ECE661: Homework 7

Fall 2016

Deadline : November 1, 2016 , 1:30 pm

Turn in your solution via Blackboard. Additional instructions given at [I].

1 Introduction

This homework will introduce you to the notions of depth images, point clouds, and a very popular image registration technique called Iterative Closest Point (ICP) algorithm [II]. Although this homework is an addition to the set of homeworks that were previously assigned for this course, and no previous submissions are available on the course webpage, I have described the ICP algorithm here in sufficient detail so that you can do the homework without much difficulty. The basic concepts that go into ICP have already been covered in class.

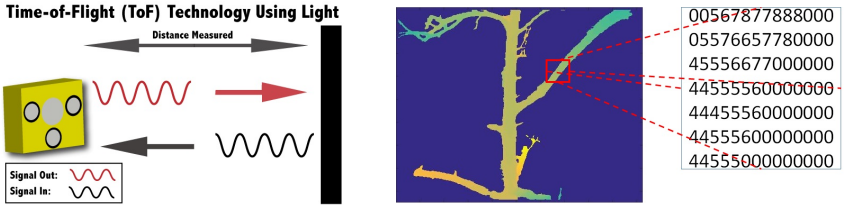
2 Background

2.1 Depth image

The most common form of images is the one we can capture from our regular cameras in which each pixel of the foreground object and the background clutter in the scene is represented by its RGB value. But using more sophisticated cameras we can also capture another not-so-common form of images which are called "depth images". In depth images instead of three channel RGB values each pixel is represented by the distance or depth of the object from the camera or sensor.

One of the most popular depth sensors is the Microsoft Kinect 2 sensor. This sensor works on the principle of Time-of-Flight, in which a light beam is thrown into the scene which is reflected from the object and the time taken by the beam to hit the object and return back to the sensor is used to calculate the distance of the object from the sensor. This is illustrated in Figure 1a.

Two depth images of a leafless tree captured from slighlty different viewpoints using Kinect 2 sensor are provided at the course webpage [I]. The visualization of one of those two depth images is given in Figure 1b. The numbers shown in that figure are the depth values for a patch in the image. For example, the depth value at position (row, col) = (1, 3) of the image patch is 5 which means that the object present at this pixel location is 5 meters away from the sensor. Note that the blue color in the image indicates the background and has pixel value equals 0.

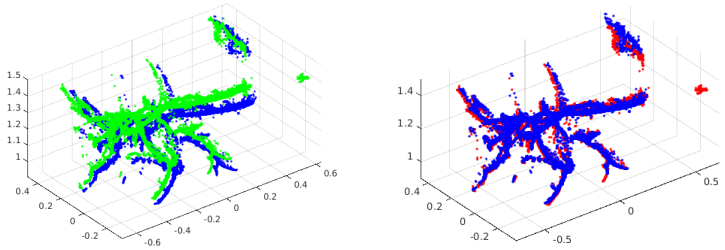


(a) (b)

Figure 1: Illustration of ToF principle and a sample depth image

2.2 Point Cloud

A point cloud is a set of data points in some coordinate system. We are interested in the 3-D point cloud representation of objects in our scene. In 3-D coordinate system, these points are defined by X, Y, and Z coordinates, and are often intended to represent the external surface of an object. For example, if a 3-D point cloud is given as $\{(1.2, 2.3, 1.8), (1.4, 2.1, 1.9), (1.1, 2.1, 2.0), (3.1, 2.0, 1.1)\}$, then this means that there are four points in the point cloud located at their respective (x^p, y^p, z^p) coordinates.



(a) before alignment (b) after alignment

Figure 2: Illustration of ICP alignment on pair of point clouds

The conversion of depth image to a 3-D point cloud representation is very straightforward. Let $\mathbf{u} = (x, y, 1)$ be the homogeneous coordinate representation of an image pixel coordinate (x, y). Also let $\mathbf{D}(\mathbf{u})$ be the depth value at $\mathbf{u}$, and $\mathbf{K}$ represents the "intrinsic camera matrix" of the Kinect 2 sensor. You will learn about the "intrinsic camera matrix" this Thursday. For now, just use it as described below. The values for the 3×3 $\mathbf{K}$ matrix are provided in Section 4.2.

Then, the 3-D point $\mathbf{P(u)}$ or the (x^p, y^p, z^p) coordinates of the point in the point cloud corresponding to $\mathbf{u}$ is given by:

$$\mathbf{P(u)} = (x^p, y^p, z^p) = \mathbf{D(u)K^{-1}u} \tag{1}$$

Note that there are several 3-D visualization functions [III], [IV], [V] that you can use for visualizing the point cloud. Two such point clouds are shown in Figure 2a with different colors.

2.3 ICP Algorithm

Let $\mathbf{P}$ and $\mathbf{Q}$ be two point clouds constructed for the same object from two different viewpoints. Each point cloud constructed in its own frame of reference. We now wish to transform $\mathbf{Q}$ so that it exists in $\mathbf{P}$'s frame of reference. The transformation $[R, t]$ that makes the common portions of $\mathbf{P}$ and $\mathbf{Q}$ congruent is unknown. The goal of ICP is to find this transformation. Following are the steps that you need to follow in order to implement ICP algorithm:

Step 1:

The first step in ICP is to find the points in $\mathbf{P}$ that have the least Euclidean distance to the points in $\mathbf{Q}$. You can think of it as obtaining the correspondence pairs except that instead of images, you now have point clouds. Note that if the distance is greater than some threshold δ, then we reject the correspondence pair. You can set $\delta = 0.1$ meters.

As a result you will get point clouds $\mathbf{P'}$ and $\mathbf{Q'}$ which have dimensions $3 \times N$ where N is the total number of correspondence pairs. Note that the points are arranged in the point clouds such that the ith point in the array of $\mathbf{P'}$ has the closest corresponding point at the ith position in the array of $\mathbf{Q'}$.

Step 2:

The next step is to estimate the rotation and translation matrices which can transform the point cloud $\mathbf{Q}$ to frame of reference of $\mathbf{P}$.

For this, we first calculate the centroid of the two point clouds by summing over all the points in the point cloud and dividing by N. Therefore, $\mathbf{P}_c = \sum_i^N \frac{\mathbf{P'}(i)}{N}$ and $\mathbf{Q}_c = \sum_j^N \frac{\mathbf{Q'}(j)}{N}$.

Now subtract the centroid from all the points in the point cloud. The resulting point clouds are called $\mathbf{M_P}$ and $\mathbf{M_Q}$ and are calculated as follows: $\mathbf{M_P} = \{\mathbf{P'}(i) - \mathbf{P}_c\}_{i=1}^N$ and $\mathbf{M_Q} = \{\mathbf{Q'}(i) - \mathbf{Q}_c\}_{j=1}^N$. Note that every point $\mathbf{P'}(i)$ and $\mathbf{Q'}(j)$ has three coordinates X, Y, and Z. Therefore, the point clouds $\mathbf{M_P}$ and $\mathbf{M_Q}$ have dimensions equal $3 \times N$.

Next, we need to compute the correlation matrix C which is very straightforward and is given as the product of the two point clouds: $C = \mathbf{M_Q} * \mathbf{M_P}^\mathbf{T}$. Note that C is a 3×3 matrix.

We can now estimate rotation matrix R using SVD on the correlation matrix C. Specfically, if applying SVD on C gives $C = U \sum V^T$ then the matrix R is given by the product: $R = V * U^T$.

The translation vector t is defined as a difference between $\mathbf{P}_c$ and rotated $\mathbf{Q}_c$: $t = \mathbf{P}_c - R * \mathbf{Q}_c$.

Step 3:

The third step is to form a transformation matrix using the R and t matrices found in the previous step and apply the transformation to $\mathbf{Q}$. The 4×4 transformation matrix is given by:

$$T = \begin{pmatrix} R & t^T \\ 0 & 1 \end{pmatrix}$$

Transform the model point cloud $\mathbf{Q}$ using the transformation matrix T. The transformed $\mathbf{Q}$ is given by: $\mathbf{Q}_{trans} = T * \mathbf{Q}$. You will need to convert the points in $\mathbf{Q}$ into homogeneous coordinates. Also, note that you need to apply the transformation to the original point cloud $\mathbf{Q}$ and not to $\mathbf{Q}$' which only contains the points of $\mathbf{Q}$ which are 'very close' to points of $\mathbf{P}$.

Step 4:

Iterate the steps 1, 2, and 3 using the original point cloud $\mathbf{P}$ and the transformed point cloud $\mathbf{Q}_{trans}$ until the maximum number of iterations M has reached. You can empirically set $M = 20$. Note that you will need to convert $\mathbf{Q}_{trans}$ from homogeneous coordinate to physical coordinate representation.

For more information on ICP, please refer to [VI].

3 Goal of this homework

Let $\mathbf{P}$ and $\mathbf{Q}$ be the point cloud representations of the two depth images captured by a sensor from two slightly different viewpoints. Then, the goal of ICP algorithm is to register the two point clouds. By point cloud registration, we simply mean that we want to align the two point clouds by applying a transformation $[R, t]$ to one of them such that it transforms into the coordinate frame of the other point cloud.

Visual illustrations of point clouds before and after alignment are given in Figures 2a and 2b. In Figure 2a, the blue and the green points represent the two different point clouds before alignment. While in Figure 2b, the blue and the red points represent those two point clouds after alignment. Note that the red point cloud of Figure 2b is the transformed version of the green point cloud of Figure 2a.

4 Tasks

4.1 Load depth images

For the purpose of this homework you are provided with two depth images of a tree in TXT file format. You can load the images into the computer using regular TXT file reading functions. If you are a MATLAB user you can use the *dlmread*() function. Note that the depth image, as stored in the TXT file will be in the form of a 424×512 dimensional array. Also note that the elements of this array have floating point values. For visualizing the array of the depth image you can use *imagesc*() function in MATLAB. Load, visualize and include the two depth images in your report.

4.2 Convert depth images to point cloud

You need to convert the depth images into their point cloud representations. You can use Eq. (1) to obtain the point cloud representation. The "intrinsic camera matrix" for Kinect 2 sensor is given as:

$$\mathbf{K} = \begin{pmatrix} 365 & 0 & 256 \\ 0 & 365 & 212 \\ 0 & 0 & 1 \end{pmatrix}$$

After forming the two point cloud representations plot them using the library functions [III], [IV], [V] and include the output in your report. Note that you need to plot the two point clouds in the same figure as shown in Figure 2a. This is the plot before ICP alignment. Notice that the two point clouds are plotted with different colors to make sure that they are visually distinguishable.

4.3 Apply ICP algorithm to register two point clouds

Follow the steps mentioned in the Section 2.3 to implement your own ICP algorithm to register the two point clouds of the tree. Apply the ICP algorithm on the two point clouds that you obtained in Task 4.2. Plot the point clouds after ICP alignment. This plot will have the two point clouds merged

together as shown in Figure 2b. Again choose different color for plotting the two point clouds. Include the plot you obtain after alignment in the report.

5 Submission

1. Turn in a typed pdf of your report via Blackboard.
2. Your pdf must include
   - A description of your implementation of ICP algorithm and how you applied it to register the point clouds.
   - A description of how you converted the depth images to point clouds.
   - Your observations on the performance of your ICP algorithm.
   - The depth images, point clouds before alignment, and point clouds after alignment.
   - Your source code.

References

[I] `http://web.ics.purdue.edu/~sakbar/ECE661/`
[II] Besl, Paul J.; N.D. McKay (1992). "A Method for Registration of 3-D Shapes". IEEE Trans. on Pattern Analysis and Machine Intelligence. Los Alamitos, CA, USA: IEEE Computer Society. 14 (2): 239256.
[III] `http://pointclouds.org/`
[IV] `http://matplotlib.org/mpl_toolkits/mplot3d/tutorial.html`
[V] `https://www.mathworks.com/help/matlab/ref/scatter3.html`
[VI] `https://engineering.purdue.edu/kak/distICP/ICP-2.0.html`