An Automated Assembly of 3D Point Clouds using Coupling Matching and Path Planning Algorithm by Reinforcement Learning

Dianthika Puteri Andini[1,*] Muhammad Yusuf Fadhlan[1]

[1]*Department of Electrical Engineering, Bandung State Polytechnic, Bandung, Indonesia*
[*]*Corresponding author. Email: dianthika@polban.ac.id*

ABSTRACT
3D point cloud techniques have been playing critical roles in industrial automation. Applications focused on automated assembly are thus becoming important in manufacturing industry to solve some particular problems. To implement automated assembly, Reinforcement Learning is employed in for planning the optimized assembly path. The process structure is separated into training stage and testing stage. In the training stage, one of the objects is matched to an assembled model by using registration methods, RANdom SAmple Consensus and Iterative Closest Point, to determine the transformation between the two point clouds. To realize planning assembly path, the training is based on Q-learning method by reinforcement learning. In the testing stage, the optimized assembly path can be computed from the Q table obtained by training stage. The tested objects are three 3D point cloud data with the coupling matching part. The sockets which are female and male part are identified to match each couple part. Every object is firstly trained for every zone and next for testing. The working environment is a fixed with x, y, z coordinate which is converted to the position. The first object is trained and tested for all zones. The complete optimized assembly path planning task has been successfully achieved for representative objects. The feasibility and effectiveness of the proposed approaches have been validated by experiments.

Keywords: *3D Point Cloud, Assembly Path Planning, Coupling Matching, Object Assembly, Point Cloud Registration, Reinforcement Learning.*

1. INTRODUCTION

Point cloud is an essential thing for solving many tasks, in the fundamental of robotic system and 3D computer vision, classification, localization, mapping, object assembly, recognition, defect detection, and many kind of applications recently. Since 3D point clouds has the capability of providing greatly important information cues for analyzing objects and environments. It can be applied on many kind of application with different difficulty of each task. Simulation-based methods are becoming increasingly dominant in order to improve the productivity; it allows verification of installability and functionality before beginning the actual construction of implementation [1]. However, it is very challenging task if it is combined with machine learning [2] which there has been a few references study of machine learning with adding point cloud for automatically doing assembly. The simulation can be used to successfully learn some policies on purpose to explore a virtual world [3]. There is an approach to learn dynamic skills in high-dimensional state as input, some parameters input, and steps for action as output [4].

Some applications introduce solely learning algorithm based on RL like intelligent system without human assistance. So that algorithm only becomes the teacher to train the system to predict and make decision of the movement to give higher quality of next iteration [5]. If compared to supervise learning in artificial intelligence, reinforcement learning system is trained by experience to mimic human capabilities and attempt to fulfill what human cannot do.

There are challenges to understand compound behaviours like human being. For some applications in context of robotic simulations [6], observability proposes a new challenge to train in a simulation and real case. Specifically, by exploring the behaviours, it might cause damage in training phase for the real case.

1.1. Objective

The objective of this research is to perform object coupling matching assembly based on 3D point clouds. The developed algorithm is to be implemented using Python and the Open3D. The library can be accessed by everyone that is available for great software establishment that considers with 3D data. The research aims to apply the machine learning method to train the object automatically doing assembly process by using point cloud data. The chosen method is RL to achieve the goal of assembly part by using reward with the state and action. The learning algorithm will receive the reward signal to make a decision. Based on this, the algorithm modifies its strategy in order to obtain the highest reward. The proposed algorithm exhibits automatically couple part of the object using an assembled object as a model for the next processed into learning path for training and testing. Transformation from registration method can also be used to recognize which part of the object for assembly. In addition, path planning is applied by reinforcement learning to achieve the goal to reach the target. Path planning is an essential primitive for automation system that objects finds optimal between two points or objects. The optimal paths can illustrate as paths that reduce amount of movement or what kind of specific requirement from the system. It requires the environment to represent its location. The algorithm generates path from initial position to the target or it can be called as point-to-point trajectory

1.2. Paper Structure

The rest of the paper is organized as follows. Section 2 introduces the preliminaries used in this paper, which include 3D Point Cloud, Registration, and Reinforcement Learning. Then, the framework is extended system description and algorithm in Section 3. Section 4 develops experimental result and analysis. Finally, Conclusion and Recommendation concludes the paper and presents direction for future research.

2. BACKGROUND

2.1. 3D Point Cloud

A point cloud is a structure of data employed to illustrate a collection of multi-dimensional points and is general to be used for three dimensional (3D) data. In the system, there will be several objects which are each of them will have three point cloud data as fixed target, moving object called source, and a whole assembled object for doing registration. The moving object will be matched to the assembled object to get transformation matrix that will be used to assembly with the fixed target.

Feature Description, commonly, 3D local descriptors are used for specific applications such as object recognition, registration, and local surface classification. The information geometric of the entire 3D point cloud is encoded by global descriptor which is calculated for subsets of point cloud object [7]. Global feature is related to the whole model shape, whereas the neighborhood characteristics of the feature point are only encoded by local feature. Local feature is more suitable for partially matching overlapped point cloud compared to global feature [8]. State of the art of 3D point cloud descriptor evolution based on performance generally used in practice in term of robustness, effectiveness, and efficiency.

Fast Point Feature Histograms (FPFH) is principal description of a geometric identified point. Fast and easy to compute are the advantages of FPFH because of not too much detail. The Point Feature Histogram for a given point cloud is the conceptual calculation complexity. FPFH is used to determine the initial possible correspondences firstly [9].

2.2. Registration

Point cloud registration is a process to align two point clouds of data to get transformation. This matches object to consistent fixed model object. This involves two steps which are rough and fine registration. The rough registration has purpose to initiate an initial transformation between two point clouds. From this step, the process obtains the transformation matrix that can be used to know the matching object.

RANdom SAmple Consensus (RANSAC) is a method to fit data by inliers or in order to eliminate such outliers and extent matching process speed. There are two steps in which are the first step; the input dataset randomly selects a sample member set that involves minimal data items, and in the second step; the algorithm scrutinizes which parts of the whole dataset are compatible with the model represented by the approximated model variables acquired from the first step.

Iterative Closest Point (ICP) algorithm was introduced by Besl and McKay [10] which uses two point clouds alignment for point set registration method. Coarse estimation pose is firstly started, the iteration steps are followed to find the closest point next to correspondence of best transformation, and the last is to transform the dataset [11]. It is the method that most used as an accurate matching. It can enhance convergence speed without changing initial value [12].

2.3. Reinforcement Learning

Reinforcement Learning (RL) is an algorithm to learn how to do, how to plan actions, and in order to maximize a numerical reward value. The most essential thing is two characteristics of this algorithm which are trial and error to search and delay reward. This becomes a special distinguishing feature of reinforcement learning [13-14]. In fact, these algorithms can be unstable when applied at the same time with combination of general approximation system like system of memory-based [15]. It supposes an agent is located in an environment; an action is performed by the agent at each step [16]. The algorithm is based on a tuple of goal, state, action, reward iteratively based on fundamental system of Markov Decision Process, Q-learning, and Experience Replay.

3. ALGORITHM

3.1. Working Environment

Working Environment is a working space that point cloud data can be placed and moved or the environment that give feedback as state from action of object. A working environment is set up as 5×5×5 XYZ location like a cubic frame. Figure 1 shows details of position in each coordinate. The reason to divide zone into three parts is to prevent large state spaces which is related to too long learning and too much training data. Table 1 shows zone with each position and angle used in the system. The first zone there will be combination of rotation and translation with 90˚, the next is only rotation with 30º and 1º.

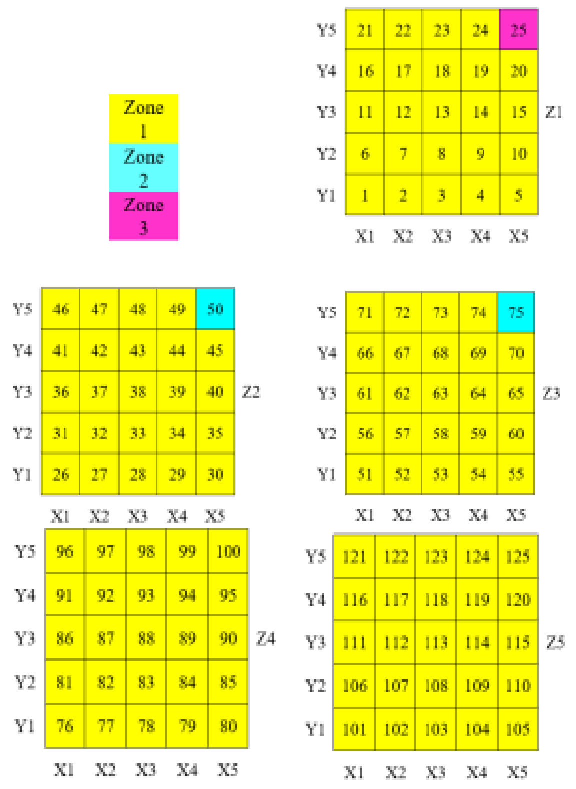


Figure 1. (a) Zone (b) environment 1 (c) environment 2 (d) environment 3 (e) environment 4 (f) environment 5

Table 1. Position and angle in every step

Zone	Total of		
	Position	Angle	State
1	122	4^3=64	122×64=7808
2.1	-	4^3=64	64
2.2	-	4^3=64	64
2.3	2	-	2
3	1	-	-

Location represents coordinate of XYZ axis.

Position is conversion from XYZ axis location that becomes a specific position number in working space; every step has the different position.

State Table a list of states which represents position with combination of angle or only combination angle or only rotation one for every zone

Q Table a list of Q value with relation between states and actions used to decide which action that will be taken for the state.

Max Training number is input to give random initial random position and angle value for training.

Max Episode number is input for total episode in one training.
Timestep are how many states for object in environment from initial position to reach the target in one episode.

3.2. Algorithm

The goal in this system is to be successfully doing assembly automatically. Automatically means that the object learns how to reach the target without human assistance by only given the input data and follows the rules to generate the path. In the system, there will be divided into two steps, first is matching and second is learning part with the first is using random value as initial position. Training and testing process will be applied for those steps to know the object is able to complete assembly task. The point cloud data (PCD) will be input as a whole data. The objects are specifically used for industrial assembly with some requirement parameter and each detail. Every object has to be matched with the couple part of another part of object which is called coupling matching.

By using the general view of the system to combine the 3D point cloud matching and machine learning part using reinforcement learning method, this becomes a continuity process starting from matching until successful assembled object. The training and testing part will be important for the system to know whether it adequately works well or not. At training part, the object will find and memorize the step by exploring the path. On the contrary, at testing part, the object will directly jump to the optimum path by experienced learning. So the learning way takes longer time than testing.

System Process:

1. Input 3 PCD (target, source, assembled object)
2. Do random initial position
3. Do RANSAC and ICP for registration.
4. Get Transformation Matrix (***T***)
5. Do Reinforcement Learning
6. Output

There will be loaded three input of PCD which are used for fixed target, moving object or called as source, and also target for coupling matching. The hyperparameters are used to input for learning system only in the training. These are related to learning rate, discount factor, and exploration value. These parameters are essential to make the system convergence and learn the rules that given efficiently. A fixed target will be placed in the position based on coordinate to let the source establish path way to find the target. This rule will give Q value updated to the table. Additionally, Q value will be generated and updated by learning system sequentially.

The system will iterate based on the maximum training value, episode, and step. These parameters also can be threshold for the training system. According to the parameter, the bigger value can affect to better result but cost time to be done. After all are completed, the result which is Q table should be tested to know the model is good and showing acceptable result.

Training Part, Q-learning algorithm is applied to train the object using the that the object learns how to reach the target by using path planning with some rules which use reward and penalty to update Q table value. Changing some parameters in the Q-learning system makes a good result in learning way of object. There will be ε, α, γ, maximum training, maximum episode, and also maximum step for each training as parameter of training or parameter tuning. ε, as epsilon greedy, is used to decide what is policy of the system that is chosen between exploration and exploitation. If the value is 0, it means that exploitation and 1 for exploration otherwise. α, as learning rate, is a probability interpretation of succeed in every step. A factor of 0 is for object learns nothing and vice versa. γ, as discount factor, is a short-sighted or long-term high reward. If 0, the object only sees current reward except strive for long-term reward. Maximum training, episode, and step number can be selected depend on the system. However, more number of thresholds of parameter will precisely obtain the result

The learning algorithm is authorized to interact with the environment. Particularly, at each time step, it can select an action to take based on the current state. Afterwards, based on the action that is selected and the current state, the environment experiences a new state, which is scrutinized by the learning algorithm at the subsequent time step. The sequence of experienced states and actions is known as a trajectory. The trajectories depend on the initial state and transition probability distributions and the way action is selected based on the current state called policy. The goal of the learning algorithm is to find a policy such that the expected cumulative cost of states over all time steps is minimized, where the expectancy is taken with respect to the assortment over trajectories

The goal of training is to have stabilized or converged result. The applied training is episodic way which is plotting episode returns during training. The training loop can be seen whether it seems adequate to satisfy the goal or no. There is also threshold to limit step, episode, and training loop and also a function of automatically stop. This process collects the training data throughout the training process and stores all our experiences and observed rewards. The training data are used for the Q-learning model. So, for every taken action, it will be stored the experience along with a terminated flag. Interaction with the system and can be fulfilled and stored in memory.

1. Load 3D point cloud file (.pcd). There are three files used in the system. First is target model, second is object 1 which is fixed, and the last one is object 2 as source which will be moved by translation and rotation to the fixed object. Target is an assembled point cloud, PCD 1 is a fixed target and PCD 2 is a moving object called source.
2. Q table is prepared filled by random value for the initialization for creating the table. After those available, the tables will be loaded into the system. If table is not available, it should be create an initialization table. This is the process to create and fill table with the combination of position and angle in the state table.
3. Maximum training number is input to the system with random initial position and angle. That random initial position is transformed into a matrix ***T_random*** for initial state of point cloud data. Object PCD 1 is fixed as a target and PCD 2 is the moving object which will be matched partially to the whole assembled model as target.
4. To know that state, registration methods are employed by using RANSAC as initial rough matching and ICP as fine registration.
5. RL will do the training based on policy which is related to giving reward and hyperparameter calculation. This can establish path and save as experience by always updating value in Q table in learning step that will be used for testing step. This is the

main part of learning process for the object to know the rules how to do path planning for reaching the target.

6. Updating state position and angle is related to current position, next position, and action. The reward is updated to Q table based on how good the translation and rotation achievement. Total reward will be computed from translation reward and rotation reward. The translation reward will get one point if it has reached designated target position and give zero if not reach target. Rotation reward is obtained from calculation of error between next angle and current angle using transformation matrix.

Testing Steps

1. Load PCD file.
2. Load Q table.
3. Random initial pose.
4. Do registration.
5. Get transformation.
6. Show the maximum step

Algorithms:

1. Reinforcement Learning

RL has parameters setting that are employed for training. They are α as learning rate, γ as discount factor, and ε as epsilon policy or exploration rate. Besides those, there are number of training, episode, and step that are input for maximum iteration. The parameter setting value for learning rate is between 0 and 1, which is for not learning anything and 1 for learning the policy or rules. Discount factor means probability to succeed or to survive in the environment. It is reflecting value of good start, valuing rewards that received higher than those received later. The value between 0 and 1, a closer value to 1 means that to prioritize rewards in the distant future. On the other hand 0 will consider rewards only in the immediate future. It will effect on how to use the rewards. A discount factor of 0 would mean that you only care about immediate rewards. The higher your discount factor, the farther your rewards will propagate through time. Exploration rate is a value which is used to make an exploration or exploitation system. It is quite a tricky value that should be assigned. The application is closer value to 1 means object will explore with random action and vice versa for closer to 0, it will go for choosing action based on highest Q value in between the options of actions. Maximum number of training, episode, and step are taken action as part of iteration to make the convergence. Maximum training, episode, and step is threshold parameter in training

In the RL algorithm which becomes essential part to establish the system because it will make whether the learning algorithm to be successful or no. This is a fundamental thing to make how the system works. This is a calculation for updating the Q value in Q table; it depends on the hyperparameter or tuning parameter that we use. The calculation involves Q value for the current state and next state. In this algorithm, there is also process to update the state based on position, action, target, and related transformation.

Choosing action algorithm is used also in the testing algorithm which is to let the object know how to choose action based on exploration rate that will go for exploration or exploitation in the system. The action needs Q table as input to know the action that should be taken. State will be used as input for the Q table that will give output as action. Q value is considered as action value in order to choose action in Q table. This algorithm uses Q value in the part of Q-learning on RL. The goal from this algorithm is to maximize Q value to get good action and result. The reward will be obtained from achieved specific task. So the strategy to plan a path is assigned to make a good result.

2. Update Status

Update state algorithm is part of reinforcement learning algorithm that plans a path from current position to the next position and also gives the reward. For giving the reward, it is based on translation and rotation reward. Reward comes as evaluation for updating the Q table; it returns from the action.

3. Continue Update State

Calculate the total reward is a section to combine reward of translation and rotation, so there will be positive and negative reward in total.

4. Choose Action

Exploration rate is specified as ε to randomly explore how to choose action. The process of generating a random number is to know it is bigger or lower from ε, then if the number is less than ε, so it will do exploration or another is exploitation. Exploration rate is equally important as choosing action in the step of training. Choosing an action given the current state is not as difficult as we think. The option is to act with random action with some small probability or the best action seen so far based on Q value from table or taking action predicted by the model. This way can be performed by trial and error to obtain different experiences in the system. The object will be succeed if it learns the interaction between states, actions and subsequent rewards which is called action policy and also determine which is the best action to choose given which is ε greedy policy.

5. Do Registration

This algorithm is a general of registration method to do coupling matching and obtain transformation between two point clouds. This is the step that is illustrating how to do initial matching and refine registration. From this step, we can get transformation matrix ***(T registration***).

4. ANALYSIS

4.1. Result

To make an assembled object, we need a couple part of object that have the same similarity to be matched as an assembly part together. The PCD files show which part of the object used to assembly. To make the system learns how to mimic human perception, we apply the coupling matching method for giving initialization for assembly which is used path planning as completed step by step.

The illustration of PCD file placement of object in the working environment is shown in the figure 2. The PCD file is used as a representation of real object. This is an instance of power electric socket object. A female part of socket becomes a fixed target that always be placed in the same position which is coordinate. As well a male object will be placed randomly in the working space to automatically assembly to the target based on the rules of learning system.

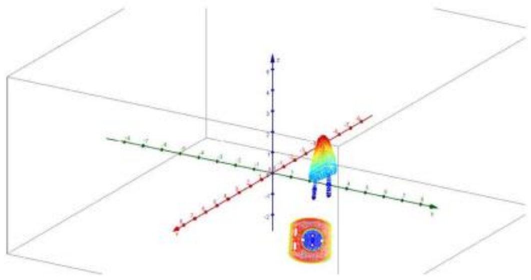


Figure 2. Object illustration in working space

Training Results are graphs and data which are references to undertake action for assembly planning. Accordingly to the goal of RL is to maximize the reward, the important thing is related to rule is exploration or exploitation of system. Exploration is finding more information about the environment. Exploitation is exploiting known information to maximize the reward. However, if it only focuses on reward, the object will never reach the big reward. Instead, it will only exploit the nearest source of rewards, even if this source is small (exploitation). But if the agent does a little bit of exploration, it can find the big reward. The parameters of Q-learning are α = 0.9, γ = 0.5, and ε = 0.2. First, we fill the table by random value. We would like to make the object learns a lot from experience. As a result, every parameter will take a particular role in turning out the results.

From the training result, we can analyze the system from its convergence shown in the figures. Figure 3 shows training result for every zone one by one. Zone 1 in figure depicts zone 1, zone 2 as zone 2.1, zone 3 as 2.2, zone 4, as zone 2.3. These are the example results of learning step from initial position to reach the target.

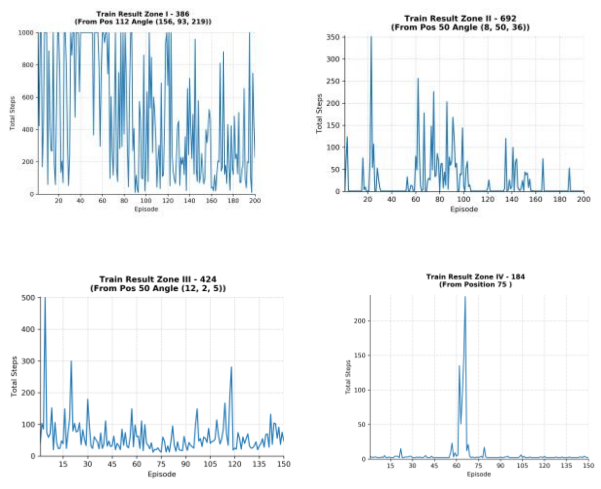
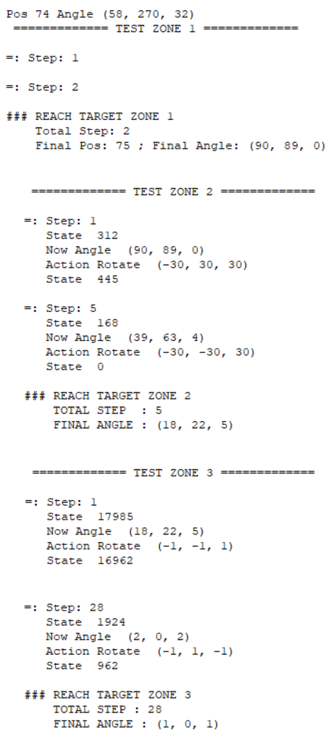


Figure 3. Training result of every zone

Testing Result in the path assembly algorithm, this step becomes essential part to evaluate the system to know whether the learning algorithm is successful or no. It can better specify the learning capability of algorithm. A sequence of specified actions to achieve the target will be explained below. Results of testing step are based on training step. From this step, we can perceive result from training step too. For that reason, the training step leads main part for the successful assembly goal. Training step is run for each zone with three different objects. All are placed in the same working environment. The path planning algorithms are tested to the objects with firstly do training. The testing part is completed by doing observation for how many paths from initial position to the target. Testing is sequentially evaluated zone by zone continuously as we can see from the Figure 4.



```
Pos 74 Angle (58, 270, 32)
 ============= TEST ZONE 1 =============

=: Step: 1

=: Step: 2

### REACH TARGET ZONE 1
    Total Step: 2
    Final Pos: 75 ; Final Angle: (90, 89, 0)


   ============= TEST ZONE 2 =============

  =: Step: 1
     State  312
     Now Angle  (90, 89, 0)
     Action Rotate  (-30, 30, 30)
     State  445

  =: Step: 5
     State  168
     Now Angle  (39, 63, 4)
     Action Rotate  (-30, -30, 30)
     State  0

  ### REACH TARGET ZONE 2
      TOTAL STEP  : 5
      FINAL ANGLE : (18, 22, 5)


   ============= TEST ZONE 3 =============

  =: Step: 1
     State  17985
     Now Angle  (18, 22, 5)
     Action Rotate  (-1, -1, 1)
     State  16962


  =: Step: 28
     State  1924
     Now Angle  (2, 0, 2)
     Action Rotate  (-1, 1, -1)
     State  962

  ### REACH TARGET ZONE 3
      TOTAL STEP : 28
      FINAL ANGLE : (1, 0, 1)
```

Figure 4 Testing result

The indicator of accomplished research algorithm that we can see in the path planning analysis, we offer table for initial position, total steps for achieving target, and achieved target in every zone. From the table and results, we can see the object successfully matched to the target with a few total of steps. From these results, we can analyze the initial position that represents coordinate and angle. Next is the target in every zone as in the first zone is the rough movement for translation and rotation. The next zone which is zone 2 has three parts of movements one by one that are two kinds of rotation and the last is translation to the target in the zone 3. The rotations are separated into 90^o, 30^o, 1^o. The reason is to make the object is easier to reach the target as step by step from rough movement into precise one. All the results of the table are obtained from testing result in which indicates the training result at the same time. The algorithm is examined by testing and computing the initial position to reach the target for assembly. The different angle and position are taken from registration method and the next is learning step to know it has reached the target by showing the different angle as 0^o or 1^o and position 25 in coordinate 5,5,1 (x, y, z) as target.

4.2. Conclusion

The paper sets out to find methods to coupling matching and path planning by using reinforcement learning to see analysis of system's convergence. The system runs based on saved experience in the training to know the path to the target by achieving rewards. The system is divided into three zones with every zone has each initial pose for testing. The working principle is when the object in firstly given random initial position, it should reach the target position or it can be called a stop position in the next zone. The commands are given about million times, and when it finally reaches target, step by step down and finally the reward will be treated. The learning type is trial and error.

The tested objects are three 3D point cloud data with the coupling matching part. The sockets which are female and male part are identified to match each couple part. Every object is firstly trained for every zone and next for testing. The working environment is a fixed with x, y, z coordinate which is converted to the position. The first object is trained and tested for all zones

The results depend on the working environment, turning parameter, and rules of algorithm. Training time and computer processor also affect to the result. For the parameters should be tried several combination matched values. The algorithms and tuning parameter settings will be different on every each particular problem. As a result, a parameter adjustment and experimentation take long processes which are often required trial and error for every application.

Recommendation of this work can be further improved as the methods used to choose action and larger working environment to easier reach the target. Another future progress is the implementation of test which has to be more focused on integration system with specific parameters that can influence performance and efficiency of the system.

REFERENCES

[1] D. Yoon, H. J. Kang, J. Choi, D. E. Kim, and X. Huang, "3D point cloud data basis shape management for assembly of modularized large and complicated marine structures," International Journal of Manufacturing Engineering, pp. 1–8, Feb. 2015.

[2] W.-C. Chang and V.-T. Pham, "An efficient neural network with performance-based switching of candidate optimizers for point cloud matching," in Proc. of 2018 the 6th International Conference on Control, Mechatronics and Automation, Tokyo, Japan, Oct. 2018.

[3] W. Yu, J. Tan, C. K. Liu, and G. Turk, "Preparing for the unknown: Learning a universal policy with online system identification," arXiv preprint arXiv:1702.02453, Feb. 2017.

[4] X. B. Peng, G. Berseth, and M. Van de Panne, “Terrain-adaptive locomotion skills using deep reinforcement learning,” ACM Transactions on Graphics (TOG), vol. 35, no. 4, p. 81, Jul. 2016.

[5] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. Van Den Driessche, T. Graepel, and D. Hassabis,“Mastering the game of go without human knowledge,” Nature, vol.550, no.7676, p.354,Oct. 2017.

[6] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, “Benchmarking deep reinforcement learning for continuous control,” in Proc. of the 33rd International Conference on Machine Learning-Volume 48, New York, USA, Jun. 2016, pp. 1329–1338.

[7] A. Aldoma, Z. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. B. Rusu, S. Gedikli, and M. Vincze, “Tutorial: Point cloud library: Threedimensional object recognition and 6 DOF pose estimation,” IEEE Robotics & Automation Magazine, vol. 19, no. 3, pp. 80–91, Sep. 2012.

[8] J. Yang, Z. Cao, and Q. Zhang, “A fast and robust local descriptor for 3d point cloud registration,” Information Sciences, vol. 346, pp. 163–179, Jun. 2016.

[9] J. Zhao, C. Li, L. Tian, and J. Zhu, “FPFH-based graph matching for 3D point cloud registration,” in Proc of the 2017 Tenth International Conference on S. Legg, and D. Hassabis, “Human-level control through deep reinforcement learning,” Nature, vol. 518, no. 7540, p. 529, Feb. 2015.

[10] P. J. Besl and N. D. McKay, “A method for registration of 3-D shapes,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, no. 2, pp. 239–256, Feb. 1992.

[11] Z. Liu, G. Liu, J. Li, and D. Ye, “Pose estimation of rigid object in point cloud,” in Proc. of International Congress on Image and Signal Processing,
BioMedical Engineering and Informatics (CISP-BMEI). Shanghai, China: IEEE, Oct. 2016, pp. 708–713.

[12] Y. He, B. Liang, J. Yang, S. Li, and J. He, “An iterative closest points algorithm for registration of 3D laser scanner point clouds with geometric features,” Sensors, vol. 17, no. 8, p. 1862, Aug. 2017

[13] R. Sutton, A. Barto, and F. Bach, Reinforcement Learning: An Introduction. MIT Press, Mar. 1998.

[14] A. L. Strehl, L. Li, and M. L. Littman, “Reinforcement learning in finite mdps: Pac analysis,” Journal of Machine Learning Research, vol. 10, pp. 2413–2444, Nov. 2009.

[15] L. Baird, “Residual algorithms: Reinforcement learning with function approximation,” in Proceedings of the 12th International Conference on Machine Learning 1995. California, US: Elsevier, July 1995, pp. 30–37.

[16] X. B. Peng, G. Berseth, and M. Van de Panne, “Dynamic terrain traversal skills using reinforcement learning,” ACM Transactions on Graphics (TOG), vol. 34, no. 4, p. 80, Jul. 2015.