Adaptive Iterative Closest Keypoint using ORB Features and Clustering

Johan Ekekrantz [1], Andrzej Pronobis[1], John Folkesson[1] and Patric Jensfelt[1]

***Abstract*— Add abstract. Bla bla bla bla.**

I. INTRODUCTION

Sensor data is essential to robotics. More sensors are producing better data at faster rates. At the same time the computational power of the robot is limited and not able to look at all the data it has collected when making control decisions. That data is therefore put into a representation that allows it to be more easily used. These invariably require so called data registration or placing the data taken at different locations into the same frame of reference. Data registration between RGBD frames is the focus of this paper.

Since the introduction of cheap RGBD cameras in 2010 these sensors have become very popular in the field of robotics. A lot of research has been done on data registration of 2D and 3D range data captured by laser scanners and sonars. There has been even more work on RGB camera image registration. The combining of RGB data with the depth, 'D', range data in one sensor introduces new challenges and new opportunities. The data registration problem is equivalent to finding the transformation between the sensor poses for each frame of sensor data.

For range data, either 2D slices or 3D point clouds, the most widely used method is the iterative closest point, *ICP*, algorithm [1]. This method requires an initial guess for the transformation and will not converge correctly if started too far away. ICP works on sets of geometric points in either 2D or 3D. It requires only the locations of the points relative to the sensor in each frame. From this it returns the transformation between the two sensor frames.

For RGB image data the registration can be done using image features or keypoints. The process is to first detect points of interest in the image. Then (optionally) compute a descriptor from the context (ie neighboring pixels) of the point. Registration is done by matching these 'keypoints' either by using the descriptors or the geometric constraints or both. As the camera only gives a bearing to the points the geometric constraints are more complicated than for range sensors.

Having RGBD sensors allows us to both use the simpler geometric registration and the selective power of the RGB descriptor. In [2] we presented the Adaptive Iterative Closest Keypoint (AICK) algorithm. This data registration algorithm uses an adaptive distance metric based on both visual similarity and geometric distance between keypoints to improve the registration. It thereby combines range ICP with camera keypoint based registration methods. In this paper we further investigate the AICK algorithm to see how one might relax some of the more computationally expensive parts without unduly sacrificing the quality of the registration. Our aim is an algorithm for real-time systems where both low computational load and high performance is a requirement.

The contributions of this paper are the investigation of the AICK method's sensitivity to using weaker keypoint features and in using a faster but not exhaustive search strategy for the keypoint matching. The method is described in detail in Section III-A.

To validate the performance of the AICK algorithm we compare it against state of the art methods for registration; generalized ICP (GICP) [3] and the 3D normal distribution transform (3D-NDT) [4]. We also compare several variations of the AICK to be described later. The evaluation is performed on a publicly available data set[1] [5] using an established benchmarking procedure and performance measure [5]. The results show a significant decrease in computational load while still maintaining performance close to that of our previous results for the AICK while outperforming both the GICP and the 3D-NDT.

In Section II we provide an overview of popular registration methods for 3D point clouds. Section III provides details of the AICK algorithm. Section III-A describes the contribution of this paper. Section IV covers the setup for the experimental evaluation. To round off the paper, we present the results of the experimental evaluation in Section V.

II. RELATED WORK

The Iterative Closest Point (ICP) algorithm introduced in [1] forms the basis for a large fraction of all point cloud registration methods. ICP starts from a transformation between the frames and iterates over two main steps,

1) for each point in frame A transformed to frame B's sensor coordinate system, find the closest point in frame B,

[1]The authors are with the Centre for Autonomous System at KTH Royal Institute of Technology, SE-100 44 Stockholm, Sweden {ekz,pronobis,johnf,patric}@csc.kth.se

[1]http://vision.in.tum.de/data/datasets/rgbd-dataset

2) calculate an update to the transformation of frame A that minimises the sum of squared distances between the point pairs

This is repeated until a stopping criteria is reached which could be a certain number of iterations is reached, the mean squared error is below some threshold, or the difference between the transformations in two consecutive steps is small. Numerous ways to improve ICP has been suggested in the literature, typically along one of these directions [6] a) what points to select in the two frames, b) how to do the matching, c) how to weigh the different matches, d) what to use as an error metric and f) how to minimise the error.

The most expensive part of ICP is typically finding the closest points. The standard way of performing the matching is to use nearest neighbour matching in Euclidean space. This has a complexity of $\mathcal{O}(N^2)$ in a naive implementation. A common way to speed this up is to use a kd-tree (or a set of trees) which reduces the complexity to $\mathcal{O}(Nlog(N))$.

Each point cloud is a sample of the real world. Even small perturbations in the sensor pose can lead to sampling different structures or parts thereof. A common way to address this is to make a parametric model and then, for example, fit the points in one frame against planes in the other as in [7] or more recently in the GICP algorithm [3] where both point clouds are models with planar surfaces. GICP is currently one of the most widely used algorithms due to its good performance and being readily available in the PCL library. As with all sensors the data is subject to noise. The 3D normal distribution transform (3D-NDT) [4], [8] fit Gaussian ellipsoids to the data which both address the issue of noise and reduces the dimensionality of the data, thus speeding up the processing. This also alleviates the need to make a commitment regarding the correspondence of individual points, instead probability distributions are matched. Similar work has been presented in [9].

All of the previously mentioned methods, with the exception of [10] were developed when 3D point clouds were typically acquired by a laser scanner which only provided geometric information. A common denominator between many of the registration methods is that they require a good initial guess for the transformation, close enough to the global minimum not to get stuck in a local minimum. As a way to address this, [11] extend the matching metric to a weighted sum of the Euclidean distance and the distance between geometric shape features extracted from the point cloud. The descriptor information allows the system to find the transformation even without an initial guess.

In a track parallel to the work with lasers, vision based systems have achieved impressive results. Also here the registration problem is a key issue. A common way to perform the matching has been frames is to extract so called key points and to assign descriptors to these which allows for reliable and accurate matching. Using key points reduces the need to treat all pixels to only treat the key points and using the descriptors allows for more reliable associations. SIFT [12] and later SURF[13] are till widely used and give very good results. While being discriminative and somewhat invariant to scale and rotation SURF and especially SIFT are relatively expensive to compute. Several simpler but faster to compute features have been suggested such as BRIEF [14] and later ORB [15] which extends BRIEF with invariance to rotation. The detection of keypoints is often done by FAST [16]. Additional recent descriptors include BRISK[17] and FREAK [18]. In [10] the Kinect Fusion algorithm is presented. It takes a completely opposite approach to extracting few distinctive features and instead uses a dense, non-parametric, representation for the reference frame from which an artificial point cloud is sampled and compared to the new point cloud.

So far we have discussed registration between consecutive frames. However, an equally, if not more, important and even harder data association problem is that of looking for a match between one frame and all frames previously seen. This is key to successful implementation of SLAM. Here the question is first if the two frames match at all and if so what the transformation is. While it is often technically possible to limit the search space for the possible matches one reduces robustness by relying on the position when looking or matches. Matching feature by feature in each frame is prohibitively slow. A common approach taken is to make use of visual vocabularies [19]. The basic idea is to form clusters in descriptor space and assign a label to each cluster or word. The discretisation of descriptors into words allows means that features matching can be done by comparison two integers indices (the label of the word). An image can be described by a set (bag) of words and represented by a histogram counting the number of times a certain word occurs in the image. Matching two images has then been reduced from a $\mathcal{O}(N^2)$ matching operation where high dimensional descriptors are compared to a constant time operation of comparing two histograms. This has laid the foundation for FAB-MAP [20] and its follow-ups.

A. *Contribution*

In our previous work [2] we presented AICK which addressed the issue of point selection by only using the points in the point cloud with an associated keypoint descriptor. We also build on the work in [11] and leverage on the RGB information in the frames to suggest a metric that combines the Euclidean distance with the distance in visual descriptor space between points. The weight assigned to the two components of the metric is adaptive over the course of the registration, starting with assigning all weight to the visual descriptor to achieve robustness to poor initial guesses and converging towards assigning all weight to the Euclidean distance to achieve accuracy. In the current paper we expand the analysis of AICK and investigate how important the choice of keypoint descriptor is. More specifically we investigate if we can trade the robust SURF feature used in [2] for something less descriminative but computationally less intense. We also suggest a way to reduce the search space when finding matching by exploiting part of the machinery used in bag of words approaches. Finally, we show through experimental validation how the suggested method performs on par with our previously suggested method, performing much better than GICP and 3D-NDT and at a fraction of the computational cost.

III. AICK Data Registration

The AICK algorithm is an efficient and accurate way to register two frames of RGB-D data. It exploits keypoints that have both a 3D position in space as used by ICP and a descriptor which characterizes the surrounding context of the point. In contrast to ICP, it is able to find a good registration even when no initial guess is given. AICK is an iterative algorithm that adaptively changes from emphasizing the descriptor match to emphasizing the geometric fit between the points in the two frames. At the later stages it becomes essentially ICP but having avoided the local minima that result from incorrect initial matches. The results are thus as for ICP with less failures.

As said, in ICP one must start with an initial guess of the transformation between the two frames. One then finds all the matching pairs of points. The matching criteria is the smallest Euclidean distance, d_e between the 3D points. After finding all matches where d_e is below a threshold, the transformation is recomputed to minimize the sum of these distances.

The main strength of the ICP method is that it gives very accurate transformations when the matches are correct. It is most suitable for dense point clouds where sampling artifacts are not significant.

The main weakness of ICP is that if the initial guess leads to too many incorrect matches the solution can get 'stuck' trying to make those fit. It needs most of the initial matches to either be correct or at least on the correct smooth surfaces. The need to have a good guess to start with is rather problematic as it is just this transformation that we are after. It would be better if the method did not require any initial guess. In AICK the initial match is independent of the transformation as it is based solely on the descriptor information.

AICK does not match dense point clouds but rather keypoints which are detected points of interest with features that give a descriptor as vector of values. Two similar features or the same feature seen from different angles will have descriptors that are close in this descriptor space. This way we reduce the number of points to consider for matching[2] to only those points that have a key point associated with it. This then addresses the problem of which points to select as well.

AICK does the same two phases, match and optimize, as ICP but it uses a different matching metric which adapts over the course of the iterations. Instead of d_e we use d_i,

$$d_i = (1 - \alpha^i) d_e + \alpha^i d_d, \tag{1}$$

where $i \in \{0, 1, 2, 3, \ldots\}$ is the iteration number, d_d is the distance, L2 norm, in descriptor space and the constant parameter $\alpha \in [0, 1]$ is the decay factor to move from pure descriptor distance, $(i = 0)$ to nearly only Euclidean distance, $(\alpha^i << 1)$

One technical problem that can arise from combining two 'distances' in this way is the arbitrary scale implicit in the distances. Eq.(1) clearly depends on the units for the distances. It can be necessary to scale one of the distances to make it numerically similar to the other. This is particularly important when applying the threshold for outlier rejection, ie. deciding there is no match to a point.

A. Deeper Investigation of AICK

In our original paper on AICK we compared AICK using SURF features to GICP, NDT and a method based on RANSAC matching. We found that AICK could outperform those methods. The use of SURF features was sensible for the first work as they are known to give strong and robust descriptors. Here we will demonstrate that most of the gains from using the AICK method are not dependent on those strong descriptors. We will therefore compare using the more easily computed but weaker ORB features.

We also observed that the performance of the AICK algorithm seemed to degrade very slowly with lowering the number of keypoints used. This was true from 1000 to 200 keypoints after which the results started to degrade noticeably[3]. That observation led us to consider ways in which we might make the algorithm more efficient by trading off the expensive step of finding all the matches that fall below our threshold. If we could limit the search to only a small subset of the points we might miss some matches but that should not matter based on the above observation. This is one of the assumptions tested in this work.

To do this we use the method of learning a 'vocabulary' of words as in the bag of words method.[4] Then match only to nearby words. The learning step is based on different 'training' data than the registration is done on. Learning is essentially clustering the descriptors from all the training images into a predetermined number of clusters. Then the vocabulary consists of the mean descriptor for each cluster.

We then find for each keypoint in a frame the nearest words to it. This requires fewer comparisons if the number of words is less than the number of keypoints or if the words are arranged in a tree structure that speeds this search. More importantly, if we match this frame to many other frames we need not recompute the nearest words. Thus, for each frame we store the nearest words for each keypoint and the nearest keypoints for each word. We then make the initial selection of what points to try to match to those keypoints that are near the same words. By near we mean within our match threshold. The result of this is that instead of all having to match all points to all points we only match each point to a (small) subset of the points in the other frame.

As the descriptor space is vast most of the space will not be near any word. We are therefore relying on the words being a

[2]A typical RGB-D frame has hundreds of thousands of points

[3]See for instance Fig. 6 in [2]

[4]We do not use the 'bags' in this work only the words. The bags might be useful to chose which two frames to try to register to one another which is a question not addressed here.

good representation for most of the common keypoints that are found in images. We expect that the keypoint clusters will be mostly tight and few points will be far from any word. This is one of the assumptions being tested here. In particular we use the same threshold on deciding keypoint to word match as we used for keypoint to keypoint matching. We will look at how the number of clusters (or words) effects the accuracy and success rate of our method.

To summarise we select only points with both depth information and an associated keypoint to be used in ICP. Instead of matching all key points to all key points as in our original work we only consider keypoints in the other frame that belongs to similar words. As we will show this can speed up the expensive association step by an order of magnitude in most case. In contrast to our previous work we use ORB features instead of the provenly reliable SURF feature which also gives a significant reduction in computational time, and as we will see, we lose relatively little in performance.

IV. EXPERIMENTAL SETUP

A. Benchmark Data and Performance Measure

B. Algorithms tested

1) GICP:

2) 3D-NDT:

C. Experimental Procedure

V. EXPERIMENTAL RESULTS

A. Convergence and Speed

B. Visual inspection

VI. SUMMARY AND CONCLUSIONS

ACKNOWLEDGMENT

This work was funded by SSF through its Centre for Autonomous Systems.

REFERENCES

[1] P. Besl and N. McKay, "A method for registration of 3-d shapes," *IEEE Trans. on Pattern Analysis and Machine Intel.*, no. 2, pp. 239–256, 1992.

[2] J. Ekekrantz, A. Pronobis, J. Folkesson, and P. Jensfelt, "Adaptive closest keypoint," in *Submitted to: Proc. of the International Conference on Intelligent Robot Systems (IROS)*, 2013.

[3] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp," in *Proceedings of Robotics: Science and Systems*, (Seattle, USA), June 2009.

[4] M. Magnusson, A. Lilienthal, and T. Duckett, "Scan registration for autonomous mining vehicles using 3d-ndt," *Journal of Field Robotics*, vol. 24, pp. 803–827, 2007.

[5] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[6] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pp. 145–152, IEEE, 2001.

[7] Y. Chen. and G. Medioni, "Object modeling by registration of multiple range images," in *Proc. of the 1992 IEEE Intl. Conf. on Robotics and Automation*, pp. 2724–2729, 1992.

[8] T. Stoyanov, M. Magnusson, and A. J. Lilienthal, "Point Set Registration through Minimization of the L2 Distance between 3D-NDT Models," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 14–19 2012.

[9] L. Montesano, J. Minguez, and L. Montano, "Probabilistic scan matching for motion estimation in unstructured environments," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, 2005.

[10] R. Newcombe, A. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pp. 127–136, IEEE, 2011.

[11] G. Sharp, S. Lee, and D. Wehe, "Icp registration using invariant features," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, pp. 90 –102, jan 2002.

[12] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[13] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer Vision–ECCV 2006*, pp. 404–417, Springer, 2006.

[14] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: binary robust independent elementary features," in *Computer Vision–ECCV 2010*, pp. 778–792, Springer, 2010.

[15] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: an efficient alternative to sift or surf," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 2564–2571, IEEE, 2011.

[16] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Computer Vision–ECCV 2006*, pp. 430–443, Springer, 2006.

[17] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 2548–2555, IEEE, 2011.

[18] A. Alahi, R. Ortiz, and P. Vandergheynst, "Freak: Fast retina keypoint," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 510–517, IEEE, 2012.

[19] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pp. 1470–1477, IEEE, 2003.

[20] M. Cummins and P. Newman, "FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.