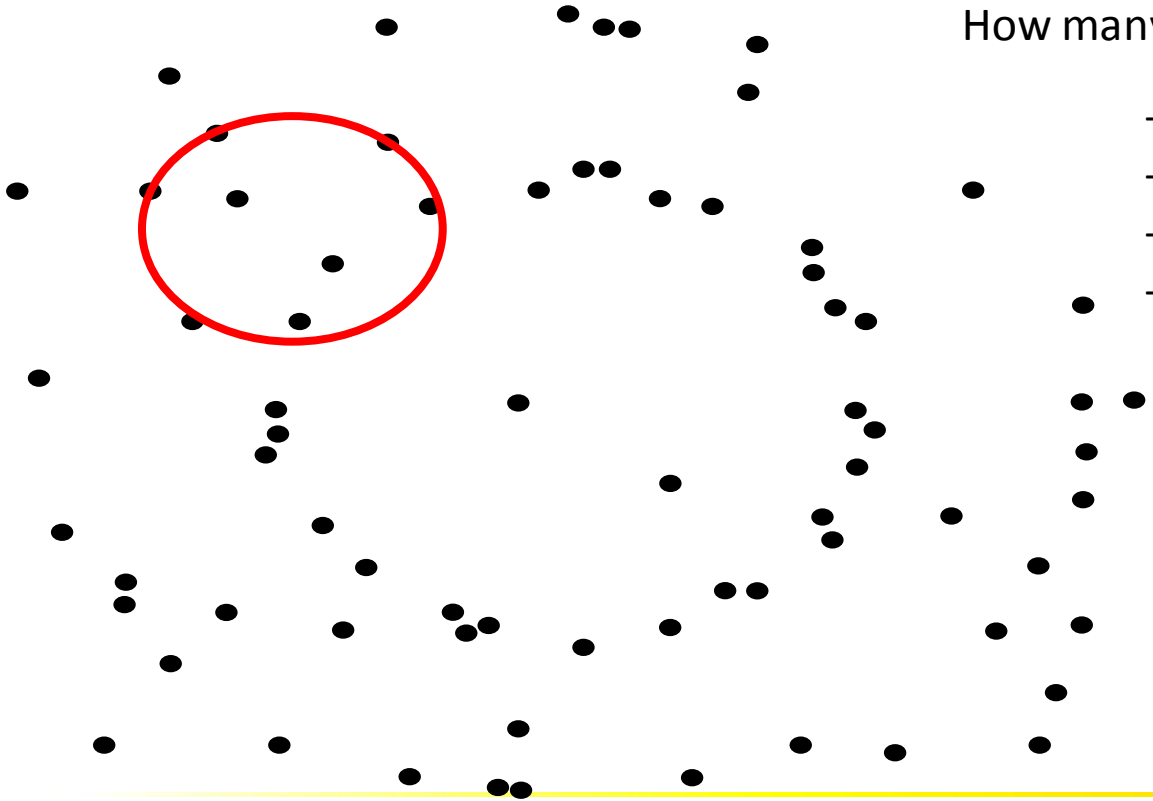
RANSAC

- Think about
  - The pros and cons of RANSAC
  - How to apply it for ellipse detection, plane detection, parallel line detection, etc?





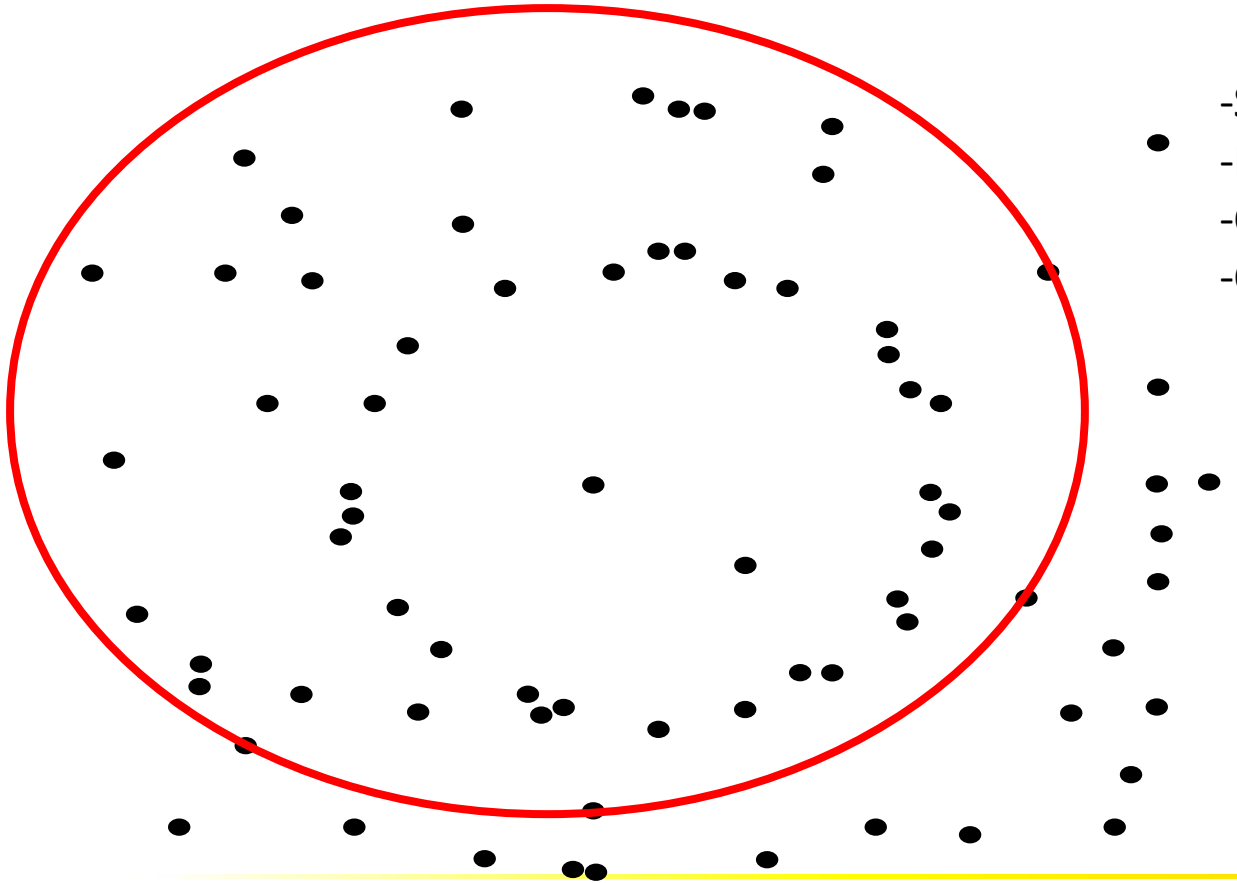
RANSAC



How many points for a circle?

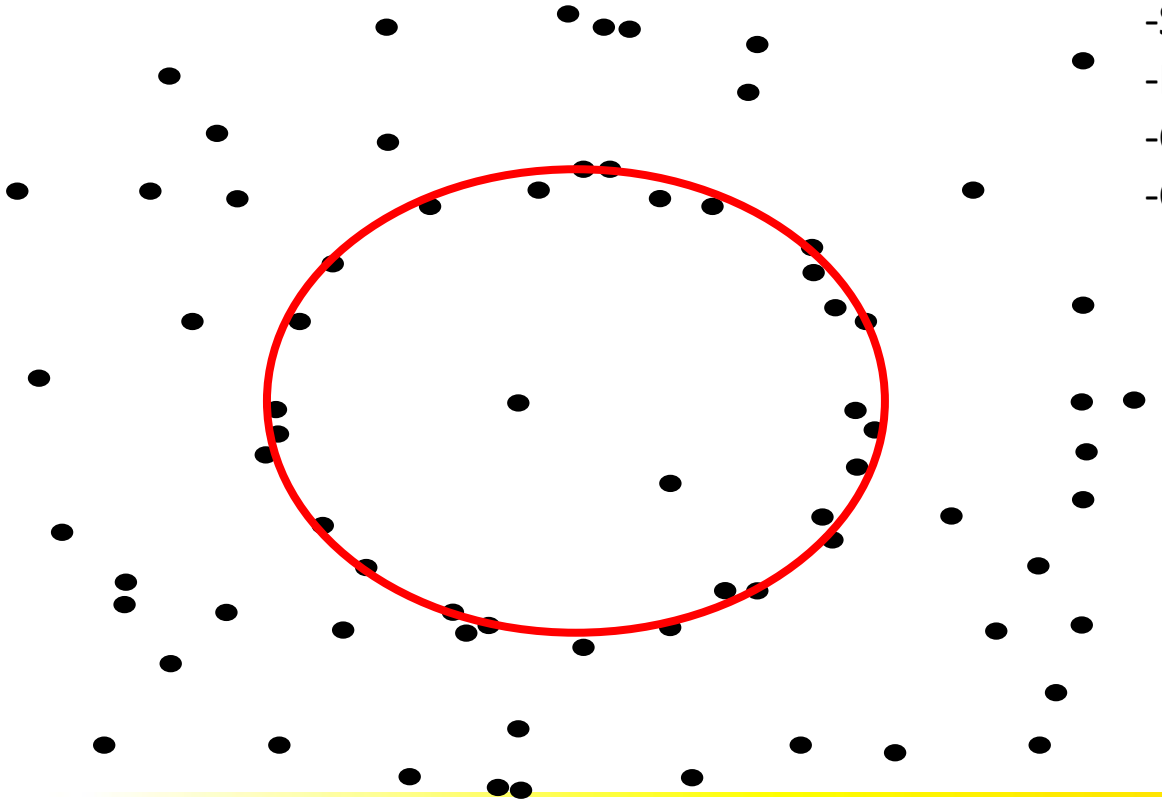
-Select 3 points
-Fit the circle
-Compute the distances
-Count the nb of inliers
-5

RANSAC



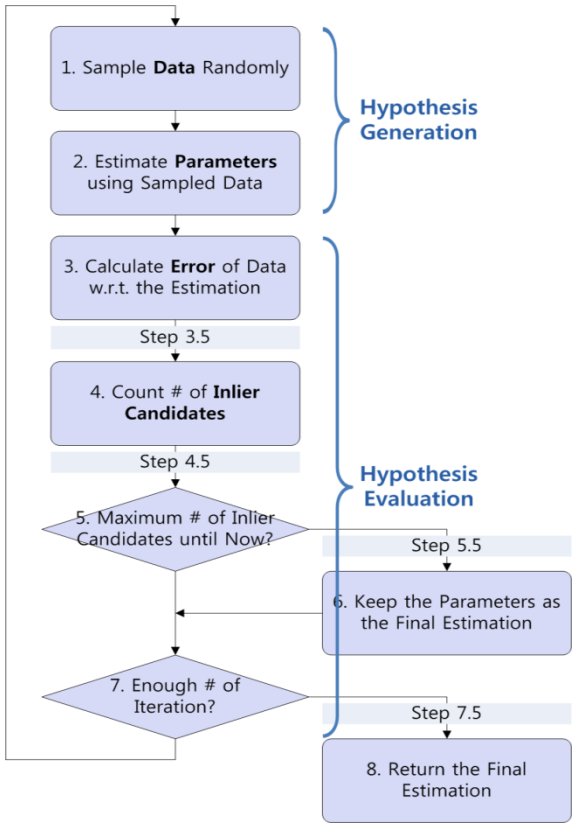
-Select 3 points
-Fit the circle
-Compute the distances
-Count the nb of inliers
-4

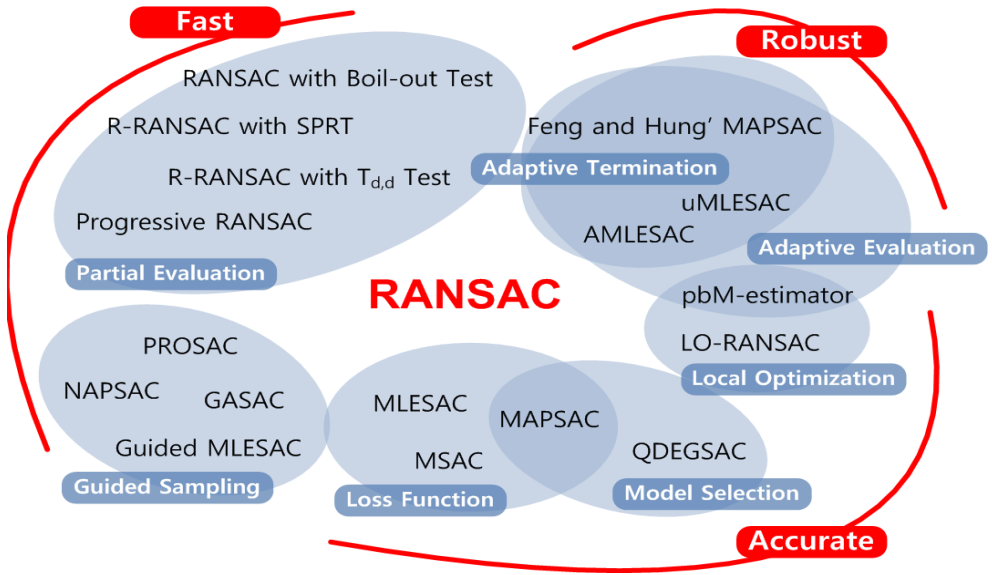
RANSAC



-Select 3 points
-Fit the circle
-Compute the distances
-Count the nb of inliers
-23

RANSAC family



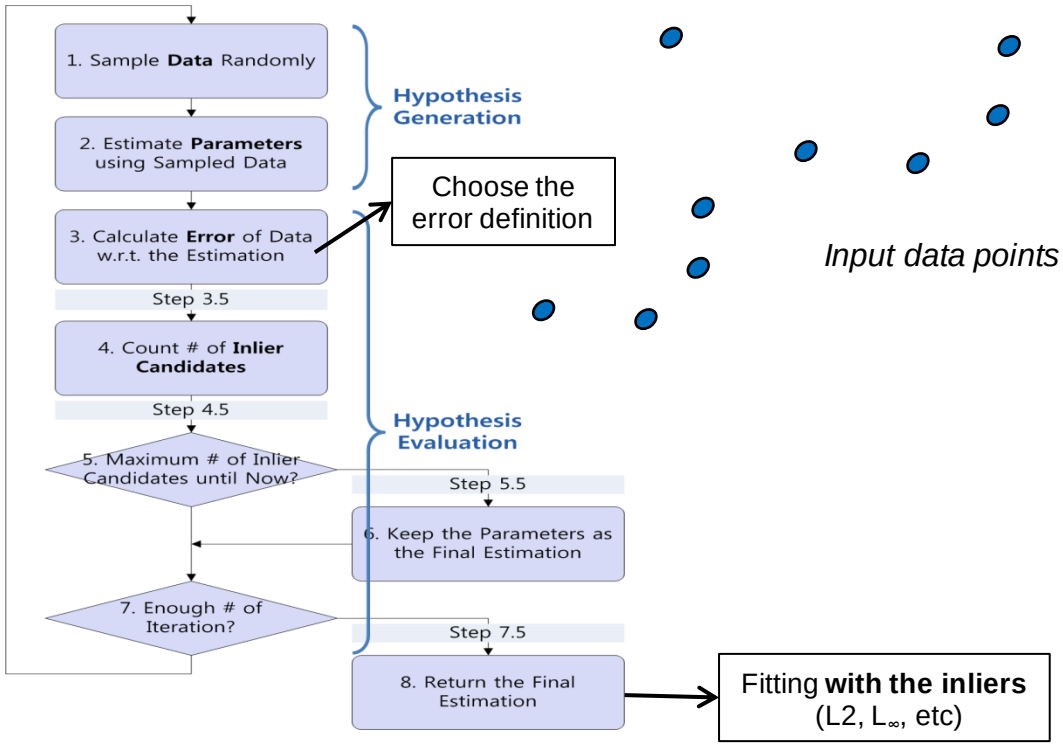


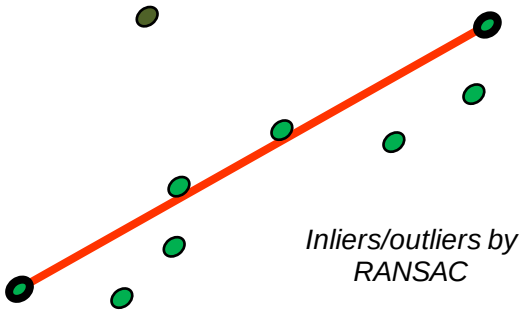
From "Performance Evaluation of RANSAC Family", by Choi et al., BMVC, 2009

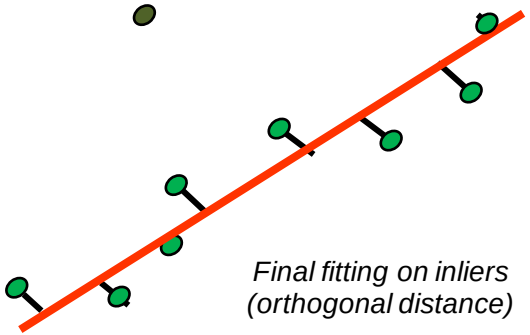
RANSAC family - references

- D. Capel. “An effective bail-out test for RANSAC consensus scoring”, BMVC, 2005
- S. Choi and J.-H. Kim. “Robust regression to varying data distribution and its application to landmark-based localization”, IEEE SMC, 2008 [uMLESAC]
- O. Chum and J Matas. “Randomized RANSAC with $T_{d,d}$ test”, BMVC, 2002
- O. Chum and J. Matas. “Matching with PROSAC - Progressive Sample Consensus”, CVPR, 2005
- O. Chum and J. Matas. “Optimal randomized RANSAC”, TPAMI, 2008.
- O. Chum, J. Matas, and S. Obdrzalek. “Enhancing RANSAC by generalized model optimization”, ACCV, 2004.
- C.L. Feng and Y.S. Hung. “A robust method for estimating the fundamental matrix”. In Proc. of the 7th Digital Image Computing: Techniques and Applications, 2003
- J.-M. Frahm and M. Pollefeys. “RANSAC for (quasi-)degenerate data (QDEGSAC)”, CVPR, 2006.
- A. Konouchine, V. Gaganov, and V. Veznevets. “AMLESAC: A new maximum likelihood robust estimator”, Int. Conf. on Computer Graphics and Vision (GrapiCon), 2005.
- J. Matas and O. Chum. “Randomized RANSAC with sequential probability ratio test”, ICCV, 2005
- D.R. Myatt, P.H.S Torr, S.J. Nasuto, J.M. Bishop, and R. Craddock. “NAPSAC: High noise, high dimensional robust estimation - it’s in the bag”, BMVC, 2002
- D. Nister. “Preemptive RANSAC for live structure and motion estimation”, ICCV, 2003
- R. Raguram, J.-M. Frahm, and M. Pollefeys. “A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus”, ECCV, 2008
- V. Rodehorst and O. Hellwich. “Genetic Algorithm SAmple Consensus (GASAC)- a parallel strategy for robust parameter estimation”, CVPR Workshop, 2006
- R. Subbarao and P. Meer. “Beyond RANSAC: User independent robust regression”, CVPR Worksop, 2006
- B. J. Tordoff and D. W. Murray. “Guided-MLESAC: Faster image transform estimation by using matching priors”, TPAMI, 2005
- P.H.S. Torr and A. Zisserman. “MLESAC: A new robust estimator with application to estimating image geometry”, CVIU, 2000
- And many others

RANSAC



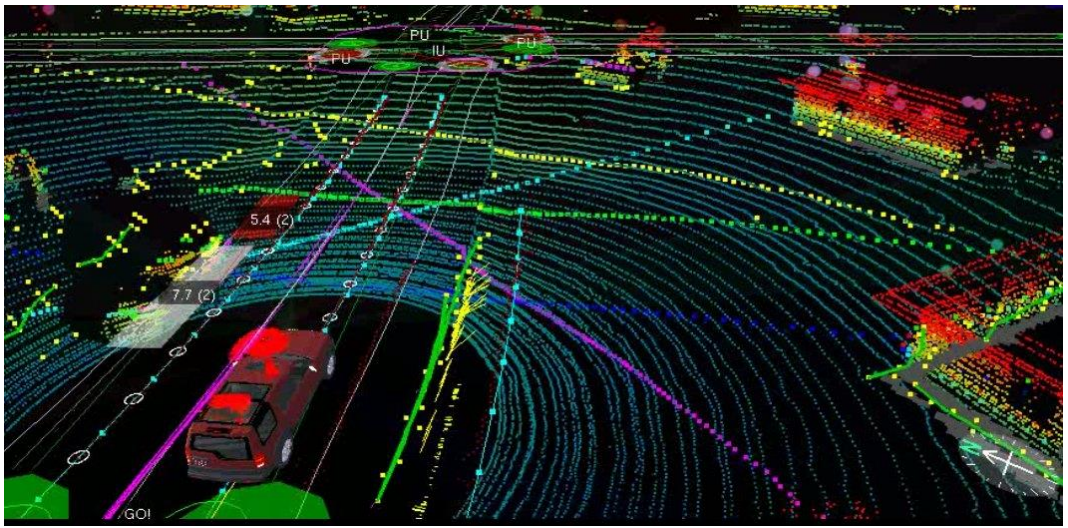




From "Performance Evaluation of RANSAC Family", by Choi et al., BMVC, 2009

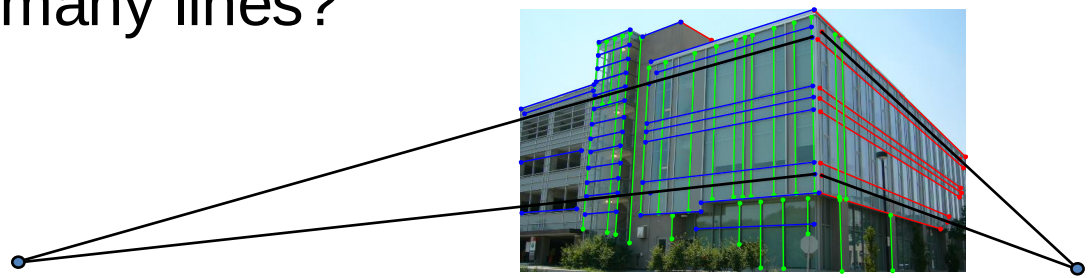
3D plane extraction

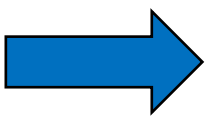
- Ground plane extraction on 3D point cloud
  - How many points?

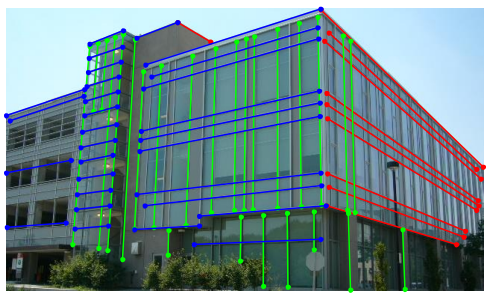


Line clustering and VPs

- Orthogonal vanishing point in intrinsically calibrated images. How many lines?





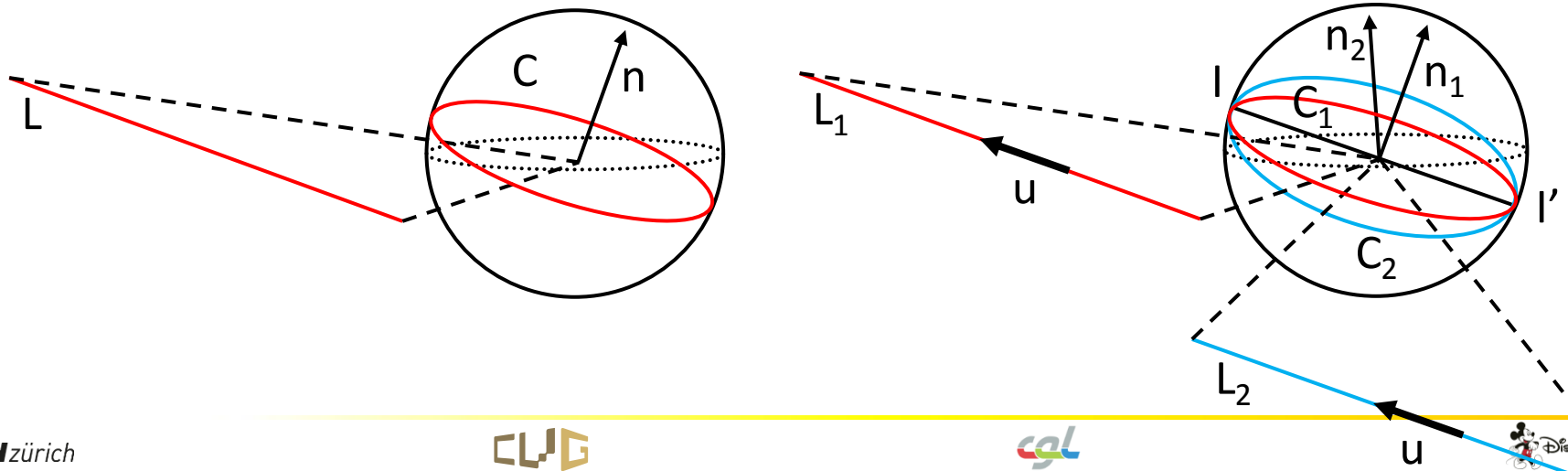


Input: a set of lines (manually or automatically) extracted in calibrated images

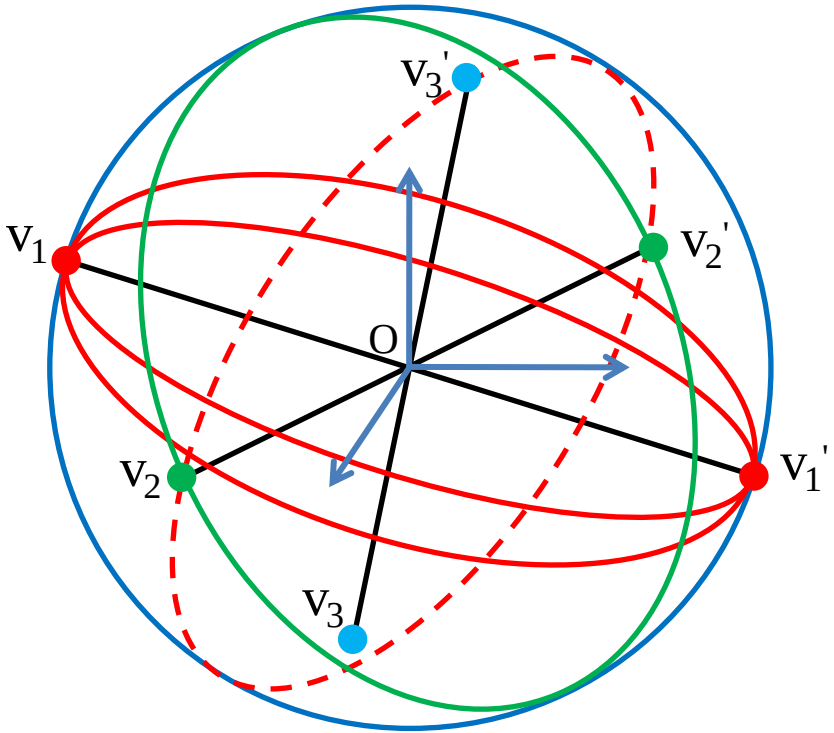
Output: line clustering with respect to their (unknown-but-sought) orthogonal VPs

Equivalent sphere

- Interesting geometric properties
  - A world line is a great circle in the sphere
  - World parallel lines intersect in two antipodal points in the sphere
    - Correspond to VPs
  - VP direction is the world line direction
  - Orthogonal sets of world parallel lines lead to orthogonal VP



“3-line RANSAC”



Select 2 lines

↓

Compute their intersection
→ VP_1

↓

Create the orthogonal VP definition set

↓

Select 1 line

↓

Intersect the definition set and the line
→ VP2

↓

Compute VP3 by cross-product

↓

Compute R given the VPs [Horn]

Line clustering and VPs

Representative results obtained for the York Urban database

VPs on Google Street View

Google Street View images kindly provided by Google

Omni-video stabilization

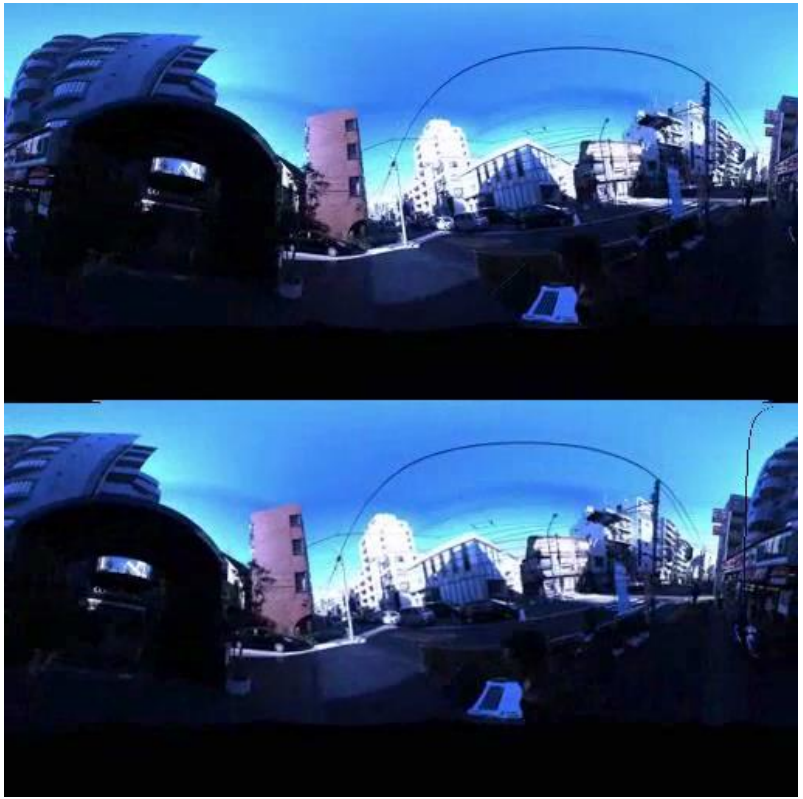
Ladybug3 from pointgrey:
- 6 x 2Mpixels camera
- at 15fps

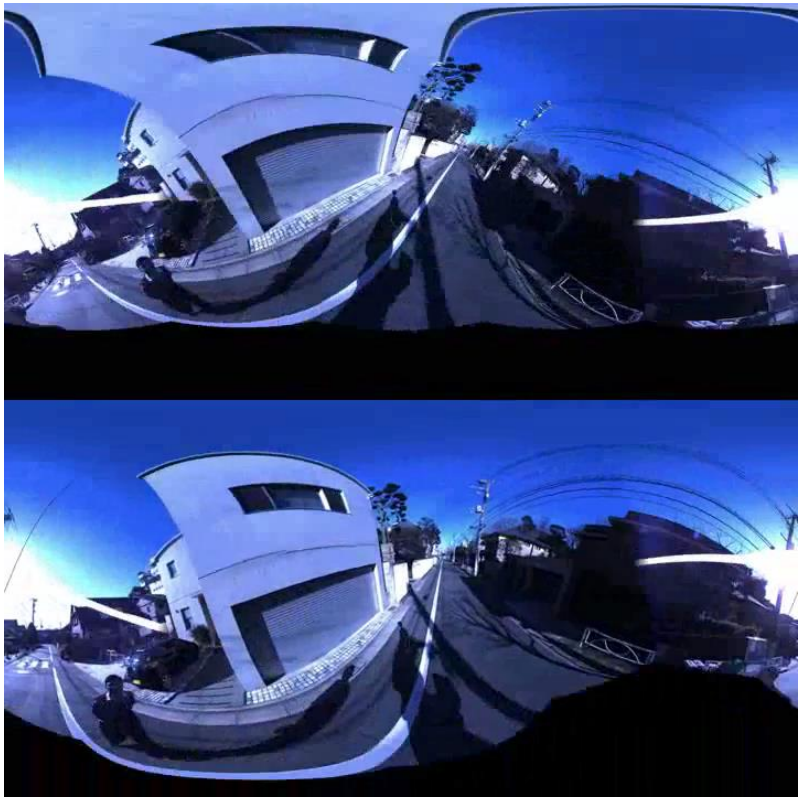
Omni-video stabilization

input

output

Omni-video stabilization





IRLS – a step back

- Center of gravity

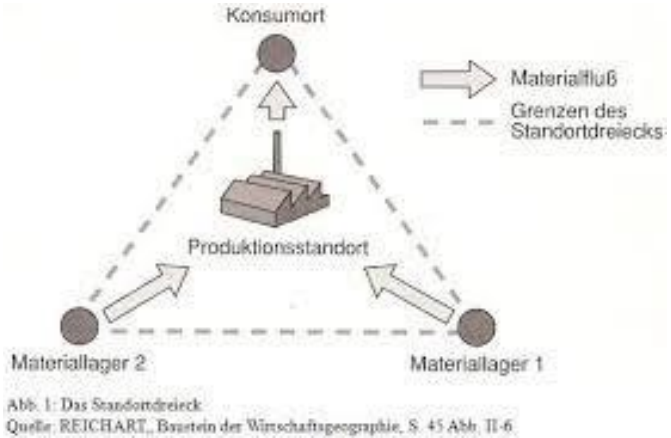
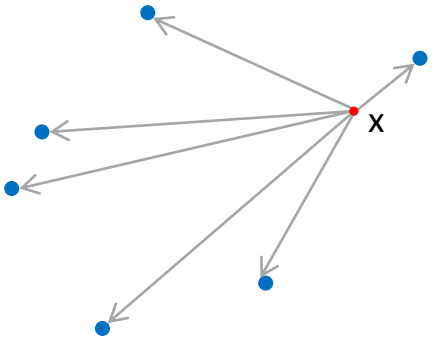


Abb. 1: Das Standortdreieck
Quelle: REICHART, Bausteine der Wirtschaftsgeographie, S. 45 Abb. II-6

Alfred Weber

IRLS – a step back



Example in 2D

$$x^* = \arg\min_x \sum_{i=1}^{n} d(x, y_i)$$

Weiszfeld's algorithm:
- minimizing the sum of distances to a set of points
- repeatedly finds a point improving the sum of distances until no more improvements can be made
- an iterative algorithm to find L1 minimum point of a set of points
  - L1 here is the “non-squared” version

$$\min_{\mathbf{x}} \sum_{i=1}^{n} \|\mathbf{x} - \mathbf{y}_i\|_2$$

A globalized world ;-)

E. Weiszfeld, « Sur le point pour lequel la somme des distances de n points donnés est minimum », Tôhoku Mathematics Journal 43 (1937)

IRLS – a step back

Don't mix up with **squared** geometrical distance (squared L2):
- the centroid or center of mass (and center of gravity in a uniform gravitational field)

$$\min_{\mathbf{x}} \sum_{i=1}^{n} \|\mathbf{x}-\mathbf{y}_i\|_2^2 \neq \min_{\mathbf{x}} \sum_{i=1}^{n} \|\mathbf{x}-\mathbf{y}_i\|_2$$

$$\min_{\mathbf{x}} \sum_{i=1}^{n} \|\mathbf{x}-\mathbf{y}_i\|_2^2 = \min_{\mathbf{x}} \sum_{i=1}^{n}\sum_{j=1}^{m} (x_j - y_{ij})^2$$

$$= \min_{\mathbf{x}} \sum_{i=1}^{n}\sum_{j=1}^{m} (x_j^2 - 2x_j y_{ij} + y_{ij}^2)$$

$$\frac{\partial C}{\partial x_j} = \sum_{i=1}^{n}\sum_{j=1}^{m} (2x_j - 2y_{ij}) = 0 \quad \Rightarrow \quad \sum_{i=1}^{n}\sum_{j=1}^{m} (x_j - y_{ij}) = 0 \quad \Rightarrow \quad x_j = \frac{\sum_{i=1}^{n} y_{ij}}{n}$$

Simply the average

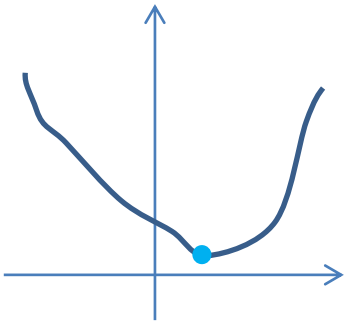
IRLS

- Does it converge?

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{i=1}^{n} d(\mathbf{x}, \mathbf{y}_i)$$

$$C(\mathbf{x}) = \sum_{i=1}^{n} d(\mathbf{x}, \mathbf{y}_i) = \sum_{i=1}^{n} \|\mathbf{x} - \mathbf{y}_i\|_p$$

Convex. Why?
So single minimum



Gradient descent algorithm

$$\nabla C(\mathbf{x}) = \sum_{i=1}^{n} \frac{(\mathbf{x} - \mathbf{y}_i)}{\|\mathbf{x} - \mathbf{y}_i\|_p}$$

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \gamma^t \sum_{i=1}^{n} \frac{(\mathbf{x}^t - \mathbf{y}_i)}{\|\mathbf{x}^t - \mathbf{y}_i\|_p}$$

Valid for p=1,2,…

step-size

direction

IRLS

- Iteratively reweighted least squares

$$\arg\min_{\Theta} \sum_{i=1}^{n} w_i(\Theta)\,(y_i - f(\Theta, x_i))^2$$

where $w_i(\Theta^{(t)}) = \left(y_i - f(\Theta^{(t)}, x_i)\right)^2$

$$\boxed{\Theta^{(t+1)} = \arg\min_{\Theta} \sum_{i=1}^{n} w_i(\Theta^{(t)})\,(y_i - f(\Theta, x_i))^2}$$

General IRLS

1. Identify a weighted optimization problem that can be solved optimally (e.g. in closed form)

$$C(\mathbf{x}, \mathbf{w}) = \sum_{i=1}^{n} w_i f_i(\mathbf{x})$$

Written without the squares

2. Solve iteratively: at each step, define weights (how)

$$w_i^t = w_i(\mathbf{x}^t)$$

Define weights

and define

$$\mathbf{x}^{t+1} = \arg\min_{\mathbf{x}} C(\mathbf{x}, \mathbf{w}^t)$$

$$= \arg\min_{\mathbf{x}} \sum_{i=1}^{n} w_i^t f_i(\mathbf{x})$$

Minimize weighted cost

3. Hope that it converges to what you want

How to choose the weights

- Assume we can minimize the cost

$$C(\mathbf{x}, \mathbf{w}) = \sum_{i=1}^{n} w_i f_i(\mathbf{x})$$

No square

- We wish to minimize

$$C_\rho(\mathbf{x}, \mathbf{w}) = \sum_{i=1}^{n} \rho \circ f_i(\mathbf{x})$$

Robust cost function

- We want

$$\nabla_{\mathbf{x}} C(\mathbf{x}, \mathbf{w}) = 0 \text{ if and only if } \nabla_{\mathbf{x}} C_\rho(\mathbf{x}, \mathbf{w}) = 0$$

- So

$$\nabla_{\mathbf{x}} w_i f_i(\mathbf{x}) = \nabla_{\mathbf{x}} (\rho \circ f_i(\mathbf{x}))$$

$$w_i \nabla_{\mathbf{x}} f_i(\mathbf{x}) = \rho'(f_i(\mathbf{x})).\nabla_{\mathbf{x}} f_i$$

$$\boxed{w_i^t = \rho'(f_i(\mathbf{x}^t))}$$

Required weights

Example L_1

- Let

$$f_i(\mathbf{x}) = d(\mathbf{x}, \mathbf{y}_i)^2$$

$$\rho(s) = \sqrt{s}$$

$$C_\rho(\mathbf{x}) = \sum_{i=1}^{n} \rho \circ f_i(\mathbf{x}) = \sum_{i=1}^{n} d(\mathbf{x}, \mathbf{y}_i)$$

Sum of distances

- Then

$$\begin{aligned} w_i &= \rho'(f(\mathbf{x})) \\ &= \frac{1}{2} f(\mathbf{x})^{-1/2} \\ &= \frac{1}{2} d(\mathbf{x}, \mathbf{y}_i)^{-1} \end{aligned}$$

Example L_q

- Let

$$f_i(\mathbf{x}) = d(\mathbf{x}, \mathbf{y}_i)^2$$

$$\rho(s) = s^{q/2}$$

$$C_\rho(x) = \sum_{i=1}^{n} \rho \circ f_i(\mathbf{x}) = \sum_{i=1}^{n} d(\mathbf{x}, \mathbf{y}_i)^q$$

- Then

$$\begin{aligned} w_i &= \rho'(f(\mathbf{x})) \\ &= \frac{q}{2} f(\mathbf{x})^{(q-2)/2} \\ &= \frac{q}{2} d(\mathbf{x}, \mathbf{y}_i)^{q-2} \end{aligned}$$

Moving least squares

- Method for smoothing, interpolating, fitting data
  - “use the neighbors”
  - Very powerful
  - **Local** approximation
  - Introduced by Lancaster and Salkaukas

See the previous lecture

$$\min_{\Theta} \sum_{i=1}^{n} g(\mathbf{x} - \mathbf{x_i}) \| f_i - f(\Theta, \mathbf{x_i}) \|^2$$

MLS <u>at the point ***x***</u>

Note the difference with LS, WLS, etc…

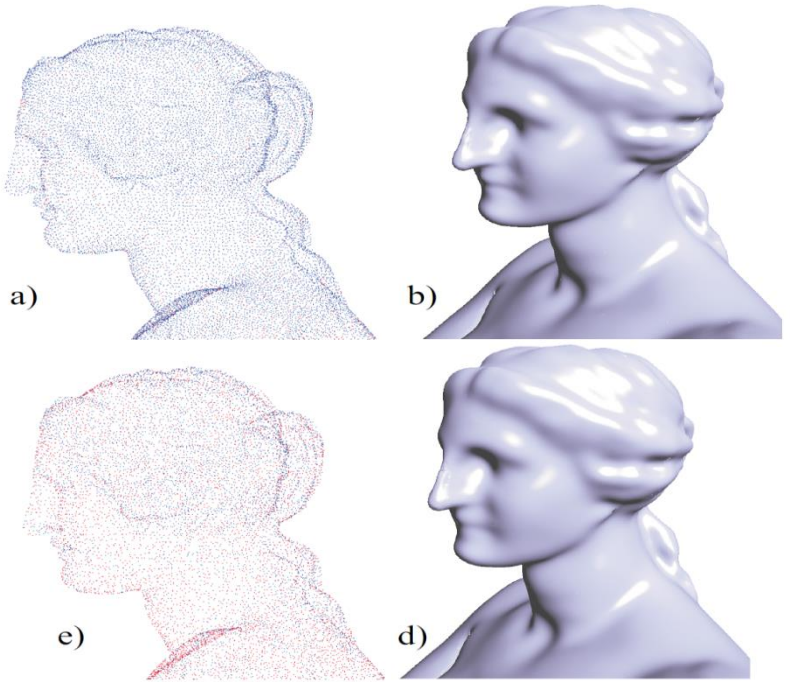
$$\min_{\Theta} \sum_{i=1}^{n} g(\bar{\mathbf{x}} - \mathbf{x_i}) \| f_i - f(\Theta, \mathbf{x_i}) \|^2$$

Lancaster, P., and Salkauskas, K. “Surfaces generated by moving least squares methods”, *Mathematics of Computation 87*, 1981.

Applications of MLS

Figure 1: A point set representing a statue of an angel. The density of points and, thus, the accuracy of the shape representation are changing (intentionally) along the vertical direction.

Surface reconstruction from points



Down-sampling from 37K to 20K points

Alexa et al., “Computing and rendering point set surfaces”. *IEEE TVCG* , 2003.
Alexa et al., “Point set surfaces”, *IEEE Visualization*, 2001

Applications of MLS

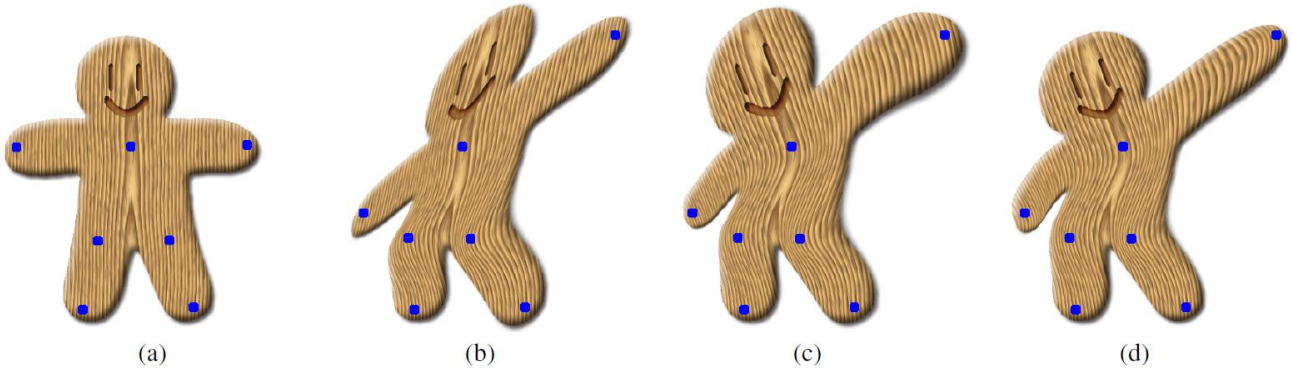


Figure 1: Deformation using Moving Least Squares. Original image with control points shown in blue (a). Moving Least Squares deformations using affine transformations (b), similarity transformations (c) and rigid transformations (d).

$$\sum_i w_i \left| l_v(p_i) - q_i \right|^2 \quad \text{with} \quad w_i = \frac{1}{|p_i - v|^{2\alpha}}$$

input/output control points

Models for the deformation l_v:
- Affine transformation
- Similarity transformation
- Rigid transformation
- etc

Why “MLS”?
- The weights w_i depend on the evaluation point
- One transformation $l_v(x)$ for each pixel/vertex v of the original image (mapping function)

Schaefer et al., “Image deformation using moving least squares”, *SIGGRAPH*, 2006

Applications of MLS

Input data

C. Kuster et al., “Spatio-temporal geometry fusion for multiple hybrid cameras using moving least squares surfaces”, Eurographics, *2014*

Applications of MLS

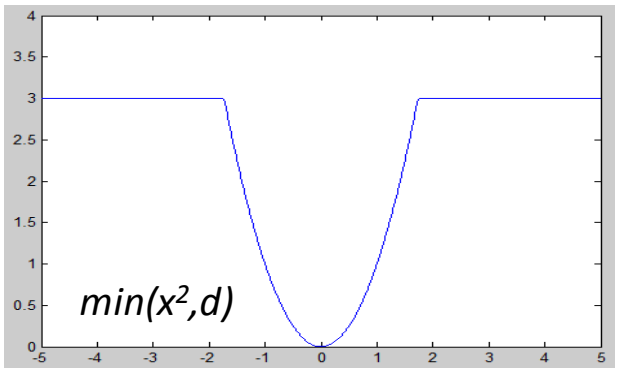
Fast-forward video

C. Kuster et al., “Spatio-temporal geometry fusion for multiple hybrid cameras using moving least squares surfaces”, Eurographics, *2014*

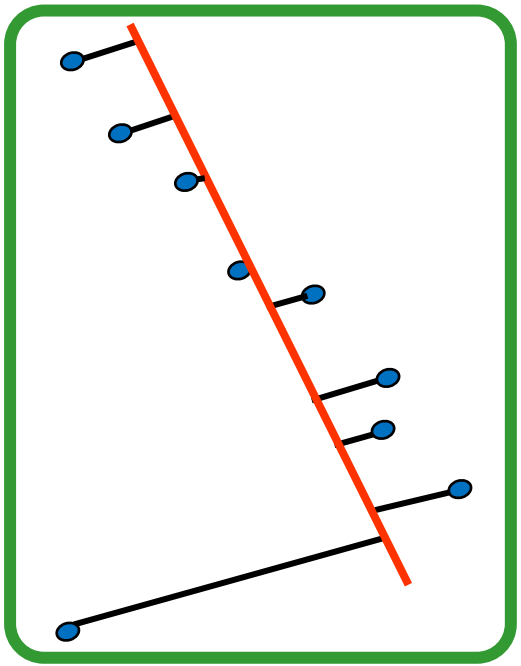
Summary

Norms & cost

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$



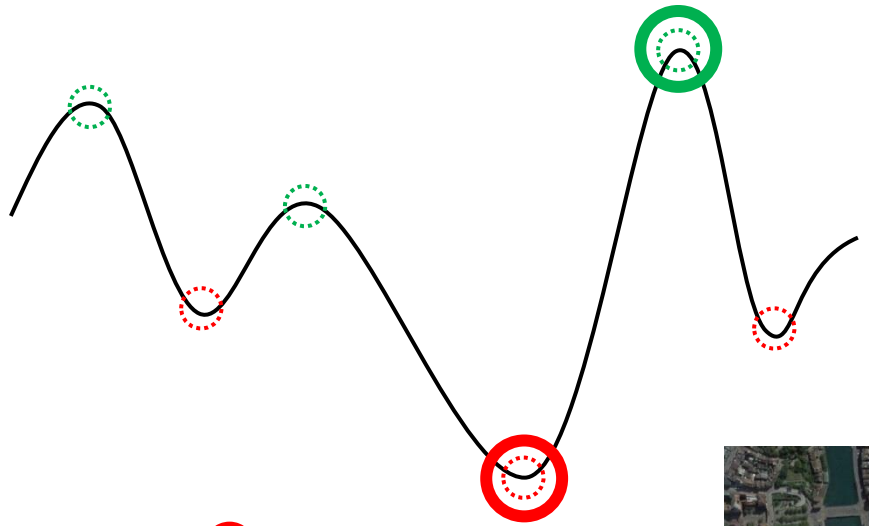
RANSAC



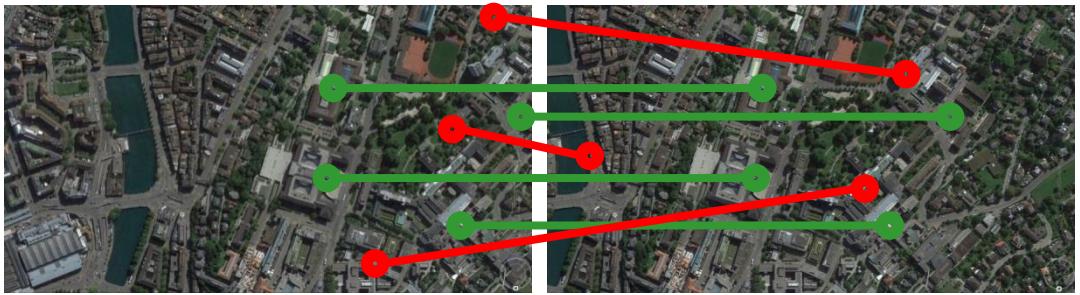
MLS

$$\min_{\Theta} \sum_{i=1}^{n} g(x - x_i) \, \|f_i - f(\Theta, x_i)\|^2$$

Next lecture



Perez et al., “Poisson image editing”, SIGGRAPH’03



Questions?