Photogrammetry & Robotics Lab

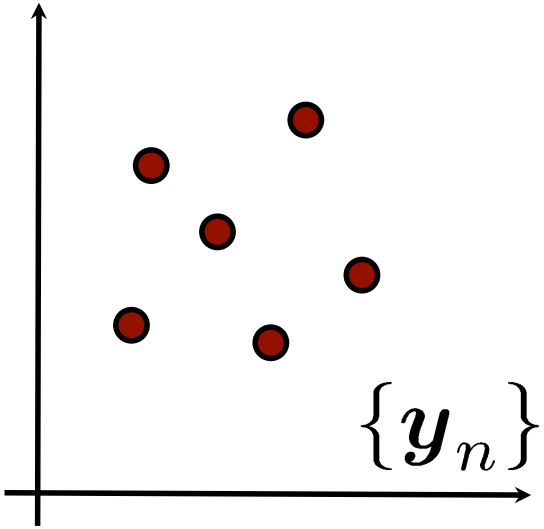
Absolute Orientation Problem: Derivation of the Solution

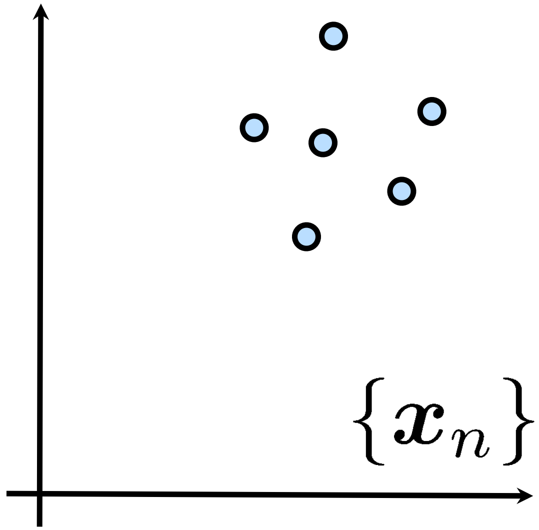
Cyrill Stachniss

5 Minute Preparation for Today

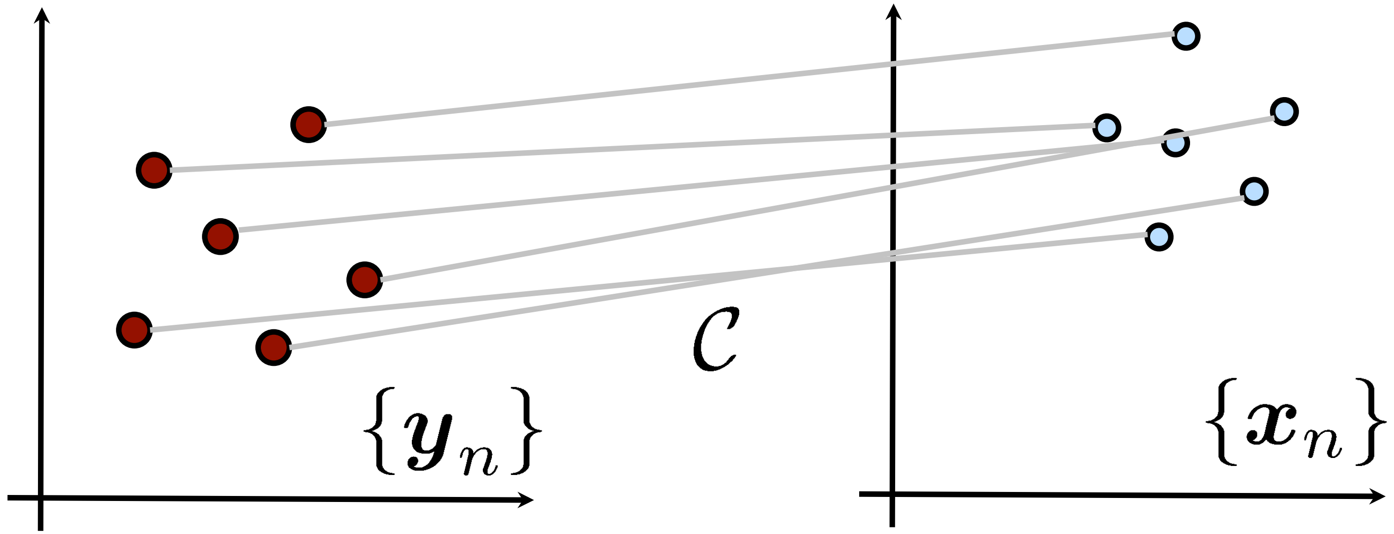
https://www.ipb.uni-bonn.de/5min/

3D Point Set Registration

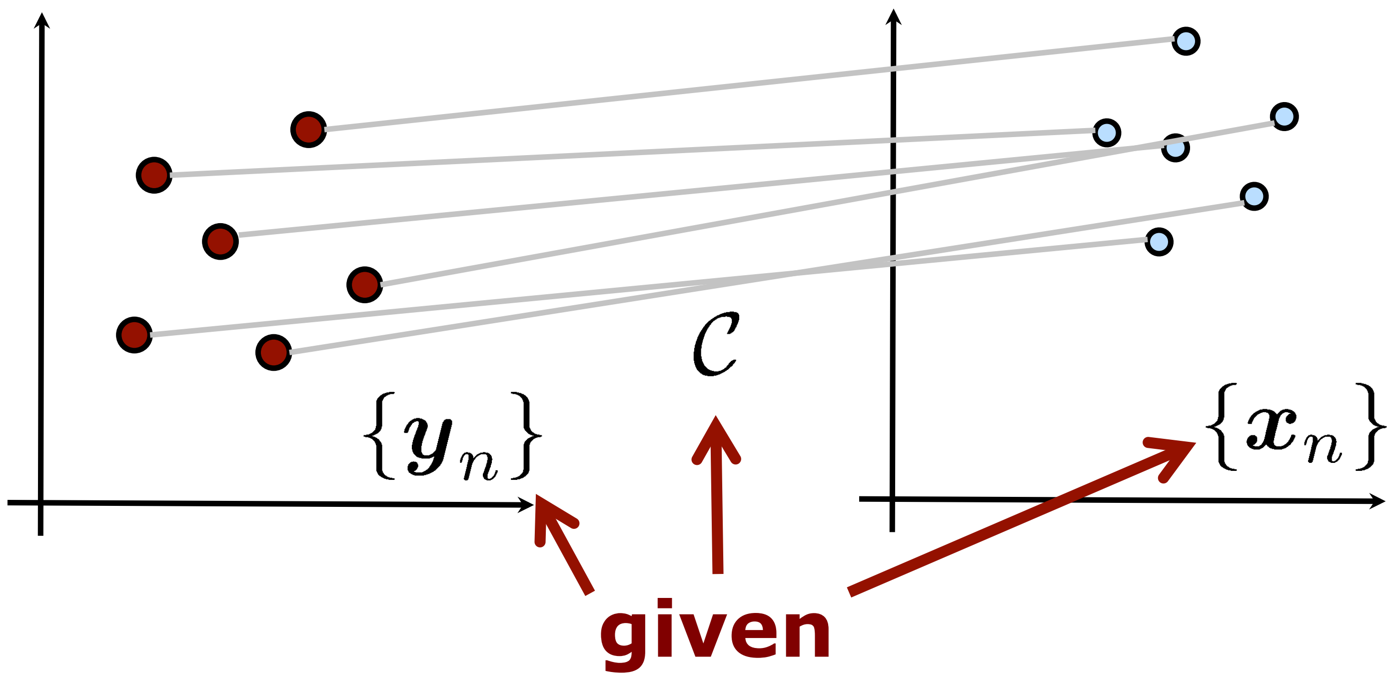




3D Point Set Registration

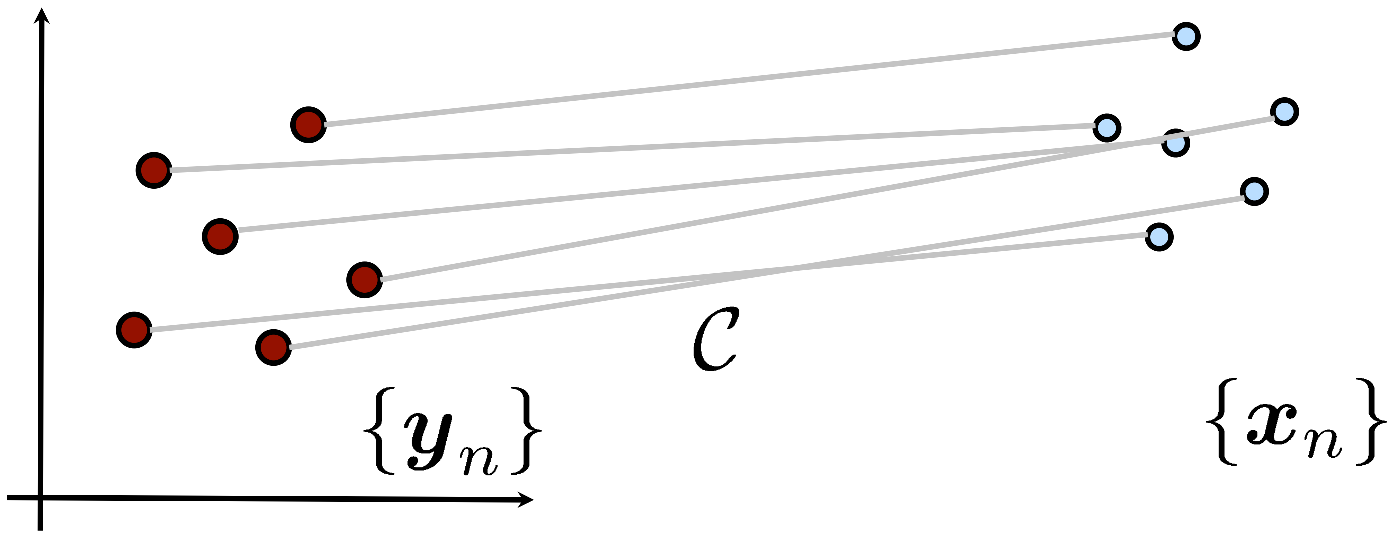


3D Point Set Registration



3D Point Set Registration

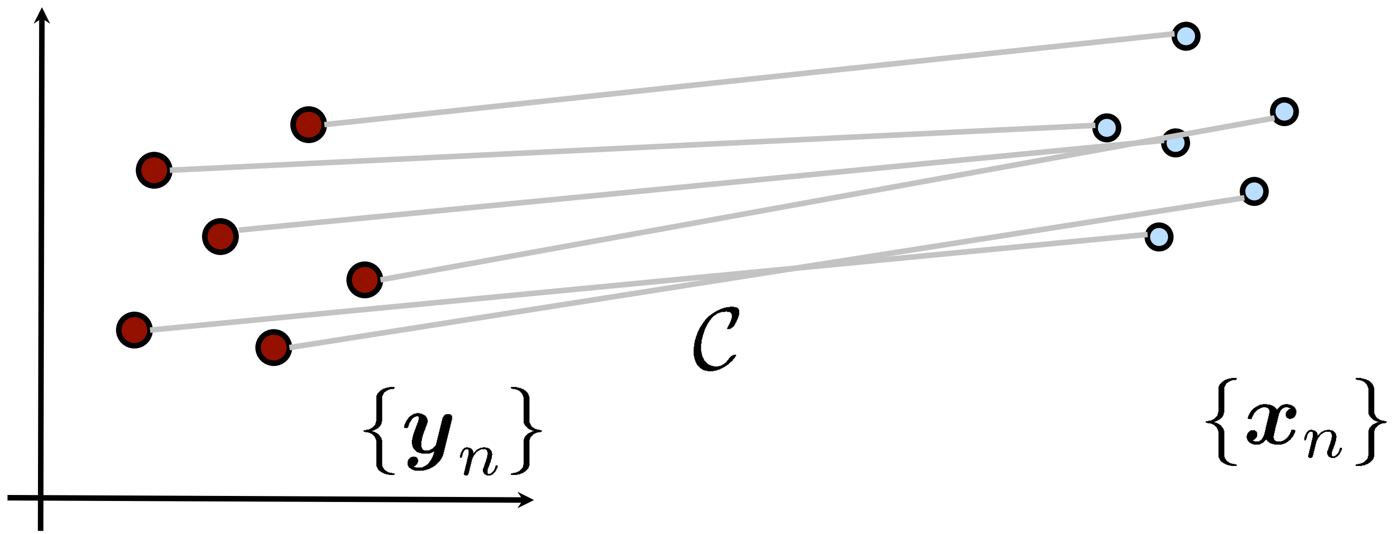
$$\bar{\boldsymbol{x}}_n = \lambda R \boldsymbol{x}_n + \boldsymbol{t}$$



3D Point Set Registration

to be estimated

$$\bar{\boldsymbol{x}}_n = \lambda R \boldsymbol{x}_n + \boldsymbol{t}$$

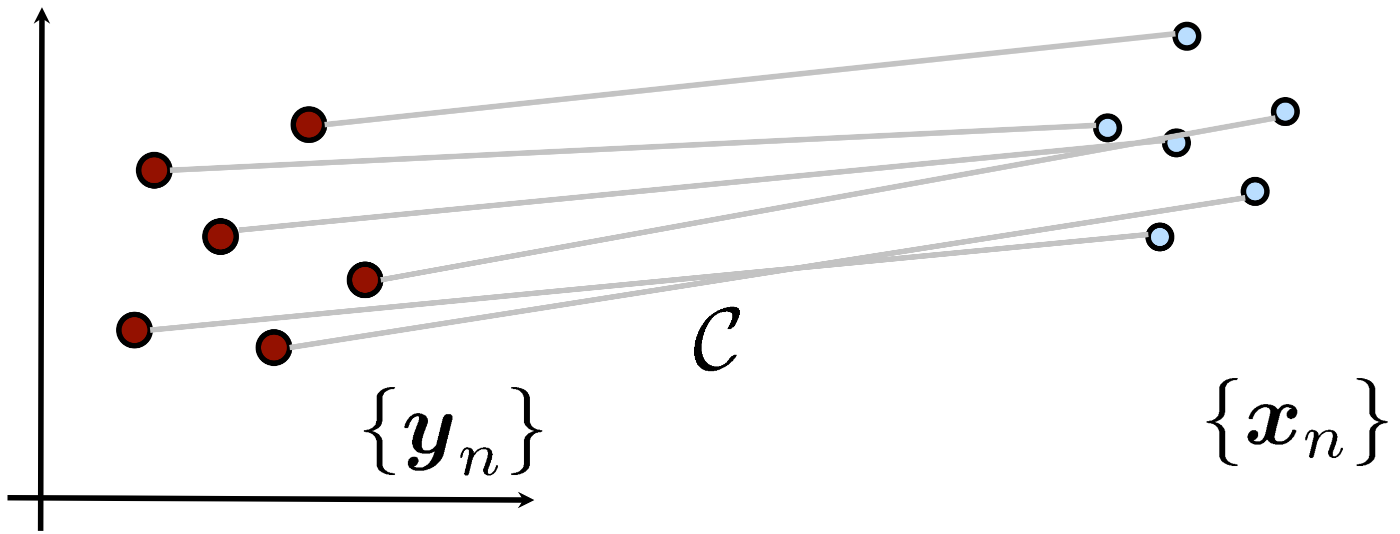


3D Point Set Registration

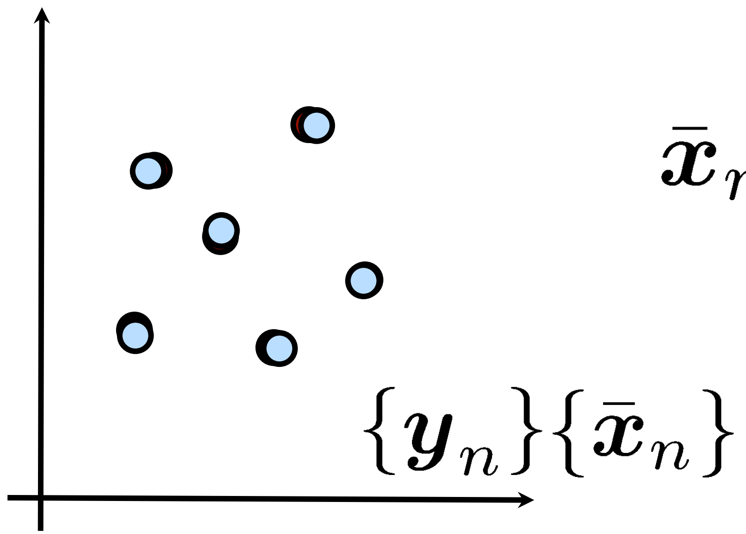
transformation should yield

$$\bar{\boldsymbol{x}}_n = \lambda R \boldsymbol{x}_n + \boldsymbol{t}$$

$$\sum ||\boldsymbol{y}_n - \bar{\boldsymbol{x}}_n||^2 \rightarrow \min$$



3D Point Set Registration



$$\bar{\boldsymbol{x}}_n = \lambda R \boldsymbol{x}_n + \boldsymbol{t}$$

Absolute Orientation

- Find the similarity transform

$$\bar{\boldsymbol{x}}_n = \lambda R \boldsymbol{x}_n + \boldsymbol{t}$$

- that transforms a set of points $\{\boldsymbol{x}_n\}$
- so that the points $\{\bar{\boldsymbol{x}}_n\}$ will be as close as possible to the point set $\{\boldsymbol{y}_n\}$
- Known correspondences
- Minimize the squared error

$$\sum ||\boldsymbol{y}_n - \bar{\boldsymbol{x}}_n||^2 \to \min$$

Why is That Relevant?

For camera data

- Camera pairs can only compute a 3D model defined up to a similarity transform (photogrammetric model)
- A.O. anchors the model

For LiDARs / point clouds

- A.O. is a key ingredient of the iterative closest point algorithm (ICP)

Example:
Absolute Orientation
Used on Image Data

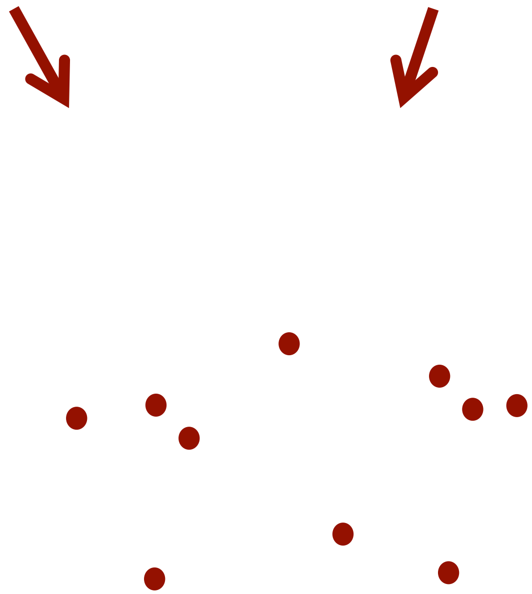
Absolute Orientation

- A similarity transform maps the photogrammetric model into the object reference frame

$$ {}^{o}\boldsymbol{X}_n = \underline{\lambda R}\, {}^{m}\boldsymbol{X}_n + \underline{\boldsymbol{T}} $$

- 7 DoF for the similarity transform (3 rotation, 3 translation, 1 scale)
- **Control points** are required (3+)

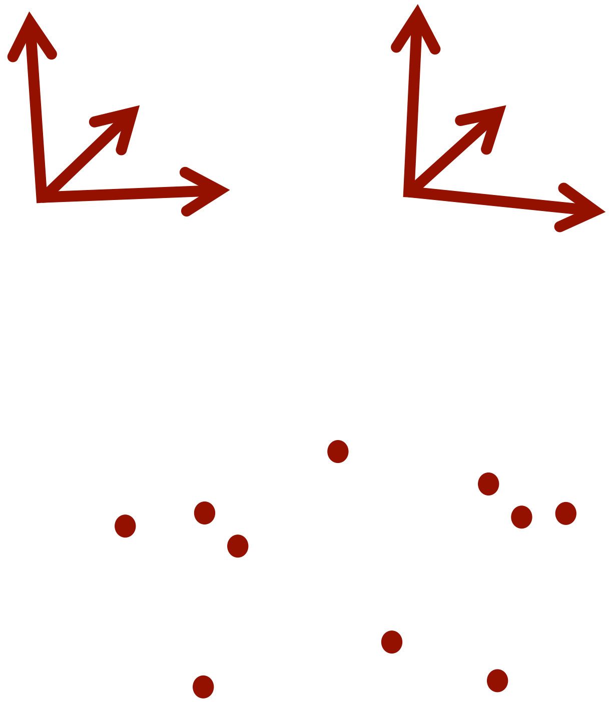
Relative Orientation



two cameras

corresponding points in both images

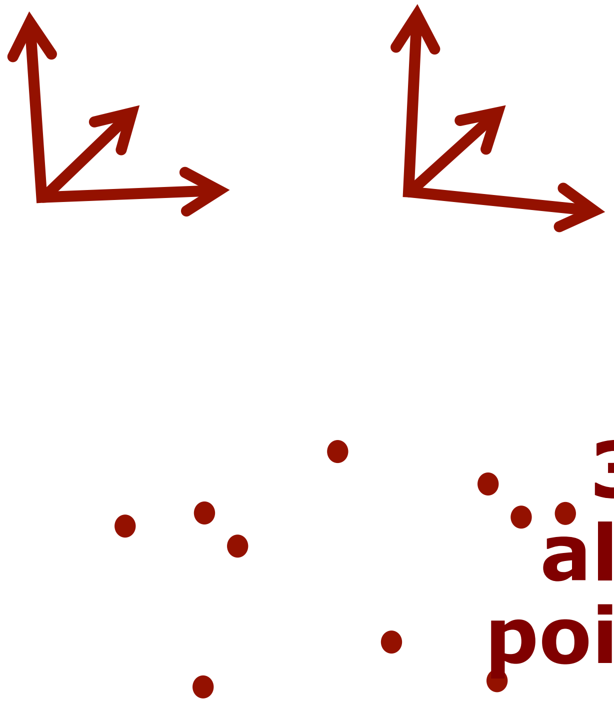
Relative Orientation



relative orientation

corresponding points in both images

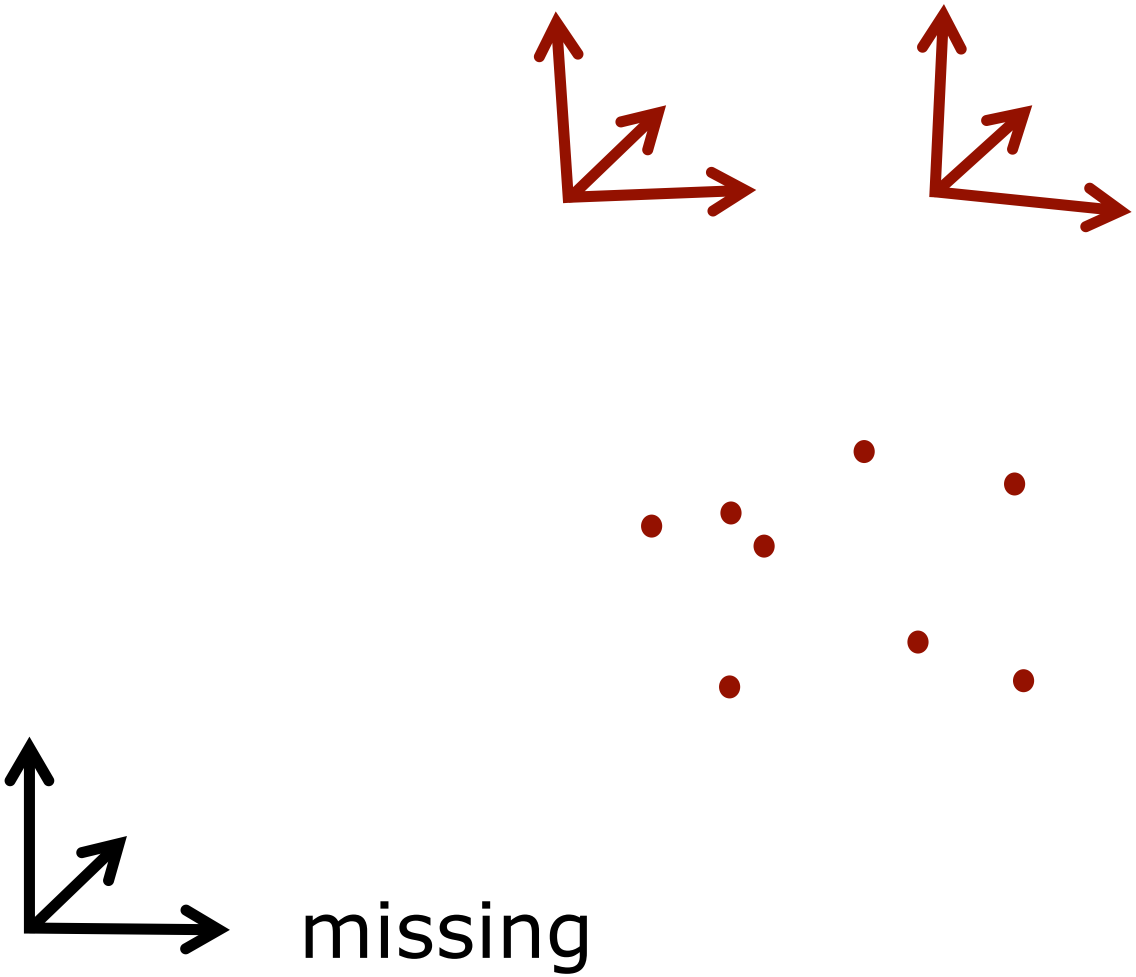
Triangulation



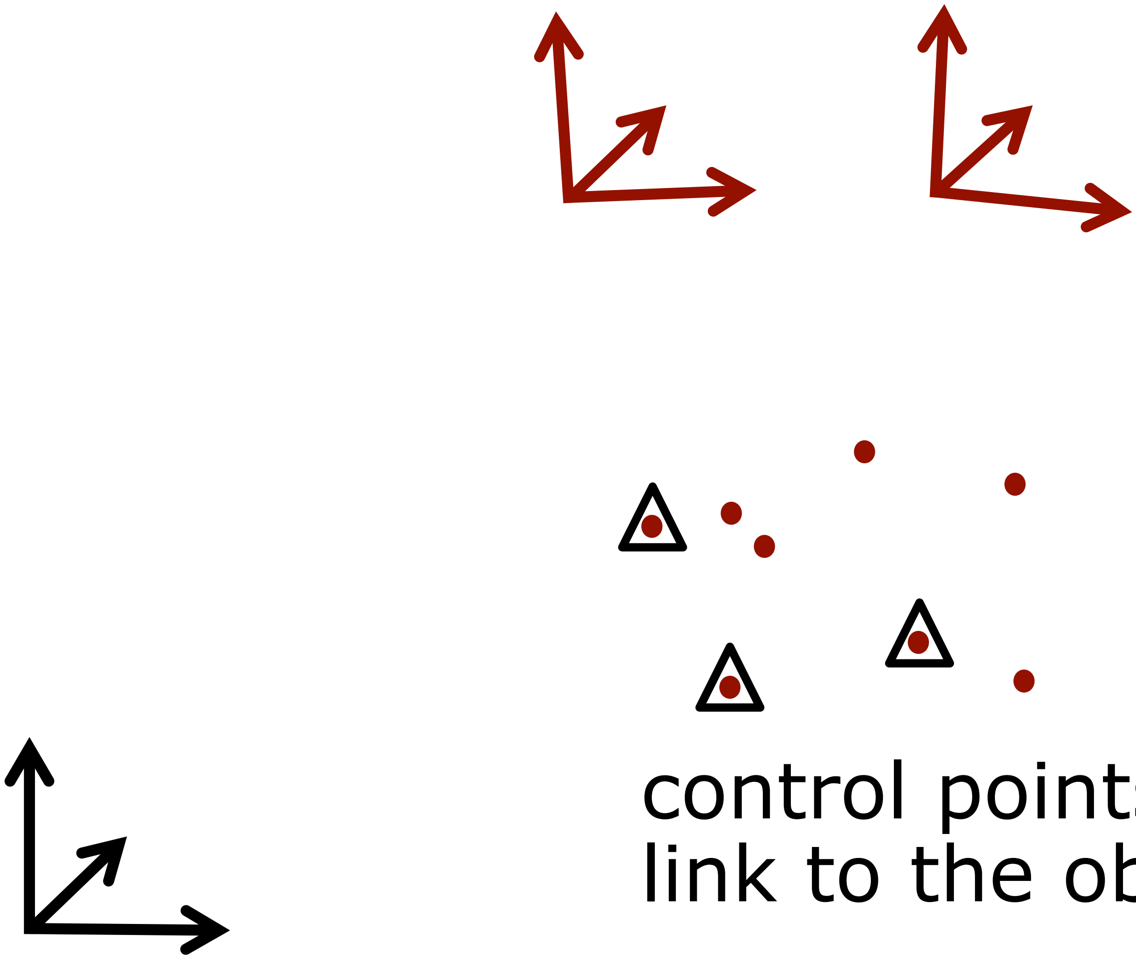
relative orientation

3D locations of all corresponding points (local frame)

Photogrammetric Model



Object Reference Frame



control points create the
link to the object frame

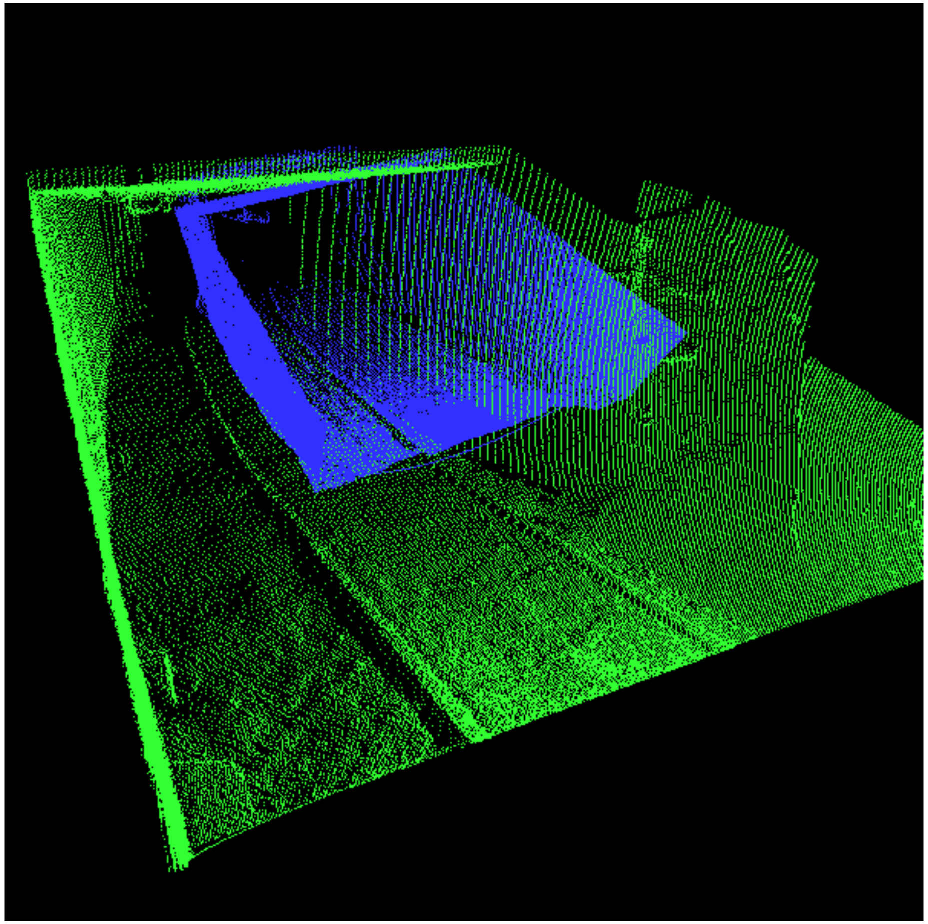
Example:
Absolute Orientation
Used on Point Cloud Data

Point Cloud Registration (ICP)

- Find the rigid body transform

$$\bar{\boldsymbol{x}}_n = R\boldsymbol{x}_n + \boldsymbol{t}$$

- to transform one point cloud or surface into the other
- Equal scale



Video courtesy: Nuechter

Problem Definition

- Given corresponding points:

$$\boldsymbol{y}_n, \boldsymbol{x}_n \qquad n = 1, \ldots, N$$

- and weights (optional):

$$p_n \qquad n = 1, \ldots, N$$

- Find the parameters $\lambda, R, \boldsymbol{t}$ of the similarity transform so that

$$\bar{\boldsymbol{x}}_n = \lambda R \boldsymbol{x}_n + \boldsymbol{t} \qquad n = 1, \ldots, N$$

- so that the squared error is minimized

$$\sum ||\boldsymbol{y}_n - \bar{\boldsymbol{x}}_n||^2 \, p_n \to \min$$

Derivation of a Direct Solution to the Absolute Orientation Problem

Use Local Coordinate System

- We use local coordinates defined by the point set $\{\boldsymbol{y}_n\}$
- We set the origin as weighted mean of $\{\boldsymbol{y}_n\}$ computed by

$$\boldsymbol{y}_0 = \frac{\sum \boldsymbol{y}_n p_n}{\sum p_n}$$

- so that we minimize

$$\sum ||\boldsymbol{y}_n - \boldsymbol{y}_0 - \lambda R \boldsymbol{x}_n - \boldsymbol{t} + \boldsymbol{y}_0||^2 \, p_n \rightarrow \min$$

does not change the problem

Rewrite Translation Vector

- Start with $\bar{\boldsymbol{x}}_n = \lambda R \boldsymbol{x}_n + \boldsymbol{t}$
- and use the shift of the origin

$$\bar{\boldsymbol{x}}_n - \boldsymbol{y}_0 = \lambda R \boldsymbol{x}_n + \boldsymbol{t} - \boldsymbol{y}_0$$

- to rewrite the translation vector

$$\bar{\boldsymbol{x}}_n - \boldsymbol{y}_0 = \lambda R(\boldsymbol{x}_n + \underline{R^\top \boldsymbol{t} - R^\top \boldsymbol{y}_0})$$

- Introduce a new variable $\boldsymbol{x}_0$:

$$\bar{\boldsymbol{x}}_n - \boldsymbol{y}_0 = \lambda R(\boldsymbol{x}_n - \boldsymbol{x}_0)$$

- with $\boldsymbol{x}_0 = R^\top \boldsymbol{y}_0 - R^\top \boldsymbol{t}$

Rewrite Scale Term

- Start with $\bar{\boldsymbol{x}}_n - \boldsymbol{y}_0 = \lambda R(\boldsymbol{x}_n - \boldsymbol{x}_0)$
- Divide both sides by $\lambda^{\frac{1}{2}}$
- This leads to

$$\underline{\lambda}^{-\frac{1}{2}}(\bar{\boldsymbol{x}}_n - \boldsymbol{y}_0) = \underline{\lambda}^{\frac{1}{2}} \underline{R}(\boldsymbol{x}_n - \underline{\boldsymbol{x}_0})$$

unknowns parameters to determine

- **Goal:** Find parameters such that

$$\sum ||\boldsymbol{y}_n - \bar{\boldsymbol{x}}_n||^2 \, p_n \to \min$$

Define Objective Function

- Minimize the fct. $\Phi(\boldsymbol{x}_0, \lambda, R)$
- defined by

$$\Phi(\boldsymbol{x}_0, \lambda, R) = \sum \left[\lambda^{-\frac{1}{2}}(\boldsymbol{y}_n - \boldsymbol{y}_0) - \lambda^{\frac{1}{2}} R(\boldsymbol{x}_n - \boldsymbol{x}_0)\right]^\top \left[\lambda^{-\frac{1}{2}}(\boldsymbol{y}_n - \boldsymbol{y}_0) - \lambda^{\frac{1}{2}} R(\boldsymbol{x}_n - \boldsymbol{x}_0)\right] p_n$$

- Minimizes a weighted squared error

How to minimize this function?

Minimize Objective Function

- Minimize the objective function

$$\Phi(\boldsymbol{x}_0, \lambda, R) = \sum \left[\lambda^{-\frac{1}{2}}(\boldsymbol{y}_n - \boldsymbol{y}_0) - \lambda^{\frac{1}{2}} R(\boldsymbol{x}_n - \boldsymbol{x}_0)\right]^\top \left[\lambda^{-\frac{1}{2}}(\boldsymbol{y}_n - \boldsymbol{y}_0) - \lambda^{\frac{1}{2}} R(\boldsymbol{x}_n - \boldsymbol{x}_0)\right] p_n$$

By

- Computing the first derivatives
- Setting derivatives to zero
- Solving the remaining equation(s)

Rearrange the Terms

- Rearrange the objective function

$$\begin{aligned}\Phi(\boldsymbol{x}_0, \lambda, R) &= \sum \left[\lambda^{-\frac{1}{2}}(\boldsymbol{y}_n - \boldsymbol{y}_0) - \lambda^{\frac{1}{2}} R(\boldsymbol{x}_n - \boldsymbol{x}_0)\right]^\top \\ &\quad \left[\lambda^{-\frac{1}{2}}(\boldsymbol{y}_n - \boldsymbol{y}_0) - \lambda^{\frac{1}{2}} R(\boldsymbol{x}_n - \boldsymbol{x}_0)\right] p_n\end{aligned}$$

- to

$$\begin{aligned}\Phi(\boldsymbol{x}_0, \lambda, R) &= \sum \lambda^{-1}(\boldsymbol{y}_n - \boldsymbol{y}_0)^\top(\boldsymbol{y}_n - \boldsymbol{y}_0)\, p_n \\ &\quad + \sum \lambda(\boldsymbol{x}_n - \boldsymbol{x}_0)^\top(\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n \\ &\quad - 2\sum(\boldsymbol{y}_n - \boldsymbol{y}_0)^\top R(\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n\end{aligned}$$

Rearrange the Terms

- Rearrange the objective function

$$\Phi(\boldsymbol{x}_0, \lambda, R) = \sum \left[\lambda^{-\frac{1}{2}}(\boldsymbol{y}_n - \boldsymbol{y}_0) - \lambda^{\frac{1}{2}} R(\boldsymbol{x}_n - \boldsymbol{x}_0)\right]^\top \left[\lambda^{-\frac{1}{2}}(\boldsymbol{y}_n - \boldsymbol{y}_0) - \lambda^{\frac{1}{2}} R(\boldsymbol{x}_n - \boldsymbol{x}_0)\right] p_n$$

- to

$$\begin{aligned}\Phi(\boldsymbol{x}_0, \lambda, R) = & \sum \lambda^{-1}(\boldsymbol{y}_n - \boldsymbol{y}_0)^\top(\boldsymbol{y}_n - \boldsymbol{y}_0)\, p_n \leftarrow \text{no } \boldsymbol{x}_0, R \\ & + \sum \lambda(\boldsymbol{x}_n - \boldsymbol{x}_0)^\top(\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n \leftarrow \text{no } R \\ & -2\sum(\boldsymbol{y}_n - \boldsymbol{y}_0)^\top R(\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n \leftarrow \text{no } \lambda\end{aligned}$$

Solve w.r.t. $\boldsymbol{x}_0$

Derivative with respect to $\boldsymbol{x}_0$

- Compute first derivative of

$$\begin{aligned}\Phi(\boldsymbol{x}_0, \lambda, R) &= \sum \lambda^{-1} (\boldsymbol{y}_n - \boldsymbol{y}_0)^\top (\boldsymbol{y}_n - \boldsymbol{y}_0)\, p_n \\ &+ \sum \lambda (\boldsymbol{x}_n - \boldsymbol{x}_0)^\top (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n \\ &-2 \sum (\boldsymbol{y}_n - \boldsymbol{y}_0)^\top R (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n\end{aligned}$$

- with respect to $\boldsymbol{x}_0$

$$\begin{aligned}\frac{\partial \Phi(\boldsymbol{x}_0, \lambda, R)}{\partial \boldsymbol{x}_0} &= -2 \sum \lambda (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n \\ &+2 \sum R^\top (\boldsymbol{y}_n - \boldsymbol{y}_0)\, p_n\end{aligned}$$

Set Derivative to Zero

- Set first derivative to zero: $\frac{\partial \Phi}{\partial \boldsymbol{x}_0} = 0$

$$0 \quad = \quad -2 \sum \lambda (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n + 2 \sum R^\top (\boldsymbol{y}_n - \boldsymbol{y}_0)\, p_n$$

- This simplifies to

$$\sum \lambda (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n \quad = \quad R^\top \sum (\boldsymbol{y}_n - \boldsymbol{y}_0)\, p_n$$

Set Derivative to Zero

- Set first derivative to zero: $\frac{\partial \Phi}{\partial \boldsymbol{x}_0} = 0$

$$0 \quad = \quad -2 \sum \lambda (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n + 2 \sum R^\top (\boldsymbol{y}_n - \boldsymbol{y}_0)\, p_n$$

- This simplifies to

$$\sum \lambda (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n \quad = \quad R^\top \underline{\sum (\boldsymbol{y}_n - \boldsymbol{y}_0)\, p_n}$$

equals zero as $\boldsymbol{y}_0$ is the weighted mean of $\boldsymbol{y}_n$

$$\sum \lambda (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n \quad = \quad 0$$

Unknown $\boldsymbol{x}_0$ is the Weighted Mean of the Points to Transform

- As $\sum \lambda(\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n \;=\; 0$
- We obtain $\sum \boldsymbol{x}_n\, p_n - \sum \boldsymbol{x}_0\, p_n \;=\; 0$
- This leads to

$$\boxed{\boldsymbol{x}_0 \;=\; \frac{\sum \boldsymbol{x}_n\, p_n}{\sum p_n}}$$

- The optimal value for $\boldsymbol{x}_0$ is the **weighted mean** of the points $\boldsymbol{x}_n$

Solve w.r.t. λ

Derivative with respect to λ

- Compute first derivative of

$$\begin{aligned}\Phi(\boldsymbol{x}_0, \lambda, R) &= \sum \lambda^{-1} (\boldsymbol{y}_n - \boldsymbol{y}_0)^\top (\boldsymbol{y}_n - \boldsymbol{y}_0)\, p_n \\ &\quad + \sum \lambda (\boldsymbol{x}_n - \boldsymbol{x}_0)^\top (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n \\ &\quad - 2 \sum (\boldsymbol{y}_n - \boldsymbol{y}_0)^\top R (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n\end{aligned}$$

- with respect to λ

$$\begin{aligned}\frac{\partial \Phi(\boldsymbol{x}_0, \lambda, R)}{\partial \lambda} &= -\lambda^{-2} \sum (\boldsymbol{y}_n - \boldsymbol{y}_0)^\top (\boldsymbol{y}_n - \boldsymbol{y}_0)\, p_n \\ &\quad + \sum (\boldsymbol{x}_n - \boldsymbol{x}_0)^\top (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n\end{aligned}$$

Set Derivative to Zero

- Set first derivative to zero: $\frac{\partial \Phi}{\partial \lambda} = 0$

$$
\begin{aligned}
0 &= -\lambda^{-2} \sum (\boldsymbol{y}_n - \boldsymbol{y}_0)^\top (\boldsymbol{y}_n - \boldsymbol{y}_0)\, p_n \\
&\quad + \sum (\boldsymbol{x}_n - \boldsymbol{x}_0)^\top (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n
\end{aligned}
$$

- Directly leads to

$$
\boxed{\lambda = \sqrt{\frac{\sum (\boldsymbol{y}_n - \boldsymbol{y}_0)^\top (\boldsymbol{y}_n - \boldsymbol{y}_0)\, p_n}{\sum (\boldsymbol{x}_n - \boldsymbol{x}_0)^\top (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n}}}
$$

Solve w.r.t. R

Compute R That Minimizes Φ

- Only the 3rd term of Φ depends on R

$$\begin{aligned}\Phi(\boldsymbol{x}_0, \lambda, R) \quad = \quad & \sum \lambda^{-1} (\boldsymbol{y}_n - \boldsymbol{y}_0)^\top (\boldsymbol{y}_n - \boldsymbol{y}_0)\, p_n \\ & + \sum \lambda (\boldsymbol{x}_n - \boldsymbol{x}_0)^\top (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n \\ & \boxed{-2 \sum (\boldsymbol{y}_n - \boldsymbol{y}_0)^\top R (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n}\end{aligned}$$

- So we need to find R that maximizes

$$R^* = \operatorname*{argmax}_R \sum (\boldsymbol{y}_n - \boldsymbol{y}_0)^\top R (\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n$$

- with $R^\top R = I$

Exploit Information So Far

- Given we know $\boldsymbol{x}_0$, compute reduced coordinates as

$$\begin{aligned}\boldsymbol{a}_n &= (\boldsymbol{x}_n - \boldsymbol{x}_0)\\ \boldsymbol{b}_n &= (\boldsymbol{y}_n - \boldsymbol{y}_0)\end{aligned}$$

- This leads from

$$R^* = \underset{R}{\operatorname{argmax}} \sum (\boldsymbol{y}_n - \boldsymbol{y}_0)^\top R(\boldsymbol{x}_n - \boldsymbol{x}_0)\, p_n$$

- to

$$R^* = \underset{R}{\operatorname{argmax}} \sum \boldsymbol{b}_n^\top R \boldsymbol{a}_n\, p_n$$

Rewrite Using the Trace

- We can directly rewrite

$$R^* = \operatorname*{argmax}_R \sum \boldsymbol{b}_n^\top R \boldsymbol{a}_n \, p_n$$

- using the trace as

$$R^* = \operatorname*{argmax}_R \operatorname{tr}(RH)$$

- with the cross covariance matrix

$$H = \sum (\boldsymbol{a}_n \boldsymbol{b}_n^\top) \, p_n$$

- **Goal: Find R that maximizes $\operatorname{tr}(RH)$**

Maximization Using SVD

- To find R that maximizes $\mathrm{tr}\,(RH)$, we can exploit the SVD
- SVD gives us

$$\mathrm{svd}(H) = UDV^\top$$

- with

$$U^\top U = I \qquad V^\top V = I \qquad D = \mathrm{diag}(d_i)$$

Maximization Using SVD

- Let's see what happens if we set

$$R = VU^\top$$

- Then, we obtain

$$\mathrm{tr}\left(RH\right) = \mathrm{tr}\left(V \underbrace{U^\top U}_{I} D V^\top\right) = \mathrm{tr}\left(VDV^\top\right)$$

- and can rewrite this as

$$\mathrm{tr}\left(VDV^\top\right) = \mathrm{tr}\left(VD^{\frac{1}{2}}D^{\frac{1}{2}}V^\top\right)$$

Maximization Using SVD

- Given that D is diagonal, we get

$$\operatorname{tr}\left(VD^{\frac{1}{2}}D^{\frac{1}{2}}V^{\top}\right) = \operatorname{tr}\left(VD^{\frac{1}{2}}(D^{\frac{1}{2}}V)^{\top}\right)$$

- and with the definition $A = VD^{\frac{1}{2}}$

$$\operatorname{tr}(RH) = \operatorname{tr}\left(AA^{\top}\right)$$

- with A being a positive definite matrix, details see: Arun et al (1987)

Exploit Schwarz Inequality

- For every pos. definite matrix A holds

  $$\operatorname{tr}\left(AA^\top\right) \geq \operatorname{tr}\left(R'AA^\top\right)$$

  for any rotation matrix R'
- Result of the Schwarz inequality
- This means

  $$\operatorname{tr}(RH) = \operatorname{tr}\left(AA^\top\right) \geq \operatorname{tr}\left(R'AA^\top\right) = \operatorname{tr}(\underline{R'RH})$$

  any other rotation matrix
- Thus, our choice $R = VU^\top$ was optimal as it maximizes the trace

Proof that $\mathrm{tr}\left(AA^\top\right) \geq \mathrm{tr}\left(R'AA^\top\right)$

optional

Lemma: For any positive definite matrix AA^t, and any orthonormal matrix B,

$$\text{Trace}\,(AA^t) \geq \text{Trace}\,(BAA^t).$$

Proof of Lemma: Let a_i be the ith column of A. Then

$$\begin{aligned}\text{Trace}\,(BAA^t) &= \text{Trace}\,(A^tBA)\\ &= \sum_i a_i^t(Ba_i).\end{aligned}$$

But, by the Schwarz inequality,

$$a_i^t(Ba_i) \leq \sqrt{(a_i^ta_i)(a_i^tB^tBa_i)} = a_i^ta_i.$$

Hence, Trace $(BAA^t) \leq \Sigma_i\, a_i^ta_i =$ Trace (AA^t). Q.E.D.

Let the SVD of H be:

See: Arun et al (1987) "Least-Squares Fitting of Two 3D Point Sets." IEEE T-PAMI 9(5), 698–700.

Optimal R

- The rotation matrix minimizing Φ is

$$\boxed{R = VU^\top}$$

- with $\mathrm{svd}(H) = UDV^\top$
- and $H = \sum (\boldsymbol{a}_n \boldsymbol{b}_n^\top)\, p_n$

Unique Solution?

- SVD provides the decomposition

$$\mathrm{svd}(H) = UDV^\top$$

- The matrices U, V are 3 by 3 matrices
- U, V are rotation matrices
- Diagonal matrix $D = \mathrm{Diag}(d_1, d_2, d_3)$
- Only if $\mathrm{rank}(H) = 3$, the rotation minimizing Φ is unique

Translation Vector

- Based on $\boldsymbol{x}_0$ and R, we obtain the translation vector $\boldsymbol{t}$ of our transform
- Starting from

$$\boldsymbol{x}_0 = R^\top \boldsymbol{y}_0 - R^\top \boldsymbol{t}$$

- directly leads to

$$\boxed{\boldsymbol{t} = \boldsymbol{y}_o - R\boldsymbol{x}_0}$$

We Derived the Solution for the Absolute Orientation Problem

- Scale: $$\lambda = \sqrt{\frac{\sum(\boldsymbol{y}_n - \boldsymbol{y}_0)^\mathsf{T}(\boldsymbol{y}_n - \boldsymbol{y}_0)p_n}{\sum(\boldsymbol{x}_n - \boldsymbol{x}_0)^\mathsf{T}(\boldsymbol{x}_n - \boldsymbol{x}_0)p_n}}$$
- Rotation: $$R = VU^\top$$
- Translation: $$\boldsymbol{t} = \boldsymbol{y}_0 - R\boldsymbol{x}_0$$
- with $$\boldsymbol{y}_0 = \frac{\sum \boldsymbol{y}_n p_n}{\sum p_n} \qquad \boldsymbol{x}_0 = \frac{\sum \boldsymbol{x}_n p_n}{\sum p_n}$$

$$H = \sum (\boldsymbol{x}_n - \boldsymbol{x}_0)(\boldsymbol{y}_n - \boldsymbol{y}_0)^\top p_n \qquad \mathrm{svd}(H) = UDV^\top$$

Note: Two Different Variants...

- There are two (sometimes confusing) variants of the problem formulation
- Variant 1:

$$H = \sum (\boldsymbol{x}_n - \boldsymbol{x}_0)(\boldsymbol{y}_n - \boldsymbol{y}_0)^\top p_n \qquad R = V U^\top$$

- Variant 2:

$$H = \sum (\boldsymbol{y}_n - \boldsymbol{y}_0)(\boldsymbol{x}_n - \boldsymbol{x}_0)^\top p_n \qquad R = U V^\top$$

- Both are equivalent!

Summary

- Solving the absolute orientation problem is key in photogrammetry and point cloud processing
- Computes the similarity transforms between two point sets
- Direct solution that minimizes the squared error
- Efficient to implement
- Effective and popular approach