Synchronous Dual-Arm Manipulation by Adult-Sized Humanoid Robot

Hanjaya Mandala, Saeed Saeedvand, and Jacky Baltes

***Abstract*—This paper introduces a synchronous dual-arm manipulation with obstacle avoidance trajectory planning by an adult-size humanoid robot. In this regard, we propose a high precision 3D object coordinate tracking using LiDAR point cloud data and adopting Gaussian distribution into robot manipulation trajectory planning. We derived our 3D object detection into three methods included auto K-means clustering, deep learning object classification, and convex hull localization. Therefore, a lightweight 3D object classification based on a convolutional neural network (CNN) has been proposed that reached 91% accuracy with 0.34ms inference time on CPU. In empirical experiments, the Gaussian manipulation trajectory planning is applied adult-sized dual-arm robot, which shows efficient object placement with obstacle avoidance.**

***Index Terms*—3D perception, deep learning, manipulation, humanoid robot**

I. Introduction

Object placement problem is the most interesting manipulation problem in robots with arm and it has been utilized to tackle a variety of problems. Therefore, the optimal trajectory planning for pick and placement is a critical issue to be investigated by single-arm robots. There are numerous industrial oriented applications and several famous manipulation competitions as the benchmark for single-arm robot [1]. Those applications represent object manipulation importance, which in the last decade many solutions have been proposed [2].

Unlike the single-arm manipulation robot, a dual-arm manipulation robot is more complex. This complexity is because of the required synchronization procedure in object grasping and placement with dual-arm manipulators. In this regard, different items should be considered such as two arms collision and amount of torque applied on the dual-arm object grasping. Therefore, the dual-arm configuration is needed for complex operations and can be useful for the load sharing of a big or heavy object. Recently, there have been many research studies on dual-arm manipulations.

In [3], authors introduced a manipulation planning of dual-arm industrial manipulators (HiroNX) to integrate grasp and object placement planner. Manipulation planning in this work presented transferring a simple solid object from one to the other hand as a sequential task by using a transit-transfer path. In [4], the authors used a dual-arm robot for specific applications in harvesting tomato using binocular vision and hand-eye coordination system. The dual-arm robot system implemented in this work is a cooperative work, where one arm grasps the tomatoes, and the other arm cut the tomatoes. In [5], the authors designed an anthropomorphic, compliant, lightweight dual-arm for aerial manipulation. They demonstrated a high accuracy visual servoing of the combination dual-arm with a hexarotor drone platform that capable to grasp a lightweight object with the provided object grasping point. In [6], the authors presented dual-arm compliance control coordination to solve the undetermined force balance for grasping objects. They used a master-slave force control strategy, where the trajectory and operational force of the slave arm can be calculated through a dynamic force balance equation from master arm forces input.

Most previous literature studies addressed the dual-arm manipulation problem as a sequential task or they just considered object grasping and object placement as a separate study. In [2], the authors presented an extensive survey for dual-arm manipulation. They concluded that dual-arm manipulations can be divided into the un-coordinated (asynchronous) task and coordinated (synchronous) task. Accordingly, in existing methods, most of the study in literature assumed that the grasping objects tasks are required using a special robot hand or gripper.

Motivated by shortcomings in stated studies, in this paper, we introduce a heuristic based approach using 3D point clouds perception to deal with synchronous dual-arm object placement that considers obstacle avoidance in adult-sized humanoid robot arms. Moreover, we use a human-like robot hand to generalize for many different types of object shape. Therefore, the grasping object and trajectory planning task is performed in high-dimensional (3D) coordinate system.

In general, the main contributions of the paper are as follows. (i) Proposing a light-weight and real-time two-stage 3D object detection based on point clouds data. (ii) Proposing a Gaussian trajectory for motion planning. (iii) Implementing the proposed algorithm on the real-robot with dual-arm synchronize manipulation including obstacle avoidance.

The rest of this paper is organized as follows. Section II presents the system architecture of the THORMANG-Bear adult-sized humanoid robot platform. Section III describes the proposed methods with two algorithms in detail. Section IV provides the experimental result of the proposed method on the dual-arms humanoid robot. Finally, Section V concludes and discusses the future research of the paper.

II. System Architecture

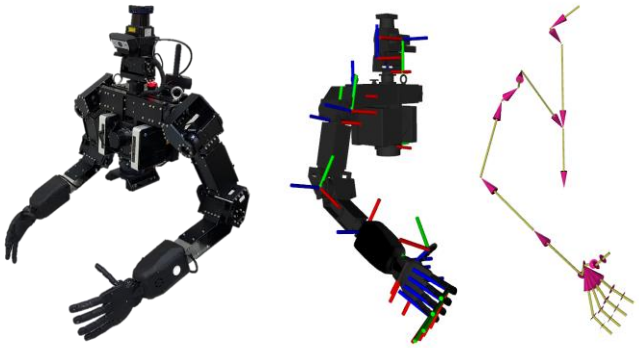
In this section, the proposed system architecture details for an adult-sized humanoid robot to accomplish the manipulation task is derived.

Hanjaya Mandala and Jacky Baltes are with the with the Department Electrical Engineering, National Taiwan Normal University, Taipei 10610, Taiwan (e-mail: ahanjaya@gmail.com and jacky.baltes@ntnu.edu.tw).

Saeed Saeedvand is with the Faculty of Electrical Engineering and Computer Engineering, University of Tabriz, Tabriz, 5166616471, Iran (e-mail: saeedvandr@tabrizu.ac.ir).

A. THORMANG-Bear Robot

We used a modified upper body of the THORMANG humanoid robot platform named THORMANG-Bear. The main modification was applied on the robot's hands, as the original robot used gripper, we replaced it with Seed Robotics dexterous robot hand. Also, we replaced its camera with a wide-ranging field of view web camera Logitech C930E.



(a) (b) (c)

Fig. 1. The modified upper-body of THORMANG3 platform (a) THORMANG-Bear robot (b) Right arm URDF and (c) Kinematic chains.

The THORMANG-Bear robot has 27 Degree of Freedom (DOF) in total: 2 DOF on the head, 1 DOF on the torso, and 12 DOF on each arm. It equipped with two minicomputers for motion and perception controls, one mono-vision camera, and one LiDAR scanner.

B. Forward and Inverse Kinematics

Kinematics plays an important role in humanoid robot development for different applications to derive a mechanism of motion [7], especially for manipulation tasks. We used the Unified Robot Description Format (URDF) for representing kinematics and dynamic model of the robot. URDF in many respects is similar to the Denavit-Hartenberg convention but with significant additional enhancement 0(b). Therefore, formulating the humanoid robot kinematic can be divided into forwarding kinematics and inverse kinematics.

Forward Kinematics (FK) refers to calculate the position and orientation of the end-effector given by a set of joint configurations. Solving FK problems is more straightforward and there is no complexity to derive the equations. Conversely, Inverse Kinematics (IK) calculates a set of joint variables to generate the end effector at the desired position. Solving IK solutions is more complex due to the position and orientation of a link and the joint angles humanoid robot are represented by nonlinear equations.

The most typical way to compute the IK solution from a joint link constrained is based on the numerical approach. Therefore, one of the famous open-source IK solvers uses Jacobian Pseudo Inverse (numerical method) to derive the relationship between the position and rotation of a link and joint angles into linear equations as Orocos Kinematic Dynamic Library (KDL) [8]. This approach could give IK and FK solutions based on the kinematic chain rule (URDF). 0(c) illustrates the kinematic chains of the right arm THORMANG-Bear robot as an example.

III. PROPOSED METHOD

In this section, the design of the proposed algorithm to solve synchronous dual-arm manipulation are deliberate into (i) 3D object detection and (ii) trajectory planning. Therefore, in 0 we illustrated the block diagram of the proposed method. Each step of the shown block diagram in 0, will be described individually in the rest of this section.

A. 3D Object Detection

Point clouds data (PCD) refers to a set of points in three-dimensional geometric coordinates that can represent objects or space. These points represent the X, Y, and Z in cartesian coordinates. In this regard, the robot pelvis link is used as an origin coordinates system for the processing reference. Therefore, we acquired PCD by using the LiDAR scanner from the robot. As a result, those PCD were utilized as the main robot vision for 3D object detection in this paper.

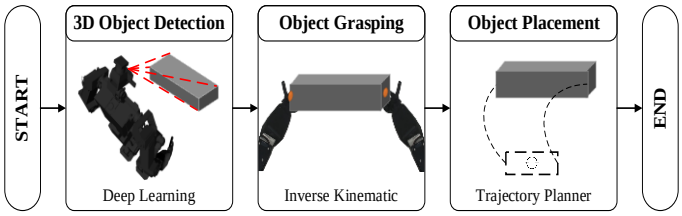


Fig. 2. The block diagram of the proposed algorithm dual-arm synchronous manipulation.

Recently, the most famous algorithm of the Convolutional Neural Network (CNN) has successfully addressed the 3D object detection problem in a comprehensive survey provided by [9]. However, in those implementations, employing the CNN algorithm in 3D data is still facing some drawbacks such as the requirement of a high computing platform or graphic processing unit (GPU) to achieve real-time performance. Therefore, in this paper, we propose a lightweight 3D object detection to tackle this problem that can run on the central processing unit (CPU). In 0, we illustrated the general flowchart of our proposed object detection algorithm.

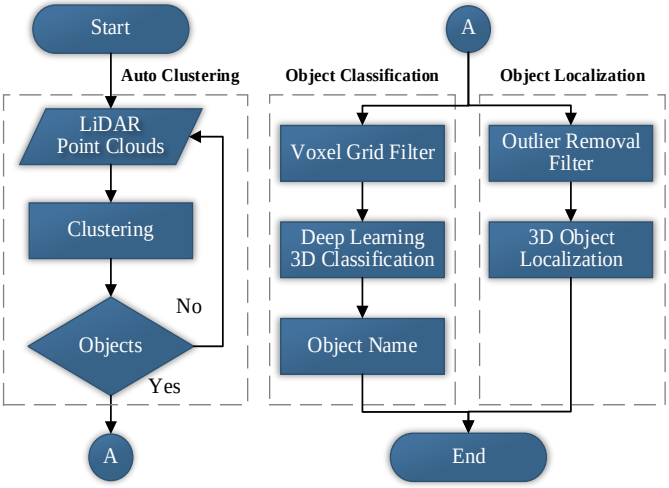


Fig. 3. Flowchart of the proposed two-stage 3D object detection.

Our proposed 3D object detection algorithm is consisting of a two-stage process. In the first stage, the LiDAR PCD is obtained from the robot head scanning process where it depicted in 0(a). Then, we apply a heuristic algorithm based on Euclidean distance to select point clouds located on robot workspace only as shown in 0(b). Next, those data are processed into the K-means clustering algorithm [10] to segment objects with the table. In the second stage, as there is no fixed amount number of point clouds in different scans form LiDAR we utilized a Voxel Grid (VG) filter to down sample the point cloud data into fixed shape occupancy grid [11]. As a result, a fixed shape binary voxel grid shape is feed into our proposed deep learning algorithm for the proposed 3D classifications algorithm to predict the objects' class.

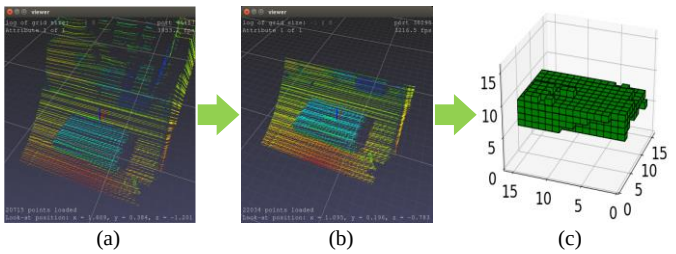
In the second stage, the segmented objects from the K-means clustering result are also fed into the object localization process. In this step, the set of point clouds is applied to a statistical outlier removal filter to remove noise points from the cluttered environment. Then, our proposed heuristic 3D object localization algorithm takes the filtered data to calculate the object position. Finally, the output from the 3D object classification with 3D object localization is merged to produce a complete 3D object detection procedure. So, to describe our proposed two-stage 3D object detection in detail, we explain it in three sub-section in the following.

1) Auto Clustering

Clustering points into sensible grouping are the key points of the pre-processing to achieve 3D point cloud object detection. Therefore, we use K-means algorithm as feature extraction to clustering the PCD located in the robot workspace. The elbow method is one of the heuristic methods by iterating all possible k candidates into the auto clustering problem [12]. We use the inertia attribute to identify the sum of squared error (SSE) distances of k samples to the nearest cluster center. As k increases, the Sum Squared Error (SSE) distance tends to decrease towards zero. Then, by visualizing the SSE value with a line chart for each k candidate. When the chart looks like an arm, the "elbow" on the arm is the k parameter do define the total number of clusters in the given point cloud data.

2) Object Classification

As mentioned earlier, deep learning (DL) models have required a fixed shape of input data. In 0, it shows the voxel grid filter process to down sample the result from K-means clustered objects. The voxel data represent the fixed size of the discretized binary voxel occupancy grid at 16×16×16, where each voxel has a binary state of occupied or unoccupied.



(a) (b) (c)

Fig. 4. The example of down-sample point cloud data using the voxel grid method from (a) raw data goes to (b) filtered workspace (c) and voxelized.

In this paper, a CNN classifier is proposed to predict the class of the down-sample voxel grid point clouds. In general, a CNN classifier consists of a convolution layer, a pooling layer, and a fully connected (FC) layer. Our CNN architecture is inspired by the famous 3D shape recognition VoxNet [13], that integrating a volumetric occupancy grid representation using 3D CNN. The detail of the proposed CNN network architecture is shown in 0. Our proposed CNN-classifier model takes the input of the volumetric occupancy grid 16 × 16 × 16. This model uses 3D convolutional layers with a total of 3 layers. These layers use 16 filters of size 3, with stride 1 and Rectified Linear Unit (ReLU) activation functions followed by Batch Normalization (BN) along with two max-polling layers. The ReLU layer works as a sub-linear function that will output the input directly if it is positive, otherwise, it will be resulting zero. Then, the BN normalizes the output of a previous activation layer by subtracting the batch mean and dividing it with the batch standard deviation. Whereas, the pooling layer works as a subsampling layer to selects a maximum value from the previous layer with specific window size and stride. Finally, there are two fully connected layers at the end layers of the model, where the final layers use the SoftMax function to normalize the classifier output prediction. During the training, we utilized Adam optimizer with a learning rate of 0.0005 for updating the model weights.

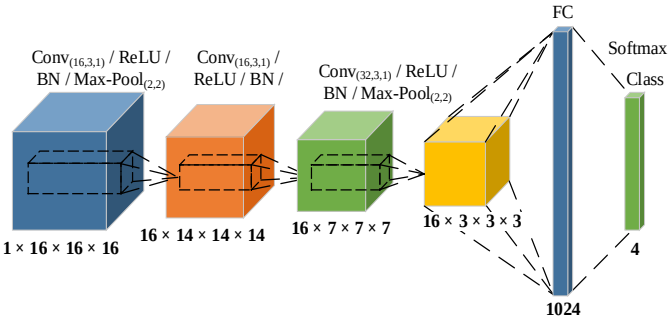


Fig. 5. The proposed 3D classification of deep learning models.

3) Object Localization

In general, LiDAR scanners generate PCD based on the range of point densities. During the scanning process, there is a possibility of measurement error or occlusion that leads to sparse outlier data. Those outlier data are critical since they can clutter object detection results (false or corrupted result). Therefore, we used a statistical outlier removal (SOR) filter to tackle those problems [11]. SOR filter uses point neighborhood statistics to filter outlier data. This filter calculates the distribution of neighbor's point distances in the given PCD. For each point, we compute the mean distance from it to all its neighbors. By assuming that the resulted distribution is Gaussian with a mean and a standard deviation, all points whose mean distances are outside an interval defined by the global distances mean and standard deviation can be considered as outliers and trimmed from the dataset.

0 shows our SOR filter implementation in PCD data. To simplify the filter process, we implemented a lightweight SOR based on a two-dimensional filter. By looking from the top view of the clustered 3D PCD drawn in 0(a), we can aim 3D

PCD that laying in the single-axis plane shown in 0(b), which is represented as 2D PCD. This method is efficient, as we can filter the noise point cloud located out of the object without losing important information of the object.

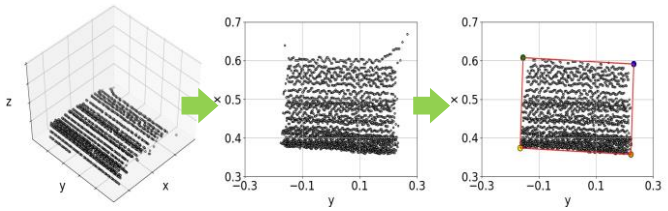


Fig. 6. Statistical outlier removal filter of point clouds.

For object localization, we propose a heuristic 3D point cloud localization based on the 2D convex hull algorithm. The convex hull refers to a method to find the farthest outlier of the given set of points. There are many techniques to calculate convex hulls. The Quick Hull (QHull) algorithm is one of the most efficient methods because of its simplicity and speed [14]. It uses a divide and conquer approach similar to the quicksort algorithm. Therefore, we utilized convex hull results to define the position of extreme points and divided the exterior points into four groups bounded by rectangles illustrated in 0.

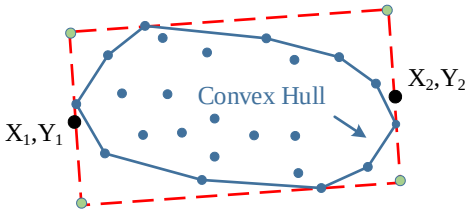


Fig. 7. Convex hull on 2D point clouds.

As shown in 0. the green dots represent the 4 corner points of the rectangle bounding box. With those points, we also can aim the rotation of the objects from the slope of rectangle midpoints given by (1).

$$\theta = \tan{-1}\left(\frac{y_2 - y_1}{x_2 - x_1}\right) \tag{1}$$

We can obtain the 2D object position and rotation using the above-mentioned method. Moreover, we assumed that our target object has a solid shape. Therefore, by calculating the minimum and maximum height of the object point cloud we can match the bounding rectangle in the 3D perspective.

B. *Object Grasping*

Dual-arm synchronization to grasp an object is a significant problem to tackle for a humanoid robot. The challenge is comprised of identifying the grasping points and synchronous movement for both arms. We introduce an efficient algorithm to select a grasping point of the object based on 3D PCD. The result of object detection is utilized to calculate the grasping point and illustrated in 0.

To calculate the grasping point, we used a geometric equation to calculate the middle point from given two points of each axis in a three-dimensional plane.

$$M = \left(\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2}, \frac{z_1 + z_2}{2}\right) \tag{2}$$

Where M , is the middle point for one arm grasping point. Therefore, for calculating the dual-arm grasping point, the same equation is used to calculate for the other arm. However, to allow synchronous grasping, we use an affine transformation matrix to offset the reference grasping point for each arm. As a result, the robot will synchronize grasp the object with additional offset, also concerning the rotation of the object. The detailed affine transformation is given by the equation.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{3}$$

Where x', y' is the new grasping point, θ is the object rotation, x, y is the offset distance in the 2D axis. Note that, we are not transforming the z axis, we assume the grasping height stayed in the middle of the object.

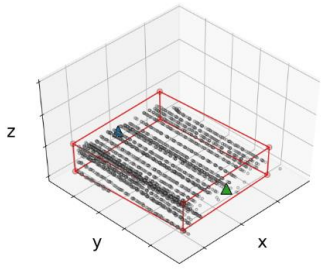


Fig. 8. The coordinates grasping point is represented by a triangular symbol on each side (blue triangle for the left arm and green triangle for right arm).

C. *Object Placement*

We maintained synchronous manipulation by referring to object width using the Euclidian distance of the left arm and right arm grasping points. So, after the robot grasps the object, it can move the object to a predefined target location. However, using a direct trajectory planner faced an issue when there is an obstacle in the trajectory path. Therefore, we propose a manipulation trajectory planner adopted from a normal distribution graph for obstacle avoidance as shown in the following equation.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{4}$$

Where μ is the mean of distribution or obstacle position, σ is the standard deviation or obstacle width, and x is the Euclidean distance of the trajectory path. Based on the focus of this paper, the path avoidance of Gaussian distribution is applied only on the z-axis and illustrated in 0.

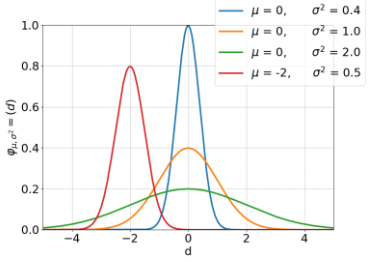


Fig. 9. The sample of Gaussian distribution plots.

IV. Experiment Results

To evaluate the performance of the proposed algorithm, first, we describe the experimental setup. Then, we conducted and evaluated the proposed methods into 3D object detection and synchronous dual-arm manipulation trajectory.

A. Experimental Setup

All experimental processes are tested using the real THORMANG-Bear robot (see Section **錯誤! 找不到參照來源。**). In the robot, we used the Robot Operating System (ROS) as the main software architecture. As a result, an additional operating computer (OPC) is required to integrate with two existing computers inside the robot. OPC acts as the central processing to execute the overall proposed algorithm in this paper with Intel i7-8750H CPU @ 2.20GHz.

Our experiments are conducted on the typical box objects that are available in daily life with four different shapes and illustrated in 0. Moreover, our entire code was implemented using ROS with the Python programming language. In this regard, the standard Python deep learning library (Keras), point cloud library (PCL), and mathematics library (SciPy) were used to implement the proposed algorithm.

Fig. 10. Experimental objects.

B. 3D Object Detection Result

In this subsection, we divided our two-stage 3D object detection into three-part. First, we showed our auto k-means clustering. As we mentioned earlier, we use the elbow method to choose k the parameter and drawn the graph result in 0. Moreover, we assumed the given point datasets are in two-dimensional shown in 0. Therefore, the 2D points are obtained from the top perspective of the 3D data. This method can reduce computer power, where the K-means algorithm runs on lower-dimensional data.

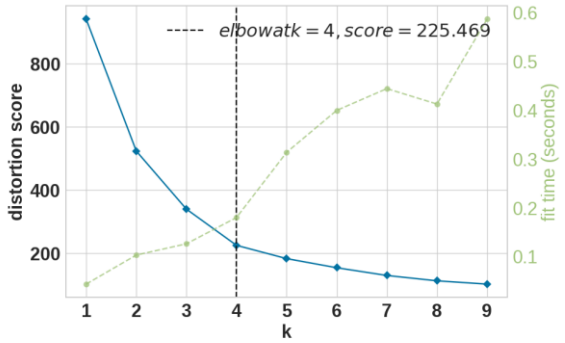


Fig. 11. Auto clustering using the elbow method.

Auto K-means run the clustering on the elbow method for a range of k values from 1 to 9 and computes an average score for each cluster. Therefore, the distortion score is computed by the sum of square error distances from each PCD to its assigned center. This approach is shown in 0 in which a blue line resembles an arm and the "elbow" (the point of inflection on the curve) is indicating the underlying model fits the best point. So, the "elbow" will be annotated with a dashed vertical line.

The objective of the elbow algorithm is to find the best k parameter to cluster point clouds based on multi-object detection. We showed the result of multi-object clustering based on k value selected by the elbow algorithm in 0. In total, we use 4 experimental objects shown in 0, where each object is segmented by four different colors.

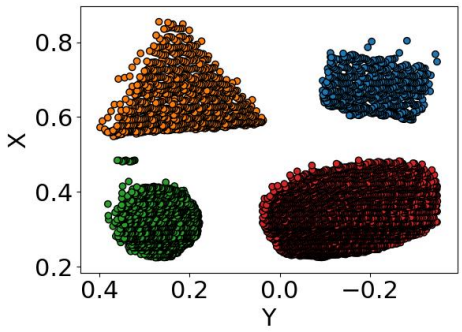
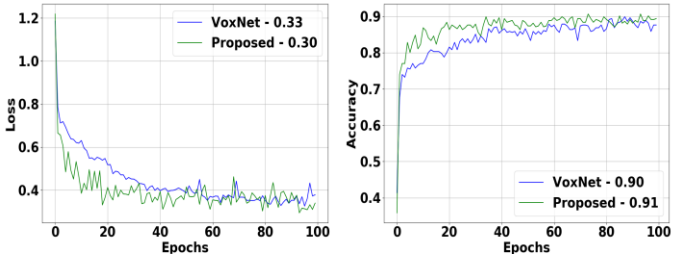


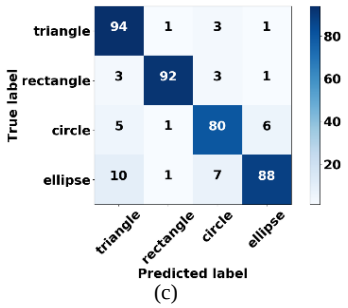
Fig. 12. Clustered point clouds result from the auto k-means algorithm.

Second, we collected a 1200 PCD dataset based on the experimental object show in 0. The key performance to evaluate a CNN-classifier model is by looking at loss and accuracy value. Therefore, trends of loss value should go down to interpret how badly the model in predicting each training iteration. In contrast, the accuracy value shows how accurate was the model's prediction on each epoch. A higher accuracy value is the essence of a classifier model. Moreover, to clarify our CNN-classifier model performance, we demonstrated a confusion matrix to evaluate the model on the validation dataset. So, in 0, we illustrated the result of our classification model in terms of loss, accuracy, and confusion matrix respectively.

0 shows the result of our proposed model is comparable with the VoxNet model [13]. As the original VoxNet model use volumetric occupancy grid with size 32×32×32, we changed into size of 16×16×16. This adjustment is to make an equal comparison with our proposed model by using the same size input data. Therefore, while the result of those two models is equivalent to 90% accuracy and 0.3 in loss value. However, our proposed model is superior in terms of the model's weight size, as it only has 0.5 MB of the weight parameter size. This means the model only has a 38981 trainable parameter which is lower ~40% from the adopted VoxNet model. As a result, it achieves an inference time of 0.34ms, running on the CPU of our experimental computer.



(a) (b)



(c)

Fig. 13. Multi-object classification model's performance based on (a) loss, (b) accuracy, and (c) confusion matrix.

In total, 396 validation dataset or (33%) of the collected dataset is used to evaluate the confusion matrix. Therefore, as shown in flowchart 0, for real-time usage of the clustering, results from the K-means algorithm are feed sequentially to our 3D classification model to detect the object name.

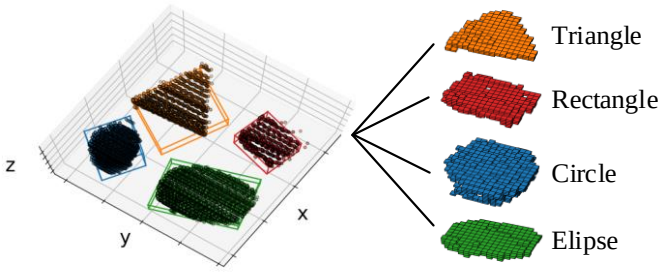


Fig. 14. 3D multi-object detection result.

Third, while the object classification process is running, the result from clustering segmentation which is also provided to the 3D object localization process parallel. Therefore, by calculating the convex of the point clouds, we can draw the object localization by the 3D bounding rectangle. Additionally, we utilized the result of the bounding box by considering the object rotation to determine the object grasping points. Wherefore, the width of the object is used for the reference parameter in the dual-arm synchronization algorithm.

Overall, those three stages represent our two-stage 3D object detection heuristic algorithm in using point clouds data that is based on the K-means clustering, deep learning 3D classification, and convex hull localization in 0.

C. Synchronous Manipulation Trajectory

In this section, we discuss the result of the experiment in the real environment that utilizing a real THORMANG-Bear robot. First, we set the object placement target location, then using the result of 3D object detection for defining the start point, and finally using the obstacle width and location for the μ and σ parameter in the given (4) for the synchronous manipulation trajectory. The procedure of the proposed trajectory planning is shown in 0.

0 shows our trajectory planning from the starting point in $x = 0.1$, $z = 0.62$, $ly = 0.08$ and $ry = -0.08$. Then, we set target location by changing only in x axes into 0.5 m. The green cylindrical shape in the middle represents an obstacle to avoid during object placement trajectory planning, at location $x = 0.3$ with 0.05 m width. Therefore, by taking those defined parameters, the gaussian trajectory planning result is the carriage in z axes with obstacle height offset of 0.03 m. So, the highest peak is located at 0.72 m.

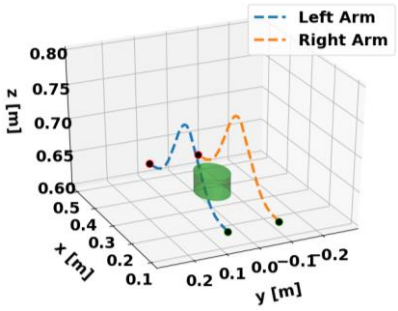
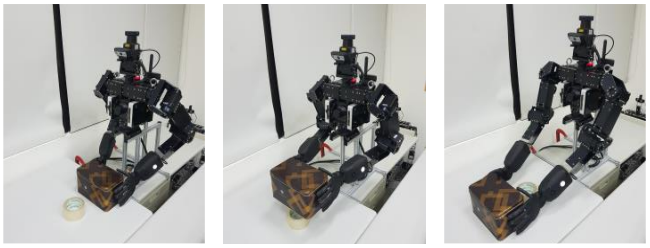


Fig. 15. 3D-plot of the gaussian trajectory planning for obstacle avoidance.

We illustrated the trajectory planning of the Gaussian trajectory with obstacle avoidance in 0. In this regard, first, the robot grasps the detected object with both arms, then with using gaussian trajectory planning in 0, the synchronous dual-arm manipulation moves the object to the predefined target location with obstacle avoidance. A brief video of experiments is available at shorturl.at/frFJV.



(a) (b) (c)

Fig. 16. Real-view robot grasping (a), avoiding obstacle (b), and object placement.

V. CONCLUSIONS

In this paper, we presented synchronous dual-arm manipulation methods in which we used a modified upper body of THORMANG3 robot to place an object with obstacle avoidance. The proposed method consists of two algorithms. First, on the light-weight two stages 3D object detection includes point clouds auto clustering, deep-learning object classification, and convex hull localization is presented. Second, a heuristic synchronous dual-arm manipulation based on the Gaussian distribution to calculate the trajectory for obstacle avoidance is proposed. In the experiments results, the 3D object classification algorithm was able to detect four different shape objects that can run on CPU reached 91% accuracy, with 0.34ms inference time. Moreover, the manipulation trajectory adopted from Gaussian distribution avoids obstacles in the single-axis successfully. The presented method can easily extend to multi-dimensional. In the future work multi objects obstacle avoidance can be taken into consideration.

ACKNOWLEDGMENT

This work was financially supported by the 'Chinese Language and Technology Center' of National Taiwan Normal University (NTNU) from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan, and Ministry of Science and Technology, Taiwan, under Grant Nos. MOST 108-2634-F-003-002, MOST 108-2634-F-003-003, and MOST 108-2634-F-003-004 (administered through Pervasive Artificial Intelligence Research (PAIR) Labs) as well as MOST 107-2811-E-003-503. We are grateful to the National Center for High-performance Computing for computer time and facilities to conduct.

REFERENCES

[1] J. Baltes, Y. Sun, and H. Moon, "2017 Competitions: Magical, Manipulating, Mercurial Robots [Competitions]," *IEEE Robotics & Automation Magazine*, vol. 25, no. 2, pp. 8-15, 2018, doi: 10.1109/MRA.2018.2822045.

[2] C. Smith *et al.*, "Dual arm manipulation—A survey," *Robotics and Autonomous systems*, vol. 60, no. 10, pp. 1340-1353, 2012.

[3] K. Harada, T. Tsuji, and J.-P. Laumond, "A manipulation motion planner for dual-arm industrial manipulators," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China, 2014: IEEE, pp. 928-934, doi: 10.1109/ICRA.2014.6906965.

[4] X. Ling, Y. Zhao, L. Gong, C. Liu, and T. Wang, "Dual-arm cooperation and implementing for robotic harvesting tomato using binocular vision," *Robotics and Autonomous Systems*, vol. 114, pp. 134-143, 2019.

[5] A. Suarez, P. R. Soria, G. Heredia, B. C. Arrue, and A. Ollero, "Anthropomorphic, compliant and lightweight dual arm system for aerial manipulation," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 24-28 Sept. 2017 2017, pp. 992-997, doi: 10.1109/IROS.2017.8202266.

[6] L. Yan, Z. Mu, W. Xu, and B. Yang, "Coordinated compliance control of dual-arm robot for payload manipulation: Master-slave and shared force control," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 9-14 Oct. 2016 2016, pp. 2697-2702, doi: 10.1109/IROS.2016.7759419.

[7] S. Saeedvand, M. Jafari, H. S. Aghdasi, and J. Baltes, "A comprehensive survey on humanoid robot development," *The Knowledge Engineering Review*, vol. 34, p. e20, 2019, Art no. e20, doi: 10.1017/S0269888919000158.

[8] P. Beeson and B. Ames, "TRAC-IK: An open-source library for improved solving of generic inverse kinematics," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, 3-5 Nov. 2015 2015, pp. 928-935, doi: 10.1109/HUMANOIDS.2015.7363472.

[9] A. Ioannidou, E. Chatzilari, S. Nikolopoulos, and I. Kompatsiaris, "Deep Learning Advances in Computer Vision with 3D Data: A Survey," *ACM Comput. Surv.*, vol. 50, no. 2, p. Article 20, 2017, doi: 10.1145/3042064.

[10] J. Yadav and M. Sharma, "A Review of K-mean Algorithm," *International journal of engineering trends and technology*, vol. 4, no. 7, pp. 2972-2976, 2013.

[11] X.-F. Han, J. S. Jin, M.-J. Wang, W. Jiang, L. Gao, and L. Xiao, "A review of algorithms for filtering the 3D point cloud," *Signal Processing: Image Communication*, vol. 57, pp. 103-112, 2017.

[12] D. J. Ketchen and C. L. Shook, "The application of cluster analysis in strategic management research: an analysis and critique," *Strategic management journal*, vol. 17, no. 6, pp. 441-458, 1996.

[13] D. Maturana and S. Scherer, "VoxNet: A 3D Convolutional Neural Network for real-time object recognition," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 28 Sept.-2 Oct. 2015 2015, pp. 922-928, doi: 10.1109/IROS.2015.7353481.

[14] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Trans. Math. Softw.*, vol. 22, no. 4, pp. 469–483, 1996, doi: 10.1145/235815.235821.

Hanjaya Mandala received his B. App. Sc degree in Mechatronic Engineering Study Program of Electrical Engineering from Polytechnic State Batam in 2017. He received his M.Sc degree in the Electrical Engineering Department from the National Taiwan Normal University in 2020. He has been worked on the humanoid kid-size robots since 2014, and he obtained 1st place at IEEE IROS 2019 Humanoid Application Challenge of robot magic and 3rd places at RoboCup 2018 kid-size humanoid robot competitions. Also, three-rows of 1st place at Indonesia National Humanoid Robot Competitions from 2016-2018. His research interest includes a humanoid robot, computer vision, artificial intelligence, and machine learning.

Saeed Saeedvand received his B.S. degree in Computer Engineering in 2012. He received his M.S. degree in Computer Engineering from the University of Tabriz in 2014. Currently, he is a Ph.D. candidate at the University of Tabriz. He has been a lecturer at the University of Tabriz since 2014. He has been worked on the humanoid adult-size, and kid-size robots since 2009, and he obtained many 1sts to 3rd places at the well-known humanoid robot competitions. He is a member of the technical committee of RoboCup-IranOpen robotic competitions since 2016. His research interest includes artificial intelligence, robotics, machine learning, and computer vision.

Jacky Baltes received his Ph.D. degree in 1996 from the University of Calgary in Artificial Intelligence. From 1996 to 2002, he worked as a senior lecturer at the University of Auckland in Auckland, New Zealand. Since 2002, he has been a professor in the Department of Computer Science at the University of Manitoba in Winnipeg, Manitoba. He has worked as an Outstanding Professor and head of Educational Robotics Center at the National Taiwan Normal University, Taiwan, since 2016. He has also been a vice president of the FIRA robotic soccer association, chair of the HuroCup competition, and a member of the RoboCup executive committee. His research interests are intelligent robotics, artificial intelligence, machine learning, and computer vision.