Computer Vision 2 – Assignment 1
Iterative Closest Point - ICP

Partha Das, Wei Zeng

April 3, 2020

Deadline: 19-04-2020, 23:59:59 (Amsterdam time)

General guidelines

Students should work on the assignments in **groups of 2**. Students are supposed to work on this assignment for two weeks. Some minor additions and changes might be done during these three weeks. Students will be informed for these changes via Canvas announcements.
Any questions regarding the assignment content can be discussed on Canvas Discussions (Let us know if you don't have access to it).
For this assignment, you can choose your prefered language for implementation, however support will be provided **only for Matlab**. In either cases, please provide requirements.txt with all external dependencies listed and README.txt file with **clear descriptions on how to reproduce the results**.
Source code and report must be handed in together in a zip file (ID1_lastname1_ID2_lastname2.zip) before the deadline by uploading through the Canvas submission system. For full credit, make sure your report follows these guidelines:

- Include an introduction and a conclusion to your report.
- The maximum number of pages is 10 (single-column, including tables and figures). Please be concise. The number of words does not necessarily correlate with how well you understand the concepts.
- Answer all given questions. Briefly describe what you implemented. Motivate your choices where applicable. Reports comprising of solely results will have the grades lowered (Avoid reports like, "Result 1 - check code outputs").
- Try to understand the problem as much as you can. When answering a question, give evidences (qualitative and/or quantitative results, references to papers, etc.) to support your arguments.
- Analyze your results and discuss them, e.g. why algorithm A works better than algorithm B in a certain problem.
- Tables and figures must be accompanied by a brief description. Do not forget to add a number, a title, and if applicable name and unit of variables in a table, name and unit of axes and legends in a figure.

Late submissions are not allowed. Assignments that are submitted after the strict deadline will not be graded. In case of submission conflicts, TAs' system clock is taken as reference. We strongly recommend submitting well in advance, to avoid last minute system failure issues.
Plagiarism note: Keep in mind that plagiarism (submitted materials which are not your work) is a serious crime and any misconduct shall be punished with the university regulations. For the assignment (and all subsequent future assignments), you are allowed to discuss on the respective Canvas Discusssion board. You are encouraged to discuss with your class mates regarding the problem. However, this should NOT include any code snippets.
Points: Maximum points are mentioned in the section heading. There are **no bonus points**. Total possible points in this assignment is **30 points**.

1 Data

README.txt file attached to this assignment contains the information about the provided data.

2 Iterative Closest Point - ICP (12 points)

ICP algorithm was first introduced by Besl and McKay (See [1] for details). Given two point-clouds A_1(base) and A_2(target), ICP tries to find a spatial transformation that minimizes the distance (e.g. Root Mean Square (RMS)) between A_1 and A_2 which can be defined as

$$RMS(A_1, A_2, \psi) = \sqrt{\frac{1}{n} \sum_{a \in A_1, b \in A_2} \|a - \psi(b)\|^2} \quad , \tag{1}$$

$$\min_{\psi: A_1 \to A_2, t \in \Re^d} \sum_{a \in A_1, b \in A_2} \|Ra - t - \psi(b)\|^2 \quad . \tag{2}$$

where R and t are the rotation matrix and the translation vector in d dimensions, respectively. ψ is one-to-one matching function that creates correspondences between the elements of A_1 and A_2. R and t, that minimize the above equation, are used to define the camera movement between A_1 and A_2.

2.1 Implementation

In this assignment, students will implement the ICP algorithm and some of its variants to estimate the camera poses for given point-clouds (*#####.pcd*). First, the basic ICP algorithm will be implemented:

1. Initialize $R = I$ (identity matrix), $t = 0$.
2. Transform the point cloud with R and t.
3. Find the closest points for each point in the base point set (A_1) from the target point set (A_2) using brute-force approach.
4. Refine R and t using Singular Value Decomposition (Please check `https://igl.ethz.ch/projects/ARAP/svd_rot.pdf` for details).
5. Go to step 2 unless RMS is unchanged.

There are different ways to improve the efficiency and the accuracy of ICP. Some of these techniques are discussed in [3]. In this assignment, students should also analyze various aspects such as (a) accuracy, (b) speed, (c) stability and (d) tolerance to noise by changing the point selection technique. Using (a) all the points, (b) uniform sub-sampling, (c) random sub-sampling in each iteration and (d) sub-sampling more from informative regions can be used as the point selection technique. Students are expected to implement these variants and report their findings in the final report.

Note: You can use the provided data (source.mat and target.mat) to test the correctness of the estimated transformation matrix.

3 Merging Scenes

3.1 (5 points)

(a) Estimate the camera poses using two consecutive frames of given data (ICP implemented in Section 2.1). Using the estimated camera poses, merge the point-clouds of all the scenes into one point-cloud and visualize the result. Does the merging produce sufficient result? Discuss why.

(b) Now, estimate the camera pose and merge the results using every 2^{nd}, 4^{th}, and 10^{th} frames. For instance, in case of choosing the frame sampling rate to be 2 and given N frames ($frame_1$, $frame_2$, .., $frame_N$):

- First, you will need to estimate the camera pose of $frame_3$ (target) from $frame_1$ (base) and use it in merging $frame_1$ with $frame_3$ to get $frame_{1,3}$.
- Next, estimate the camera pose of $frame_5$ (target) from $frame_3$ (base) and use it to merge $frame_{1,3}$ with $frame_5$ to get $frame_{1,3,5}$ and so on until reaching $frame_N$.

Does the camera pose estimation change?

3.2 (5 points)

Iteratively merge and estimate the camera poses for the consecutive frames (point-clouds). Suppose there are N frames ($frame_1$, $frame_2$,. . ., $frame_N$):

- First, estimate the camera pose of $frame_2$ (target) from $frame_1$(base).
- Next, merge $frame_1$ and $frame_2$ into $frame_{1,2}$.
- Then, estimate the camera pose of $frame_3$(target) from $frame_{1,2}$(base) and use it to merge $frame_{1,2}$ and $frame_3$ into $frame_{1,2,3}$ and so on until reaching the final frame ($frame_N$).

Do the estimated camera poses change in comparison with the previous estimates (Section 3.1)? Does this estimation produce better results?

4 Questions (5 points)

1. What are the drawbacks of the ICP algorithm?
2. How do you think the ICP algorithm can be improved, beside the techniques mentioned in [3], in terms of efficiency and accuracy?

5 Additional improvements (3 points)

Feel free to improve the results of ICP using your suggestions to the questions above. You are free to explore other references outside those which are provided.

6 Self-Evaluation

Please briefly describe in an appendix to the report about the contributions of each team member to this assignment. Self-evaluation will be taken into consideration in cases where the contributions are significantly different.

7 Supplemental code

- For point cloud visualization fscatter3 `https://nl.mathworks.com/matlabcentral/fileexchange/2993-fscatter3-m` can be considered.
- readPcd.m is for reading provided pcd files from Matlab. Please check README.txt for more details. As a test please verify that

  ```
  A = readPcd("data/0000000038.pcd")
  ```

  returns a matrix of size 59596x4, but not something else.

8 Further Reading

For further exploration on the topic, please check A Symmetric Objective Function for ICP [2] `https://gfx.cs.princeton.edu/pubs/Rusinkiewicz_2019_ASO/symm_icp.pdf`

References

[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.

[2] Szymon Rusinkiewicz. A symmetric objective function for ICP. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 38(4), July 2019.

[3] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the ICP algorithm. In *Third International Conference on 3D Digital Imaging and Modeling (3DIM)*, 2001.