UNIVERSITY OF AMSTERDAM

Computer Vision 2

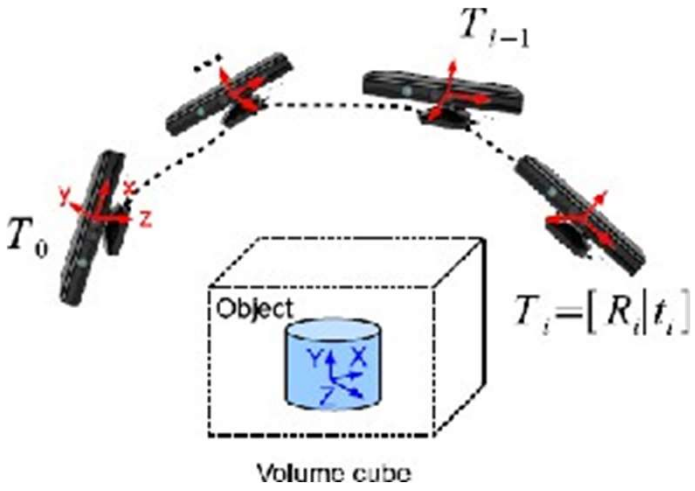
Practical Session #1

Iterative Closest Point (ICP)

Weijie Wei, Rick Groenendijk

Background

3D Reconstruction (Multi-view Active Stereo)



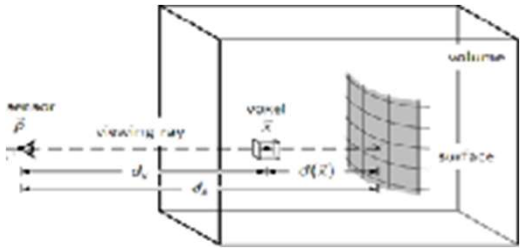
Overview of 3D reconstruction algorithm

Point Cloud

View Registration algorithm

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} [R|t] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

For the first frame, there is no rotation and translation



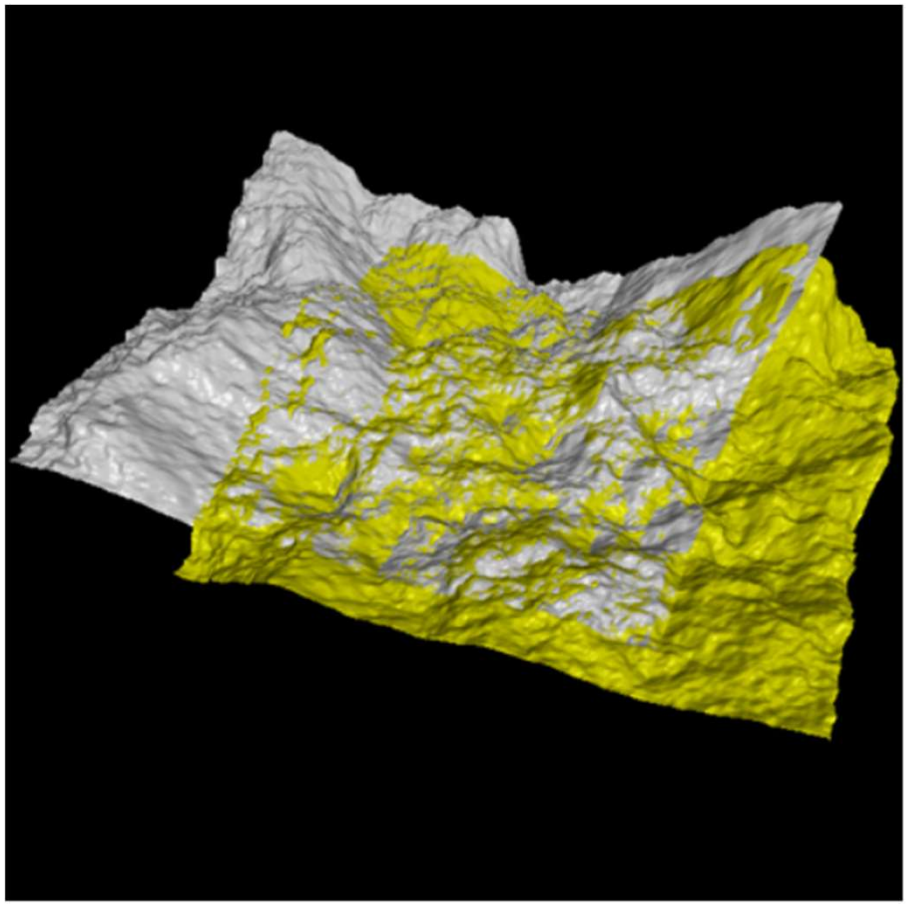
Volume stitching

Background

ICP (Iterative Closest Point) algorithm was first introduced by Besl and McKay[1].

Given initial guess, it can help us align two partially-overlapping meshes.

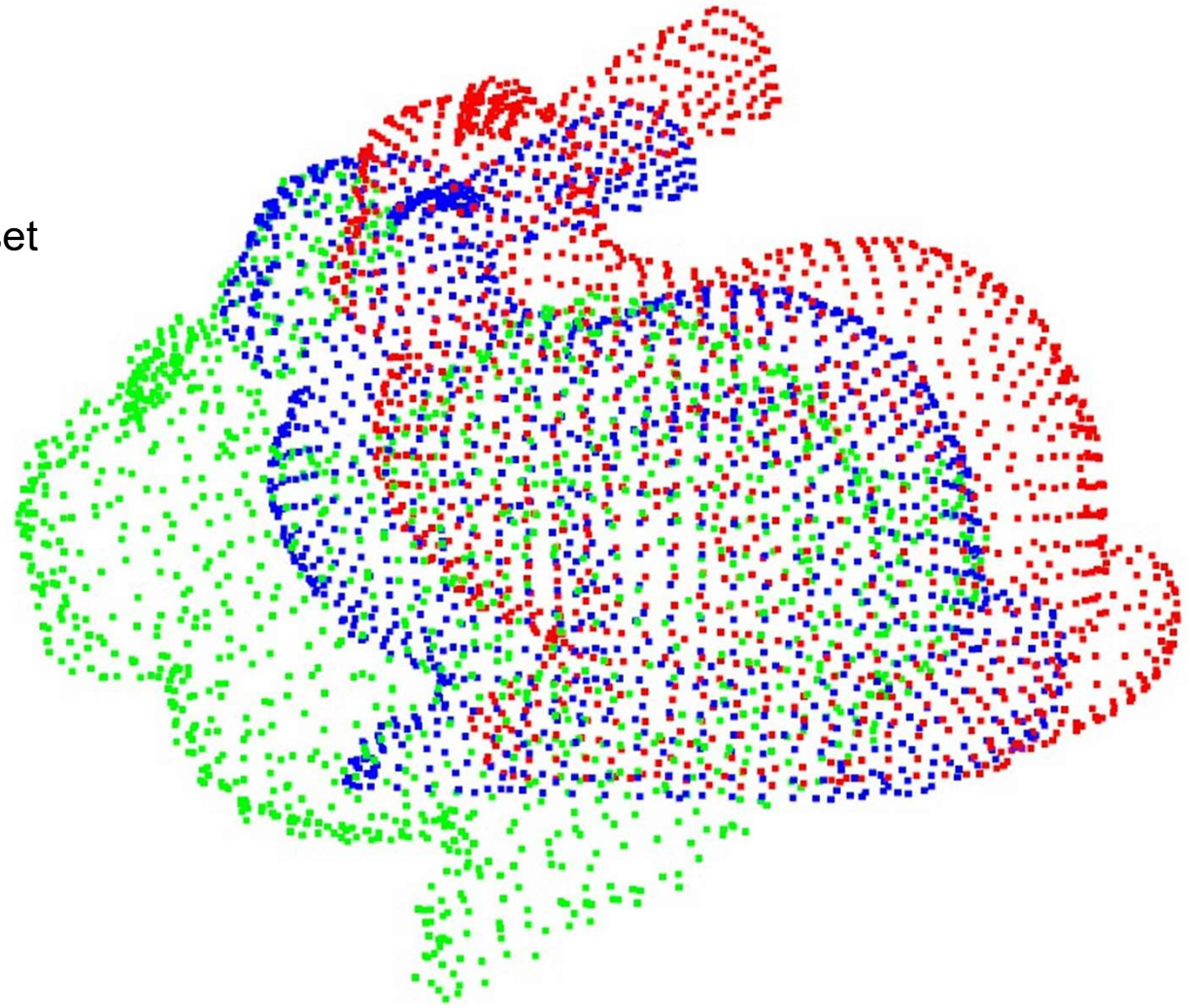
It is an iterative algorithm in which we continuously find better rotation matrix and translation matrix transforming source to target.



[1] P.J. Besl and Neil D. McKay. A method for registration of 3-d shapes. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 14:239–256, 1992

Iterative Closest Point (ICP)

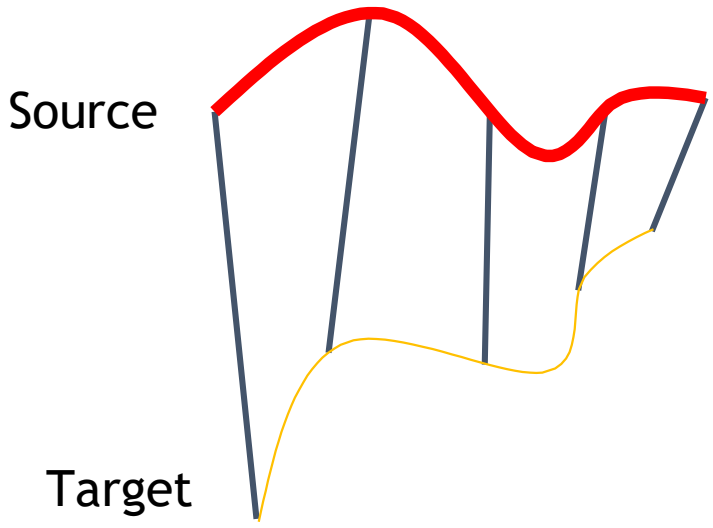
- Red: source point set
- Green: target point set
- Blue: iteratively updated point set



https://github.com/neka-nat/probreg

Iterative Closest Point (ICP)

- Let’s start from a simple condition:
- Assume the correspondences are known, how to find the best rotation and translation? (The correspondences indicate the point one-to-one matching function.)

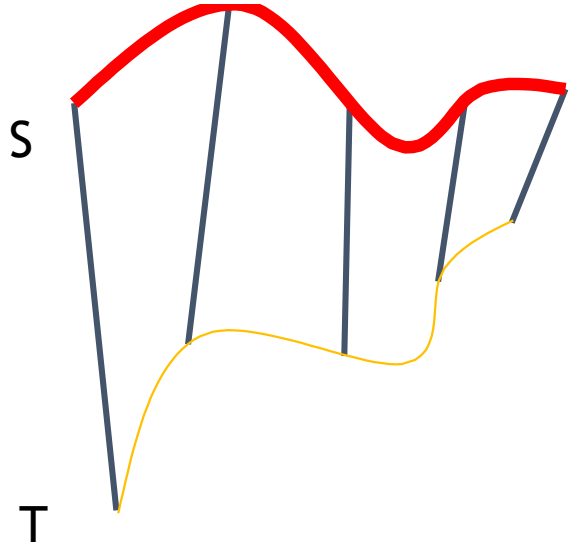


Iterative Closest Point (ICP)

- Let S be a source point set.
- Let T be a target point set.

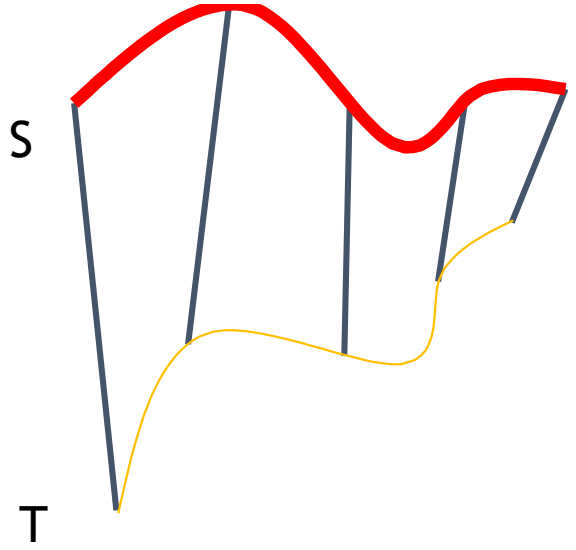
We assume:

1. $N_S = N_T$.
2. Each point S_i correspond to T_i.



Iterative Closest Point (ICP)

- Let S be a source point set.
- Let T be a target point set.



We assume:

1. $N_S = N_T$.
2. Each point S_i correspond to T_i.

❑ Iteration Process:

Step 1: Minimize the RMS objective function

$$R^*, t^* = argmin\left(\sqrt{\frac{1}{N_s}\sum_{i=1}^{N_s}\|T_i - (RS_i + t)\|^2}\right)$$

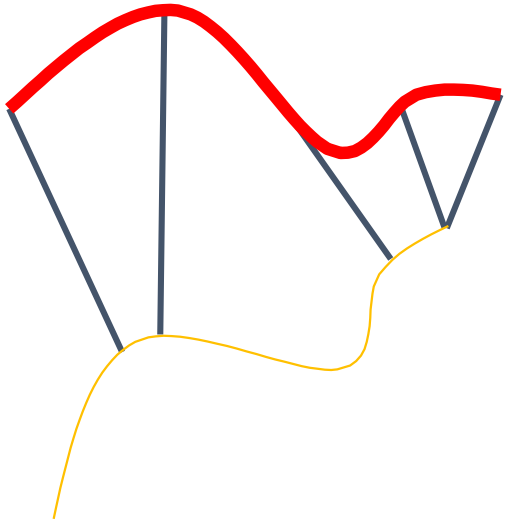
Step 2: Update the source point set

$$S_i{}^* = R^*S_i + t^*$$

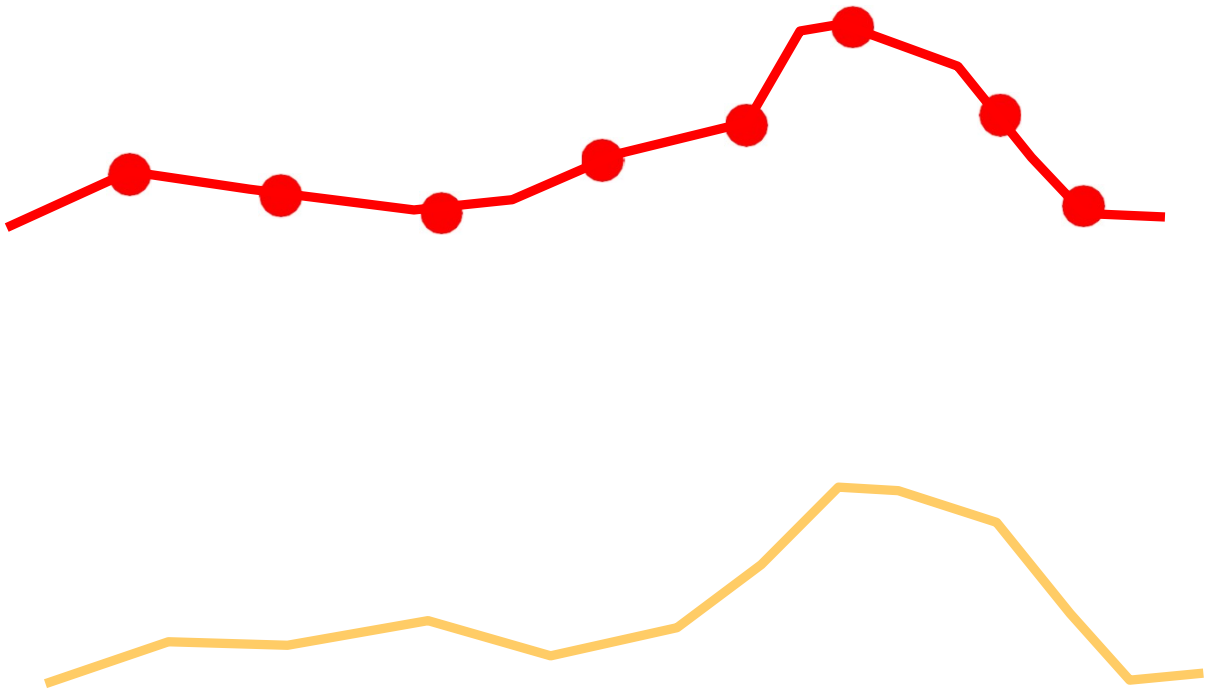
Step 3: Check the measure error until RMS is unchanged or we reach a certain number of iteration.

Iterative Closest Point (ICP)

- What if the correspondences are unknown? How to find the corresponding points?
- Previous systems are based on user input, feature matching, surface signatures, etc.
- In ICP, we define the correspondences based on the space distance between two points.

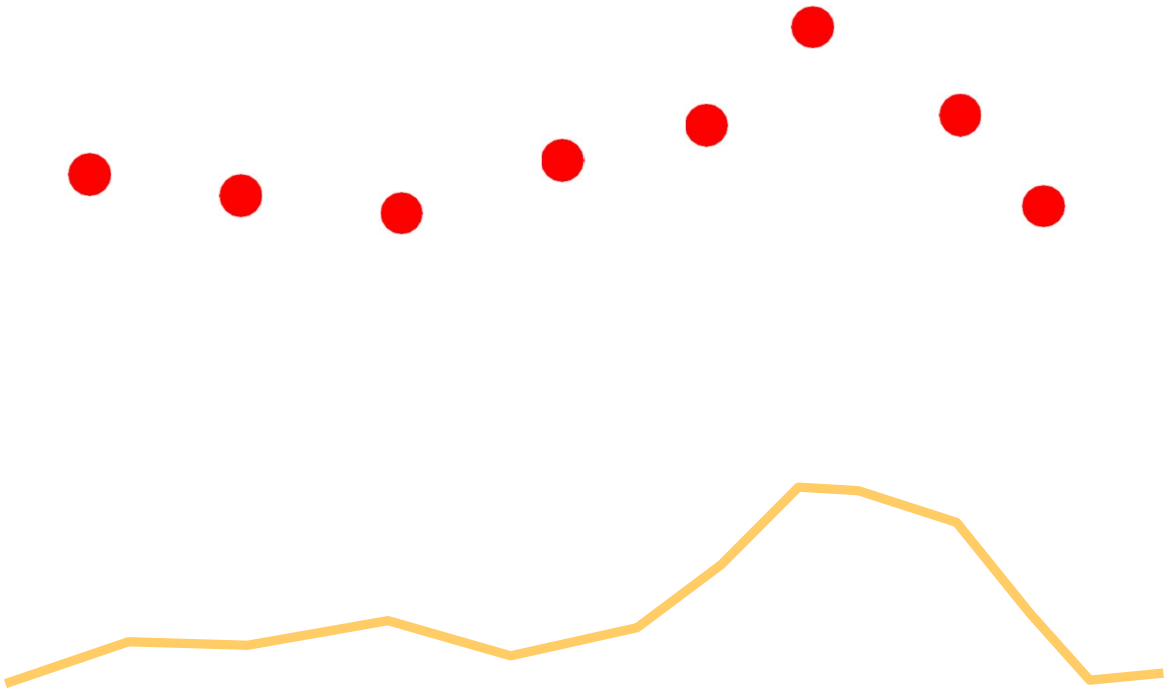


Iterative Closest Point (ICP)



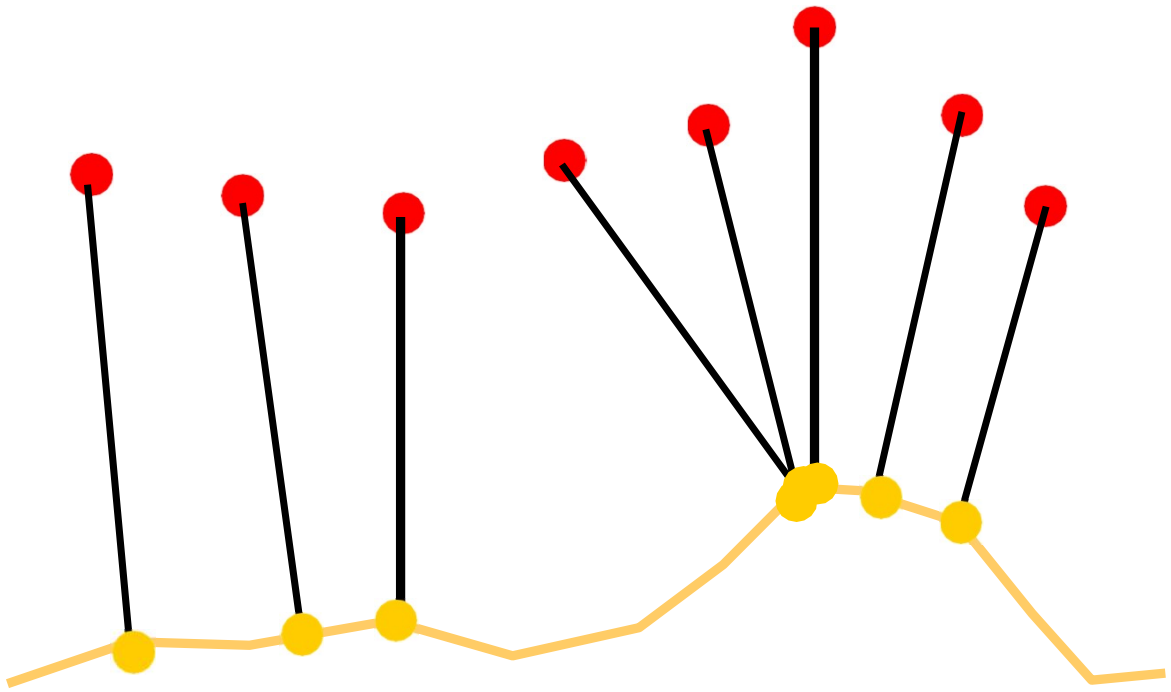
Sample N random points

Iterative Closest Point (ICP)



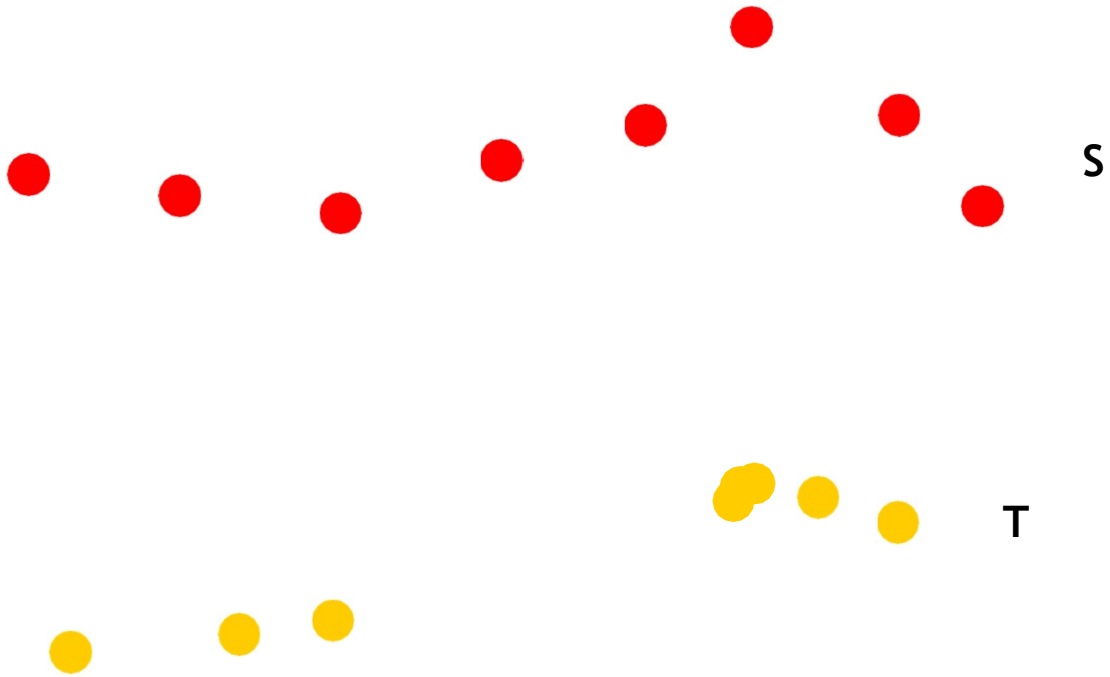
Points become a proxy for the source line

Iterative Closest Point (ICP)



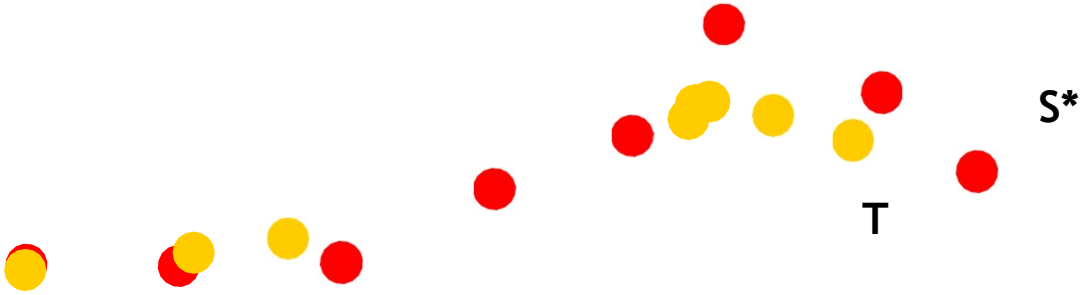
Match each point to the closest point of the target line

Iterative Closest Point (ICP)



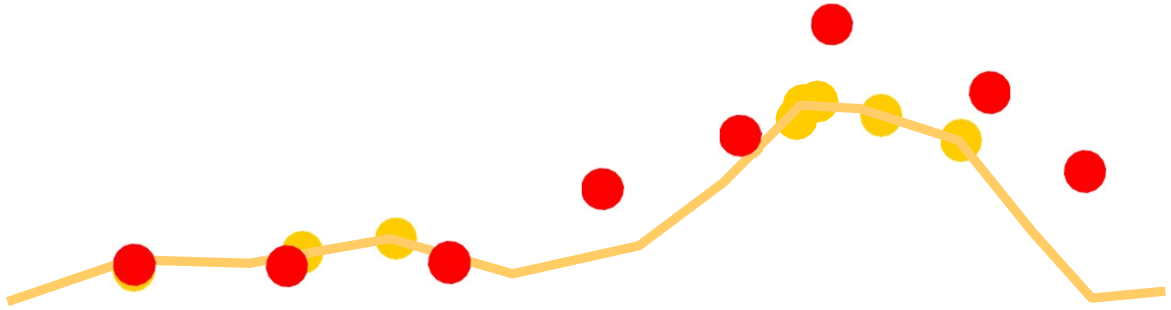
Estimate rigid motion (Rotation and translation) $E = \sum_{i=1}^{N_S} \|T_i - (RS_i + t)\|^2$

Iterative Closest Point (ICP)



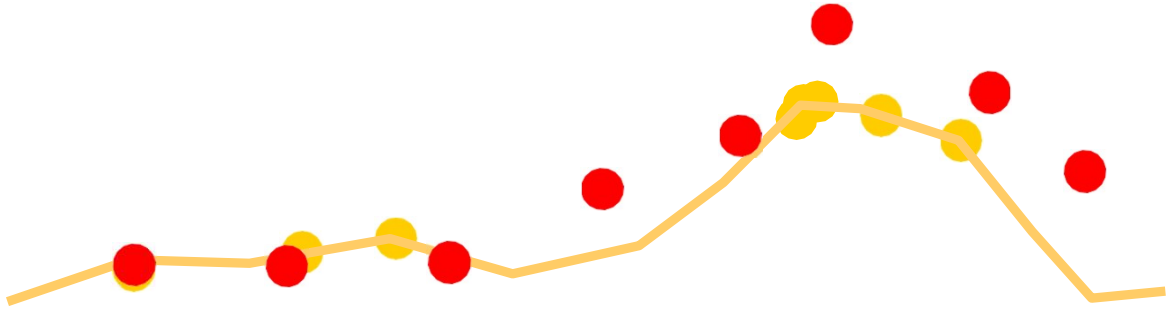
New positions of source points

Iterative Closest Point (ICP)



Restore target line

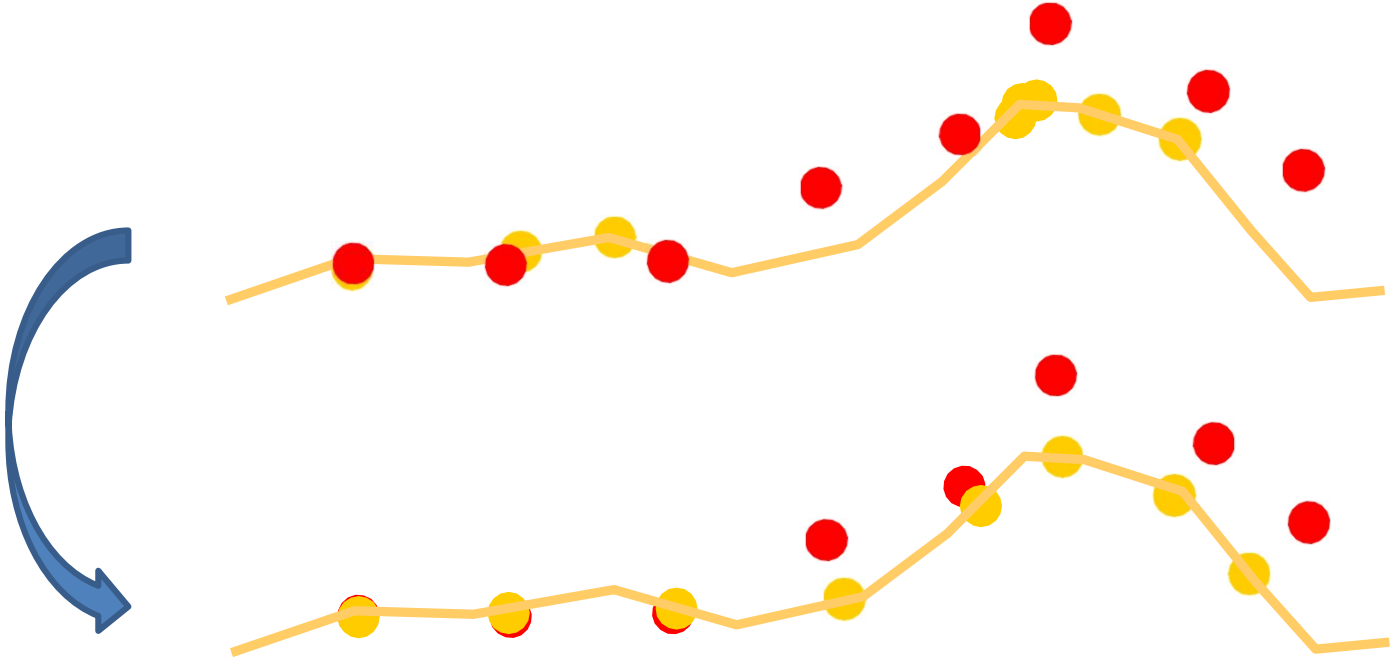
Iterative Closest Point (ICP)



Find (new) closest points

Iterative Closest Point (ICP)

Find (new) closest points



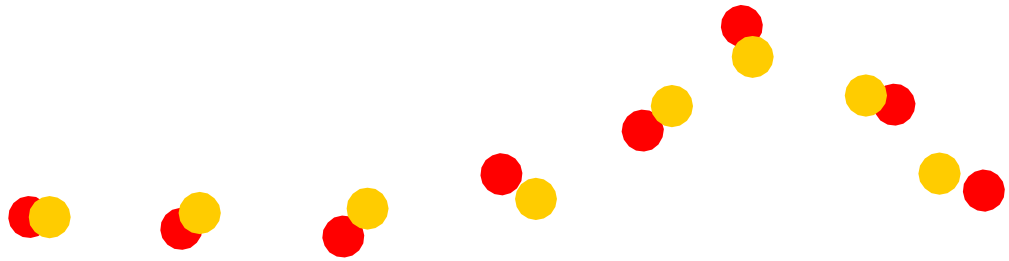
Iterative Closest Point (ICP)

Remove stationary line

Iterative Closest Point (ICP)

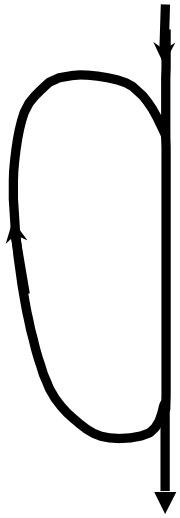
Estimate rigid motion (Rotation and translation) $E = \sum_{i=1}^{N_S} \|T_i - (RS_i + t)\|^2$

Iterative Closest Point (ICP)

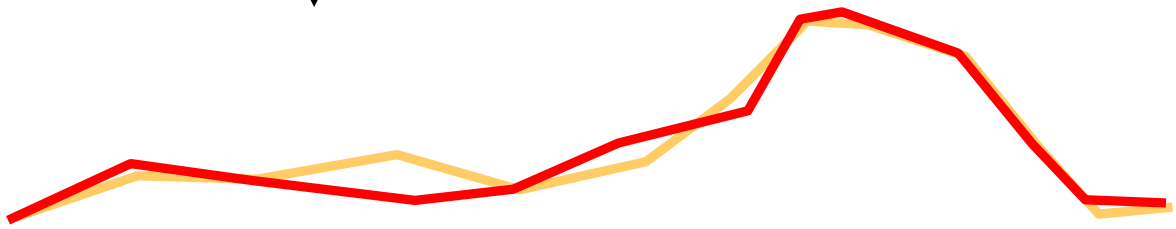


Update Points, and so on…

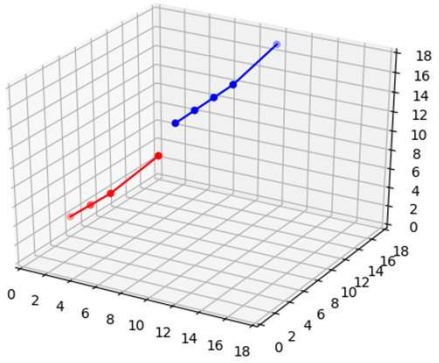
Iterative Closest Point (ICP)



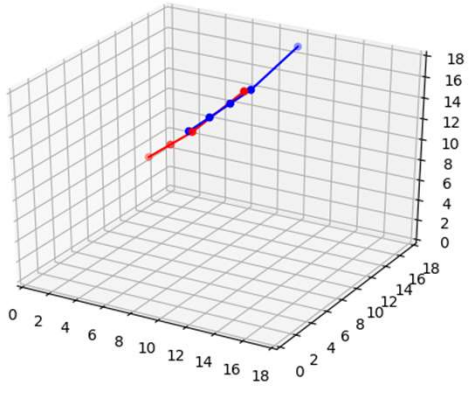
- **Find closest points and measure total distance**
- **Compute the rotation and translation, and apply this transformation to the source points**
- **Check the measure error until RMS is unchanged or we reach a certain number of iteration.**



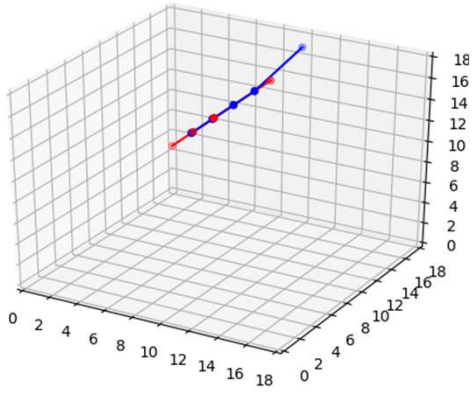
Different Initialized Position



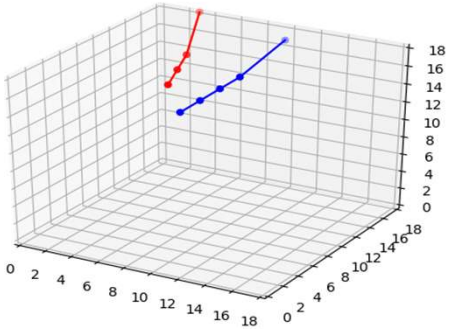
Initialization



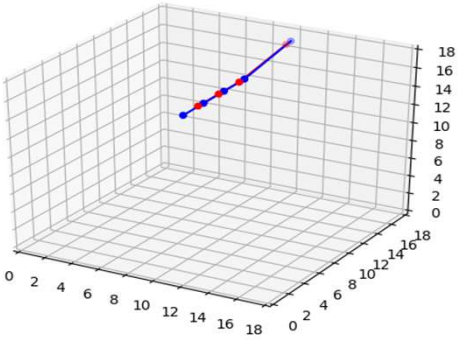
iter= 1



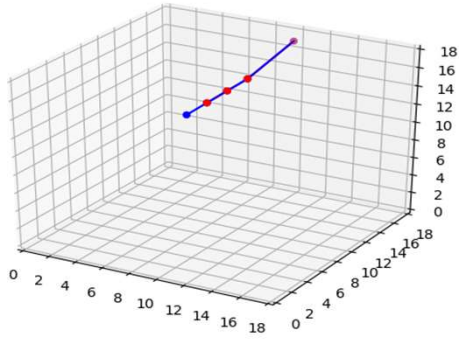
iter= 10



Initialization*



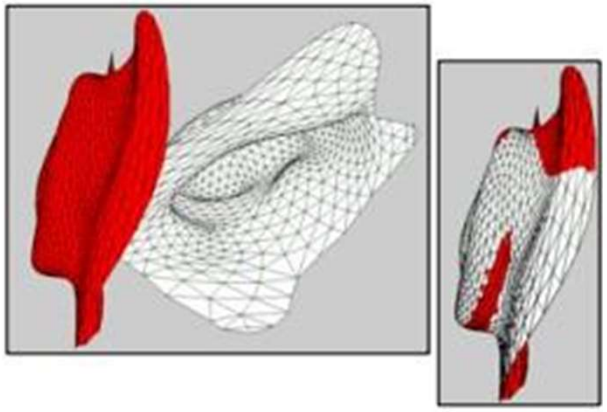
iter= 1



iter= 10

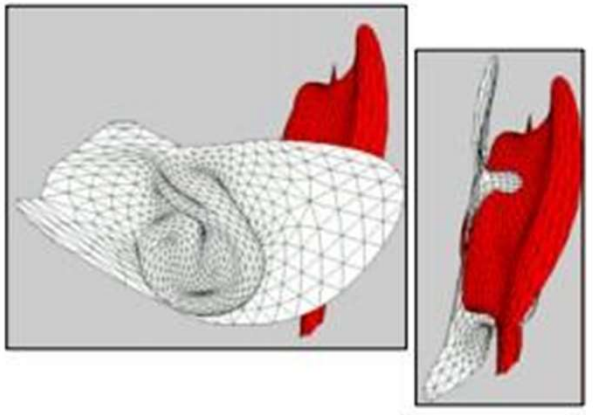
Iterative Closest Point (ICP)

Good initial guess



Converge to global minimum

Bad initial guess



Converge to local minimum

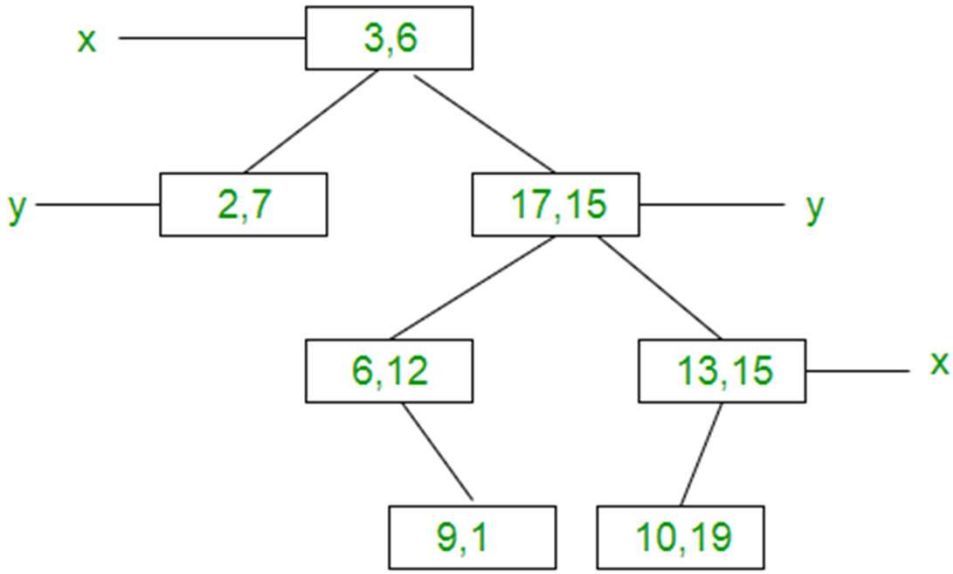
https://www.programmersought.com/article/48863656076/

Iterative Closest Point (ICP)

- Find closest points

  Use Kd-tree to speed up the process.

  Given t points (3, 6), (17, 15), (2, 7), (13, 15), (6, 12), (9, 1), (10, 19)



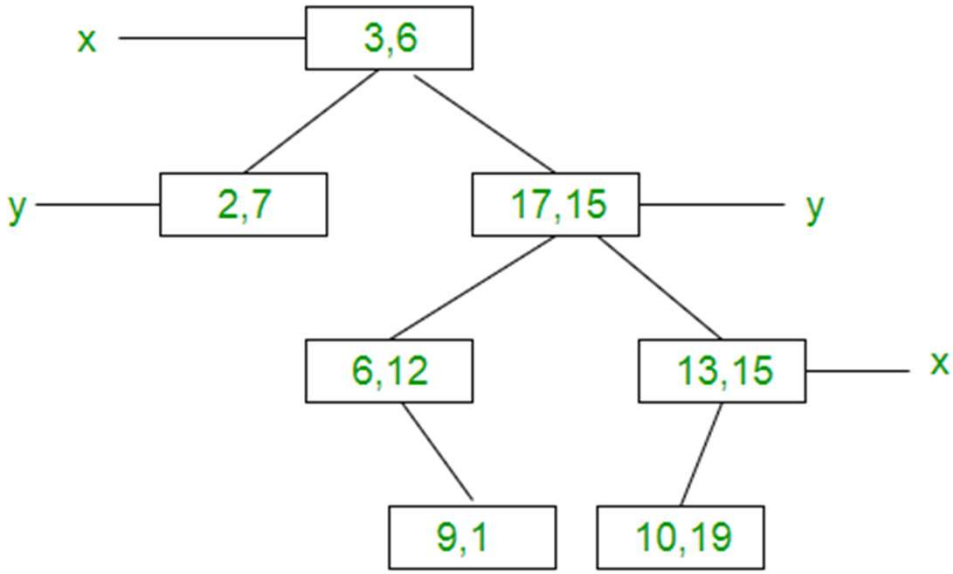
https://www.geeksforgeeks.org/k-dimensional-tree/

Iterative Closest Point (ICP)

- Find closest points

  Use Kd-tree to speed up the process.

  Given t points (3, 6), (17, 15), (2, 7), (13, 15), (6, 12), (9, 1), (10, 19)



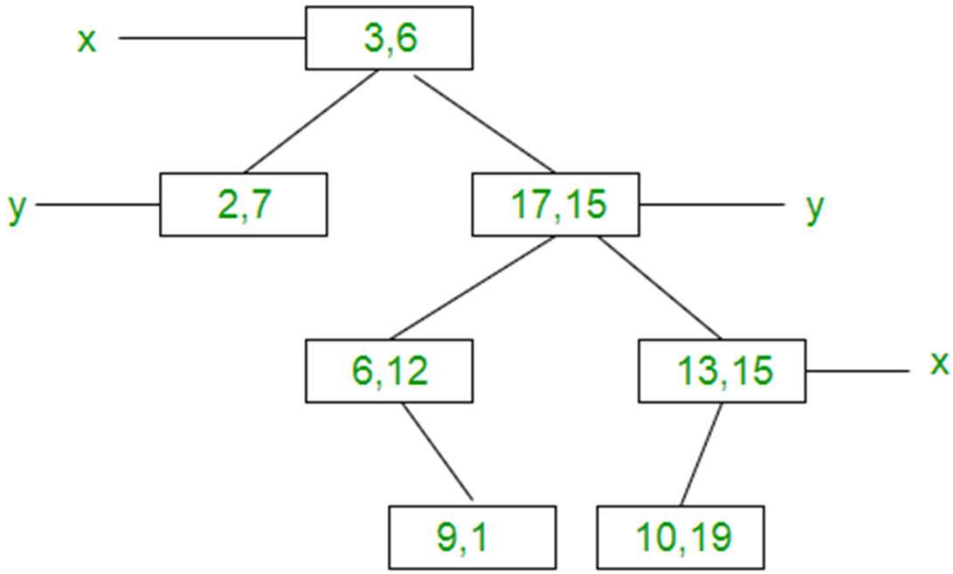
https://www.geeksforgeeks.org/k-dimensional-tree/

Iterative Closest Point (ICP)

- Find closest points

  Use Kd-tree to speed up the process.

  Given t points (3, 6), (17, 15), (2, 7), (13, 15), (6, 12), (9, 1), (10, 19)



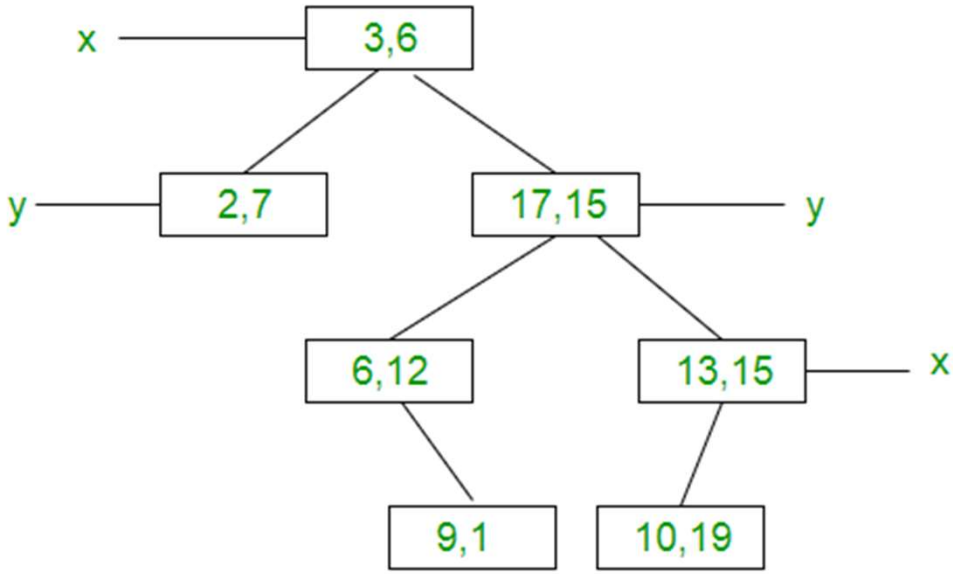
https://www.geeksforgeeks.org/k-dimensional-tree/

Iterative Closest Point (ICP)

- Find closest points

  Use Kd-tree to speed up the process.

  Given t points (3, 6), (17, 15), (2, 7), (13, 15), (6, 12), (9, 1), (10, 19)



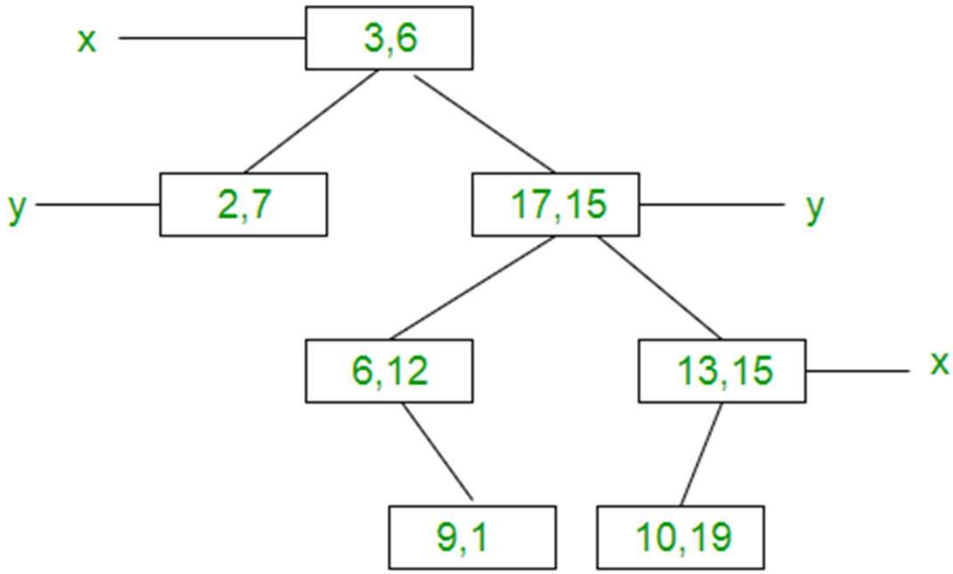
https://www.geeksforgeeks.org/k-dimensional-tree/

Iterative Closest Point (ICP)

- Find closest points

  Use Kd-tree to speed up the process.

  Given t points (3, 6), (17, 15), (2, 7), (13, 15), (6, 12), (9, 1), (10, 19)



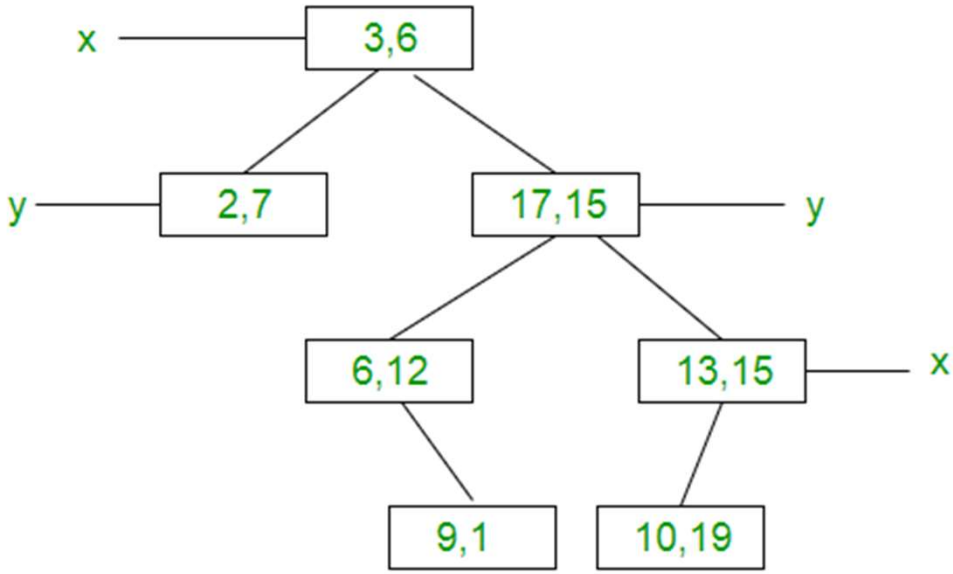
https://www.geeksforgeeks.org/k-dimensional-tree/

Iterative Closest Point (ICP)

- Find closest points

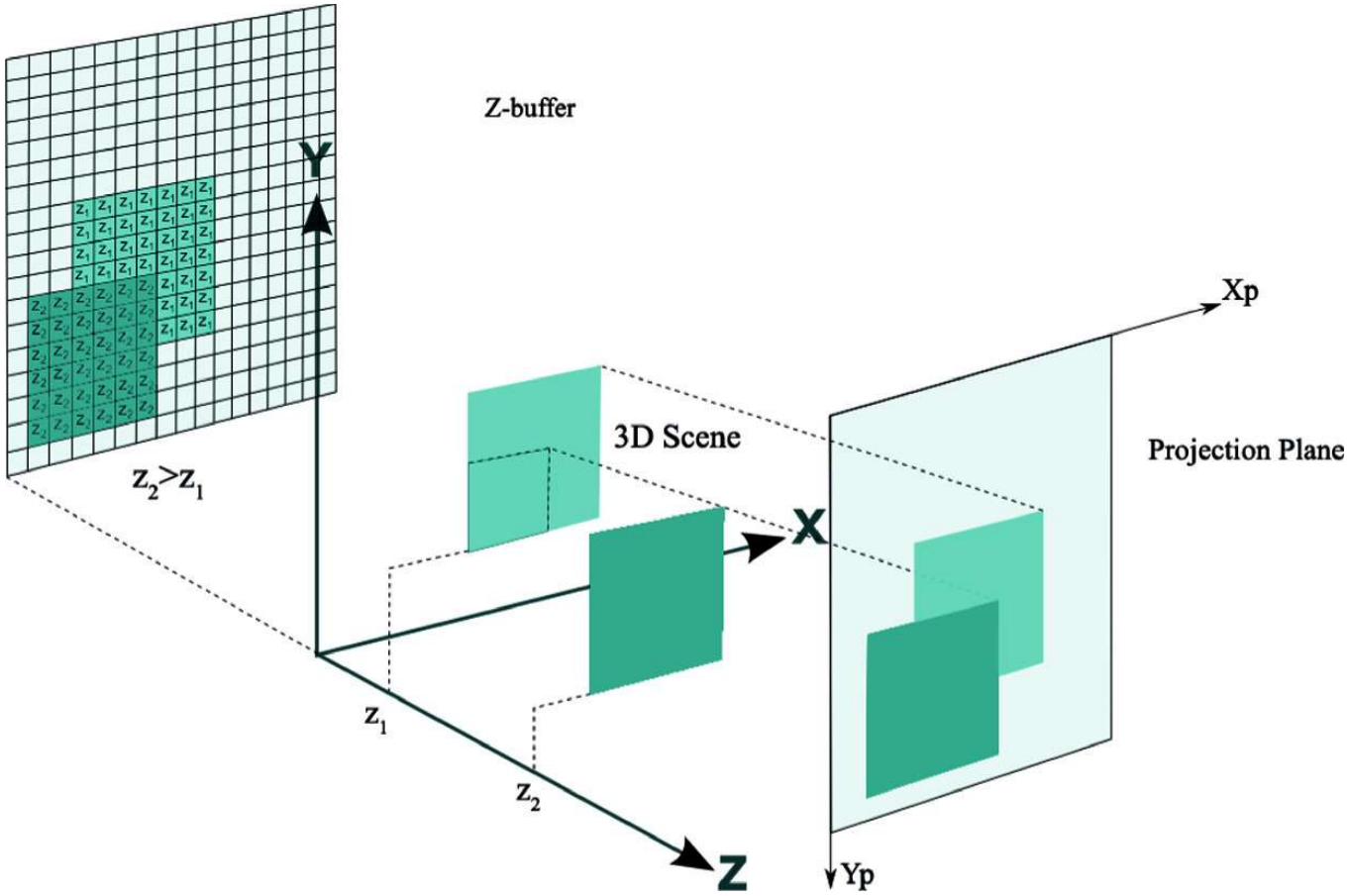
  Use Kd-tree to speed up the process.

  Given t points (3, 6), (17, 15), (2, 7), (13, 15), (6, 12), (9, 1), (10, 19)



https://www.geeksforgeeks.org/k-dimensional-tree/

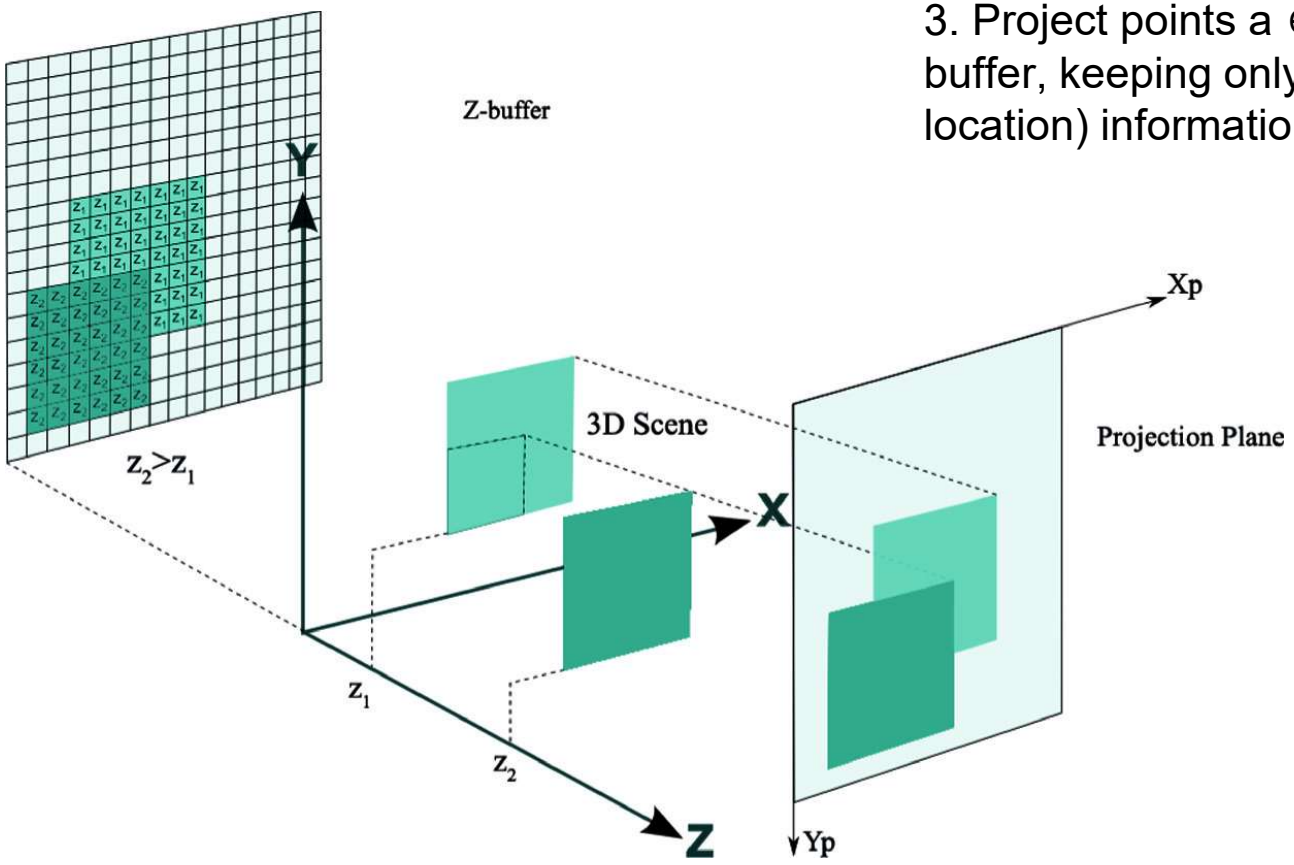
Speed up via z-buffer



[2] R. Benjemaa and F. Schmitt, “Fast global registration of 3d sampled surfaces using a multi-z-buffer technique,” Image and Vision Computing, vol. 17, no. 2, pp. 113–123, 1999.

Speed up via z-buffer

1. Take the union of A1 , A2 and determine the minimum enclosing box on the x,y-plane.
2. Divide the minimum enclosing box into H × W rectangular cells.
3. Project points a ∈ A1 orthogonally into the cells of the source buffer, keeping only the nearest one but saving all geometric (i.e. location) information. You thus keep a single point per cell.



[2] R. Benjemaa and F. Schmitt, "Fast global registration of 3d sampled surfaces using a multi-z-buffer technique," Image and Vision Computing, vol. 17, no. 2, pp. 113–123, 1999.

Speed up via z-buffer

4. Project points a ∈ A2 orthogonally into the cells of the target buffer, keeping only the nearest one but saving all geometric (i.e. location) information.
5. For each cell at h, w in the source buffer, obtain the closest point in the target buffer by comparing it to the points in an m × m window centered at h, w.

Speed up via z-buffer

4. Project points a ∈ A2 orthogonally into the cells of the target buffer, keeping only the nearest one but saving all geometric (i.e. location) information.
5. For each cell at h, w in the source buffer, obtain the closest point in the target buffer by comparing it to the points in an m × m window centered at h, w.

Iterative Closest Point Variants

- Variants on the following stages of ICP have been proposed [3]:

1. Selecting source points (from one or both meshes)
2. Matching to points in the other mesh
3. Assigning an error metric to the current transform
4. Minimizing the error metric

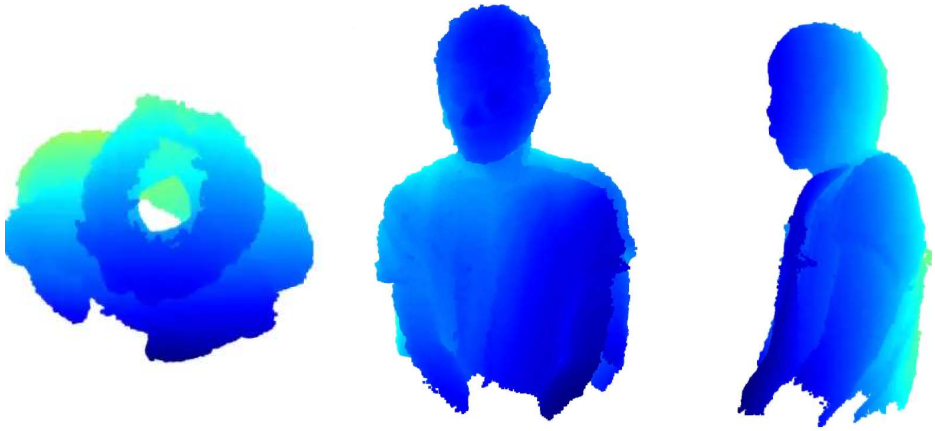
[3] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," *in Third International Conference on 3D Digital Imaging and Modeling (3DIM)*, 2001.

Assignment 1 requirements

Given series of adjacent frames
(point cloud)

R
G
B

Expected output



Reference

[1] P. Besl and N. D. McKay, “A method for registration of 3-d shapes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 239–256, 1992.

[2] R. Benjemaa and F. Schmitt, “Fast global registration of 3d sampled surfaces using a multi-z-buffer technique,” Image and Vision Computing, vol. 17, no. 2, pp. 113–123, 1999.

[3] S. Rusinkiewicz and M. Levoy, “Efficient variants of the ICP algorithm,” *in Third International Conference on 3D Digital Imaging and Modeling (3DIM)*, 2001.

End