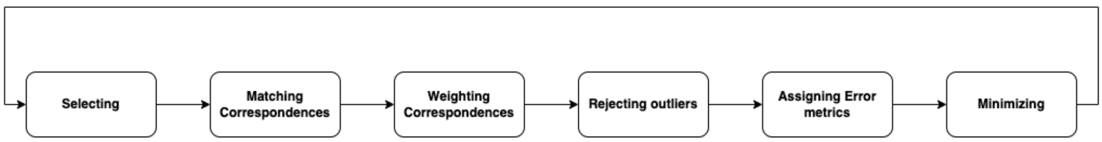
Project Proposal

"Analysis of the ICP Algorithm"

1 Abstract

The Iterative Closest Point (ICP) algorithm is a common choice for registering 3D scans, e.g. for mapping tasks in robotics. It was proposed in 1992 by Besl and McKay [1] and many variations have been developed since. Starting with an initial guess for the transform, the ICP algorithm iteratively refines the rotation and translation by repeatedly choosing pairs of corresponding points and minimizing an error metric.



It can be seen in the above figure, that the ICP algorithm has many steps, and a variation in one of those steps can affect the final result. In our project, we would like to explore the ICP algorithm and evaluate the speed and precision of different variants for real data. Our work has been mainly inspired by Rusinkiewicz and Levoy [2], who focused on the efficient variants of the ICP. In this work, we will not limit our analysis only to the efficient versions of the ICP. We will also compare more recent variants, like Symmetric-ICP [3], inspect how the color information can affect the precision of some steps, and lastly, test the effects of a new accelerated approach, Multi-Resolution ICP [4], combined with the popular ICP variants. More precisely, we will consider the following variations of the ICP algorithm:

1. **Selection Step:** Use all available points [1], or random sampling [5].
2. **Matching Step:** Find the closest point by using an accelerated data structure [6], or project the source point cloud onto the target mesh (which only works for structured data, e.g. RGB-D scans) [7].
3. **Weighting Step:** Constant weight, adding lower weights to pairs with greater point-to-point distances [2], weighting based on compatibility of colors [8], or weighting based on compatibility of normals [2].
4. **Rejecting Step:** Rejection of points based on a fixed distance apart, or rejection based on the angle between the normals.
5. **Error Metric Step:** The options are Point-to-Point ICP [1], Color ICP [9], Point-to-Plane ICP [10], and Symmetric ICP [3].
6. **Minimization Step:** Linear least squares, or non-linear least squares using the Levenberg-Marquardt method [11].

As an optional step, instead of only outputting the rotation, translation, and aligned point clouds at the end, we would like to visualize the point clouds at different iterations of the ICP using a live visualization. This could be a nice way to show how the point clouds are being aligned iteratively. One option is to use the Point Cloud Library [12], which offers ways to render point clouds.

2 Requirements

To compare the different variants of ICP, we need a dataset and our goal is to compare the performance for real-world data. Therefore, the Laser Registration Dataset from ETH Zurich [13] is one good option as it contains real scans of indoor and outdoor scenes. To make the usage of the datasets easier, and to potentially use additional datasets, we may use a small tool developed by Simone Fontana from the IRALab of the University of Milan [14]. It can transform scenes from different sources, including the ETH dataset, into a common format.

In addition, the required libraries are, the common Eigen [15] library for linear algebra calculations, Ceres [16] for the non-linear optimization step, along with FLANN [17] and glog [18] as well as FreeImage [19] and trimesh2 [20] for data processing.

3 Team

Florian Donhauser (Mat. Nr. 03695568), Cuong Ha (Mat. Nr. 03749410), Panagiotis Petropoulakis (Mat. Nr. 03751104), and Suren Sritharan (Mat. Nr. 03752962).

4 Milestones

- 13 Dec - 17 Dec: Adapt given ICP code to be modular (skeleton for variants); Prepare data and pipeline for speed and convergence analysis.
- 20 Dec - 24 Dec: Implement 2 selection, 1 matching, 2 weighting, 2 rejection, and 2 error metrics (using both minimization methods) steps.
- 27 Dec - 31 Dec: Implement Symmetric objective function and Projective variant of the matching step.
- 03 Jan - 07 Jan: Finish all variants apart from Color and Multi-Resolution ICP.
- 10 Jan - 14 Jan: Implement Color ICP and start doing some analysis.
- 17 Jan - 21 Jan: Implement Multi-resolution ICP; Start the optional live visualization.
- 24 Jan - 28 Jan: Complete analysis of speed, convergence, and output quality; Finish the optional live visualization, if possible.
- 31 Feb - 04 Feb: Start writing the report and prepare the presentation.

References

[1] P.J. Besl and Neil D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[2] Szymon Rusinkiewicz and Marc Levoy. Efficient Variants of the ICP Algorithm. In *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, 2001.

[3] Szymon Rusinkiewicz. A Symmetric Objective Function for ICP. In *ACM Transactions on Graphics*, 2019.

[4] Timothe Jost and Heinz Hgli. Multi-Resolution ICP with Heuristic Closest Point Search for Fast and Robust 3D Registration of Range Images. In *Proceedings of International Conference on 3-D Digital Imaging and Modeling*, 2003.

[5] Takahiko Masuda, Katsuhiko Sakaue, and Naokazu Yokoyaz. Registration and integration of multiple range images for 3-D model construction. In *Proceedings of 13th International Conference on Pattern Recognition*, 1996.

[6] David A. Simon and Takeo Kanade. *Fast and Accurate Shape-Based Registration.* PhD thesis, Carnegie Mellon University, USA, 1996. AAI9838226.

[7] Gerard Blais and Martin D. Levine. Registering multiview range data to create 3D computer objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):820–824, 1995.

[8] Guy Godin, Marc Rioux, and Rejean Baribeau. Three-dimensional registration using range and intensity information. In *Proceedings International Society for Optics and Photonics*, 1994.

[9] Andrew E. Johnson and Sing Bing Kang. Registration and integration of textured 3-D data. In *Proceedings International in 3-D Digital Imaging and Modeling*, 1997.

[10] Yang Chen and Gerard Medioni. Object modeling by registration of multiple range images. In *International Journal of Image and Vision Computing*, 1992.

[11] Jorge J. Mor. The Levenberg-Marquardt algorithm: Implementation and theory. In *Watson G.A. (eds) Numerical Analysis. Lecture Notes in Mathematics, vol 630. Springer, Berlin*, 1977.

[12] Radu Rusu and Steve Cousins. 3D is here: Point cloud library (PCL). *IEEE International Conference on Robotics and Automation 2011 (ICRA 2011)*, 05 2011.

[13] François Pomerleau, M. Liu, Francis Colas, and Roland Siegwart. Challenging data sets for point cloud registration algorithms. *The International Journal of Robotics Research*, 31(14):1705–1711, 12 2012.

[14] Simone Fontana, Daniele Cattaneo, Augusto L. Ballardini, Matteo Vaghi, and Domenico G. Sorrenti. A benchmark for point clouds registration algorithms. *Robotics and Autonomous Systems*, 140:103734, Jun 2021.

[15] Benot Jacob and Gal Guennebaud et al. Eigen. `https://eigen.tuxfamily.org/`.

[16] Sameer Agarwal and Keir Mierle et al. Ceres solver. `http://ceres-solver.org`.

[17] Marius Muja and David G. Lowe. Flann. `https://github.com/flann-lib/flann`.

[18] Google. Glog. `https://github.com/google/glog`.

[19] Freeimage. `https://freeimage.sourceforge.io`.

[20] Szymon Rusinkiewicz. trimesh2. `https://gfx.cs.princeton.edu/proj/trimesh2`.