DOCTORAL THESIS

Adaptive Differential Evolution Methods on 3D Range Image Registration and Object Tracking

Author:
Linh TAO

Supervisor:
Professor Dr. Eng
HASEGAWA HIROSHI

July 17, 2017

Declaration of Authorship

I, Linh TAO, declare that this thesis titled, "Adaptive Differential Evolution Methods on 3D Range Image Registration and Object Tracking" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

_____

Date:

_____

Abstract

Robots have gained its popularity in many areas from industrial, household to medical applications. To perform their tasks efficiently, they must be equipped with ability of localizing themselves in the environment and determining handling objects' positions. Those remains the most challenging problems for the computer vision and robotics researchers.

The thesis presents efforts of continuing applying recent advanced natural inspired optimization methods in general and adaptive differential evolution methods in particular for above tasks. It contributes to the research area in the important way of proposing new pipelines for applying the optimization algorithms into 3D Range Image Registration. It also proves the effectiveness of those optimization algorithms in applying for object tracking problem with 2D cameras.

The proposed methods presented in this thesis have been fully implemented and empirically evaluated. The first demonstration related to registering different 3D scenes to archive the transformation matrix of camera movement. The fast, accurate, and robust results show that the proposed algorithms significantly improves on the registration problem over state-of-the-art algorithms. The second demonstration presents the model-based tracking system for textureless objects by using 2D camera. Experiment results show the ability of the methods in solving object detection and tracking problems.

Acknowledgements

Firstly, I would like to express my sincere gratitude to my advisor Prof. Hiroshi Hasegawa for the continuous support of my study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my study.

Last but not the least, I would like to thank my family: my parents, my wife and my little son for supporting me spiritually throughout writing this thesis and my life in general.

Contents

List of Figures

List of Tables

List of Abbreviations

ICP	**I**terative **C**losest **P**oint
EM-ICP	**M**ulti **S**cale **I**terative **C**losest **P**oint
PFH	**P**oint **F**eature **H**istograms
SIFT	**S**cale **I**nvariant **F**eature **T**ransform
RANSAC	**RAN**dom **SA**mpe **C**consensus
EAs	**E**volution **A**lgorithms
SA	**S**imulated **A**nnealing
DE	**D**ifferential **E**volution
ISADE	**I**mproved **S**elft **A**daptive **D**ifferential **E**volution
NNGADE	**N**eural **N**etwork **G**enetic **A**lgorithm **D**ifferential **E**volution

Chapter 1

Introduction

In Japan and other developed countries, aging population is becoming a great concern for society [1] . As life expectation increase, so does the chance of people becoming physically and cognitively limited or disabled. In the same way, requirements for caring services and people who can give those services. The social phenomenon forces us to find out new solutions including technologies for promoting independent living, protecting elderly from disabilities, increasing their participation in daily life. Which can help them to improve their health-state and prevent mitigating to care-giving state.

Recent research program in Japan [2] and USA [3] proposes key robot technologies for prolonging the independence of elderly people. Those robots assist people in performing their daily activities. Some robots could bring objects, sorting dishes [4] , set table or warn people in case of forgetting something. Indeed a robot capable of performing pick-and-place tasks for the objects of daily use might already be of substantial use. Assistive robots need to be equipped with necessary perceptual capabilities. The robots have to detect, recognize, localize and track the position of manipulating objects in order to perform their task competently.

This dissertation thesis continues investigating the core technologies for robot localization methods, image registration, by using the recent developed optimization algorithms as the searching engines. We also apply those algorithms into tracking

problem to find object positions which enables robots to handle objects.

1.1 Intelligent Evolutionary Optimization Algorithms

Optimization takes part into every aspect of our lives. Our working schedules need to be optimized, transportation routes need to be optimized to minimize possibility of traffic jams [5], household wives need to optimize their expenses, biological system need to be optimized to adapt with environment changes [6]. The fascinating of optimization area is not only because of its algorithmic or theoretical content, but also its universal applicability. In computer vision task, searching algorithms work on complicated and nonlinear searching spaces, using intelligent evolutionary algorithms is necessary for solving problems.

Computers has been dramatically developed from the first versions, they are good at what we poorly do, like calculating. However, they would long to be able to perform tasks that humans can do well, like recognizing a face. This led to attempts to mimic biological behavior in an effort to make computers better at such tasks. These efforts resulted in technologies like Fuzzy systems [7], Neural networks [8], Genetic Algorithms [9], and other evolution algorithms (EAs). EAs are therefore considered to be a part of the general category of computer intelligence.

This section introduce some intelligent characteristics included in EAs: adaptation, randomness, communication, feedback, exploration, and exploitation. These are the characteristics that implemented in EAs in searching for intelligent algorithms [10].

1.1.1 Adaptation

We usually consider adaptation to changing environments as a feature of intelligence [11]. In our society, if you are intelligent, you will be able to learn things faster

and change yourself following different conditions. However, if you are not so intelligent, then you need helps from someones else or you are slower in adapting yourself.

However, we do not consider adaptive controllers intelligent [12, 13], or a virus that can survive extreme environments intelligent. We thus conclude that adaptation is a necessary but not sufficient condition for intelligence. We want our EAs be able to adapt with a wide class of problems. Adaptability in an EA is only one of many criteria for a successful EA.

1.1.2 Randomness

Ones usually think of randomness in negative terms. We want everything in your control and trajectory, we try to avoid unpredictable things and we also try to control our environment. However, some degree of randomness is a necessary component of intelligence [14] . Think of a zebra running to escape from a lion. If the zebra runs in a straight line and at a constant speed, it will be easy to catch. But an intelligent zebra will zigzag and move unpredictably to avoid its predator. Conversely, think of a lion that is trying to catch a zebra. If the lion waits at the same bush and at the same time every day, it will be easy to avoid. But an intelligent lion will strike at different places and different times and in an unpredictable way. Randomness is a characteristic of intelligence.

Too much randomness will be counterproductive. If the zebra randomly decides to lie down while being chased, we would be right to question its intelligence. If a lion randomly decides to dig a hole in its search for a zebra, we would be right to question its intelligence. So randomness is a feature of intelligence, but only within limitations [15].

EAs designs will include some component of randomness. If we exclude randomness, EAs will not work well. But if we use too much randomness, they will

not work well either. We will need to use the right amount of randomness in EA designs. Of course, as discussed earlier, EAs are adaptable. Therefore, good EAs will perform well over a range of randomness measures. We cannot expect the EAs to be so adaptable that we can use any level of randomness, but they will be adaptable enough so that the exact randomness measure will not be critical.

1.1.3 Communication

Communication is a feature of intelligence. Consider a genius who takes an IQ test, except the genius has no way of communicating. He will fail the IQ test even though he is a genius. Many deaf, dumb, and autistic individuals fail IQ tests even though they are quite intelligent. Children who are raised without human interaction are not creative, intelligent, happy, or well adjusted [16]. Their lack of communication with others during their formative years prevents them from developing any intellectual capacity beyond a young child. Their years of isolation are irrecoverable, and they cannot learn to communicate or adapt to society.

Intelligence not only involves communication, but it is also emergent. That is, intelligence arises from a population of individuals. A single individual cannot be intelligent. It can be argued that there are many intelligent individuals in the world, and even if such an individual were isolated he would still be intelligent. However, such individuals gained their intelligence only through interaction with others. A single ant wanders aimlessly and accomplishes nothing, but a colony of ants can find the shortest path to food, build elaborate networks of tunnels, and organize themselves as a self-sustaining community [17]. Likewise, a single individual will never accomplish anything if he never has any interaction with a community. The main point here is that communication is a feature of intelligence. This is why most EAs involve more than one candidate solutions. Those candidates interact and communicate with each other and learn from each other's successes and failures. After

many loops of learning and changing, the population of individuals evolves a good solution to the optimization problem.

1.1.4 Feedback

Feedback is a fundamental characteristic of intelligent [18]. This involves adaptation, which was discussed above. A system cannot adapt if it cannot sense and react to its environment. However, feedback involves more than adaptation; it also involves learning. When we make mistakes, we change so that we don't repeat those mistakes. However, even more importantly, when others make mistakes, we adjust our behavior so that we don't repeat those mistakes. Failure provides negative feedback. Conversely, success (our's and others') provides positive feedback and influences us to adopt those behaviors to which we attribute success. We often see others who don't seem to learn from mistakes, and who don't adopt behaviors that are proven to lead to success; we don't consider such people to be very intelligent. Feedback is also the basis for many natural phenomena. The water cycle consists of an endless succession of rain and evaporation. More rain leads to more evaporation, and more evaporation leads to more rain. Since this includes a fixed amount of water, the water cycle leads to a stable amount of moisture on the surface of the earth and in the sky. If this feedback mechanism were somehow disturbed, there would be a lot of difficulties for life, including floods and drought. The sugar/insulin balance in the human body is another feedback mechanism [19] . The more sugar we eat, the more insulin our pancreas produces; the more insulin our pancreas produces, the more sugar is absorbed from the blood. Too much sugar in the blood leads to hyperglycemia, and too little sugar in the blood leads to hypoglycemia. Diabetes is the disturbance of the sugar/insulin feedback mechanism, and can lead to serious and long-term health problems. This characterization of feedback as a hallmark of intelligence is often recognized in intelligent control theory [20]. Feedback is not a

sufficient condition for intelligence. No one would call a proportional controller intelligent, and no one would call a mechanical thermostat intelligent. Feedback is a necessary, but not sufficient, condition for intelligence.

1.1.5 Exploration and Exploitation

Exploration is the search for new ideas or new strategies. Exploitation is the use of existing ideas and strategies that have proven successful in the past. Exploration is high-risk; a lot of new ideas waste time and lead to dead ends. However, exploration can also be high-return; a lot of new ideas pay off in ways that we could not have imagined. Exploitation is closely related to the feedback strategies discussed previously. Someone who is intelligent uses what they know and what they have instead of constantly reinventing the wheel. But someone who is intelligent is also open to new ideas, and is willing to take calculated risks. Intelligence includes the proper balance of exploration and exploitation. The proper balance of exploration and exploitation depends on how regular our environment is [21]. If our environment is rapidly changing, then our knowledge quickly becomes obsolete and we cannot rely as much on exploitation. However, if our environment is highly consistent, then our knowledge is dependable and it may not make sense to try very many new ideas. EA designs will need a proper balance of exploration and exploitation to be successful. Too much exploration is similar to too much randomness, which we discussed earlier, and will probably not give good optimization results. But too much exploitation is related to too little randomness. The proper balance of exploration and exploitation in EAs was called "the optimal allocation of trials" by John Holland, one of the pioneers of genetic algorithms [22].

1.2 Range Image Registration

The three-dimensional reconstruction of real objects is an important topic in computer vision [23] . Most of the acquisition systems are limited to reconstruct a partial view of the object obtaining in blind areas and occlusions, while in most applications a full reconstruction is required. Many proposed techniques such as Goicp[24] and SAICP[25] fuse 3D surfaces by determining the motion between the different views. The first step is related to obtaining a rough registration when such motion is not available. The second one is focused on obtaining a fine registration from an initial approximation. Figure 1.1 and Figure 1.2 show an example of registration result for a more completed views from six range images from PointCloudLibrary website[26].

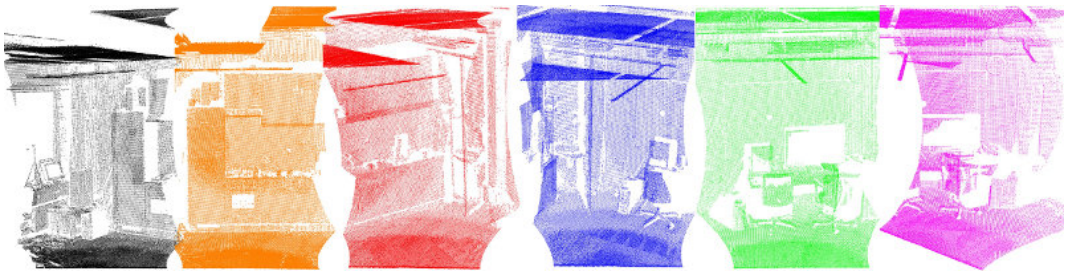

FIGURE 1.1: Six range images

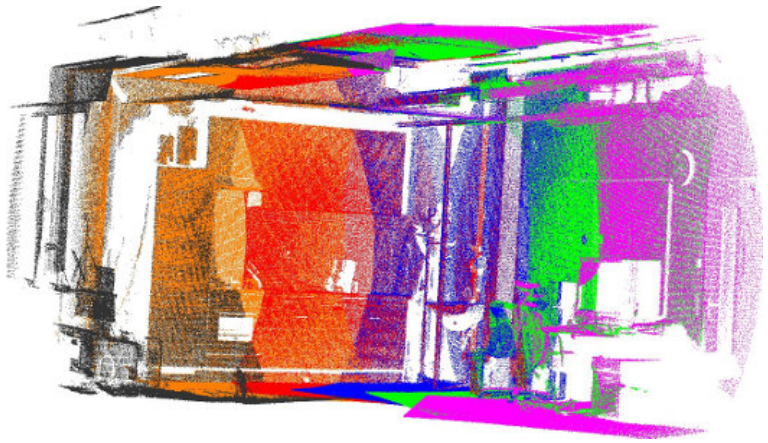

FIGURE 1.2: Registration result

1.2.1 Going 3D

The introduction of commercial depth sensing devices, such as the Microsoft Kinect and Asus Xtion, has shifted the research areas of robotics and computer vision from 2D-based imaging and laser scanning toward 3D-based depth scenes for environment processing. As physical objects or scenarios are built using more than a single image, images from different times and positions need to be aligned with each other to provide a more complete view. We call the alignment process registration, and it plays a key role in object reconstruction, scene mapping, and robot localization applications. Depending on the number of views that are processed simultaneously, registration is divided into multi-view [27] and pair-wise cases [28]. Our method focuses on the latter case for constructed range images captured by 3D cameras. From two images, called the model and the data, the registration algorithm finds the best homogeneous transformation that aligns the data and the model image in a common coordinate system.

1.2.2 Classical Approach

The iterative closest point (ICP)[29] algorithm and its variants, such as EM-ICP[30] and Generalized-ICP[31], have been indispensable tools in registration algorithms. ICP's concept and implementation are easy to understand. It derives a transformation that draws images closer to each other using their L_2 error iteratively. ICP-class algorithms have a drawback for general registration in that they require a further assumption of near-optimal initial pose transformation; otherwise, the registration process is likely to converge to local instead of global or near global optima. Some mesh and point cloud editor software programs, such as Meshlab [32], include an ICP built-in registration tool; however, they require that users perform manual pre-alignment before ICP can be applied.

To overcome the shortage of ICP-class methods, automatic registration algorithms in general perform two steps: coarse initialization and fine transformation. If two point clouds are sufficiently close, the first step can be omitted. Otherwise, researchers are faced with a big challenge. Two approaches for coarse transformation, pre-alignment estimation, or initialization exist: local and global. The former uses local descriptors (or signatures), such as PFH[33] and SIFT[34], which encode local shape variation in neighborhood points. If the key points of these descriptors appear in both registered point clouds, the initialization movement can be estimated by using sample consensus algorithms, such as RANSAC [35]. Unfortunately, it is not always guaranteed that these signatures will appear in both registered point clouds. On the other hand, global approaches, such as Goicp and SAICP, take all the points into account. The computation cost is the biggest problem in this approach. In big number data cases, the computation cost becomes large. By virtue of new search algorithms, in particular heuristic optimal methods, and the increase in computer speed achieved by using multi-core computer processor units (CPUs) and graphic computation units (GPUs) [36], it is possible to find reasonable solutions using global approaches for the registration problem. When the coarse transformation has been estimated, the ICP algorithm is an efficient tool for finding the fine transformation.

1.2.3 Optimization Algorithms with Range Image Registration

By integrating optimal search tools with an ICP algorithm, researchers have created hybrid algorithms that integrate global optimizers with ICP. However, this approach has its limitations. SAICP, a parameter-based algorithm, uses simulated annealing [37] as a search engine to find the best movement combination of rotation angles and translation. However, SA is not sufficiently effective to allow its application to a complicated fitness function, where the potential of a failed convergence is high. Goicp converges slowly, since it uses the branch-and-bound (BnB) method, a time

consuming and non-heuristic method, as a search algorithm to ensure a 100% convergence rate. In addition, ICP algorithms frequently include a kd-tree structure for searching corresponding points. Using the kd-tree nearest neighbor search method also leads to a high computation cost and a long runtime.

1.3 Object Tracking

Object tracking plays a crucial part in the field of computer vision. Its greatest application comes in the area of automated video analysis which has received large interest in recent times with the proliferation of powerful computer systems. In video analysis applications, there are three main steps that take place. These steps are: the detection of objects of interest, the tracking of the objects from one frame to another and lastly the analysis of the trajectory of the objects from one frame to another and lastly the analysis of the trajectory of the objects in question [38]. For this reason, we find object tracking being paramount in applications that deal with automated surveillance [39], video indexing, motion recognition [40], human computer interaction and traffic monitoring [41] among others.

The prior knowledge of object and its behavioral properties must be know and programmed in advance. Over the years, many great tracking algorithms and methods have been proposed. They all differ in approach to dealing with the issue of: what image feature to utilize, what is the best object representation and lastly how should the motion, shape and appearance of the object be modeled. We can divide tracking methods into three main categories which are: point tracking, kernel tracking and silhouette tracking.

1.3.1 Point Tracking

Point tracking methods involve detecting objects in consecutive frames and representing them as points [42]. The association of the points being based on the state

which can depended on the object location or motion with the system requiring an external module for detecting objects in every frame.

The formulation of correspondence o the points across different frames can be a difficult problem especially in the presence of occlusion, mis-detection, entries and exits of objects.

1.3.2 Kernel Tracking

Kernel tracking is usually performed by computing the motion of an object from one frame to other with the object being represented as primitive shape region [43]. Here the tendency is to see object motion in a form of a parametric motion such as translation, affine and the like. Otherwise we see it as a dense flow field computed in subsequent frames. The difference between two approaches is found in appearance of representation used, the number of objects being tracked and the methods used to approximate motion flow.

1.3.3 Silhouette Tracking

Since geometric transformations and illumination changes has no effect on finding keypoints, they have been widely used for matching images from slightly different viewpoints. Keypoint-based approaches work well in textured objects but texture-less objects [44] . Textured objects have various keypoints, those have high potential appearing on both images. After finding keypoints, sample consensus such as RANSAC calculates the most suitable transformation of the object from reference position to current position. The more matched keypoints, the more accurate the transformation is. On texture-less objects lack of keypoint repeatability and stability on texture-less regions neither reduces the accuracy of sample consensus method nor leads to wrong results. Like keypoints, edges are also invariant to general geometric transformations and illumination changes. Using edges are more suitable

as a general approach even with texture-less objects [45]. In early computer vision research, to find the best alignment between two edge maps, a given priori set of edge templates compare their suitability to the current edge maps to draw the mot suitable transformation. The current proposed method of chamfer distance matching enhances the cost functions enable for applying global searching algorithm into object tracking problem [46]. One drawback of using edges is that they are not distinctive enough to provide effective discrimination in complex background or occlusions, there have been efforts to enhance the previous one by unifying interest points or considering multiple but limited hypotheses on edge correspondences [44]. For consideration of multiple hypotheses in a more general sense global searching algorithm should come into consideration.

1.4 Thesis Outline and Contributions

Differential Evolution algorithms proved to be effective in solving various computer vision problems. Its simplicity, effectiveness and straightforward approach are suitable characteristics for computer vision applications which require high accuracy, small runtime and robustness with different scenarios. Those reasons lead us to continue applying currently developed Adaptive Differential Evolution Algorithms into problems of Range Image Registration and Object Tracking.

The thesis contributes new approaches for Range Image Registration and Object Tracking problems by using recently developed Adaptive Differential Evolution Algorithms. The first part, chapter II, we presented the current using methods recently developed ISADE[47], NNGADE[48]. First adaptive method, not only keeps the advantages of Differential Evolution but also enhances the effective of algorithms by applying adaptive techniques. Another more complex algorithm, uses an advanced adaptive technique based on Neural Network and Genetic Algorithms combination.

The second part of the thesis, chapter III, we proposed two pipelines for tackling Range Image Registration problem which are Point Based and Ray-casting Based approaches. The first pipeline, by using two movement technique, searching algorithms only search on two dimensional searching spaces instead of searching on conventional six dimensional ones. Dramatically reducing searching dimensions heightens the convergence rate and accuracy of the results.

With good results on benchmark functions, we believe that adaptive differential evolution algorithms are sufficient for challenging searching tasks. We propose the direct applying adaptive differential evolution algorithms to replace current hybrid approaches which use both local search and global searching tools. Our new approach only uses ISADE as a global searching tool on ray-casting, a fast error calculation method. It reduces the computation cost significantly while still archives high accuracy and robustness for Range Image Registration problem.

Chapter IV presents a potential direction of applying Adaptive Differential Evolution algorithms into object tracking problem. The results proved ability of adaptive differential evolution algorithms applying for different problems.

Finally, chapter V presented experiments and results of our proposed algorithms in various scenarios for Range Image Registration. For Object Tracking problem, our Adaptive Differential Evolution methods proved to work well in finding a position of the target object.

The disseration thesis is divided into six chapter as in Figure 1.3 including:

- Chapter 1: Introduction to the thesis topic
- Chapter 2: Adaptive Differential Evolution Algorithms
- Chapter 3: 3D Range Image Registration
- Chapter 4: Model Based Textureless Object Tracking
- Chapter 5: Experiments & Results

- Chapter 6: Conclusion and future directions

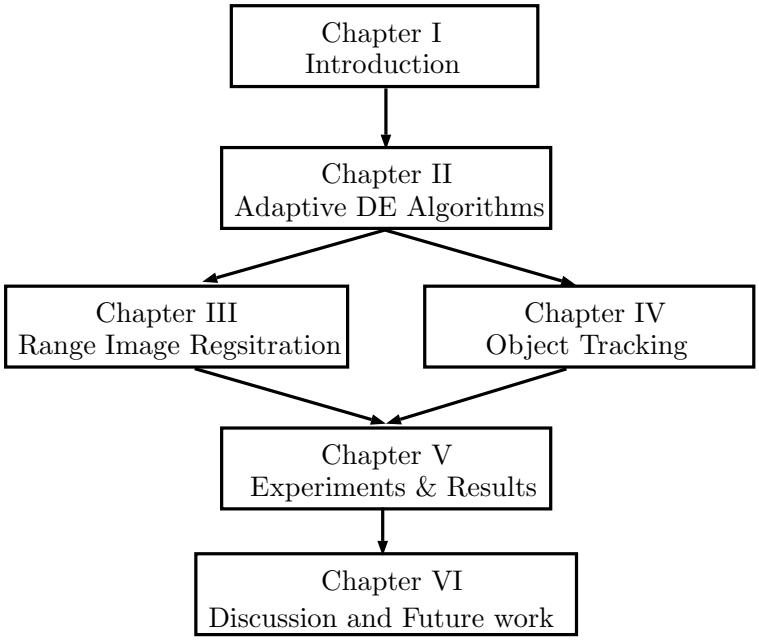

FIGURE 1.3: Thesis outline

Bibliography

[1] K. Yamazaki and R. Ueda and S. Nozawa and M. Kojima and K. Okada and K. Matsumoto and M. Ishikawa and I. Shimoyama and M. Inaba, "Home-Assistant Robot for an Aging Society", Proceedings of the IEEE, 2012, doi=10.1109/JPROC.2012.2200563.

[2] IRT (Information, Robot Technology) Foundation to Support Man, and Ag- ing Society at University of Tokyo. Mission Statement. Technical report, 2007.

[3] Computing Community Consortium. A Roadmap for US Robotics: From Internet to Robotics. Technical report, May 21 2009.

[4] D. Vischer, "Control architecture for a robot with visual and tactile capabilities skilled in sorting dishes," [Proceedings 1992] IEEE International Conference on Systems Engineering, Kobe, 1992, pp. 143-146. doi: 10.1109/ICSYSE.1992.236922

[5] Y. Wang and Y. Tian, "A Schedule Optimization for Weihai Bus System," 2015 International Conference on Service Science (ICSS), Weihai, 2015, pp. 45-48. doi: 10.1109/ICSS.2015.33

[6] O. G. Berestneva and J. S. Pekker, "Simulation and evaluation of biological systems adaptive capabilities," 2014 International Conference on Mechanical Engineering, Automation and Control Systems (MEACS), Tomsk, 2014, pp. 1-4. doi: 10.1109/MEACS.2014.6986860

[7] Novák, V., Perfilieva, I. and Mockor, J. (1999) Mathematical principles of fuzzy logic Dodrecht: Kluwer Academic. ISBN 0-7923-8595-0.

[8] McCulloch, Warren; Walter Pitts (1943). "A Logical Calculus of Ideas Immanent in Nervous Activity". Bulletin of Mathematical Biophysics. 5 (4): 115–133. doi:10.1007/BF02478259.

[9] Mitchell, Melanie (1996). An Introduction to Genetic Algorithms. Cambridge, MA: MIT Press. ISBN 9780585030944.

[10] Dan Simon, "Evolutionary Optimization Algorithms", ISBN: 978-0-470-93741-9

[11] Karl Johan Astrom and Bjorn Wittenmark. 1994. Adaptive Control (2nd ed.). Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[12] A. F. Amer, E. A. Sallam and I. A. Sultan, "Adaptive sliding-mode dynamic controller for nonholonomic mobile robots," 2016 12th International Computer Engineering Conference (ICENCO), Cairo, 2016, pp. 230-235. doi: 10.1109/ICENCO.2016.7856473

[13] H. Wang, B. Yang, Y. Liu, W. Chen, X. Liang and R. Pfeifer, "Visual Servoing of Soft Robot Manipulator in Constrained Environments With an Adaptive Controller," in IEEE/ASME Transactions on Mechatronics, vol. 22, no. 1, pp. 41-50, Feb. 2017. doi: 10.1109/TMECH.2016.2613410

[14] I. Zelinka, R. Senkerik and M. Pluhacek, "Do evolutionary algorithms indeed require randomness?," 2013 IEEE Congress on Evolutionary Computation, Cancun, 2013, pp. 2283-2289. doi: 10.1109/CEC.2013.6557841

[15] Thomas Jansen, Evolutionary Algorithms and Other Randomized Search Heuristics, 2012

[16] Newton, M. (2004). Savage Girls and Wild Boys. Picador.

[17] M. Dorigo, M. Birattari and T. Stutzle, "Ant colony optimization," in IEEE Computational Intelligence Magazine, vol. 1, no. 4, pp. 28-39, Nov. 2006. doi: 10.1109/MCI.2006.329691

[18] Benjamin, "The Intelligence Cycle: An Introduction to Direction, Collection, Analysis & Dissemination of Intelligence | Intelligence 101". www.intelligence101.com. Retrieved 2016-12-04.

[19] J. Costik, "DIY diabetes remote monitoring [Resources]," in IEEE Spectrum, vol. 52, no. 6, pp. 21-22, June 2015. doi: 10.1109/MSPEC.2015.7115551

[20] B. Ravindran, P. Kachroo and T. Hegazy, "Intelligent feedback control-based adaptive resource management for asynchronous, decentralized real-time systems," in IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 31, no. 2, pp. 261-265, May 2001. doi: 10.1109/5326.941850

[21] J. Chen, B. Xin, Z. Peng, L. Dou and J. Zhang, "Optimal Contraction Theorem for Exploration–Exploitation Tradeoff in Search and Optimization," in IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, vol. 39, no. 3, pp. 680-691, May 2009. doi: 10.1109/TSMCA.2009.2012436

[22] John Henry Holland, Adaptation in Natural and Artificial Systems, 1975.

[23] R. Rossi, X. Savatier, J. Y. Ertaud and B. Mazari, "Real-time 3D reconstruction for mobile robot using catadioptric cameras," 2009 IEEE International Workshop on Robotic and Sensors Environments, Lecco, 2009, pp. 104-109. doi: 10.1109/ROSE.2009.5355981

[24] J. Yang, H. Li, Y. Jia, Go-icp: Solving 3D registration efficiently and globally optimally, in: 2013 IEEE International Conference on Computer Vision (ICCV), 2013, pp. 1457–1464. doi:10.1109/ICCV.2013.184.

[25] J. Luck, C. Little, W. Hoff, Registration of range data using a hybrid simulated annealing and iterative closest point algorithm, in: Robotics and Automation, 2000. Proceedings of IEEE International Conference on Robotics and Automation (ICRA '00.), 2000, vol. 4, pp. 3739–3744.

[26] http://pointclouds.org/documentation/tutorials/registration_api.php#registration-api

[27] G. C. Sharp, S. W. Lee, D. K. Wehe, Multiview registration of 3D scenes by minimizing error between coordinate frames, IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(8) (Aug. 2004) 1037-1050. doi: 10.1109/TPAMI.2004.49

[28] D. Holz, A. E. Ichim, F. Tombari, R. B. Rusu, S. Behnke, Registration with the Point Cloud Library: A modular framework for aligning in 3-D, IEEE Robotics & Automation Magazine, 22(4) (Dec. 2015) 110-124. doi: 10.1109/MRA.2015.2432331

[29] P. Besl, N. D. McKay, A method for registration of 3-d shapes, Pattern Analysis and Machine Intelligence, IEEE Transactions on 14(2) (1992) 239–256. doi:10.1109/34.121791.

[30] S. Granger, X. Pennec. Multi-scale EM-ICP: A fast and robust approach for surface registration, in: European Conference on Computer Vision, vol. 2353, pp. 418–432, 2002.

[31] A. Segal, D. Haehnel, S. Thrun. Generalized-ICP, in: Proceedings of Robotics: Science and Systems, June 2009, Seattle, USA. doi:10.15607/RSS.2009.V.021.

[32] Meshlab, http://meshlab.sourceforge.net/, accessed: 2017-01-15.

[33] R. B. Rusu, Z. C. Marton, N. Blodow, M. Beetz, I. A. Systems, T. U. Mnchen, Persistent point feature histograms for 3D point clouds, in: Proceedings of the 10th International Conference on Intelligent Autonomous Systems (IAS-10), 2008.

[34] A. , M. Makaveeva, Real-time scale invariant 3D range point cloud Registration,

[35] F. Wu, X. Fang, An improved RANSAC homography algorithm for feature based image mosaic, in: Proceedings of the 7th WSEAS International Conference

on Signal Processing, Computational Geometry & Artificial Vision, ISCGAV'07, World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA, 2007, pp. 202–207. http://dl.acm.org/citation.cfm?id=1364592.1364627 doi:10.1109/ROBOT.2000.845314.

[36] D. Neumann, F. Lugauer, S. Bauer, J. Wasza, J. Hornegger, Real-time RGB-d mapping and 3-D modeling on the GPU using the random ball cover data structure, in: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), 2011, pp. 1161–1167. doi:10.1109/ICCVW. 2011.6130381.

[37] L. Ingber, Simulated annealing: Practice versus theory, Mathematical and Computer Modelling 18(11) (1993) 29 – 57. doi:http://dx.doi.org/10.1016/0895-7177(93)90204-C.

[38] A Yilmaz et al, "Object tracking: a survey", ACM Computing Surveys, Vol. 38, No. 4, Article 13, Publication date: December 2006

[39] I. Bouchrika, A. Bekhouch and A. Amirat, "Vision-based approach for people tracking using gait in distributed and automated visual surveillance," 2013 8th International Workshop on Systems, Signal Processing and their Applications (WoSSPA), Algiers, 2013, pp. 203-208. doi: 10.1109/WoSSPA.2013.6602362

[40] W. Wei and A. Yunxiao, "Vision-Based Human Motion Recognition: A Survey," 2009 Second International Conference on Intelligent Networks and Intelligent Systems, Tianjin, 2009, pp. 386-389. doi: 10.1109/ICINIS.2009.105

[41] K. Kiratiratanapruk and S. Siddhichai, "Vehicle Detection and Tracking for Traffic Monitoring System," TENCON 2006 - 2006 IEEE Region 10 Conference, Hong Kong, 2006, pp. 1-4. doi: 10.1109/TENCON.2006.343888

[42] Z. Xu, H. R. Wu and X. Yu, "Interest points based object tracking with controlled cameras," 2009 IEEE International Conference on Industrial Technology, Gippsland, VIC, 2009, pp. 1-6. doi: 10.1109/ICIT.2009.4939565

[43] D. Comaniciu, V. Ramesh and P. Meer, "Kernel-based object tracking," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 5, pp. 564-577, May 2003. doi: 10.1109/TPAMI.2003.1195991

[44] C. Choi and H. I. Christensen, "3D textureless object detection and tracking: An edge-based approach," 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura, 2012, pp. 3877-3884. doi: 10.1109/IROS.2012.6386065

[45] Y. Park, V. Lepetit and W. Woo, "Texture-less object tracking with online training using an RGB-D camera," 2011 10th IEEE International Symposium on Mixed and Augmented Reality, Basel, 2011, pp. 121-126. doi: 10.1109/ISMAR.2011.6092377

[46] M. Rauter and D. Schreiber, "A GPU accelerated Fast Directional Chamfer Matching algorithm and a detailed comparison with a highly optimized CPU implementation," 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Providence, RI, 2012, pp. 68-75. doi: 10.1109/CVPRW.2012.6238897

[47] T. Bui, H. Pham, H. Hasegawa, Improve self-adaptive control parameters in differential evolution for solving constrained engineering optimization problems, Journal of Computational Science and Technology 7(1) (2013) 59–74. doi:10.1299/jcst.7.59.

[48] Tao Ngoc Linh, Hieu Pham, Hiroshi Hasegawa, “Self-adaptive of Differential Evolution Using Neural Network with Island Model of Genetic Algorithm”, EuroSim 2016 Conference, Oulu, Finland.

Chapter 2

Adaptive Differential Evolution Algorithms

2.1 Differential Evolution

To solve complex numerical optimization problems, researchers have been looking into nature both as model and as metaphor for inspiration. A keen observation of the underlying relation between optimization and biological evolution led to the development of an important paradigm of computational intelligence for performing very complex search and optimization.

Evolutionary Computation uses iterative process, such as growth or development in a population that is then selected in a guided random search using parallel processing to achieve the desired end. Nowadays, the field of nature-inspired metaheuristics is mostly continued by the Evolution Algorithms (EAs) (e.g., Genetic Algorithms (GAs), Evolution Strategies (ESs), and Differential Evolution (DE) etc.) as well as the Swarm Intelligence algorithms (e.g., Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), etc.). Also the field extends in a broader sense to include self-organizing systems, artificial life, memetic and cultural algorithms, harmony search, artificial immune systems, and learnable evolution model.

The GAs have been applied to various complex computational problems, and its validity has been reported by many researchers [1, 2]. However, it requires a huge computational cost to obtain stability in convergence towards an optimal solution. To reduce the cost and to improve the stability, a strategy that combines global and local search methods becomes necessary. As for this strategy, current research has proposed various methods [3]. For instance, Memetic Algorithms (MAs) [4, 5, 6, 7, 8, 9] are a class of stochastic global search heuristics in which EAs-based approaches are combined with local search techniques to improve the quality of the solutions created by evolution. MAs have proven very successful across the search ability for multi-modal functions with multi-dimensions. These methodologies need to choose suitably a best local search method from various local search methods for combining with a global search method within the optimization process. Furthermore, since genetic operators are employed for a global search method within these algorithms, design variable vectors (DVs) which are renewed via a local search are encoded into its genes many times at its GA process. These certainly have the potential to break its improved chromosomes via gene manipulation by GA operators, even if these approaches choose a proper survival strategy. To solve these problems and maintain the stability of the convergence towards an optimal solution for multi-modal optimization problems with multiple dimensions, Hieu Pham et al. proposed evolutionary strategies of Adaptive Plan system with Genetic Algorithm (APGAs) [10]. It is shown to be statistically significantly superior to other EAs and MAs.

Unlike most other techniques, GAs maintain a population of tentative solutions that are competitively manipulated by applying some variation operators to find a global optimum. For non-trivial problems, this process might require high computational resources such as large memory and search times. To design efficient GAs, a variety of advances by new operators, hybrid algorithms, termination criteria, and more are continuously being achieved. Parallel GAs (PGAs) [11, 12, 13] often leads to superior numerical performance not only to faster algorithms. However, the truly

interesting observation is that the use of structured population, either in the form of a set of islands or a diffusion grid, is responsible for such numerical benefits. A PGA has the same as a serial GA, consisting in using representation of the problem parameters, robustness, easy customization, and multi-solution capabilities. In addition, a PGA is usually faster, less prone to finding sub-optimal solutions only, and able of coorperating with other search techniques in parallel.

Differential Evolutionary (DE)[14] was recently introduced and has garnered significant attention in the research literature. DE has many advantages including simplicity of implementation, reliable, robust, and in general is considered as an effective global optimization algorithm. DE operates through similar computational steps as employed by a standard EA. However, unlike traditional EAs, the DE variants perturb the current generation population members with the scaled differences of randomly selected and distinct population members. Therefore, no separate probability distribution has to be used for generating the offspring [15]. Recently, DE has drawn the attention of many researchers all over the world resulting in a lot of variants of the basic algorithm with improved performance such as Self-adaptive control parameters DE (jDE) [16] and Advanced DE (ADE) [17]. Compared with and other techniques [18], it hardly requires any parameter tuning and is very efficient and reliable.

In DE, the scaling factor F and crossover rate C_r determine the correction and speed of convergence, while another important parameter, NP, the population size, remains a user-assigned parameter to handle problem complexity.

2.1.1 Initialization in DE

The initial population was generated uniformly at random in the range lower boundary (lb) and upper boundary (ub).

$$X_i^G = lb_j + rand_j(0,1)(ub_j - lb_j) \tag{2.1}$$

where $rand_j(0,1)$ a random number $\in [0,1]$.

2.1.2 Mutation operation

In DE, there are various mutation schemes to create mutant vectors $V_i^G = (V_{i,1}^G, ..., V_{i,D}^G)$ for each individual of population at each generation G. X_i^G is target vector in the current population, D is vector dimension number.

$$DE/rand/1 : V_{i,j}^G = X_{r1,j}^G + F(X_{r2,j}^G - X_{r3,j}^G) \tag{2.2a}$$

$$DE/best/1 : V_{i,j}^G = X_{best,j}^G + F(X_{r1,j}^G - X_{r2,j}^G) \tag{2.2b}$$

$$DE/currenttobest/1 : V_{i,j}^G = X_{i,j}^G + F(X_{best,j}^G - X_{i,j}^G) + F(X_{r1,j}^G - X_{r2,j}^G) \tag{2.2c}$$

$$DE/rand/2 : V_{i,j}^G = X_{i,j}^G + F(X_{r2,j}^G - X_{r3,j}^G) + F(X_{r4,j}^G - X_{r5,j}^G) \tag{2.2d}$$

$$DE/best/2 : V_{i,j}^G = X_{best,j}^G + F(X_{r1,j}^G - X_{r2,j}^G) + F(X_{r3,j}^G - X_{r4,j}^G) \tag{2.2e}$$

$$DE/randtobest/1 : V_{i,j}^G = X_{best,j}^G + F(X_{best,j}^G - X_{r2,j}^G) + F(X_{r2,j}^G - X_{r3,j}^G) \tag{2.2f}$$

where r_1, r_2, r_3, r_4, and r_5 are randomly selected integers in the range $[1, NP]$.

2.1.3 Crossover operation

After mutation process, DE performs a binomial crossover operator on X_i^G and V_i^G to generate a trial vector $U_i^G = (U_{i,1}^G, ..., U_{i,D}^G)$ for each individual population i as shown in Equation 2.3.

$$U_{i,j}^G = \left\{ \begin{array}{ccc} V_{i,j}^G & if \quad rand_j \leqslant C_r \quad or \quad j = j_{rand} \\ X_{i,j}^G & otherwise \end{array} \right\} \tag{2.3}$$

where $i = 1, ..., NP, \quad j = 1, ..., D$, j_{rand} is a randomly chosen integer in $[1, D]$, $rand_j(0,1)$ is a uniformly distributed random number between 0 and 1 generated for

each j and $C_r \in [0, 1]$ is called the crossover control parameter. Using j_{rand} ensures the difference between the trial vector U_i^G and target vector X_i^G.

2.1.4 Selection operation

The selection operator is performed to select the better one between the target vector X_i^G and the trial vector U_i^G entering to the next generation.

$$X_i^{G+1} = \left\{ \begin{array}{lll} U_i^G & if & f(U_i^G) \leqslant f(X_i^G) \\ X_i^G & & otherwise \end{array} \right\} \tag{2.4}$$

where $i = 1, ..., NP$, X_i^{G+1} is a target vector in the next generation's population.

2.1.5 DE algorithm flowchart

Figure 2.1 shows implementation flowchart of DE algorithms.

2.2 ISADE, an efficient improved version of Differential Evolution algorithm

2.2.1 Adaptive selection learning strategy in mutation operator

With ISADE [19], authors randomly chose three mutation schemes: $DE/best/1/bin$, $DE/best/2/bin$, and $DE/randtobest/1/bin$. $DE/best/1/bin$ and $DE/best/2/bin$ have a good convergence property and $DE/randtobest/1/bin$ has a good population diverse property. The probability of applying these strategies is equal at values of $p_1 = p_2 = p_3 = 1/3$.

$$DE/best/1 : V_{i,j}^G = X_{best,j}^G + F(X_{r_1,j}^G - X_{r_2,j}^G) \tag{2.5a}$$

$$DE/best/2 : V_{i,j}^G = X_{best,j}^G + F(X_{r1,j}^G - X_{r2,j}^G) + F(X_{r3,j}^G - X_{r4,j}^G) \tag{2.5b}$$

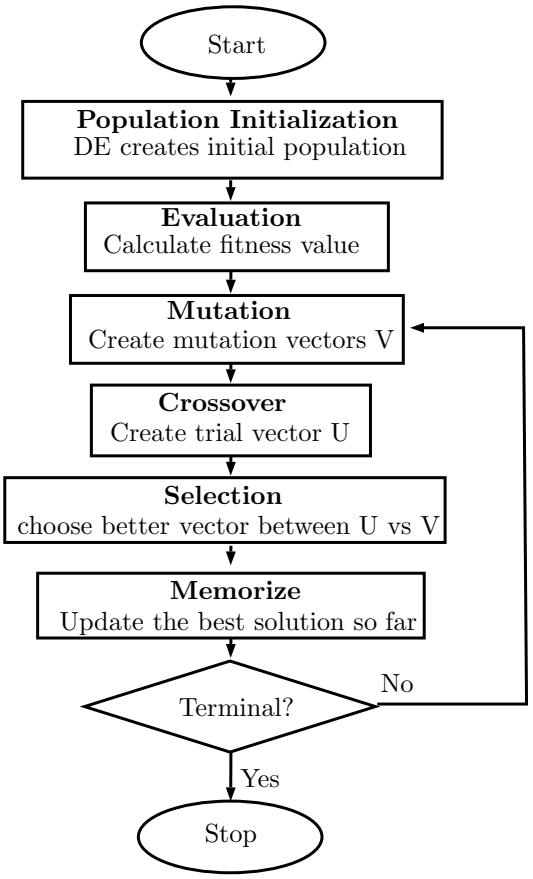

FIGURE 2.1: DE implementation process

$$DE/randtobest/1 : V_{i,j}^{G} = X_{best,j}^{G} + F(X_{best,j}^{G} - X_{r2,j}^{G}) + F(X_{r2,j}^{G} - X_{r3,j}^{G}) \quad (2.5c)$$

where r_1, r_2, r_3, r_4, and r_5 are randomly selected integers in the range $[1, NP]$, where NP is the population size.

2.2.2 Adaptive scaling factor

To achieve a better performance, ISADE gives the scale factor F a large value initially to allow better exploration and a small value after the generations to allow appropriate exploitation. Instead of using sigmoid scaling in Equation 2.6 taken from Tooyama and Hasegawa's study on APGA/VNC[20], ISADE adds a new factor to

calculate F as shown in Equation 2.7.

$$F_i = \frac{1}{1 + exp(\alpha * \frac{i-NP/2}{NP})} \tag{2.6}$$

$$F_i = \frac{F_i + F_i^{mean}}{2} \tag{2.7}$$

in which F_i^{mean} is calculated as Equation 2.8.

$$F_i^{mean} = F_{min} + (F_{max} - F_{min})(\frac{i_{max} - i}{i_{max}})^{n_{iter}} \tag{2.8}$$

where F_{max} and F_{min} denote the lower and upper boundary condition of F with recommended values of 0.15 and 1.55, respectively. i, i_{max}, and n_{iter} denote the current, max generation, and nonlinear modulation index as in Equation 2.8.

$$n_{iter} = n_{min} + (n_{max} - n_{min})(\frac{i}{i_{max}}) \tag{2.9}$$

where n_{max} and n_{min} are typically chosen in the range $[0, 15]$. Recommended values for n_{min} and n_{max} are 0.2 and 6.0 respectively.

2.2.3 Crossver control parameter

ISADE is able to detect whether the height of C_r values are useful. The control parameter C_r is assigned as

$$C_r^{i+1} = \left\{ \begin{array}{cc} rand_2 & if \quad rand_1 \leqslant \tau \\ C_r^i & otherwise \end{array} \right\} \tag{2.10}$$

where $rand_1$ and $rand_2$ are random values $\in [0, 1]$, τ represents the probability to adjust C_r, which is also updated using

$$C_r^{i+1} = \begin{cases} C_{r_{min}} & C_{r_{min}} \leqslant C_r^{i+1} \leqslant C_{r_{medium}} \\ C_{r_{max}} & C_{r_{medium}} \leqslant C_r^{i+1} \leqslant C_{r_{max}} \end{cases} \tag{2.11}$$

where $C_{r_{min}}$, $C_{r_{medium}}$, and $C_{r_{max}}$ denote a low value, median value, and high value of the crossover parameter, respectively. We use recommended values of $\tau = 0.1$, $C_{r_{min}} = 0.05$, $C_{r_{medium}} = 0.50$, and $C_{r_{max}} = 0.95$.

2.2.4 ISADE algorithm pseudo-code

ISADE eliminates tuning tasks for the problem-dependent parameters F and C_r. With simple adaptive rules, the computation complexity of this new version of the DE algorithm remains the same as that of the original version. All the above ideas and theories of ISADE algorithm is implemented as in the flowchart shown in Figure 3.11.

2.3 Self-adaptive of Differential Evolution Using Neural Network with Island Model of Genetic Algorithm

We purposed a new evolutionary algorithm called NN-DEGA that using Artificial Neural Network (ANN) for Self-adaptive DE with Island model of GA to solve large scale optimization problems, to reduce a large amount of calculation cost, and to improve the convergence towards the optimal solution.

2.3.1 Island model parallel distributed in NN-DEGA

Migration PGA, island model, are reported to have greater information compatibility, a stable design and low computational costs because they deal with GAs in

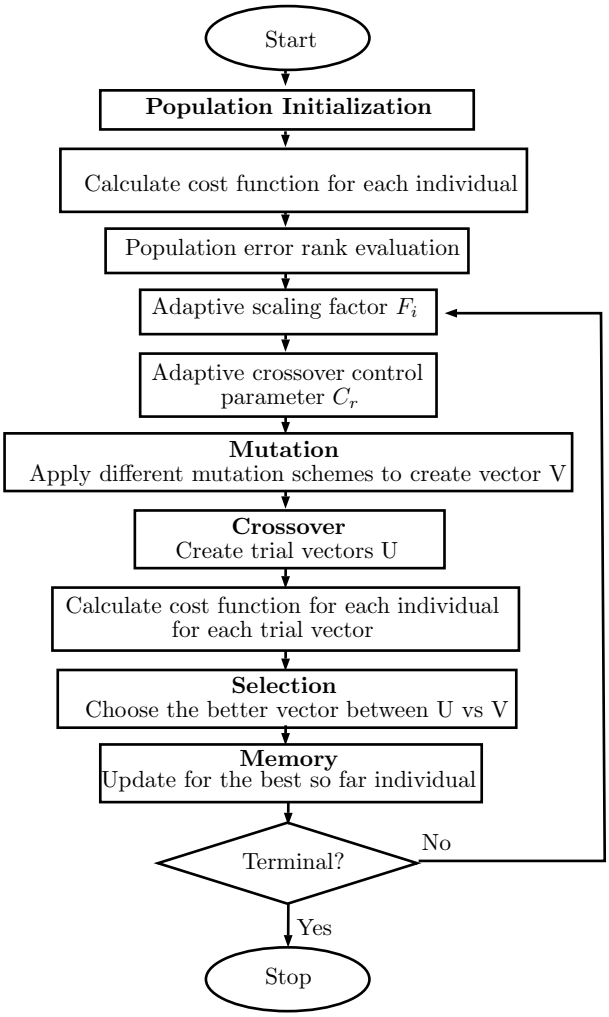

FIGURE 2.2: ISADE implementation process with ray-casting corresponding method

parallel. In NN-DEGA, optimization is conducted by applying GA and DE to each subpopulation. The control variables adjust the vicinity of the output constriction factor F between the subpopulations. The candidate control variables and the new solution come from the other subpopulation at the time of immigration, so a diversity of solutions can be expected because the migration destination is determined at random. A schematic diagram of NN-DEGA with PGA migration is shown in Fig. 2.3.

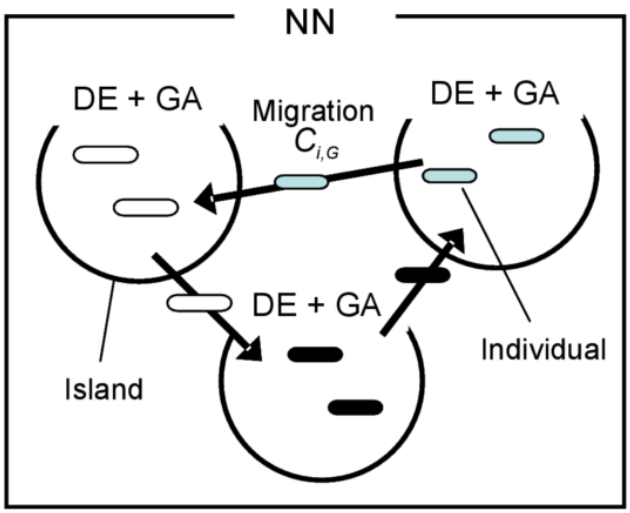

FIGURE 2.3: Island Model GA conceptual diagram in NN-DEGA.

2.3.2 Self-adaptive using Neural Network

The self-adaptive constriction factor $F(NN)$ is used for data clustering of the GA control variables using NN, which have been determined uniquely to stabilize their variation. From a viewpoint of excellent parallel processing and to ensure compatibility with multi-point search methods such as GA and DE are also used in the present method. NN is often used in combination with these techniques [24]. NN may be used to cluster and classify the data without using a signal if it is necessary to learn using a teacher signal that is also a NN. In the present method, the GA variable data clustering is controlled using unsupervised learning to determine the output scaling factor change. The initial constriction factor F is set at random and we vary its value based on the NN output. The unsupervised learning method is also a multi-layer NN, so we use NN to perform the feed-forward transfer. The number of layers is determined in a number of search points for each subpopulation. In addition, the NN is configured after it has been sorted in descending order of fitness in the subpopulations to the output side from the input side, where the weight of the transfer equation is as shown in Equation 2.12. Therefore, many subpopulations

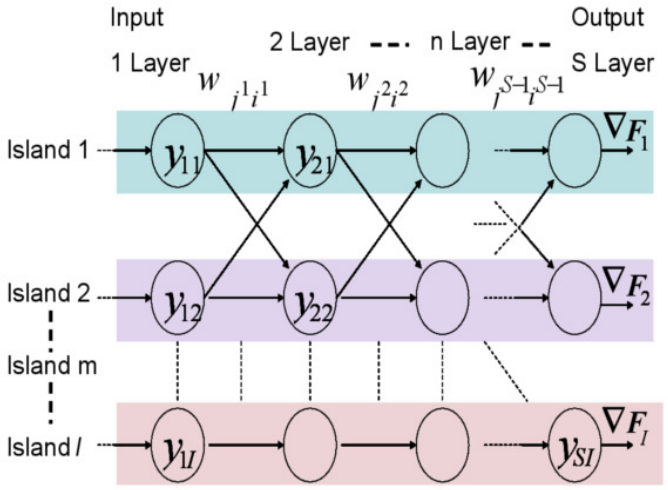

FIGURE 2.4: NN-DEGA neural network.

have highly adaptive search points with strong effects on other subpopulations. The formulation of the control variable, the transfer equation for each node in the NN and the schematic diagram of the overall NN are as follows.

$$w_{j^n i^n} = y_{nm}/y_{(n-1)m} \tag{2.12}$$

$$node_t = \sum_{i=1}^{I} SP \cdot w_{i^n j^n} out_i^{n-1}/I \tag{2.13}$$

$$SP = 2 \cdot C_{i,G} - 1 \tag{2.14}$$

$$C = [c_{i,j}, \ldots, c_{i,p}]; (0.0 \leq c_{i,j} \leq 1.0) \tag{2.15}$$

$$F_{i,G+1} = F_{i,G} - \nabla F_i \tag{2.16}$$

The GA handles control variables (CVs) and C_t is allocated to each search point, which is encoded as a 10-bit string. The order of each search point is allocated to each node of a multi-layer NN, as shown in Fig. 2.4, on the input side and the output

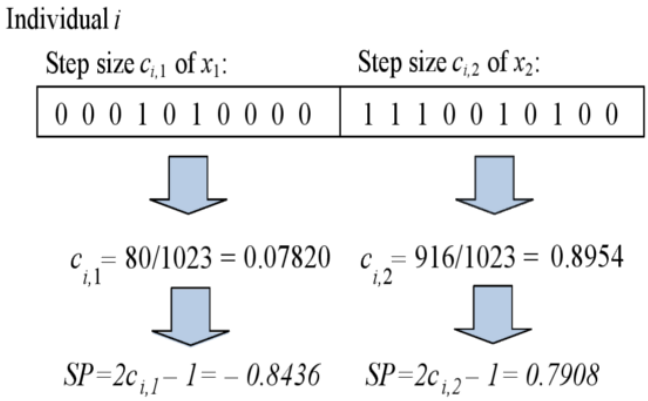

FIGURE 2.5: Step size that defined by CVs for controlling a global behavior to prevent it falling into the local optimum.

side. The weight of the NN, $w_{j^n i^n}$, which is determined from the adaption ratio of the search points, is transmitted between the nodes. C_t is the control variable that determines the step size SP as shown in Fig. 2.5 and this element determines the extent of the constraint factor change, ∇F. Therefore, the constriction factor change is an important factor, which determines the width of the overall distribution of the neighborhood of search points. Using the control variable, we can change F adaptively to facilitate more stable solution search and better control of the control variable in the NN. In addition, n is the number of NN hierarchical levels, m is the number of subpopulations, j, i is the number of neurons in NN, t is the number of individuals, S is the number of searches per island and I is the maximum number of islands.

2.3.3 Reconstruction of differential vector

Each target vector aims at the global optimal solution by updating differential vector based on its best solution has been achieved so far $pbest_{ij}$ and the best solution of all

Algorithm 1 The NN-DEGA Pseudocode

Initialize population with CVs;
Generate initial DVs;
Evaluate individuals with initial DVs;
while (Termination Condition) **do**
    Adaptive control of scaling factor $F = F(NN)$ using Neural network;
    Generate DVs via AP with new DE scheme:
    Generate a mutant vector: $V_{ij,G+1} = gbest_{j,G} + F(NN) \cdot (pbest_{ij,G} - X_{ij,G})$;
    Generate a trial vector $U_{ij,G+1}$ through binomial crossover: $U_{ij,G+1} = \begin{cases} V_{ij,G+1}, (rand_j \leq CR) \; or \; (j = j_{rand}) \\ X_{ij,G+1}, (rand_j \geq CR) \; and \; (j \neq j_{rand}) \end{cases}$;
    Evaluate the trial vector $U_{i,G}$;
    if $f(U_{i,G}) \leq f(X_{i,G})$ **then** $X_{i,G+1} = U_{i,G}$ else $X_{i,G+1} = X_{i,G}$;
    end if
    Evaluate individuals with DVs;
    Select parents;
    Recombine to produce offspring for CVs;
    Mutate offspring for CVs;
    if (Restructuring Condition) **then**
        Restructure chromosome of offspring for CVs;
    end if
end while

individuals in the population $gbest_j$ (where $j = [1, 2, \ldots, D]$, D is the dimension of the solution vector), as Equation 2.17:

$$V_{ij,G+1} = gbest_{j,G} + F \cdot (pbest_{ij,G} - X_{ij,G}) \tag{2.17}$$

We carried out the reconstruction of the control variable like considered control variables APGAs, not only control variable meet the conditions listed below, but also reconstruction of the DE differential vector by keep performing keep the global search of the search point, the appropriate solution search is always performed.

- The same value adaptation accounted for more than 80% for the entire
- The same bit-string chromosome occupies more than 80% for the entire
- The same value of scaling factor accounted for 50% of the total.

2.3.4 Elite strategy

In this method, using the diploid genetics is not proper to perform the search using the NN solution [25]. Generally, GA, information has only a single gene for one individual. However, the structure has a double recessive genetic information that does not appear in the dominant phenotype. Here, in NN, genetic information is treated as a control variable. Information dominance for the NN is elite solution closed to the control variable, as shown in the following equation. With the aim of having a strong influence in the form of dominant inheritance, enhancing the effectiveness of the control variable, advantageously advancing the solution search, elite solution against other sub-populations as the island model of GA.

$$\begin{aligned} &if\ |eSP - SP_1| - |eSP - SP_2| < 0 \qquad SP = SP_1 \\ &if\ |eSP - SP_1| - |eSP - SP_2| > 0 \qquad SP = SP_2 \end{aligned} \tag{2.18}$$

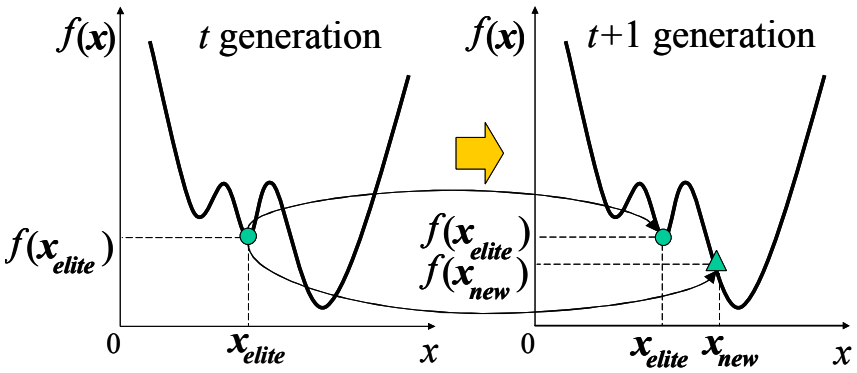

FIGURE 2.6: Elite strategy, where the best individual survives in the next generation, is adopted during each generation process.

Bibliography

[1] S. W. Mahfoud and D. E. Goldberg, *A genetic algorithm for parallel simulated annealing*, Parallel Problem Solving from Nature, 2, 301–310, 1992.

[2] D. E. Goldberg and S. Voessner, *Optimizing global-local search hybrids*, Proceedings of 1999 Genetic and Evolutionary Computation Conference, pp.220–228, 1999.

[3] http://ti.arc.nasa.gov/tech/rse/publications/

[4] Y.S. Ong and A.J. Keane, *Meta-Lamarckian Learning in Memetic Algorithms*, IEEE Transactions on Evolutionary Computation, 8(2), 99–110, 2004.

[5] Y.S. Ong, M.H. Lim, N. Zhu and K.W. Wong, *Classification of Adaptive Memetic Algorithms: A Comparative Study*, IEEE Transactions on Systems, Man and Cybnetics Part B, 36(1), 141–152, 2006.

[6] F. Neri, V. Tirronen, T. Kärkkäinen and T. Rossi, *Fitness Diversity Based Adaptation in Multimeme Algorithms: A Comparative Study*, IEEE Congress on Evolutionary Computation, pp. 2374–2381, 2007.

[7] A. Caponio, G.L. Cascella, F. Neri, N. Salvatore and M. Sumner, *A Fast Adaptive Memetic Algorithm for Online and Offline Control Design of PMSM Drives*, IEEE Transactions on Systems, Man and Cybernetics Part B, Special Issue on Memetic Algorithms, 37(1), 28–41, 2007.

[8] V. Tirronen, F. Neri, T. Kärkkäinen, K. Majava and T. Rossi, *An Enhanced Memetic Differential Evolution in Filter Design for Defect Detection in Paper Production*, Evolutionary Computation Journal, MIT Press, 16(4), 529–555, 2008.

[9] J.E. Smith, W.E. Hart and N. Krasnogor, *Recent Advances in Memetic Algorithms* Springer, 2005.

[10] Hieu Pham, S. Tooyama and H. Hasegawa, *Evolutionary Strategies of Adaptive Plan System with Genetic Algorithm* JSME Journal of Computational Science and Technology, 6(3), 129–146, 2012.

[11] E. Cantu-Paz, *A survey of parallel genetic algorithms*, Calculateurs Paralleles,10(2),1998.

[12] E. Alba and J.M. Troya, *A Survey of Parallel Distributed Genetic Algorithms*, Complexity, 4(4), 31–52,1999.

[13] R. Tanese, *Distributed genetic algorithms*, Proc. of 3rd Int. Conf. on Genetic Algorithms, pp. 434–439, 1989.

[14] K. Price, R. Storn and J. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*, Springer-Verlag, Berlin, 2005.

[15] S. Das and P.N. Suganthan, *Differential evolution - A survey of the State-of-the-Art*, IEEE Transactions on Evolutionary Computation, 15(1), 4–31, 2011.

[16] J. Brest, S. Greiner, B. Bošković, M. Mernik and V. Žumer, *Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems*, IEEE Trans. Evol. Comput., 10(6), 646–657, 2006.

[17] A. Wagdy Mohamed, H.Z. Sabry and A. Farhat, *Advanced Differential Evolution algorithm for global numerical optimization*, IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE), pp. 156–161, 2011.

[18] J. Vesterstroem and R. Thomsen, *A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems*, Proc. IEEE Congr. Evolutionary Computation, pp. 1980—1987, 2004.

[19] T. Bui, H. Pham, H. Hasegawa, Improve self-adaptive control parameters in differential evolution for solving constrained engineering optimization problems, Journal of Computational Science and Technology 7(1) (2013) 59–74. doi:10.1299/jcst.7.59.

[20] S. Tooyama, H. Hasegawa, Adaptive plan system with genetic algorithm using the variable neighborhood range control, in: Evolutionary Com- putation, 2009. CEC '09. IEEE Congress on Evolutionary Computation, CEC '09., 2009, pp. 846–853. doi: 10.1109/CEC.2009.4983033.

[21] Openmp, http://openmp.org/wp/,accessed:2017-01-15.

[22] Y. Wei Chen, A. Mimori, C.-L. Lin, Hybrid particle swarm optimization for 3-d image registration, in: 16th IEEE International Conference on Image Processing (ICIP), 2009, pp. 1753–1756. doi:10.1109/ICIP. 2009.5414613.

[23] F. L. Seixas, L. S. Ochi, A. Conci, D. M. Saade, Image registration using genetic algorithms, in: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, GECCO '08, ACM, New York, NY, USA, 2008, pp. 1145–1146. doi:10.1145/1389095.1389320.

[24] K. Kobayashi, T. Hiroyasu and M. Miki, *Mechanism of Multi-Objective Genetic Algorithm for Maintaining the Solution Diversity Using Neural Network*, The Science and Engineering Review of Doshisha University, 48(2), 24–33, 2007.

[25] M. Kouchi, H. Inayoshi and T. Hoshino, *Optimization of Neural-Net Structure by Genetic Algorithm with Diploydi and Geographical Isolation Model*, Japanese Society for Artificial Intelligence, 7(3), 509–517, 1992.

[26] D. Karaboga and B. Basturk, *A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm*, Journal Global Optimization, 39, 459–471, 2006.

Chapter 3

Range Image Registration

3.1 Range Image Registration Approaches

This part summaries some approaches for global range image registration problem up to date.

3.1.1 Registration error function and ICP

SVD and PCA[1] are integrated with ICP in classical methods and global search algorithms are integrated with ICP in most current hybrid methods. In this integration, SVD and PCA find the coarse transformation while ICP is the fine transformation estimation tool. The original version of the ICP algorithm relies on the L_2 error to derive the transformation (rotation $\mathbf{R} \in SO^3$ and translation $\mathbf{t} \in R^3$), which minimizes the L_2 type error:

$$\mathbf{E}(\mathbf{R}, \mathbf{t}) = \sum_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{e_i}(\mathbf{R}, \mathbf{t}) = \sum_{\mathbf{i=1}}^{\mathbf{n}} |\mathbf{R}\mathbf{x_i} + \mathbf{t} - \mathbf{y_{j^*}}| \tag{3.1}$$

where $X = \{x_i\}, \{i = 1, 2, 3, ..., m\}$ is the model pointset and $Y = \{y_j\}, \{j = 1, 2, 3, ..., n\}$ is the data pointset, x_i and $y_j \in R^3$ are the coordinates of the points in the pointsets, $\mathbf{R}$ and $\mathbf{t}$ are the rotation and translation matrix, respectively, $\mathbf{y_{j^*}}$ is the corresponding point of $\mathbf{x_i}$ denoting the closest point in data pointset Y. $\mathbf{R}$ and

t are determined by Roll-Pitch-Yaw movement of three rotation angles (α, β, γ) and translation values (x, y, z).

Variants of the ICP algorithm rely on different distance categories to define the closest points. Point-to-point distance and point-to-plane distance are two popular examples. Equation 3.2 presents the former case.

$$\mathbf{j}^* = \underset{\mathbf{j}\in\{\mathbf{1},\ldots,\mathbf{n}\}}{\operatorname{argmin}} ||\mathbf{R}\mathbf{x_i} + \mathbf{t} - \mathbf{y_j}|| \tag{3.2}$$

The following iterative process is designed to achieve the final transformation.

1. Compute the closest model points for each data point as in Equation 3.2.

2. Compute the transformation **R** and **t** based on the error obtained using Equation 3.1.

3. Apply **R** and **t** to the data pointset.

4. Repeat Steps 1, 2, and 3 until the error obtained using (3.1) is smaller than a set tolerance level or the procedure reaches its maximum iteration.

Step by step, the data pointset becomes closer to the model pointset and the process stops at local minima. ICP's variants, such as LMICP[2] and SICP[3], use different methods to calculate the transformation from error $\mathbf{E}(\mathbf{R}, \mathbf{t})$. A well-known accumulation registration method in the KinectFusion[4] algorithm uses ICP to register two consecutive frames. The transformation matrix for the current frame is estimated by multiplying the matrices from the previous registration steps. ICP implementation procedure is presented in Figure 3.1.

Kd-tree nearest neighbor

Kd-tree[5] is the way of organizing some number of points in k-dimension space with binary search tree. It is a powerful tool for nearest neighbor (NN) searching for the first step in ICP algorithms. Original ICP has $Olog(n)$ complexity, where n is the

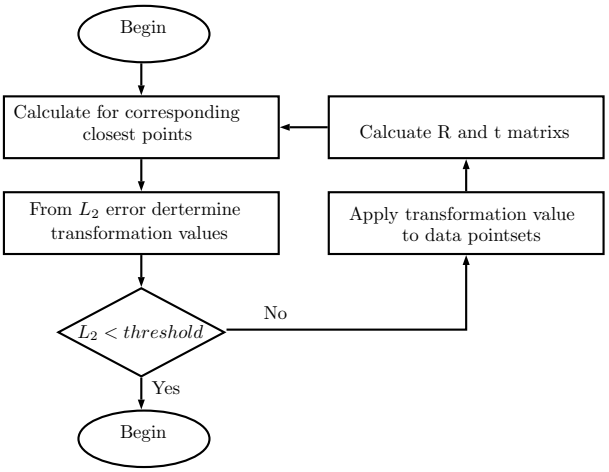

FIGURE 3.1: ICP flowchart

number of point in the pointset.

a) Building a kd-tree: At each level, a hyperplane, which is perpendicular to the corresponding axis, splits all children into two next branches of the tree. At the root of the tree, all children would be splitted based on the first dimension. The root point at each note should be the median point to balance the tree. Each level down, the tree is divided by the next dimension, returning to the first dimension once all others have been exhausted. The recurrent procedure is repeated until the last trees that contain one point. Figure 3.2-3.3 show an example of two dimension kd-tree and its presentation in binary tree.

b)Kd-tree nearest neighbor search: Nearest neighbor search algorithm aims to find the closest point of all tree point to a curtain point. By exploiting properties of kd-tree, kd-tree NN search algorithm quickly eliminates large portions of points in the searching space. Kd-tree NN algorithm in a k-d tree is presented as following:

1. Starting from the root node, the algorithm moves downward recursively, by comparing whether the point is less than or greater than the current node in the split dimension.

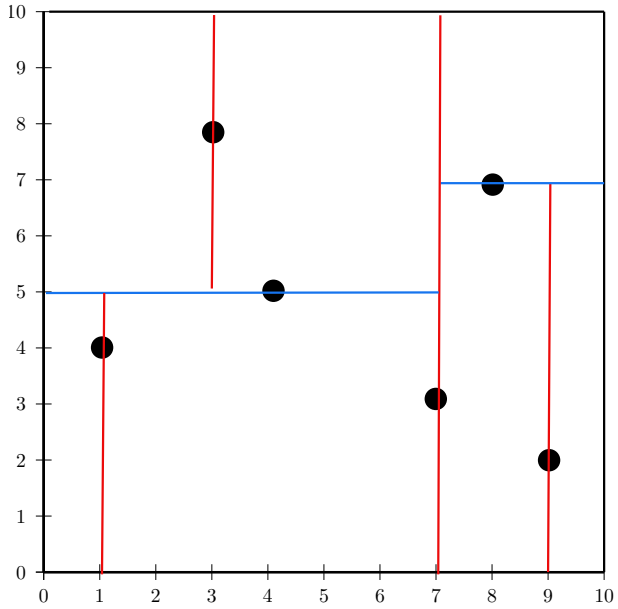

FIGURE 3.2: An example of two dimension kdtree

2. Once the algorithm reaches a leaf node, that leaf node become "current best".

3. The algorithm unwinds the recursion of the tree, performing the following steps at each node:

1. If the current node is closer than the current best, then it becomes the current best.

2. There could be other points in the other side of the splitting plane that are closer to the set point then the "current best". A hypersphere around the set points is created with "current best" on the surface.

Next, the algorithm checks whether hypersphere intersects hyperplanes by comparing the distance from the set point to those hyperplanes with radius of the hypersphere.

1. In case of the hypersphere intersects the plane, there could be a better points in the other side of the plane, then the algorithm must move downward to check on the other branch of the tree.

2. The algorithm gets rid the branch on the other side of the node and

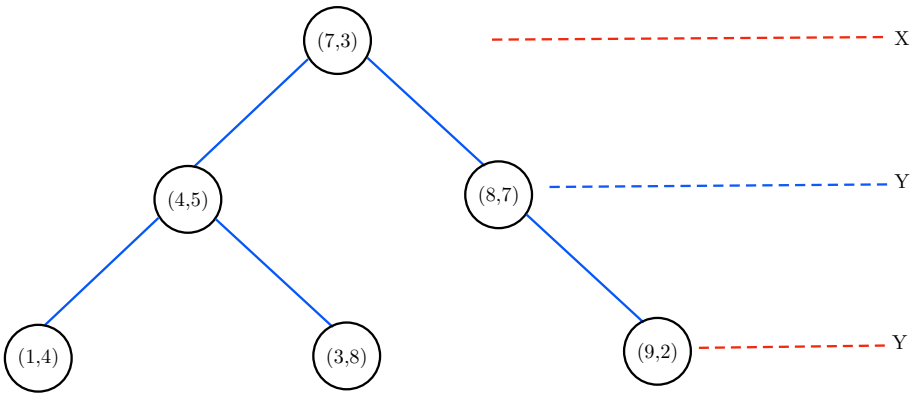

FIGURE 3.3: An example of two dimension kdtree

move upward.

4. The search draws to the NN when the algorithm finishes this process for the root node.

By using an idea of NN search in kd-tree, if we maintain k current bests instead of just one closest point we have k-nearest neighbor search algorithm of kd-tree.

3.1.2 Global hybrid registration algorithm

ICP algorithms constitute the most suitable method for registering close or pre-aligned point-cloud data. In other cases, the algorithm frequently converges incorrectly. Global search algorithms are suitable for solving this problem, since they can find the global instead of the local minima. To reduce the burden of the global search algorithm, researchers frequently flatten the search space by using ICP. Figures 3.4 and 3.5 show an example of ICP's operation as a flattening tool. In Figure 3.4, from any beginning point, after many iterations ICP finds the nearest local optima point. Figure 3.5 shows that a complex fitness function (colored black) becomes a simpler one (colored red). As a result, global search methods are able to find the global minima more effectively.

The integration is effective in the case of point-cloud data where the point number is small. For cases where the point number is large, the hybrid approach with

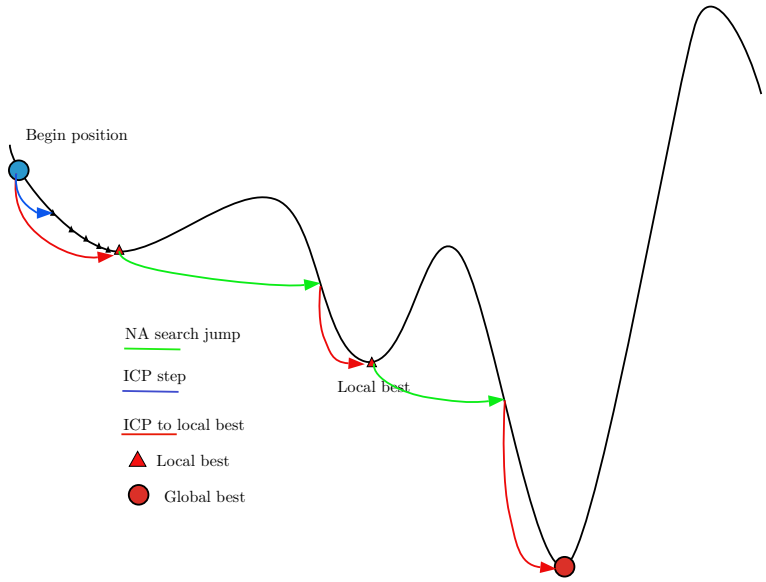

FIGURE 3.4: global searching algorithm with ICP integrated

ICP becomes slow. This method cannot therefore be implemented in real-time applications. The flowchart of hybrid approach is presented in Figure 3.6.

3.2 Point based Approach

We proposes a novel global registration approach named "Global Hybrid Point-based Registration". As a global registration method, it requires no local descriptors. The approach uses points as variables in point spaces with a global searching tools as a search engines to find the global optimal. With this approach, the searching dimension reduces from six to two and increase convergence rate as well as robustness. ICP algorithm in the method is itergrated to find local minima and error for each initialization.

3.2.1 Approach Methodology

If we assign one point of the model pointset as the based point and knowing its the corresponding on the data poinset, point based registration performs two steps

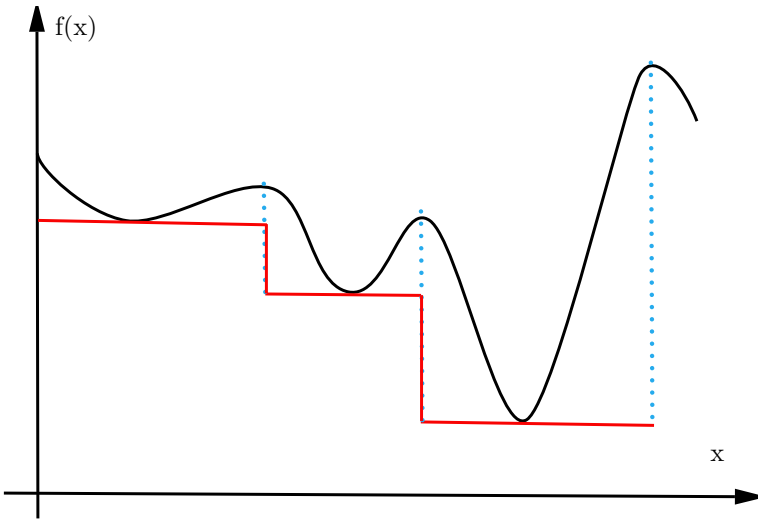

FIGURE 3.5: Example of flatten objective function after icp in red color where original function is in black.

movement which are shown in Figure 3.7, the model pointset is in red and the data pointset is in blue:

- Step 1: The first movement is to make two corresponding points and their normal vector aligned is performed. The algorithm works on assumption of existing corresponding points on data pointset. If there is some losing data on the data pointset, algorithm could fill the losing data by using nearby data points with interpolation methods.

- Step 2: To rotate the data pointset a curtain angle about the point's normal vector.

By doing so, the global searching algorithm works only on two dimensions a much more easier task.

3.2.2 Point based searching

Point and Normal adjustment

The first movement is to move data pointsets Y so a point $y_j \in Y$ coincides with a fix point $x_i \in$ model pointset X, so as their normal vectors. The first step includes

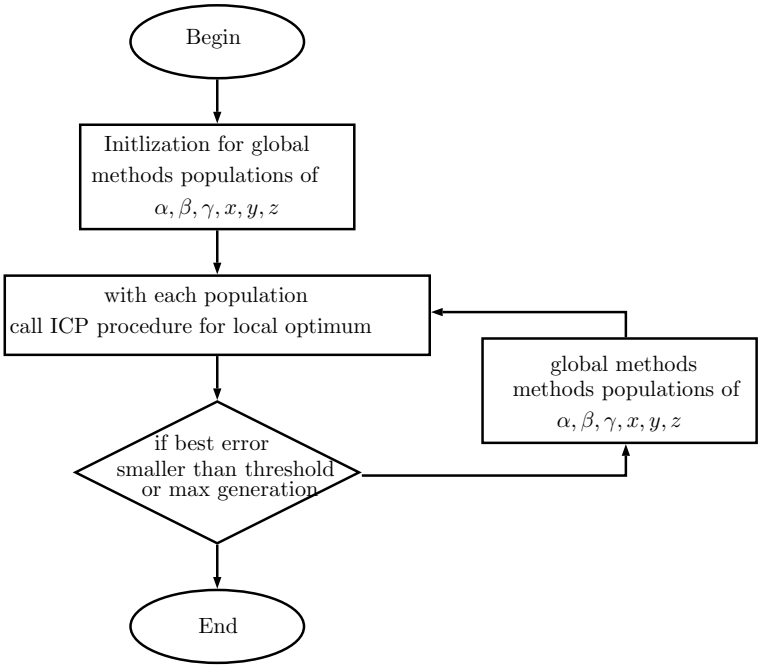

FIGURE 3.6: Implementation procedure of the hybrid approach

two small movements.

First of all, We do the translation from y_j to x_i. The translation matrix is as in Equation 3.3.

$$t = \begin{vmatrix} x_{x_i} - x_{y_j} \\ y_{x_i} - y_{y_j} \\ z_{x_i} - z_{y_j} \end{vmatrix} \tag{3.3}$$

where $x_{x_i}, y_{x_i}, z_{x_i}$ are coordinates of x_i, and $x_{y_i}, y_{y_i}, z_{y_i}$ are coordinates of y_i in Euclidean coordinate.

Then we do the rotation to make x_i and current y_i coincides. To do so, we need to find the vector, which normal to both those normal vectors by cross multiplying

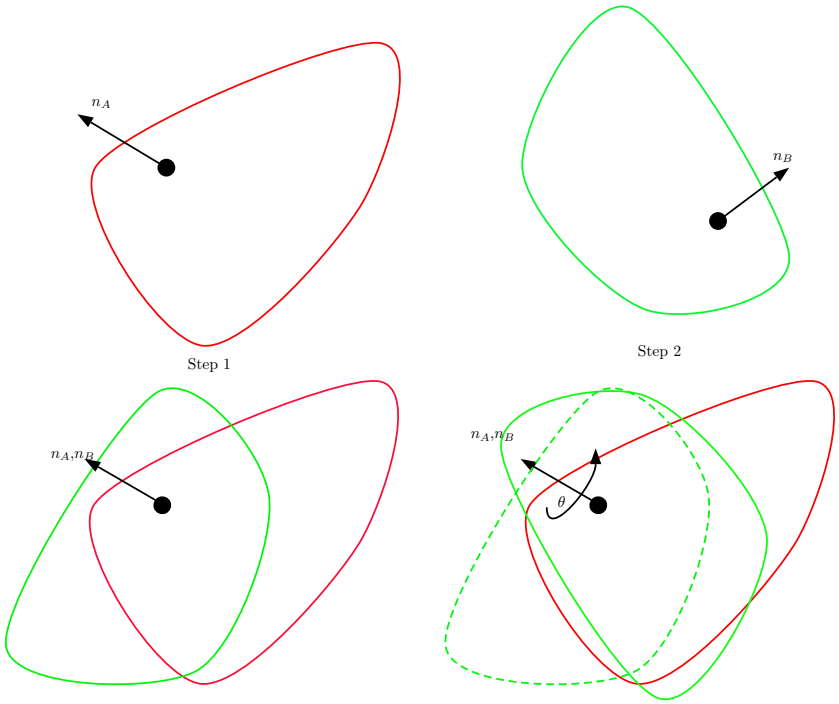

FIGURE 3.7: Point based registration method with two-step movement

those two normal vector.

$$\begin{vmatrix} u \\ w \\ v \end{vmatrix} = \begin{vmatrix} n_{x,x_i} \\ n_{y,x_i} \\ n_{z,x_i} \end{vmatrix} \times \begin{vmatrix} n_{x,y_j} \\ n_{y,y_j} \\ n_{z,y_j} \end{vmatrix} \tag{3.4}$$

where n_{x,x_i}, n_{z,x_i}, n_{z,x_i} are values of normal vector of X at x_i and n_{x,y_j}, n_{z,y_j}, n_{z,y_j} are values of normal vector of Y at y_j in three dimensions.

The angle between those two normal vector ϵ is calculated as Equation 3.5.

$$\epsilon = asin(norm \begin{vmatrix} u \\ w \\ v \end{vmatrix}) \tag{3.5}$$

There is a case when $\epsilon = 0$. We do not do any further in this step. Otherwise, the

transformation which make two normal vector coincides is presented as in Equation 3.6.

$$T = \begin{vmatrix} T_{11} & R_{12} & T_{13} & T_{14} \\ T_{21} & R_{22} & T_{23} & T_{24} \\ T_{31} & R_{32} & T_{33} & T_{34} \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{3.6}$$

where

$|u', v', w'|'$ is vector $|u, v, w|$ after normalization. $T_{11} = u'^2 + (v'^2 + w'^2)cos(\epsilon)$

$T_{12} = u'v'(1 - cos(\epsilon)) - w'sin(\epsilon)$

$T_{13} = u'w'(1 - cos(\epsilon)) + v'sin(\epsilon)$

$T_{14} = (x_{x_i}(v'^2 + w'^2) - u' * (y_{x_i}v' + z_{x_i}w'))(1 - cos(\epsilon)) + (y_{x_i}w' - z_{x_i}v')sin(\epsilon)$

$T_{21} = u'v'(1 - cos(\epsilon)) + w'sin(\epsilon)$

$T_{22} = v'^2 + (u'^2 + w'^2)cos(\epsilon)$

$T_{23} = v'w'(1 - cos(\epsilon)) - u'sin(\epsilon)$

$T_{24} = (y_{x_i}(u'^2 + w'^2) - v' * (x_{x_i}v' + z_{x_i}w'))(1 - cos(\epsilon)) + (z_{x_i}u' - x_{x_i}w')sin(\epsilon)$

$T_{31} = u'w'(1 - cos(\epsilon)) - v'sin(\epsilon)$

$T_{32} = v'w'(1 - cos(\epsilon)) + u'sin(\epsilon)$

$T_{33} = w'^2 + (u'^2 + v'^2)cos(\epsilon)$

$T_{34} = (z_{x_i}(u'^2 + v'^2) - w' * (x_{x_i}u' + y_{x_i}v'))(1 - cos(\epsilon)) + (x_{x_i}v' - y_{x_i}u')sin(\epsilon)$

After the first step, y_j changes to y_j^* and Y to Y^*.

Rotation around normal vector

The second movement is to rotate data point sets Y^* around current normal vector at y_j^* or x_i since they coincide now and calculate the best rotation angle. The transformation matrix is similar as Equation 3.6 with normal vector of X at x_i is the vector which data points rotate about.

Global searching algorithms take responsibility for searching the best corresponding point and rotating angle with each point as from Equation 3.7.

$$(\theta, j) = \underset{\theta \in [-\pi,\pi], j \in [0,n]}{\operatorname{argmin}} ||E_{x_i,y_j}(R,t)_\theta|| \tag{3.7}$$

where θ is rotating angle in the second step.

R, t are rotation and translation matrix of two steps included in transformation matrix T. After all, transformation matrix T could be calculated as Equation 3.8:

$$\mathbf{T} = T_{ntn} * T_{ran} \tag{3.8}$$

where T_{ntn} is transformation matrix which presents for movement to adjust two normals of data points. T_{ran} is transformation matrix which presents for rotation around normal vector. The flowchart for hybrid point based approach is presented in Figure 3.8.

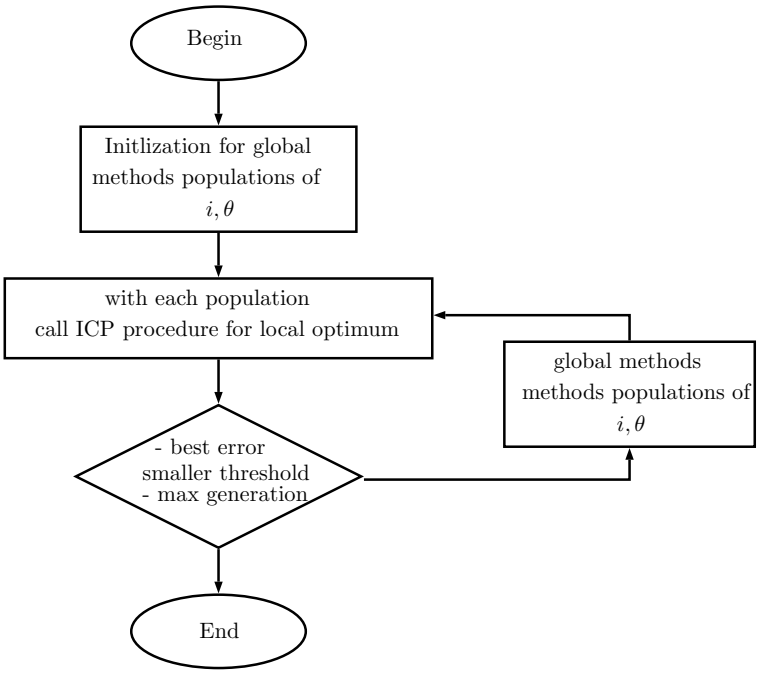

FIGURE 3.8: Implementation procedure of the hybrid approach

3.3 The new direct global approach

In this part, a new global direct registration method for 3D constructed surfaces captured by range cameras in cases with not close enough initialization is proposed.

- It eliminates the ICP algorithm from the registration process and thus becomes a direct method.

- As other global registration methods, the new method requires no local descriptors and operates directly on raw scanning data.

- The method uses the improved self-adaptive differential evolution (ISADE) algorithm [19] as a search engine to find the global minima as a direct method that does not use a fine registration procedure such as ICP.

- Furthermore, ray casting-based error calculation reduces the computation cost and runtime, because of the potential for using parallelized computation. CPU-based parallel computing procedures allow the algorithm to find the solution at a rate equivalent to the online rate.

With the newly developed global search algorithms, flattening using ICP inner loops in registration becomes redundant. Our method integrates a new global search algorithm, ISADE, which is suitable for complicated fitness functions when the flattening process is not performed, and a ray casting-based corresponding search method to accelerate the objective function calculation in the registration procedure.

3.3.1 Ray-casting for fast corresponding point determination on constructed range image

The KinectFusion algorithm, a real-time scene reconstructing pipeline, uses ICP as the only method for registering two continuous frames. The procedure requires a powerful GPU to speed up calculations and reduce runtime. However, global registration algorithms calculate a thousand times more error functions than ICP, and

thus, so that these algorithms can be applied on-line or using less powerful processors, faster error calculation methods must be included.

ICP algorithms use the kd-tree [5] structure to speed up the process of determining j^* in Equation 3.2. The complexity of the kd-tree nearest neighbor search algorithms is $O(log(n))$, where n is the set number of the search points. Figure 3.9 shows an example of the true closest corresponding points of the model and data pointsets.

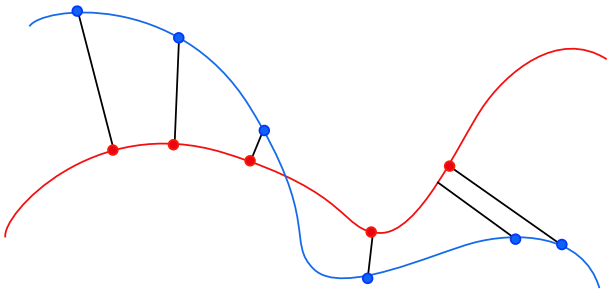

FIGURE 3.9: Closest corresponding point using kd-tree. Data points are in blue and model points are in red.

Ray casting [6] is one of the most basic of the many computer graphics rendering methods. The idea behind the ray casting method is to direct a ray from the eye through each pixel, and find the closest object blocking the path of the ray. Using the material properties and light effect in the scene, rendering methods can determine the shading of the object. Some hidden surface removal algorithms use ray casting to find the closest surfaces to the eye and eliminate all others that are at a greater distance along the same ray. The Point Cloud Library uses ray casting as a filtering method; it removes all points that are obscured by other points.

We apply ray casting to find the approximated closest point using a range camera model. Constructed range images or point-cloud data are frequently captured by a 3D range camera, where a range image can be considered a 2D gray image, G; the value of each pixel shows the depth of a point. To simplify the problem, we do not

take distortion into consideration.

$$z_{i,j} = G_{i,j} \tag{3.9}$$

where $z_{i,j}$ is the depth of the image at pixel column i and row j.

Equation 3.10 converts range image data points to real 3D depth data $\{x, y, z\}$ in R^3.

$$x_{i,j} = (i - cx)G_{i,j}/fx \tag{3.10a}$$

$$y_{i,j} = (j - cy)G_{i,j}/fy \tag{3.10b}$$

$$z_{i,j} = G_{i,j} \tag{3.10c}$$

where fx, fy, cx, and cy are the intrinsic parameters of the depth camera.

Inversely, pixel position i, j is to be calculated. Figure 3.10 shows the method's idea.

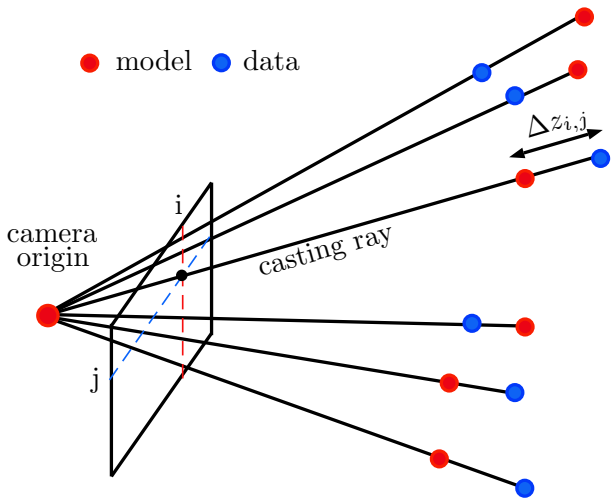

FIGURE 3.10: Ray-casting method for searching corresponding point

Using the corresponding points obtained in the ray casting step, we determine the depth difference $\Delta z_{i,j}$ for the next step of calculating the objective function for

the global search method, as

$$\Delta z_{i,j}(R,t) = \left\{ \begin{array}{cc} z_{i,j}^{X} - z_{i,j}^{Y^{R,t}} & if \ |z_{i,j}^{X} - z_{i,j}^{Y^{R,t}}| < thresthold \\ 0 & otherwise \end{array} \right\} \tag{3.11}$$

where R and t are the rotation and translation matrix, respectively, $z_{i,j}^{X}$ is the depth of the model pointset and $z_{i,j}^{Y(R,t)}$ is the depth of the data pointset after applying the rotation and translation matrix with i, j from the ray casting process.

The ray casting method is simple and fast (with a complexity of $O(1)$) and, more importantly, potentially parallel computing can be applied.

3.3.2 Objective function

Global optimization methods use fitness or objective functions to find the transformation that drives the fitness function to the smallest value. We propose a fitness function $F(R,t)$:

$$\mathbf{F(R,t)} = \mathbf{f(k)} \sum_{\mathbf{i=1}}^{\mathbf{n}} \sum_{\mathbf{j=1}}^{\mathbf{m}} (\mathbf{\Delta z_{i,j}(R,t)})^{\mathbf{2}} \tag{3.12}$$

where R and t are the rotation and translation matrix, respectively, m and n are the height and width of the image frame, and k is the inlier point number.

To gain a smaller error in a larger number of inlier points, we used an additional function $f(k)$:

$$\mathbf{f}(k) = \left\{ \begin{array}{cc} \infty & k < N/10 \\ (1 - k/N)/k^2 & k \geqslant N/10 \end{array} \right\} \tag{3.13}$$

where N is the number of points in the data pointset.

The ray casting-based method makes the algorithm run significantly faster than the kd-tree based approach. However, since a global search algorithm handles a large number of points at a huge computation cost, we take parallel implementation into consideration. Since in most computers a multi-core processor is available, using the CPU for parallel computing is convenient in most applications. In addition, CPU multi-core parallel implementation is even easier with OpenMP library [21]. Furthermore, the ray casting process adapts well to parallel computing, and the corresponding points can be calculated in different processes or threads. The whole procedure for the method is presented in Figure 3.11.

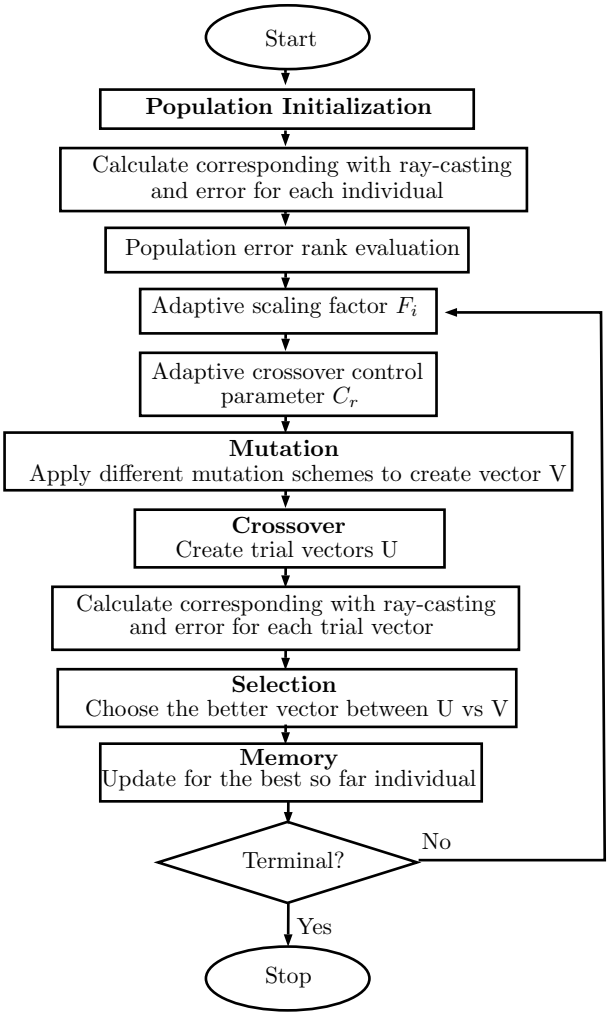

FIGURE 3.11: ISADE ray-casting implementation

Bibliography

[1] S. Marden, J. Guivant, Improving the performance of ICP for real-time applications using an approximate nearest neighbour search, in: Proceedings of Australasian Conference on Robotics and Automation, 3-5 Dec 2012.

[2] K. Lim Low, Linear least-squares optimization for point-toplane ICP surface registration, Tech. Rep. (2004).

[3] S. Bouaziz, A. Tagliasacchi, M. Pauly, Sparse iterative closest point, in: Proceedings of the Eleventh Eurographics/ACMSIGGRAPH Symposium on Geometry Processing, SGP '13, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2013, pp. 113–123. doi:10.1111/cgf.12178.

[4] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, A. Fitzgibbon, Kinect- Fusion: Real-time 3d reconstruction and interaction using a moving depth camera, in: Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology, UIST '11, ACM, New York, NY, USA, 2011, pp. 559–568. doi:10.1145/2047196.2047270.

[5] S. Chandran, Introduction to kd-trees, Tech. Rep., University of Maryland Department of Computer Science.

[6] S. D. Roth, Ray casting for modeling solids, Computer Graphics and Image Processing 18(2) (1982) 109–144. doi:http://dx.doi.org/10.1016/0146-664X(82)90169-1.

Chapter 4

Model-based Pose Estimation for Texture-less Objects

4.1 Introduction

In the last decade, object detection and recognition have gained significant improvement by using keypoint features [1] . Since geometric transformations and illumination changes has no effect on finding keypoints, they have been widely used for matching images from slightly different viewpoints [2]. Keypoint-based approaches work well in textured objects but texture-less objects. Textured objects have various keypoints, those have high potential appearing on both images. After finding keypoints, sample consensus such as RANSAC calculates the most suitable transformation of the object from reference position to current position. The more matched keypoints, the more accurate the transformation is. On texture-less objects lack of keypoint repeatability and stability on texture-less regions neither reduces the accuracy of sample consensus method nor leads to wrong results. Like keypoints, edges are also invariant to general geometric transformations and illumination changes [3]. Using edges are more suitable as a general approach even with texture-less objects. In early computer vision research, to find the best alignment between two edge maps, a given priori set of edge templates compare their suitability to the current

edge maps to draw the mot suitable transformation. The current proposed method of chamfer distance matching [4] enhances the cost functions enable for applying global searching algorithm into object tracking problem. Harris[5] and other proposed edge-based tracking systems [6] used edges and contours for visual tracking task. One drawback of using edges is that they are not distinctive enough to provide effective discrimination in complex background or occlusions, there have been efforts to enhance the previous one by unifying interest points or considering multiple but limited hypotheses on edge correspondences. For consideration of multiple hypotheses in a more general sense global searching algorithm should come into consideration. We propose an approach of using ISADE as the global searching method to continuously search for the 3D position of object in camera coordinate.

4.2 Methodology

Initialization is the most important step of tracking process. Following steps presents the implementation pipeline of the initialization:

- Canny edge image is employed to archive edge images from query images.

- Distance map or chamfer matching map is calculated from edge images.

- Adaptive Differential Evolution Algorithms search for the best fit pose of the object which create an 2D edge images fitted into the chamfer matching maps for initialization.

After initialization, narrower searching boundary is used to get the accurate results at on-line speed. If the cost function goes large, initialization is required.

4.2.1 Chamfer matching maps from query images

Canny Edge Detection

The white lines in Figure 4.1 are output of Canny[7] edge detection method from Figure 4.2. Canny method includes five different steps:

1. Apply Gaussian filter to smooth the image in order to remove the noise.

2. Find the intensity gradients of the image.

3. Apply non-maximum suppression to get rid of spurious response to edge detection.

4. Apply double threshold to determine potential edges.

5. Track edge by hysteresis: Finalize the detection of edges by suppressing all the other edges that are weak and not connected to strong edges.

OpenCV[8] library documentation gave us above implementation in detail[9].

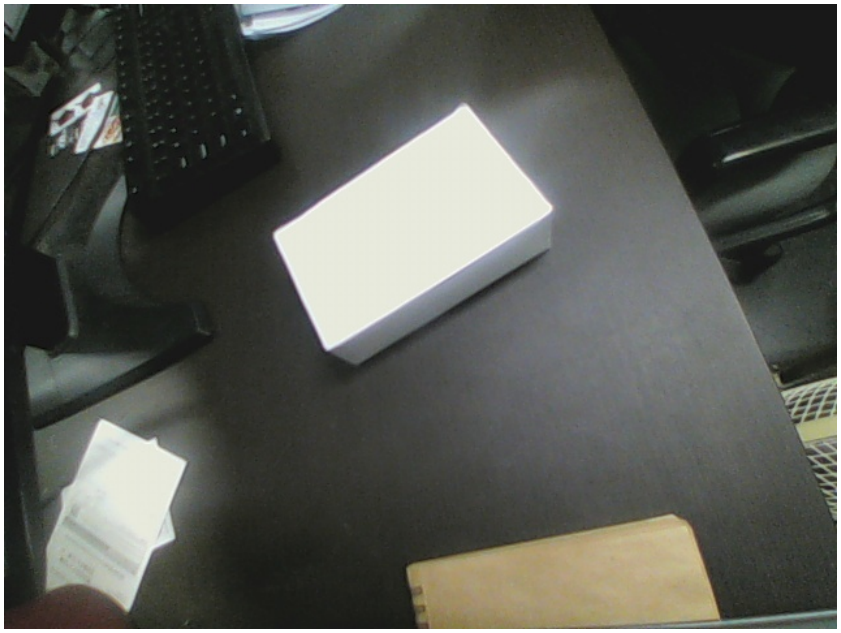

FIGURE 4.1: Input image

Chamfer Matching Maps

Building chamfer matching maps involves distance tranform method. The distance transform is an operator normally only applied to binary images. The result of the transform is a graylevel image that looks similar to the input image, except that the graylevel intensities of points inside foreground regions are changed to show the distance to the closest boundary from each point.

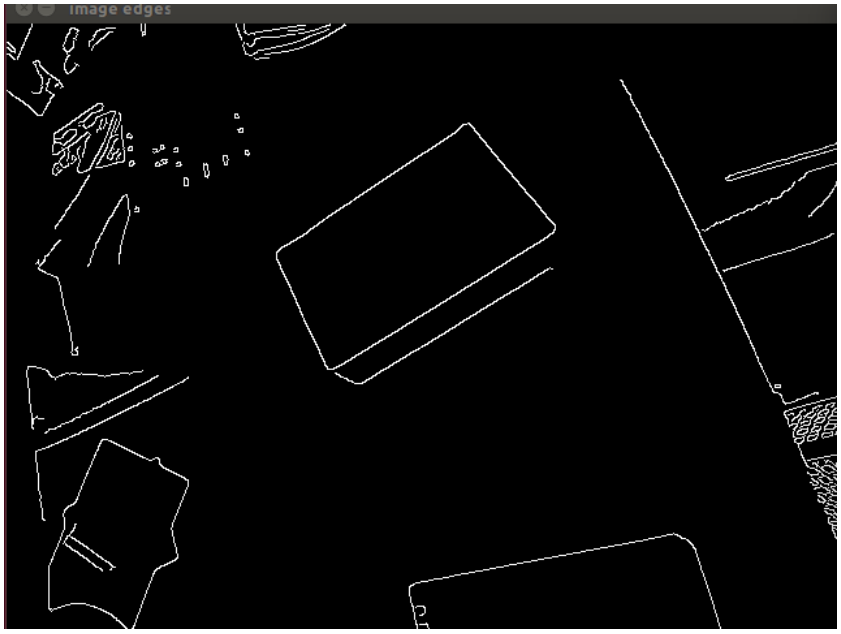

FIGURE 4.2: Edge image

One way to think about the distance transform is to first imagine that foreground regions in the input binary image are made of some uniform slow burning inflammable material. Then consider simultaneously starting a fire at all points on the boundary of a foreground region and letting the fire burn its way into the interior. If we then label each point in the interior with the amount of time that the fire took to first reach that point, then we have effectively computed the distance transform of that region.

There is a dual to the distance transform described above which produces the distance transform for the background region rather than the foreground region. It can be considered as a process of inverting the original image and then applying the standard transform as above. Figure 4.3 shows a distance transform for background region from edge maps in Figure 4.2.

4.2.2 Camera model and edges from CAD model

From a camera with prior-known configuration and object CAD model, we are able to archive ideal visible edges of objects by using camera model matrix. This matrix

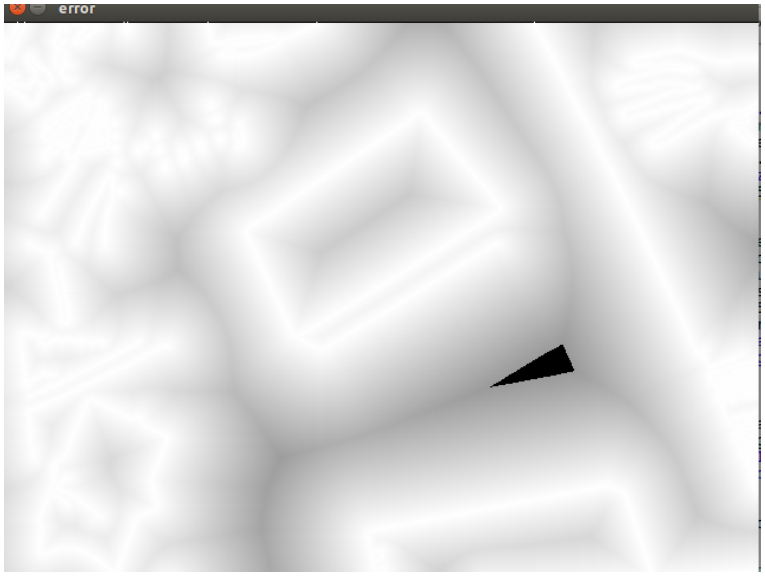

FIGURE 4.3: Charmfer matching map

convert a point with coordinate of (x, y, z) in real coordinate to a image point (u, v) as Equation 4.1 and Figure 4.4.

$$\begin{vmatrix} u \\ v \\ 1 \end{vmatrix} = \begin{vmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} \frac{x}{z} \\ \frac{y}{z} \\ 1 \end{vmatrix} \tag{4.1}$$

where f_x and f_y are focal length of camera on x and y dimension.

To determine visibility of object edges we use edge features based method from [10] . Figure 4.5 shows an edge between adjacent faces $A =< v_0, v_1, v_2 >$ and $B =< v_0, v_1, v_2 >$ with unit face normal n_A and n_B calculated as in Equation 4.2,4.3.

$$n_A = norm((v_1 - v_0) \times (v_2 - v_0)) \tag{4.2}$$

$$n_B = norm((v_3 - v_0) \times (v_1 - v_0)) \tag{4.3}$$

To determine the visibility of edge $E =< v_0, v_1 >$, we uses additional vector v_e with direction from v_0 to the camera center. E is visible edge if cross manipulation value

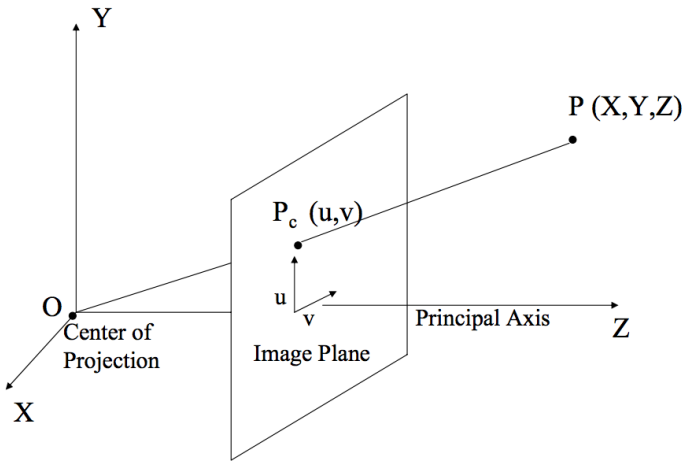

FIGURE 4.4: Pinhole camera model

of v_e with n_A or v_e with n_B is positive.

4.2.3 Initial pose searching

The cost function for global searching algorithm is a comparison result between edge maps from image and edge maps from CAD model. To gain a equivalent between ideal edge maps, a re-sampling step is applied, so the number of edge points in different ideal maps is set equally at N=200 points.

The error cost function is calculated as in Equation 4.4.

$$\mathbf{F}(\mathbf{R},\mathbf{t}) = \mathbf{f}(\mathbf{k}) \sum_{\mathbf{i=1}}^{\mathbf{m}} (\mathbf{E_i} - \mathbf{M_i})^{\mathbf{2}} \tag{4.4}$$

where $f(k)$ is function depended on number of inlier (k) as in Equation 4.5. E_i is value of real edge images at inlier i , M_i is value of CAD model edge images at inlier number i.

$$\mathbf{f}(k) = \left\{ \begin{array}{cc} \infty & if \quad k < N/10 \\ (1 - k/N)/k^2 & \quad k \geqslant N/10 \end{array} \right\} \tag{4.5}$$

The whole implementation steps are presented in Figure 4.6.

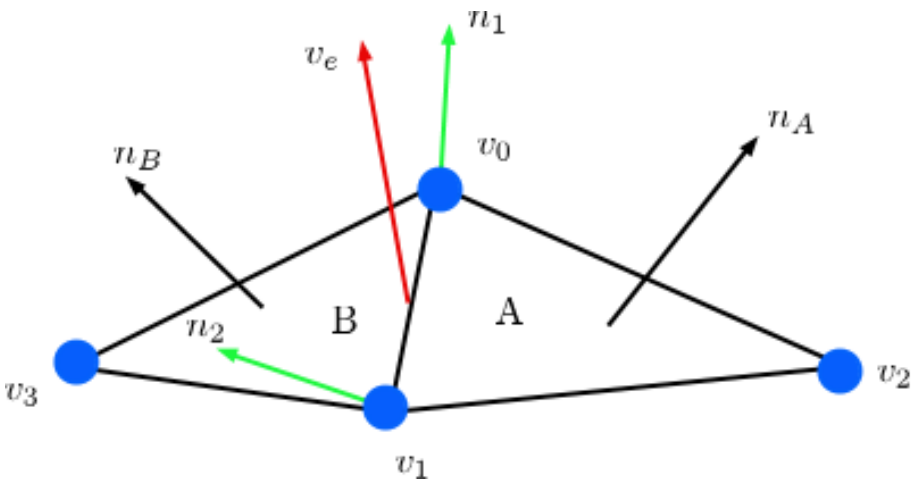

FIGURE 4.5: Visible edge identification

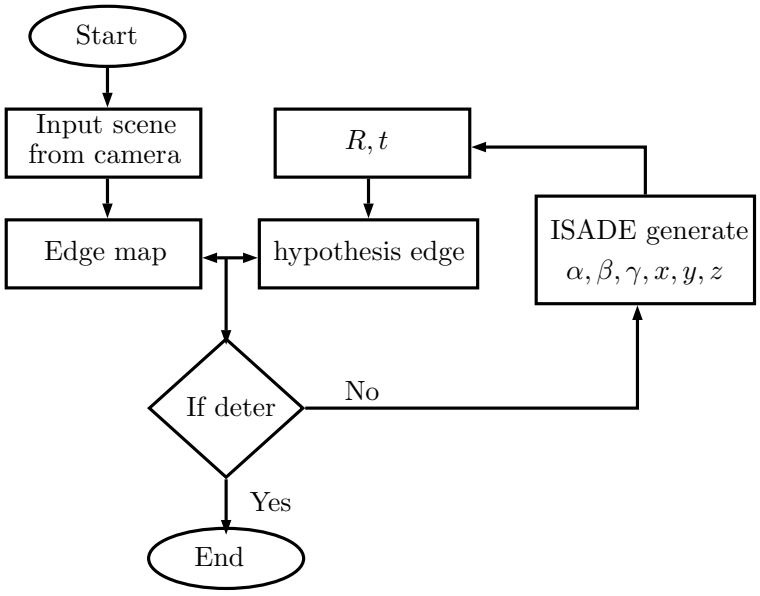

FIGURE 4.6: Visible edge identification

Bibliography

[1] [1] D. G. Lowe, “Distinctive image features from scale-invariant key- points,” IJCV, vol. 60, no. 2, pp. 91–110, 2004.

[2] Grauman and T. Darrell, “The pyramid match kernel: Discrimina- tive classification with sets of image features,” in ICCV, vol. 2, 2005, pp. 1458–1465 Vol. 2.

[3] Luc Vosters, Caifeng Shan, Tommaso Gritti, Real-time robust background subtraction under rapidly changing illumination conditions, Image and Vision Computing, Volume 30, Issue 12, December 2012, Pages 1004-1015, ISSN 0262-8856, http://dx.doi.org/10.1016/j.imavis.2012.08.017.

[4] H. Barrow, J. Tenenbaum, R. Bolles, and H. Wolf, “Parametric correspondence and chamfer matching: Two new techniques for image matching,” in IJCAI, 1977, pp. 659–663.

[5] C. Harris, “Tracking with Rigid Objects”. MIT Press, 1992.

[6] [7] A. I. Comport, E. Marchand, and F. Chaumette, “Robust model-based tracking for robot vision,” in IROS, vol. 1, 2004.

[7] Canny method http://docs.opencv.org/master/da/d22/tutorial_py_canny.html#gsc.tab=0

[8] G. Bradski, "opencv library", Dr. Dobb’s Journal of Software Tools, 2000

[9] OpenCV Edge Detection Implementation. http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/canny_detector/canny_detector.html

[10] Morgan McGuire and John F. Hughes, "Hardware-Determined Feature Edges", Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering, 2004, doi: http://doi.acm.org/10.1145/987657.987663

Chapter 5

Experiments & Results

5.1 Registration with Point based method

5.1.1 Experimental Setup

In this section, we presents our experiment and results using different algorithms of both point based and conventional approacches for different scanned surface data. The first category data are shown in Figure 5.1 which are avaible on The Stanford 3D Scanning Repository `http://graphics.stanford.edu/data/3Dscanrep`) including *Armadillo, Dragon, Stanford Bunny, Happy Buddha*. The second class is shown in Figure 5.2 which were downloaded from Queen's Range Image and 3-D Model Database `http://rcvlab.ece.queensu.ca/~qridb/QR3D/DatabasePagexyz.html` including *Old Gnome, Dinosaur, Green Pipe, Angel*. Stanford data format is **.ply* and **.wrl* is for Queen surface.

Scanning data were sub-sampled to smaller point number of 2000 points for a reasonable runtime. We use Meshlab software to subsample the scan surfaces, the sub-sampled data are showed in Figure 5.3 and Figure 5.4. The experiments arms to show the advantages in accuracy of point based approach coarse registration to conventional approach with six dimensions searching on scanning surfaces.

Algorithm 2 Hybrid Point Based Registration Algorithm

procedure SEARCHING ALGORITHM
    Initialize $X_i =$ with center point of Model pointset
    Initialization for populations with random values of (θ, j)
    while (Not reached stop criterion) **do**
        for the whole populations **do**
            Move the data surface to model surfaces using point based steps. After applying ICPs, remaining errors from Equation 3.1 are calculated.
        end for
        Sorting all populations.
        Update the best solution until the current step.
        Update for the next generation population from the current generation using suitable searching strategies of (SA, PSO, DE).
    end while
end procedure

procedure FINE REGISTRATION
    Using ICPs for $Best.solution.so.far$ from above Searching Procedure.
end procedure

FIGURE 5.1: Stanford scanning data

FIGURE 5.2: Queen range objects

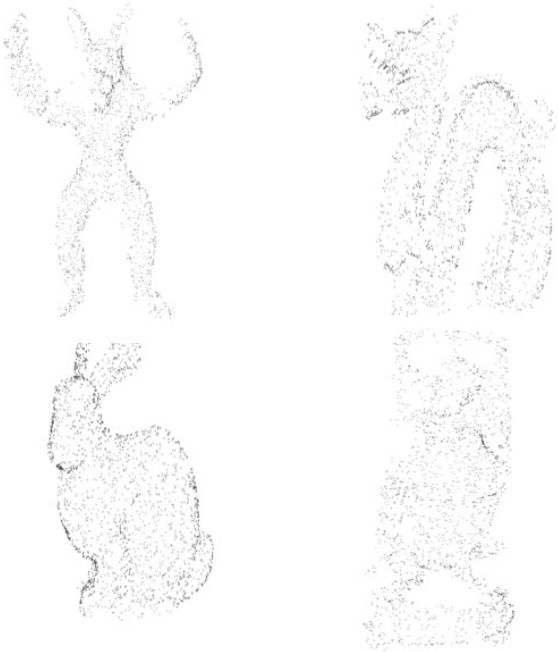

FIGURE 5.3: Stanford sub-sampled data

FIGURE 5.4: Queen sub-sampled range data

In point based algorithms, we did not set the limitation of points but the rotation angle $\theta \in [-\Pi/2, \Pi/2]$. In the conventional approach the boundaries of searching algorithm are $\alpha, \beta, \gamma \in [-\Pi/2, \Pi/2]$ and $x, y, z \in [-0.2, 0.2]$. Calculating time for error functions for algorithms were set to 2500 including 1250 searching loop of global search (1250 generations for SA, 50 generations for 25 population number with DE and PSO) and 2 ICP local minimization loops in each global search position.

Algorithm parameters are shown in Table 5.1 for both points based and parameter based algorithms. Algorithms are coded in C++, and test in a PC powered with a Intel i7 3.4GHz CPU processor.

Algorithms	SA	DE	PSO
Parameters	$\alpha = 0.996$	$F_0 = 0.8$	$elites = 1$
	$maxgen = 1250$	$C_r = 0.9$	$neighbors = 5$
		$DE/rand/1/bin$	$c1 = c2 = c3 = 2.1$
		$maxgen = 50$	$maxgen = 50$;
		$p = 25$	$p = 25$

TABLE 5.1: Evolutionary algorithms parameters

5.1.2 Experimental Results

The results presented as following arm to prove the superior of the point based registration to conventional six dimension based method. Moreover, those results gave us suggestions of a good integration for the robust hybrid algorithm. The results for Stanford dataset presents in Table 5.2 and Table 5.3 is for Queen dataset. The results are in four categories including: min, max, mean and standard deviation. The results in both tables are from coarse registration procedure in Algorithm 2. In both table, DEP, SAP and PSOP stand for Differential Evolution Point based, Simulated Annealing Point based and Particle Swarm Optimization Point based respectively. The better results are marked in bold.

The results in Table 5.2,5.3 in which all the new approach errors were in smaller means and standard deviations prove the superior of point based approaches over conventional approaches. Those results also suggest that, in three global searching algorithms, multiple agent algorithms such as DE or PSO are more suitable tool to integrate with than single agent algorithms like SA. Sometime, by chances,conventional approach could have smaller error. The reason is that the searching space is discrete for point based search and continuous for conventional search, if there are correct convergence in conventional methods, it could gain smaller errors.

Figure 5.5-5.7 visually show means and standard deviations of all methods' errors in different objects. The advantage of point based approach is clearly seen. In all objects, Gnome objects registration task is the most difficult. The object is lack of changing in shape every algorithms are false to find correct registration solutions.

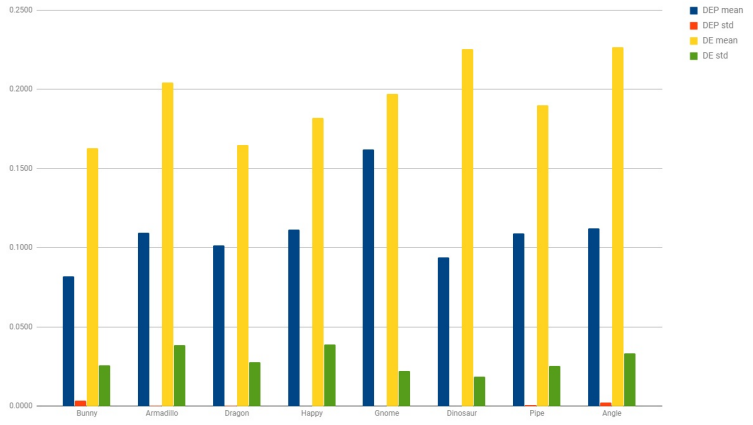

FIGURE 5.5: The results with mean and standard deviation of using DE as searching method in point based and conventional approach.

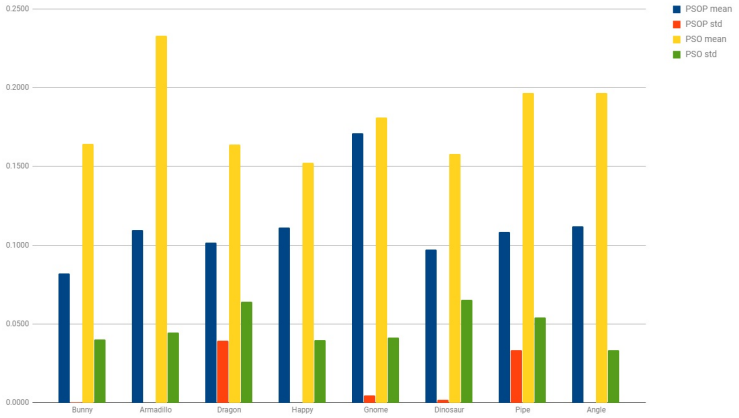

FIGURE 5.6: The results with mean and standard deviation of using PSO as searching method in point based and conventional approach.

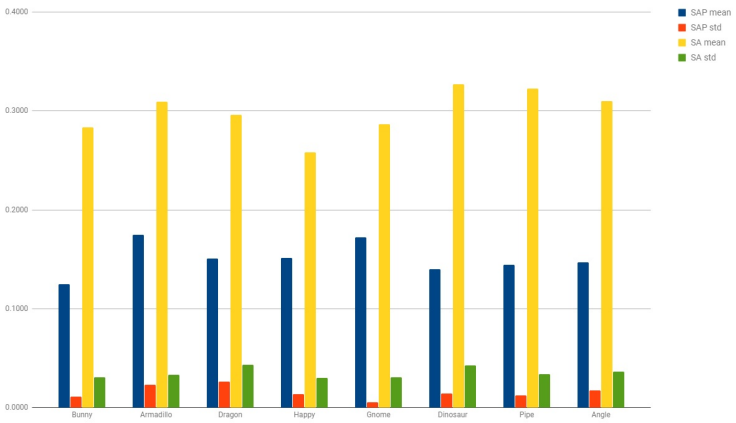

FIGURE 5.7: The results with mean and standard deviation of using SA as searching method in point based and conventional approach.

Dataset	Algorithm	Min	Max	Mean	St. dev.
	DEP	**0.1209**	**0.1472**	**0.1096**	**0.0001**
	DE	0.1565	0.2771	0.2144	0.0385
Armadillo	**SAP**	**0.1366**	**0.2011**	**0.1696**	**0.0230**
	SA	0.2559	0.3623	0.3160	0.0333
	PSOP	**0.1092**	**0.1095**	**0.1095**	**0.0001**
	PSO	0.1122	0.2601	0.2171	0.0446
	DEP	**0.1016**	**0.1020**	**0.1017**	**0.0001**
	DE	0.1340	0.2021	0.1683	0.0280
Dragon	**SAP**	**0.1219**	**0.1901**	**0.1507**	**0.0264**
	SA	0.2351	0.3869	0.2933	0.0432
	PSOP	0.1016	**0.2017**	**0.1255**	**0.0393**
	PSO	**0.0886**	0.2463	0.1618	0.0643
	DEP	**0.0616**	**0.0927**	**0.0830**	**0.0034**
	DE	0.1208	0.2167	0.1686	0.0258
Bunny	**SAP**	**0.1011**	**0.1368**	**0.1220**	**0.0111**
	SA	0.2129	0.3384	0.2812	0.0305
	PSOP	**0.0816**	**0.0831**	**0.0821**	**0.0004**
	PSO	0.0976	0.2113	0.1626	0.0402
	DEP	**0.1114**	**0.1114**	**0.1114**	**0.0000**
	DE	0.1331	0.2670	0.2670	0.1790
Happy Buddha	**SAP**	**0.1227**	**0.1701**	**0.1496**	**0.0137**
	SA	0.2349	0.3303	0.2678	0.0280
	PSOP	0.1114	**0.1114**	**0.1114**	**0.0000**
	PSO	**0.0712**	0.2046	0.1472	0.0399

TABLE 5.2: Comparison between point based and parameter based algorithm using Stanford scanning data

Dataset	Algorithm	Min	Max	Mean	St. dev.
	DEP	0.1619	**0.1620**	**0.1619**	**0.0001**
	DE	**0.1408**	0.2107	0.1863	0.0222
Gnome	**SAP**	**0.1649**	**0.1805**	**0.1721**	**0.0056**
	SA	0.2497	0.3322	0.2873	0.0310
	PSOP	0.1619	**0.1733**	**0.1687**	**0.0047**
	PSO	**0.1166**	0.2315	0.1795	0.0415
	DEP	**0.0938**	**0.0939**	**0.0939**	**0.0001**
	DE	0.2034	0.2666	0.2256	0.0186
Dinosaur	**SAP**	**0.1168**	**0.1626**	**0.1404**	**0.0143**
	SA	0.2563	0.3867	0.3205	0.0427
	PSOP	**0.0938**	**0.0972**	**0.0959**	**0.0017**
	PSO	0.1287	0.3281	0.1740	0.0692
	DEP	**0.1086**	**0.1099**	**0.1092**	**0.0005**
	DE	0.1742	0.2549	0.1960	0.0253
Green Pipe	**SAP**	**0.1249**	**0.1668**	**0.1467**	**0.0123**
	SA	0.2658	0.3740	0.3189	0.0341
	PSOP	0.1086	**0.2150**	**0.1195**	**0.0336**
	PSO	**0.0946**	0.2491	0.1824	0.0542
	DEP	**0.1042**	**0.1123**	**0.1113**	**0.0025**
	DE	0.1623	0.2519	0.2120	0.0333
Angle	**SAP**	**0.1152**	**0.1764**	**0.1486**	**0.0173**
	SA	0.2790	0.4018	0.3197	0.0366
	PSOP	**0.1120**	**0.1120**	**0.1120**	0.0000
	PSO	0.1561	0.2606	0.1990	**0.0366**

TABLE 5.3: Comparison between point based and parameter based algorithm using Queen scanning data

Figure 5.8-5.11 give us some results visually where pointsets are in red and data pointsets are in green color. They are results after raw alignment from point based and the conventional approach with DE as searching algorithms. The new method results are in the left side and the conventional method results are in the right side of figures. All experimental objects, the visual and statistical results in Table 5.6-5.7 agree with each other. The new methods gave the better performs than other conventions methods. Excepts for Gnome dataset, every methods of two approaches failed to find the good alignment.

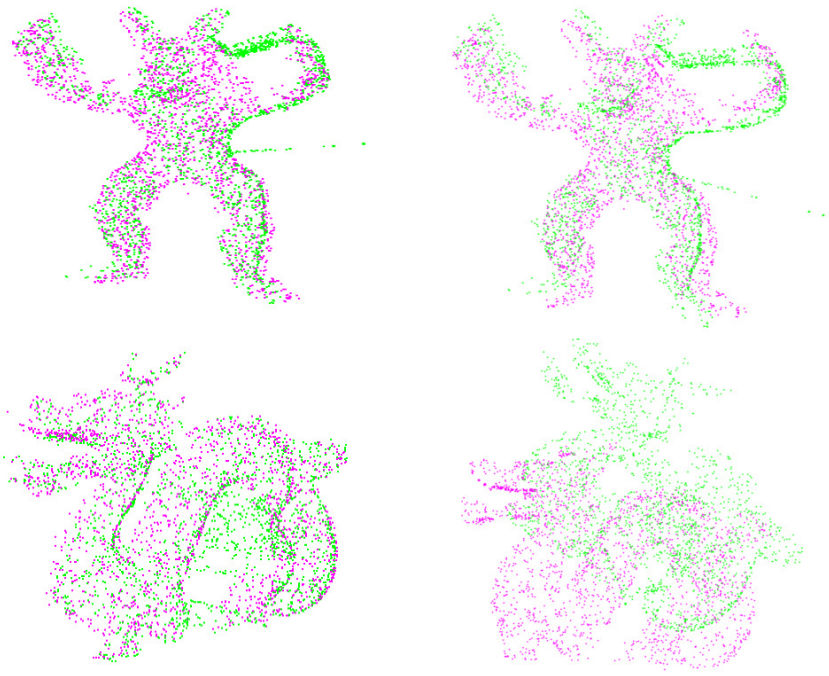

FIGURE 5.8: Alignment results of Armadillo and Dragon dataset

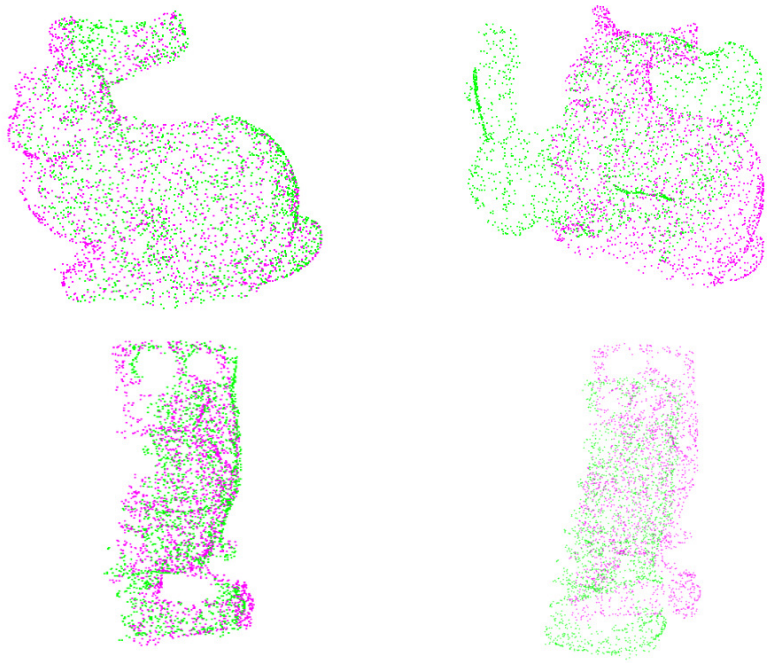

FIGURE 5.9: Alignment results of Bunny and Happy Buddha dataset

FIGURE 5.10: Alignment results of Gnome and Dinosaur dataset

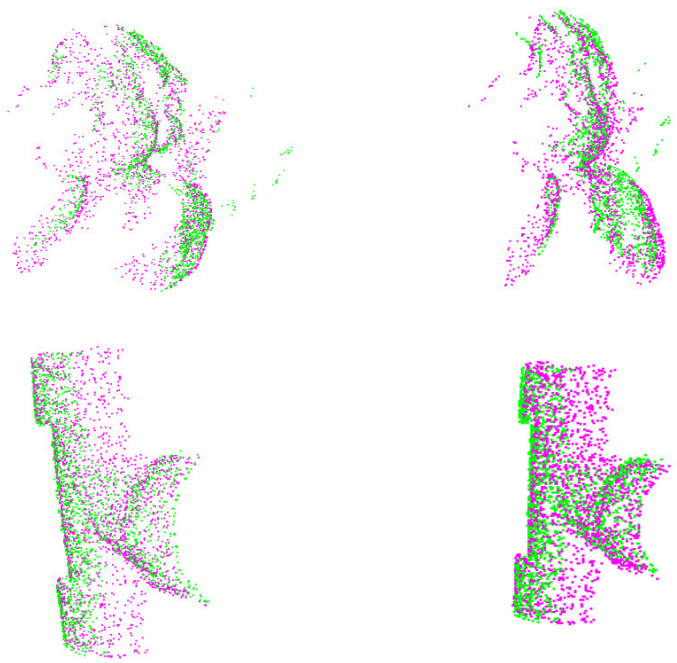

FIGURE 5.11: Alignment results of Green Pipe and Angle dataset

5.2 Ray-casting method with ISADE

This section describes experiments that were conducted using the proposed method in real range image data registration and presents the results. We integrated different global search methods with the ray casting-based algorithm in order to obtain a comparison between ISADE and the state-of-the-art methods as follows.

1) SA proposed in Luck et al.'s paper, Registration of range data using a hybrid simulated annealing and iterative closest point algorithm.

2) Particle swarm optimization (PSO) proposed in Talbi et al.'s paper, Particle swarm optimization for image processing [1].

3) Genetic algorithm (GA) proposed in Valsecchi et al.'s paper, An image registration approach using genetic algorithms [2].

4) DE proposed in Falco et al.'s paper, Differential evolution as a viable tool for

satellite image registration [3].

We also calculated the ray casting-based error of the KinectFusion and Go-ICP algorithms for further comparison. All algorithms were implemented in C++ and compiled with GNU/g++ tool.

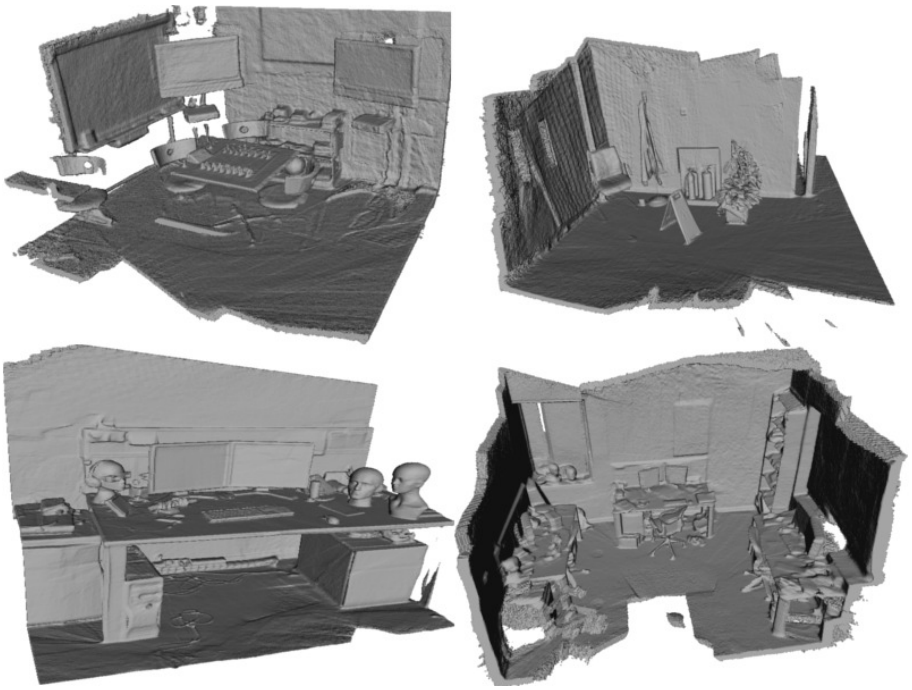

FIGURE 5.12: RGB-D Chess, Fire, Heads, Office Dataset for experiments

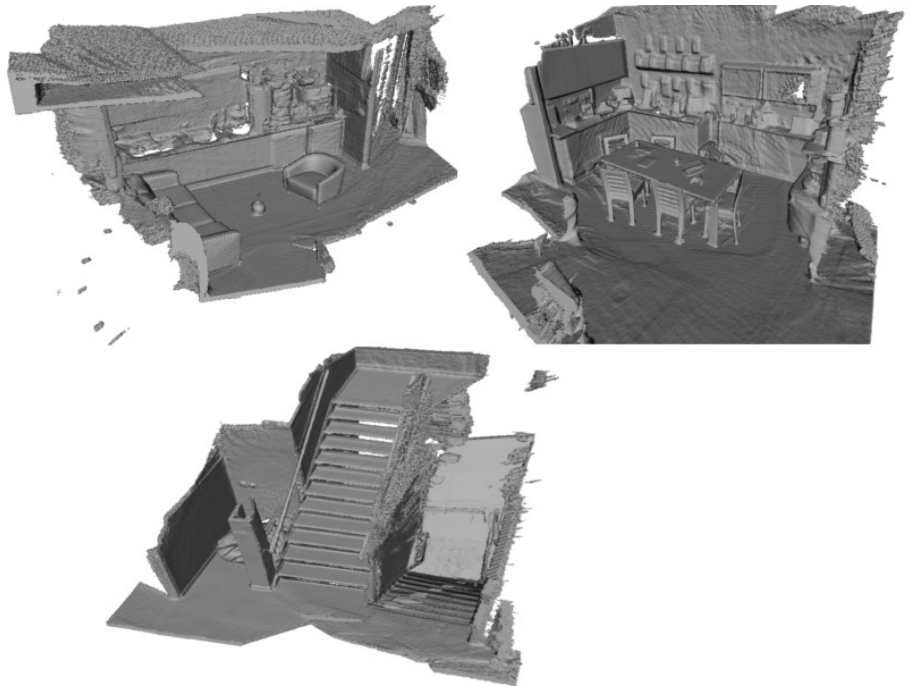

FIGURE 5.13: RGB-D Bumpkins, Red Kitchen, Stair Dataset for experiments

5.2.1 Range Image Dataset

In our experiments, a number of pair-wise registrations was conducted using well-known depth data, "RGB-D Dataset 7-Scenes", taken from the Kinect Microsoft Camera downloaded from the Microsoft Research Web site, `http://research.microsoft.com/en-us/projects/7-scenes/`. Specifically, Figures 5.12 and 5.13 show all the scenes: Chess, Fire, Heads, Office, Pumpkin, RedKitchen, and Stairs. The details of the data used in the registration experiments are as follows.

Chess dataset: image sequence 2, frame 960 vs frame 980.

Other datasets: image sequence 1, frame 000 vs frame 020.

These "PNG" format depth images are sub-sampled into a smaller resolution of 128×96, which is five times smaller than the original resolution of 640×480 in each dimension. The purpose of using a dataset with a smaller number of points is to achieve a suitable runtime while preserving robustness and accuracy.

5.2.2 Parameter Settings

For each method, thirty runs were performed. The search space had rotation angles and translation limited at $[-\pi/5, \pi/5]$ and $[-1, 1]$ separately. This means that the limitation of the rotation angles was 36 degrees and of the translation was 1 meter.

The algorithm parameters shown in Table 5.4 constitute the configuration for all the algorithms. All methods were run on a desktop PC powered with an Intel core I7-4790 CPU 3.60 GHz $\times$ 8 processor, 8 GB RAM memory and Linux Ubuntu 14.04 64-bit Operation System. The new algorithm C++ code was written based on reference from Andreas Geiger's LIBICP code [4].

TABLE 5.4: Algorithms configuration

Algorithm	DE	GA	SA	PSO	Go-ICP
parameters	$F_0 = 0.8$	$Pc = 0.95$;	$\alpha = 0.995$	$elites = 4$	$trimFraction = 0.0$
	$C_r = 0.9$	$Pm = 0.1$;		$neighbors = 5$	
	$DE/rand/1/bin$	$elites = 5$		$c1 = c2 = c3 = 2.1$	$distTransSize = 50$
maxgen	100	100	3000	100;	
population	30	30		30	subsample=1000 points

5.2.3 Comparison with KinectFusion algorithm

Accompanied by depth ranger images, "RGB-D Dataset 7-Scenes" provides homogeneous camera to world transposes at each frame calculated using the KinectFusion algorithm. We converted those camera transposes into transformation matrix between two frames as

$$T_i^j = T_i^{-1} * T_j \tag{5.1a}$$

$$T_i^j = \left[\begin{array}{ccc|c} & R_i^j & & t_i^j \\ & & & \\ 0 & 0 & 0 & 1 \end{array}\right] \tag{5.1b}$$

where T_i^j is the transformation matrix to move frame j to align with frame i, T_i and T_j are the homogeneous transpose matrix for the camera at frame i and j, respectively, and R_i^j and t_i^j are the rotation and translation matrix of T_i^j, respectively.

R_i^j and t_i^j are applied to ray casting error calculation methods for two frames, as in Equation 3.12, to describe the errors of the KinectFusion algorithm. Table 5.5 presents the mean errors of the proposed method in comparison with the error of the KinectFusion algorithm. The significantly smaller mean errors of the proposed method prove its superiority to the KinectFusion algorithm registration pipeline.

TABLE 5.5: Error comparison between new method, KinectFusion and Go-ICP algorithms

	Chess	Fire	Heads	Office	Pumpkin	RedKitchen	Stairs
Our method	**0.10230**	**0.03179**	**0.01000**	**0.03096**	**0.05563**	**0.03481**	**0.00883**
KinectFusion	22.37200	0.24311	2.99067	3.85941	0.11136	0.09836	0.01561
Go-ICP	nan	0.825212	0.01832	0.358507	inf	1.5387	2.28615

Figures 5.14 and 5.15 visually show the registration results of the proposed algorithm for the seven scenes in center and those of KinectFusion on the left hand side, to provide a visual comparison. The seven scenes included are Chess, Fire, Heads, Office, Pumpkin, RedKitchen, and Stairs. Model pointsets are colored red and data pointsets are colored green.

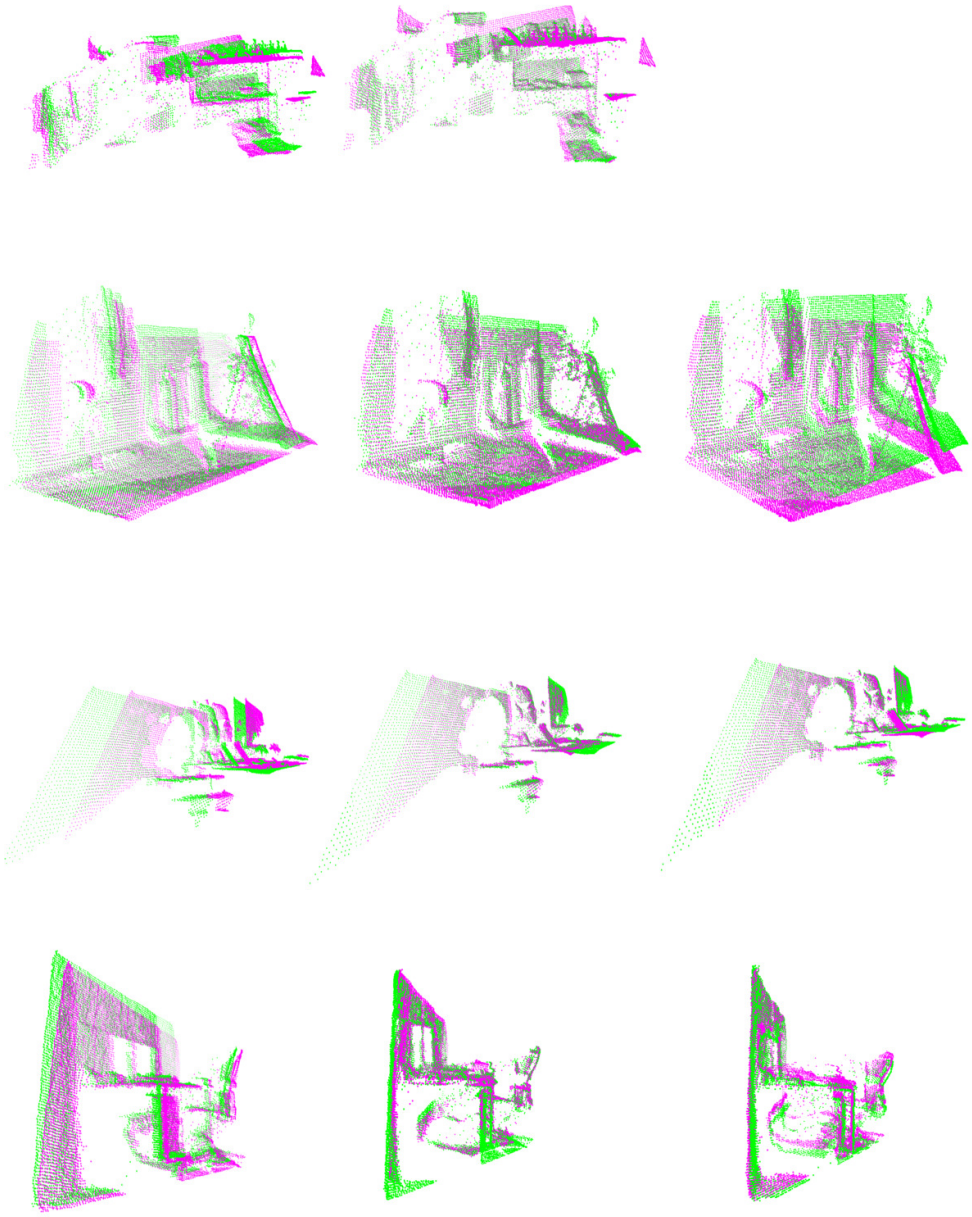

FIGURE 5.14: First 4 scenes (Chess, Fire, Heads, Office) registration output example. KinectFusion results are in the left hand side, the new algorithm's results are in the center and Go-ICP algorithm's results are on the right hand side.

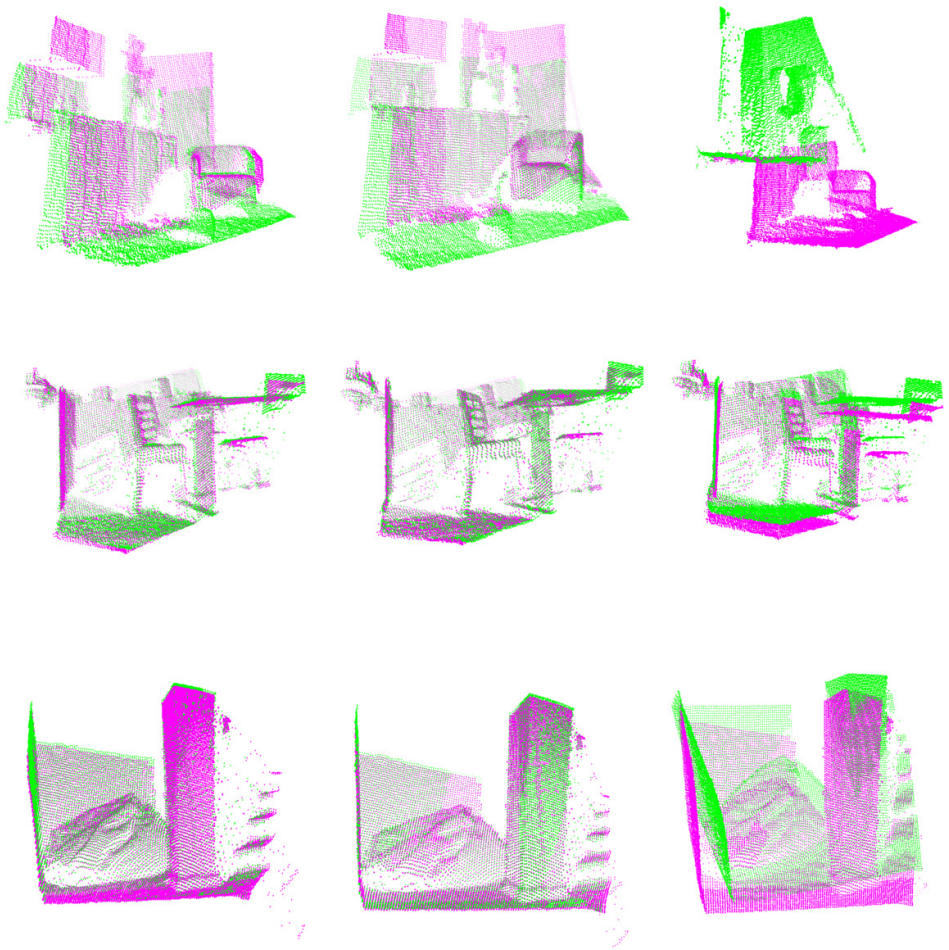

FIGURE 5.15: Last 3 scenes (Bumpkin, RedKitchen, Stairs) registration output example. KinectFusion results are in the left hand side, the new algorithm's results are in the center and Go-ICP algorithm's results are on the right hand side.

In these figures, that the proposed algorithm outperforms KinectFusion is clearly seen. Even in the best case of KinectFusion, such as Stairs or RedKitchen, the overlapping regions, where the two colors are mixed together, are not as clearly seen as in the results of the proposed algorithm.

An example of applying the new method to consecutive localizations can be seen in Figure 5.16. The pumpkin 3D scene, which is built from seven different range

images (frame 000, 020, ..., 120), visually shows the accuracy of the proposed method at various percentages of overlapping regions. The different frames are in different colors. A video at `https://www.youtube.com/watch?v=sgaUry5qsxU` gives a clearer view.

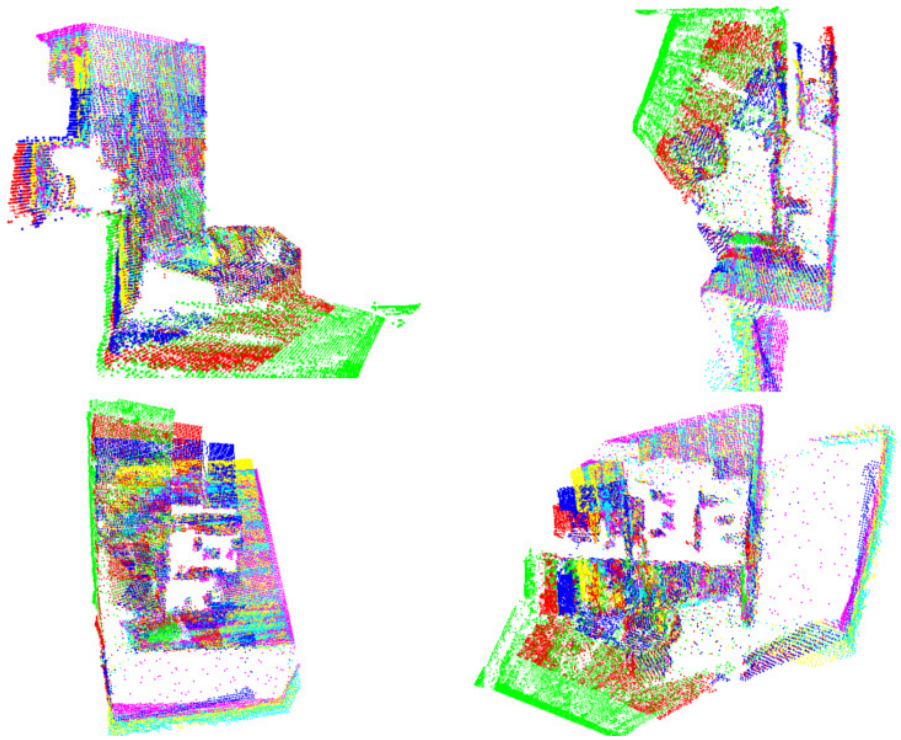

FIGURE 5.16: Office scene reconstructed results from different view angles

5.2.4 Comparison with Go-ICP algorithm

From authors contributed code[5], we performed experiments to compare our method with Go-ICP on accuracy, run time and robustness. Go-ICP configuration parameters were set as in Table 5.4 with the identical searching boundary with other methods. $distTransSize$ is the number of nodes in translation searching boundary. It was set to 50 or translation resolution is at $40\ mm$. Raising accuracy by increasing $distTransSize$ to 500 or $4\ mm$ resolution effort failed due to infinite runtime. Go-ICP were able to register Heads and Office datasets at $distTransSize$ of 100 with run

time presented in Table 5.8.

The disadvantage of big resolution could be compensated by inner ICP loops, however, the smaller resolution the more accurate the algorithm is. We set the data subsample to 1000, Go-ICP reaches infinitive runtime at the original 128×96 resolution.

Together with KinectFusion and our method errors, Table 5.5 presents the mean errors of Go-ICP algorithm where "nan" stands for undefined result in the case of infinitive runtime and "inf" stands for wrong convergence with few overlaped points. Over all, only Heads and Office showed good convergence with small error and runtime. However, those small errors are still bigger than the new method's.

Figures 5.14 and 5.15 also show the registration results of Go-ICP algorithm on the right side together with new method results in the center and KinectFusion algorithm result on the left side. From those figures, the new method better performance is clearly seen. In the case of RedKitchen dataset, the wrong convergence results of Go-ICP was observed, the error was small because of small over-lapsed percentage.

Average run time for Go-ICP on different datasets are presented in Table 5.8 where average run times of the new algorithm at different generation numbers are presented. In the table, "inf" values stand for infinitive run time. Go-ICP was fast in case of Heads dataset or extreme slow for the case of Chess dataset.

Over all, the new methods outperformed Go-ICP on experiments datasets in accuracy, runtime, and robustness.

5.2.5 Comparison between different optimization algorithms

Tables 5.6 and 5.7 show the experimental results of all the integrations and methods in four categories: min, max, mean, and standard deviation.

TABLE 5.6: Results of Chess, Fire, Heads and Office datasets

Scene name	Algorithm	Min	Max	Mean	St. dev.
Chess	**ISADE**	**0.10047**	**0.11187**	**0.10230**	**0.002821482**
KinectFusion	DE	0.17453	3.92808	0.29860	0.112087291
ref: 22.372	GA	1.44923	1.80180	2.53723	0.691936150
	SA	1.11736	2.55157	1.65871	0.400817542
	PSO	1.19899	2.58186	1.72316	0.459892382
Fire	**ISADE**	**0.03169**	**0.03196**	**0.03179**	**8.70855E-005**
KinectFusion	DE	0.03873	0.26059	0.10263	0.066038287
ref: 0.243112	GA	0.22177	3.93133	1.58268	0.913837133
	SA	0.15060	0.88670	0.45855	0.249700426
	PSO	0.11158	0.63419	0.34592	0.151824890
Heads	**ISADE**	**0.00994**	**0.01016**	**0.01000**	**7.01799E-005**
KinectFusion	DE	0.01276	0.06570	0.02205	0.012768061
ref: 2.99067	GA	0.47056	1.70316	0.97758	0.358190303
	SA	0.30740	1.01428	0.65404	0.264058658
	PSO	0.20801	1.88772	0.54401	0.463097716
Office	**ISADE**	**0.03084**	**0.03115**	**0.03096**	**8.39925E-005**
KinectFusion	DE	0.03195	0.06436	0.04373	0.009462166
ref: 3.85941	GA	0.24518	4.05346	1.88819	0.928751342
	SA	0.10385	2.67972	0.84426	0.720046753
	PSO	0.07169	2.08078	0.58507	0.686244921

TABLE 5.7: Results of Pumpkin, RedKitchen and Stairs datasets

Scene name	Algorithm	Min	Max	Mean	St. dev.
Pumpkin	**ISADE**	**0.05541**	**0.05603**	**0.05563**	**0.000175987**
KinectFusion	DE	0.06555	0.16927	0.11105	0.111050113
ref: 0.111361	GA	0.45803	3.15529	1.42922	0.775060060
	SA	0.07468	0.90335	0.49504	0.248322702
	PSO	0.11181	1.43345	0.36443	0.334116975
RedKitchen	**ISADE**	**0.03423**	**0.03759**	**0.03481**	**0.000915588**
KinectFusion	DE	0.05879	0.60304	0.17479	0.149183155
ref: 0.0983645	SA	0.52141	5.48133	2.07233	1.339500137
	GA	0.12508	1.58015	0.62601	0.441544434
	PSO	0.05515	2.48188	0.54354	0.671268667
Stairs	**ISADE**	**0.00875**	**0.00898**	**0.00883**	**0.000079463**
KinectFusion	DE	0.00975	0.04665	0.01767	0.009514675
ref: 0.0156084	SA	0.21207	2.24988	1.19252	0.627554990
	GA	0.01405	1.08881	0.29528	0.304574563
	PSO	0.04632	0.96723	0.25021	0.239971819

The smaller means and standard deviations for every dataset in comparison with the other methods show the accuracy and robustness of the new search engine as compared to the state-of-the-art search algorithms. In some cases, the experimental results show that the other integrations performed better than KinectFusion. The ICP accumulating error is the reason for this poor performance.

5.2.6 Iterations vs Convergence

Figure 5.17 we compare the robust results of convergence of the registration of the seven scenes for a small number of iterations between using ISADE and DE, where

the horizontal axis represents the iteration, and the vertical axis represents the error. In comparison with ISADE, DE required significant larger iteration number to achieve convergence. With ISADE, from 70 iterations, all the results show a flat trend and no new optimal solutions with a significant difference are found. This iteration number for DE is 120.

These results show that, if we reduce the maximum number of iterations to 70, the results remain the same. Clearly, the smaller the iteration number, the shorter is the runtime.

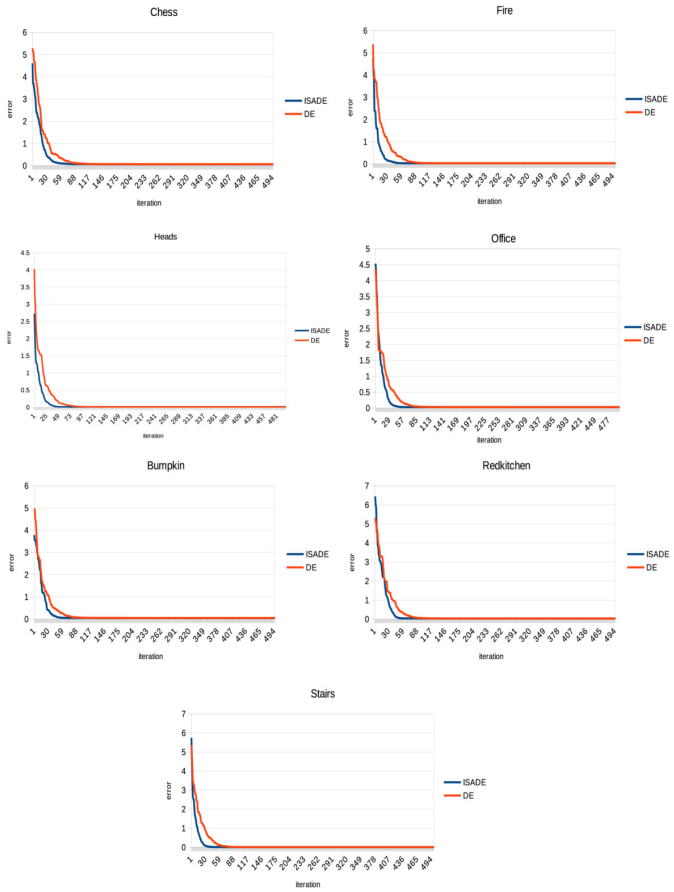

FIGURE 5.17: Fitness function as iterations of different datasets with ISADE in blue and DE in red color.

5.2.7 Results from registering in different movement patterns and frame distances

Figure 5.18 shows the values of rotation angles (α, β, γ) in radian and translation distances (x, y, z) in meter of 3D camera movement. Those values were obtained by using new algorithm to register range images from frame 001 to 060 respectively into the frame 000 of seq-01 in different datasets. The process stops if the movement values get over searching boundaries. From all datasets, we choose three typical movement of Chess, Fire and Heads datasets for rotating, sliding and forwarding with rotating movements respectively.

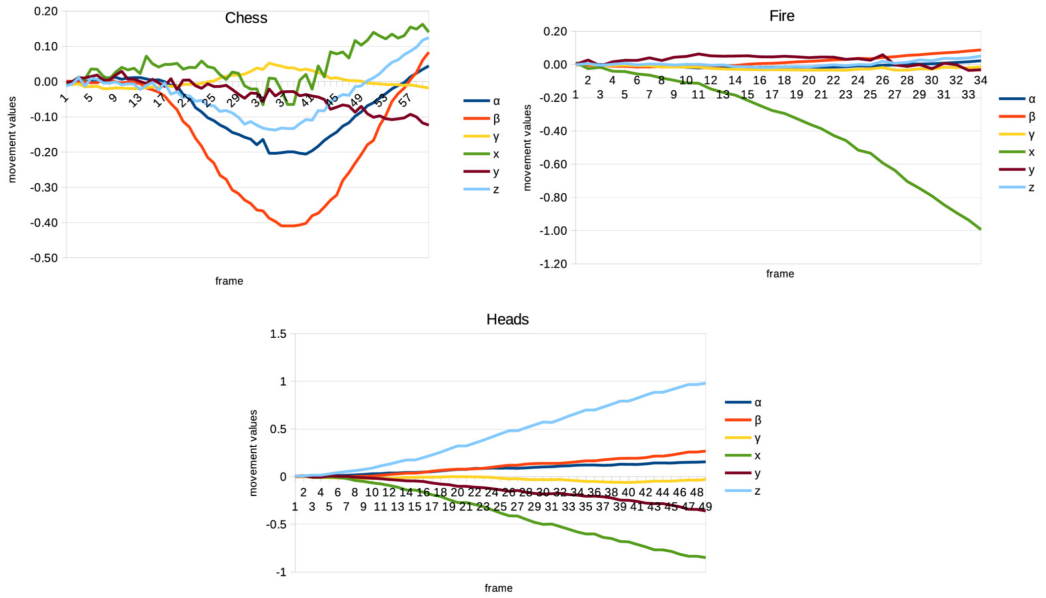

FIGURE 5.18: Movement pattern from Chess, Fire and Heads scenarios.

The results with no sudden value changing between two consecutive frames verify the feasibility of applying the new algorithm in registering range images of different movement patterns and frame distances.

5.2.8 Runtime

For the data of 128×96 resolution, average runtime for the proposed method are shown in Table 5.8. In the results, the average runtime for registration is around 0.6 s for 150 iterations of all scenes. Since the distance between two frames is 20, the registering equivalence rate is 33 frames per second (fps). At this rate, when we move the camera the algorithm are able to update the scenarios.

TABLE 5.8: Average running time (in second) on different scenes of new methods and Go-ICP

	New methods 100 generations	New method 150 generations	Go-ICP distTransSize=50	Go-ICP distTransSize=100
Chess	0.388414	0.516832	inf	inf
Fire	0.385928	0.625765	14.2786	inf
Heads	0.335828	0.562451	0.102944	0.104659
Office	0.378768	0.560734	0.030326	34.411
Pumpkin	0.410615	0.621756	104.468	inf
RedKitchen	0.415258	0.588466	30.3815	inf
Stairs	0.409834	0.597050	188.205	inf

By subsampling the data range image and remaining the model range image, the new algorithm gain smaller runtime while error level stays unchanged. Figure 5.19 shows the runtime at different level of subsample on the right hand and the errors in the left hand for the Redkitchen scenario.

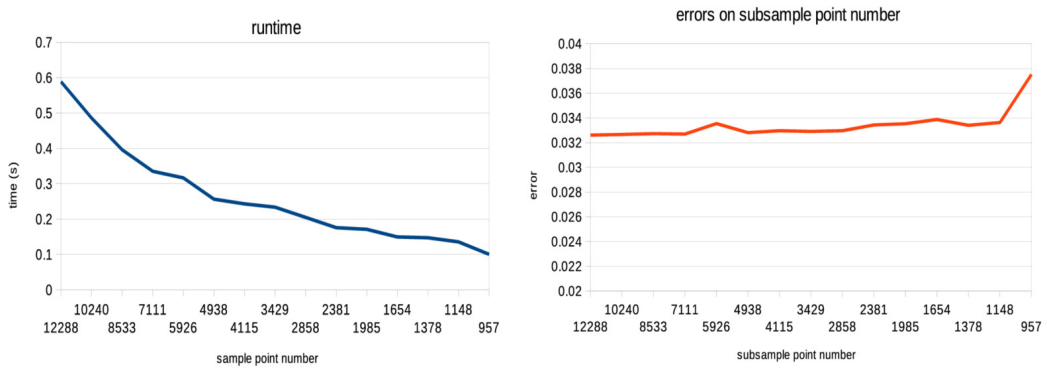

FIGURE 5.19: Runtime and error on subsample point numbers

5.3 Model based Texture-less Object Pose Estimation

To implement the algorithm, the system hardware included a iBuffalo BSW20KKM11BK camera. We used small box as a tracking object. All code is implemented on C++ code on a standard Desktop Computer powered with Intel Core i7-4790 CPU 3.6x8. In the experiment, we used the box object with size of $145 \times 95 \times 40$ (mm) in white colour as the tracking object. The object "*.ply" extension is use the input model. The searching boundary for the objects translation are in $[-100, 100]$ and $[-\pi/2, \pi/2-]$ for rotation angles of roll-pitch-raw. Parameters for ISADE searching algorithm are set as for range image registration problem to prove its adaptive property.

Fig 5.20 and Fig 5.21 show results from Canny edge detection method.

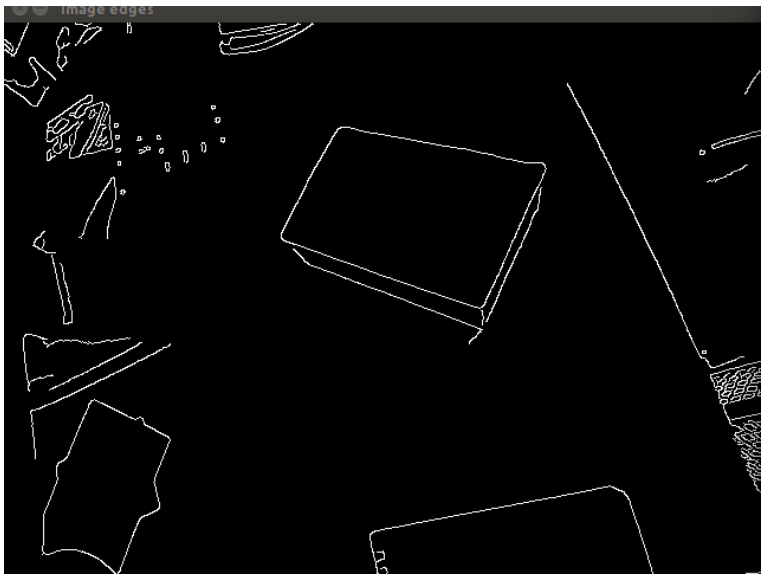

FIGURE 5.20: Edge detection result

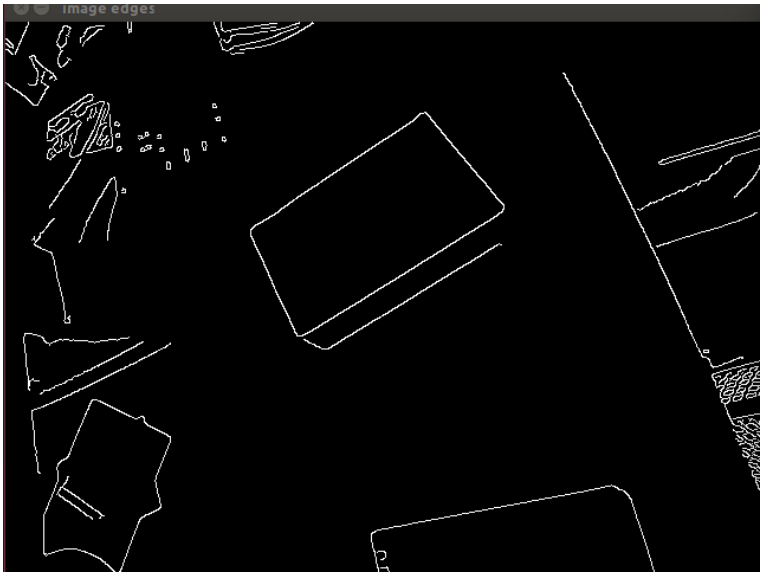

FIGURE 5.21: Edge detection result

Figure 5.22 and Figure 5.23 show results of boundary search of the box with different positions from Figure 5.20 and Figure 5.21. The box boundary is in blue colour in the left image. The result images showed that, the method were able to find the position of the object so the boundary could fit into the edge maps.

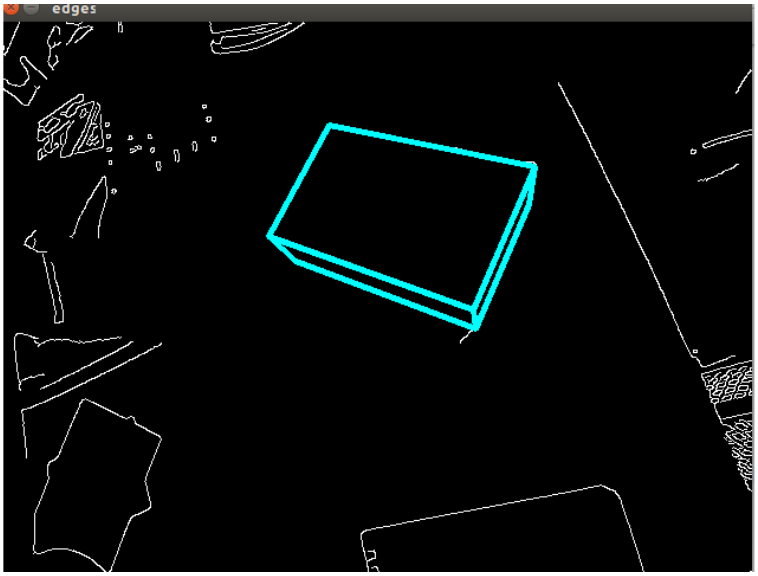

FIGURE 5.22: Tracking result

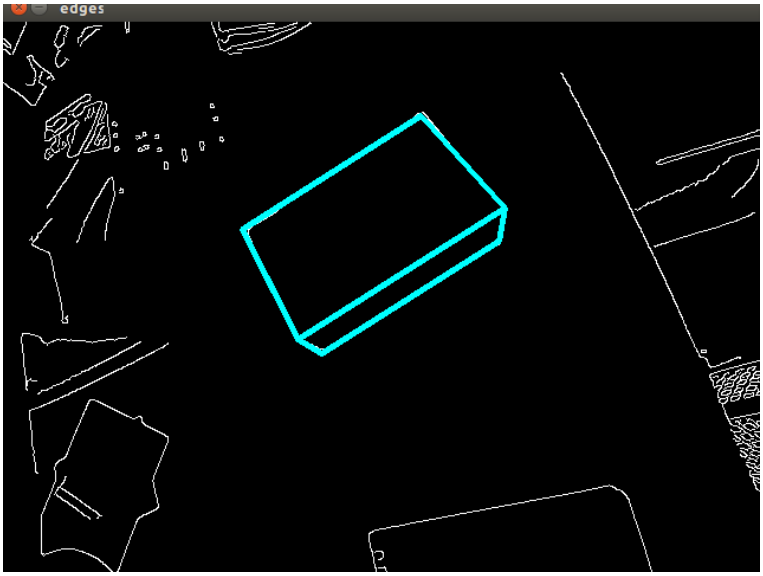

FIGURE 5.23: Tracking result

Depend on the objects, symmetric or non-symmetric, we need to set different searching boundaries for the searching algorithm (DE). Table 5.9 shows consuming time depend on number of population of DE.

TABLE 5.9: Run time on population size

Popsize	400	300	200	100	20	10
ms	2817	2190	1404	741	161	76

Bibliography

[1] Y. Wei Chen, A. Mimori, C.-L. Lin, Hybrid particle swarm optimization for 3-d image registration, in: 16th IEEE International Conference on Image Processing (ICIP), 2009, pp. 1753–1756. doi:10.1109/ICIP.2009.5414613.

[2] F. L. Seixas, L. S. Ochi, A. Conci, D. M. Saade, Image registration using genetic algorithms, in: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, GECCO '08, ACM, New York, NY, USA, 2008, pp. 1145–1146. doi:10.1145/1389095.1389320.

[3] I. D. Falco, A. D. Cioppa, D. Maisto, E. Tarantino, Differential evolution as a viable tool for satellite image registration, Applied Soft Computing 8(4) (2008) 1453–1462, Soft Computing for Dynamic Data Mining. doi:http://dx.doi.org/10.1016/j.asoc.2007.10.013.

[4] Iterative closest point implementation C++ code, http://www.cvlibs.net/software/libicp/, accessed: 2017-01-15.

[5] Go-ICP implementation C++ code, http://iitlab.bit.edu.cn/mcislab/~yangjiaolong/go-icp/, accessed: 2017-01-15

Chapter 6

Discussion and future directions

6.1 Point based Methods

The proposed hybrid point based approach to continues tackling a well-know challenging computer vision task, i.e, the registration problem for range data. The conducted experiments on different objects and datasets show the significant improvements toward having high-quality registration outcomes. Moreover, integrating between new approach with different evolutionary searching algorithm suggests a good combination with Differential Evolution. What is more important is that, this approach do not require any initial configuration related to first position of registering range datasets.

Besides good results, the method has its limitations. The method uses the surface's median point of model pointset as based point in assumption that this median point appears on dataset. That means, overlap region rate is limited at about 50%. Besides, currently normal vector at each points calculation relies on the closest pointset of sub-sampling datasets which leads to some changing in direction of normal vector. This problems can solve by preserving normal vector in sum-sampling process.

Furthermore, using original kd-tree structure to find nearest closest point for

ICP algorithm enlarger computation cost and run time. In future, fast approximated nearest neighbor searching algorithm with [1] library empowered with CPU or Graphic Card multi-core processing could be used to replace the current method.

Lastly, there are many new integrated evolutionary algorithms which have been published and proved to be far better than DE algorithm such as ISADE, APDEGA[2], AP/PSODEGA[3], etc. Those algorithms give us abundant choices of hybridizing with potential of much higher accuracy and robustness in registration task.

6.2 Global Ray-casting Method

We proposed a novel registration method in which a fast ray casting-based error calculation is integrated with a powerful self-adaptive optimization algorithm. The experimental results showed that ISADE is able to find a robust and accurate transformation matrix, while the ray casting method is fast and efficient in calculating error for global registration problems.

A more important point is that, by eliminating inner ICP loops in hybrid integrations and fine tuning procedures applied in previously proposed methods, the newly proposed method becomes the first direct, as well as the first online potential, global registration algorithm. Its robustness and accuracy were tested and verified in real 3D scenes captured by a Microsoft Kinect camera.

Currently, the algorithm is implemented using a CPU parallel procedure. In future work, the new algorithm can be implemented on a GPU to reduce its runtime and error while retaining its accuracy and robustness. Furthermore, the method can be extended for general point clouds from different sources by using a virtual camera surface and presenting it as a constructed surface. The proposed method is also potentially suitable for super resolution range images.

6.3 Model based Texture-less Object Pose Estimation

Object tracking has been always a challenging task in computer vision. Recently, evolution based global searching methods have proved its potential of tackling the tracking problem with ability of finding robust and accurate global optima solutions. We proposed a novel approach of using ISADE as a global searching method to find the best 3D position of objects. The experimental results showed promising results. ISADE was proved to be efficient working on different problems though further research need to be done.

In the future work, we would like to improve cost function with the method to narrow searching area for more accuracy but smaller generation of searching. By doing so, we expect to reduce the runtime but remains the accuracy and robustness.

Bibliography

[1] FLANN - "Fast Library for Approximate Nearest Neighbors". http://www.cs.ubc.ca/research/flann/.

[2] Hieu Pham, Tam Bui and H. Hasegawa, "Adaptive Plan system using Differential Evolution with Genetic Algorithm," 2013 IEEE International Conference on Industrial Technology (ICIT), Cape Town, 2013, pp. 40-45. doi: 10.1109/ICIT.2013.6505645

[3] Hieu Pham and H. Hasegawa, "Adaptive Plan System of Swarm Intelligent using Differential Evolution with Genetic Algorithm", Journal of Advanced Mechanical Design, Systems, and Manufacturing. vol. 7, no. 3, pp. 458-473, 2013. doi= 10.1299/jamdsm.7.458