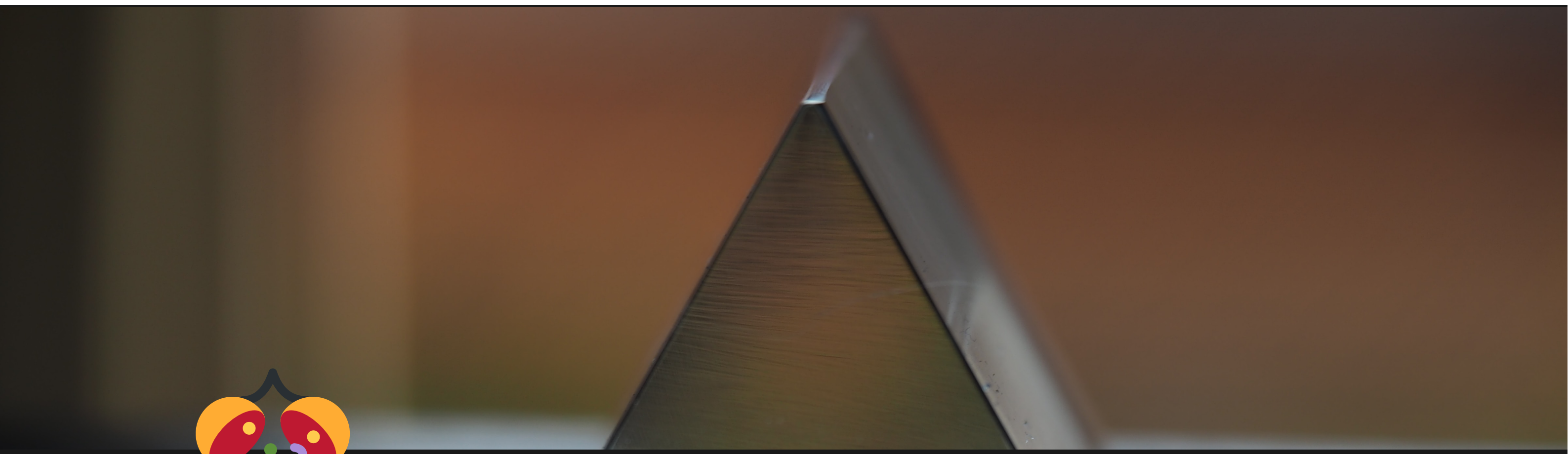

# Minimum point correspondences b/w two point clouds to solve for transformation between them

Resources
Question
Justification 1: Algorithmically
Justification 2: Geometric intuition
Some Prerequisite Theory for "Justification 3"
  Orthogonal Procrustes problem
    Solution
    When is solution to Orthogonal Procrustes unique for our point set registration problem?
  Wahba's Problem: Constrained Procrustes problem
    Solution
    When is solution to Wahba unique for our point set registration problem?
Justification 3: Theoretical
  Orthogonal Procrustes revisited
  Rank of the sum of outer products
    Solving a more standard problem: How many outer product terms make a full rank?
    Orthogonal Procrustes revisited, revisited! 💥
  Wahba's algorithm revisited
Conclusion

> Page author: Shubodh Sai

▶ Kindly comment on this Notion page itself if there are any issues in this page.

- If you want to email me for general issues, click on my name above.

## Resources

▶ Resources

## Question

> Given P1 and P2 as a set of two-point clouds in frames 1 and 2, how many point correspondences are needed between the two frames to solve for the transformation between them? Why?

Given two corresponding point sets:

$$Q = \{q_1, \ldots, q_N\}$$
$$P = \{p_1, \ldots, p_N\}$$

with correspondences $C = \{(i, j)\}$.

Wanted: Translation $t$ and rotation $R$ that minimize the sum of the squared errors:

$$E(R, t) = \sum_{(i,j) \in C} \left\| \boldsymbol{q}_i - R\boldsymbol{p}_j - t \right\|^2$$

subject to $\mathbf{R}\mathbf{R}^T = \mathbf{I}$

**The question is: What is the minimum value of $N$ above.**

🎇 **Answer: 3 point correspondences.**

## Justification 1: Algorithmically

> I made a nice ipython notebook for you to figure out the answer yourself at [this link].

See the code to get convinced that we only need 3 using Wahba algorithm (constrained Orthogonal Procrustes).

▶ Spoilers from the notebook

If you also want to understand this theoretically, look at the "Justification 3: Theoretical" below.

Geometrical justificaiton? Next section:

## Justification 2: Geometric intuition

> Before reading the following, it is a good idea to visualize it geometrically for 2 point correspondences and try to figure out yourself.

Say there are two point clouds $A$ and $B$ to be registered. Assume we just need to 2 correspondences, say 3D point $X_1$ in $A$ is corresponding to $X_2$ in $B$ and $Y_1$ in $A$ is corresponding to $Y_2$ in $B$.

Now register the two-point clouds $A$ and $B$. Now both are registered, so correspondences $X_1$ and $X_2$ are one point $X$; $Y_1$ and $Y_2$ are one point $Y$. But the problem now is, you can rotate the point cloud $B$ about the line $XY$ (the line joining the two correspondences), note that the 2 correspondences are still overlapping but point clouds are at not locked in space but are free to rotate about that line.

Therefore, for locking this additional rotation also in space, we need at least 3 non-collinear points.

This intuition can also be looked at from the perspective of 3 points forming a plane and then overlapping these 2 planes.

## Some Prerequisite Theory for "Justification 3"

Recollect that our problem is to minimize:

$$F'(R) = \| [\boldsymbol{q}'_1 \dots \boldsymbol{q}'_n] - R\,[\boldsymbol{p}'_1 \dots \boldsymbol{p}'_n] \|_F^2$$

where $R$ is the rotation matrix

## Orthogonal Procrustes problem

▸ Special case of Wahba's problem.

$$\Omega_{min} = \arg\min_{\Omega} ||\Omega A - B||_F \quad \text{subject to} \quad \Omega^T\Omega = I$$

### Solution

Consider $M = BA^T$, and do SVD of $M = U\Sigma V^T$. Then $\Omega_{min} = UV^T$.

▸ Proof

> Now that you know what Orthogonal Procrustes problem is, it is a good idea to revisit it in the context of point cloud registration. Link here for the full theory.

### When is solution to Orthogonal Procrustes unique for our point set registration problem?

In the context of our point set registration problem, $\mathrm{rank}(W)$ **has to be 3 for the optimal solution of $\mathrm{E}(R, t)$ to be unique.**

> Note that this is not exactly the same as our problem, as here $\Omega$ is an orthogonal matrix and not a rotation matrix (det = +1)

## Wahba's Problem: Constrained Procrustes problem

> This is actually our problem (a more specific version rather, just plug in $a = 1$ to get our formulation).

$$J(\mathbf{R}) = \frac{1}{2}\sum_{k=1}^{N} a_k \left\|\mathbf{w}_k - \mathbf{R}\mathbf{v}_k\right\|^2 \text{ for } N \geq 2$$

$\mathbf{R}$ is a 3 by 3 rotation matrix!

$\mathbf{w}_k$ is the k-th 3-vector measurement in the reference frame, $\mathbf{v}_k$ is the corresponding k-th 3-vector measurement in the body frame.

▶ Not to get carried away by N ≥ 2 here.

### Solution

1. Obtain a matrix $\mathbf{M}$ as follows: $\mathbf{M} = \sum_{i=1}^{n} a_i \mathbf{w}_i \mathbf{v}_i^T$. For us, $a_i$'s are 1.
2. Find the singular value decomposition of $\mathbf{M}$: $\mathbf{M} = \mathbf{U}\mathbf{S}\mathbf{V}^T$
3. The rotation matrix is simply:

   $\mathbf{R} = \mathbf{U}\mathbf{K}\mathbf{V}^T$ where $\mathbf{K} = \mathrm{diag}\left(\begin{bmatrix} 1 & 1 & \det(\mathbf{U})\det(\mathbf{V}) \end{bmatrix}\right)$

> Now this is the correct approach to our problem!

### When is solution to Wahba unique for our point set registration problem?

- Reference

Note that above in Orthogonal Procrustes, rank had to be 3 for solution to be unique.

But here, it is more relaxed:

**The solution to our problem is unique if $\sigma_2(M)$ is nonzero, where $\sigma_2(M)$ is the singular value of the matrix $M$.**

- ▶ $M$ and $\sigma$ in more detail (Toggle)
- ▶ More strictly

---

- ▶ Kabsch algorithm:

---

## Justification 3: Theoretical

If you are an online learner like me having learnt from Cyrill Stachniss/Wolfram Burgard's lectures, we need to proceed with a bit of caution here. It is a good idea to play with code in "Justification 1" above before proceeding here.

### Orthogonal Procrustes revisited

Quickly recollect steps in SVD based Orthogonal Procrustes point cloud registration method by clicking on this page for viewing the complete algorithm in detail. Only the steps are mentioned below without any explanation.

The most standard SVD based Orthogonal Procrustes approach to point cloud registration says the following:

$$\begin{aligned} Q' &= \{\boldsymbol{q}_i - \boldsymbol{\mu}_Q\} = \{\boldsymbol{q}'_i\} \\ P' &= \{\boldsymbol{p}_i - \boldsymbol{\mu}_P\} = \{\boldsymbol{p}'_i\} \end{aligned} \qquad W = \sum q'_i p'^T_i$$

If $\operatorname{rank}(W) = 3$, the optimal solution of $\mathrm{E}(R, t)$ is unique and is given by:

$$\begin{aligned} R &= UV^T \\ t &= \mu_Q - R\mu_p \end{aligned}$$

**This is what is taught in standard ICP lectures (Cyrill Stachniss link).** The content from the lecture alone will not give us our right answer of 3 point corresondences, as explained below. T*his is the point of caution I was talking about.*

Now the question is to ensure $\text{rank}(W) = 3,$ how many points do we need?

> Spoilers:
> As you will see below, we will get the answer as 4 point correspondences to ensure $\text{rank}(W) = 3$. Does that mean our geometric intuition is wrong? 😢
>
> Not really. 😃
>
> Fret not, it is not actually true that $\text{rank}(W)$ must be 3 for you to find a closed-form solution of $R, t$. Here, we are only saying if you want to apply the Orthogonal Procrustes algorithm, you need to ensure $\text{rank}(W) = 3$ for which 4 corresponding points are needed.
>
> So is there an algorithm for which only 3 points are needed (as your geometric intuition suggests)?
>
> YES! Enter Wahba Algorithm. 💥
>
> Let's not get ahead of ourselves: let us first understand why 4 points are needed for ensuring $\text{rank}(W) = 3$.

## Rank of the sum of outer products

### Solving a more standard problem: How many outer product terms make a full rank?

Before getting to our case, let us solve a more standard problem.

$$H = \sum_{(i)\in C} q_i p_i^T \qquad (0.)$$

**Above, forget about what $p$ and $q$'s are in the context of our problem. Think of them as just linearly independent vectors. We will revisit this later, don't worry.**

> **Let $p_i$'s and $q_i$'s be linearly independent vectors of dimension 3. $H$ is a $3 \times 3$ matrix. Now the question is: How many linearly independent vectors do I need to ensure $\text{rank}(H) = 3$.**

The max no. of linearly independent vectors in 3D space is obviously 3. Here the question is, can it be < 3 to ensure $\text{rank}(H) = 3$**?**

Here dimension is 3, but the following method can be extended to any N dimensional space.

Recollect: A matrix $H$ is full rank if the following condition holds:

$Hx = 0$ if and only if $x = 0$.

So let us look at when $Hx$ would be 0. Consider

$$Hx = \sum q_i p_i^T x$$

Note the inner product $p_i^T x$, which gives a scalar $f$.

$$Hx = \sum f_i q_i$$

Since $q_i$'s are independent vectors, the RHS would be 0 if and only if $f_i$'s are 0 i.e. all $p_i^T x$ are 0. Or in other words, if $Hx = 0$, then all $p_i^T x$ must be 0. $\rightarrow (2.)$

(2) above is true only for independent vectors, the maximum number of independent vectors in 3D space is 3. Let that number of linearly independent vectors be $N$. So $N \leq 3$. $\rightarrow (2.2)$

Let us rewrite it in matrix form:

$$P^T x = \begin{bmatrix} p_1^T \\ \dots \\ p_N^T \end{bmatrix} x = 0$$

The question now is can $N$ be $< 3$? *Remember that we need to prove $Hx = 0$ if and only if $x = 0$.*

Let's revisit what we did so far:

Q) **How many linearly independent vectors do I need to ensure** $\text{rank}(H) = 3$.
1. For $H$ to be full rank, the following condition must hold: if $Hx = 0$ then $x = 0$. ✅
2. If $Hx = 0$, then all $p_i^T x$ must be 0. ✅
3. If all $p_i^T x = 0$ or in other words, $P^T x = 0$, then $x$ must be 0 ❓

- 💡 If condition 3. [If $P^T x = 0 \implies x = 0$] holds, then from points 2. and 3., we can say the condition [If $Hx = 0 \implies x = 0$] holds, then we can also say $H$ is full rank which is what we want.
- ❓ But when will condition 3. [If $P^T x = 0 \implies x = 0$] hold in the first place?
- ✅ When $P^T$ is full rank, condition 3. will hold. That means above, $N \geq 3$ for $P^T$ to be full rank. $\rightarrow (3.1)$
- ❓ But when will condition 2. [If $Hx = 0 \implies p_i^T x = 0$] hold?
- ✅ We already discussed above in (2.2) that $N \leq 3$. $\rightarrow (3.2)$
- (3.1) and (3.2) then tell $N = 3$.

> **Therefore, we need 3 linearly independent vectors to ensure $\text{rank}(H) = 3$. In other words, 3 outer product terms make a full rank.**

But above, I gave you a spoiler that we need 4 points to ensure $\text{rank}(W) = 3$. But we just found out we need only 3, what is going on here⁉️

## Orthogonal Procrustes revisited, revisited! 💥

💭 Remember in (0.), we assumed $p$'s and $q$'s are linearly independent vectors?

$$H = \sum_{(i) \in C} q_i p_i^T \qquad (0.)$$

But let us look at our problem:

$$W = \sum q'_i p'^T_i$$

💭 Recollect what the "dash vectors" $p'_i$ and $q'_i$ above.

- 🎯 Let's consider 3 linearly independent $p_i$'s (and $q_i$'s) in 3D space, so basically $i : 1 \to 3$. Let us just look at $p_i$'s now.
- 💡Are the "dash vectors" $p'_i$'s too linearly independent? Pause your life now and think about it, put pen on paper and look at the geometry!

  ▶ The answer is
- ✅Therefore, if you start with 3 linearly independent $p_i$'s, you will end up with only 2 linearly independent "dash vectors" $p'_i$'s.
- 🏁Now since $W$ is a function of "dash vectors" and we need to ensure $\text{rank}(W) = 3$ for our Orthogonal Procrustes algorithm, you need **3 linearly independent "dash vectors" $p'_i$'s (and $q'_i$'s). For which, you need 4 linearly independent $p_i$'s and $q_i$'s!**
- 🥳 Therefore, we need 4 points to ensure $\text{rank}(W) = 3$ which is what **we got in our GitHub notebook experiments as well.**

> We finally understood why we need 4 points for Orthogonal Procrustes algorithm.
>
> As our geometric intuition suggests, only 3 points are actually needed for registering two point clouds: Which is what Wahba's algorithm uses.