LEARNING-BASED POSE ESTIMATION IN AUTONOMOUS ELECTRIC VEHICLE CHARGING SYSTEM USING ROBOT ARM MANIPULATION

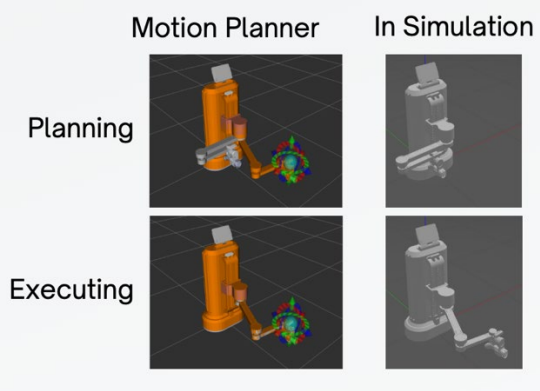
Amir Aqeel Imran Amir Mokhtar[1], Zainab Binti Asus[1], Nur Safwati Mohd Nor[1]*, Hendri Maja Saputra[2]

[1]Faculty of Mechanical Engineering, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia
[2]Research Center for Smart Mechatronics, National Research and Innovation Agency-BRIN, Jl. Sangkuriang, No. 21/154D, Bandung 40135, Indonesia

Article history
Received
11th May 2026
Received in revised form
16th June 2026
Accepted
16th June 2026
Published
20th June 2026

*Corresponding author
nursafwati@utm.my

GRAPHICAL ABSTRACT



ABSTRACT

Autonomous Electric Vehicles (AEVs) require fully automated charging systems to achieve complete autonomy. However, existing charging methods still depend on manual plug insertion by users. This study presents a learning-based control algorithm for autonomous robotic charging using a four-degrees-of-freedom robot arm. The robotic arm was designed in SOLIDWORKS, simulated in Gazebo, and controlled through the Robot Operating System (ROS). An object detection model was trained using a dataset of Type 2 EV charging sockets to identify the charging port, while a depth camera estimated its three-dimensional coordinates. The detected position was then processed by a motion planner to generate a collision-free trajectory for accurate plug alignment. Simulation results demonstrated reliable charging socket detection and robotic manipulation under various operating conditions. The proposed approach achieved a precision of 0.99 and a recall of 0.98, indicating robust detection performance. The highest detection confidence score of 0.87 was obtained at a charging socket distance of 1.1 m, while the lowest score of 0.73 was recorded at 1.0 m. These findings demonstrate the potential of the proposed framework for enabling reliable and autonomous EV charging operations.

KEYWORDS

Autonomous Electric Vehicle; Machine Learning-Based Detection; Motion Planning; Peg-in-hole

INTRODUCTION

The rapid growth of the Autonomous Electric Vehicles (AEVs) market is hampered by the challenge of building efficient, reliable charging infrastructure, which is critical to the adoption and seamless operation of these vehicles. Current charging methods often require manual intervention, undermining the autonomy of AEVs and highlighting the need for automated solutions. To address this issue, the use of robot arms for automated charging has been proposed. These robotic systems leverage learning-based control algorithms to autonomously manipulate charging plugs, adapting to various AEV models and charging port placements to ensure efficient, reliable charging without human involvement.

In the realm of object detection for robot automation, the literature reveals significant advancements. Algorithms like YOLO (You Only

Look Once) and SSD (Single Shot MultiBox Detector) are prominent for their real-time processing capabilities. YOLO is known for its speed and efficiency in processing full images quickly, making it suitable for real-time applications, though it sometimes struggles with smaller objects [1-2]. SSD, while more complex and resource-intensive, excels in detecting smaller objects due to its multi-scale detection approach [3]. Various studies have employed these algorithms for different robot applications, demonstrating their efficacy in object detection and manipulation tasks, such as peg-in-hole assembly and vision-guided robot arm control [4-7]. A study by Xiaowei Xing and Dong Eui Chang [5] employed deep reinforcement learning to train neural networks using experiences sampled from a replay buffer that was continuously updated at each time step. Meanwhile, Shen et al [6] developed a learning-based neural network, P2HNet, for high-precision peg-in-hole assembly tasks. The study focused on extracting desired landmarks for visual servoing by mimicking human behavior to improve assembly accuracy. A study by Huang et al. [8] developed a step-by-step grasping strategy and a vision-guided method to improve the assembly capabilities of the Baxter Robot. Their study used a camera mounted on the robot end-effector to estimate positioning errors, which were compensated with a proportional–integral–derivative (PID) controller during assembly.

Despite these advancements, a notable gap remains in applying these technologies to the autonomous charging of AEVs. Existing studies have focused on general object detection and robot manipulation, but have not adequately addressed the integration of these systems into an automated AEV charging infrastructure [9]. This gap underscores the need for targeted research to develop and refine robotic systems capable of autonomous charging across diverse real-world scenarios.

The objective of this research is to develop a robot arm system for charging AEVs, employing learning-based object detection algorithms to identify and manipulate charging ports and plugs. The methodology involves selecting the appropriate algorithm, training it with relevant datasets, and testing it within a simulated environment to evaluate performance. The research conditions include controlled simulation settings that replicate real-world AEV charging scenarios, ensuring that the developed system can handle various operational challenges effectively.

4-DOF Robotic Charger for AEV

Accurate 3D models ensure the fidelity and reliability of the simulation results, serving as virtual representations of the physical entities involved in the charging process. These models enable detailed analysis and refinement of the robot's actions in a controlled, simulated environment. The two primary 3D models required for the simulation are the 4-DOF robot arm and the charging socket. Figure 1 shows the 3D model of the 4-DOF robot arm. The 3D model includes all its mechanical components, allowing for precise simulation of the robot’s movements and interactions. This model replicates the kinematic and dynamic properties of the real robot, allowing the testing of control algorithms and motion planning strategies.

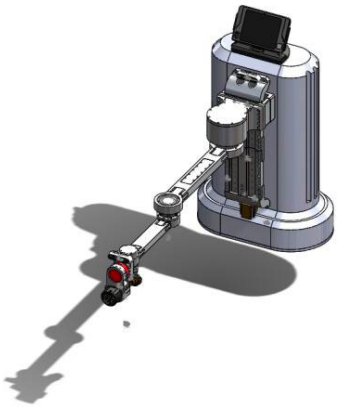


Figure 1: 4-DOF robotic charger equipped with a camera and actuator at the end effector [9]

Besides, the 4-DOF robot arm also has a Type 2 charging plug and a depth camera attached. The charging plug is located at the end of the robot arm and serves as its end effector, the component that interacts directly with the charging socket. The depth camera, on the other hand, is attached above the charging plug’s holder. This camera provides essential spatial data, enabling the robot to perceive its environment in three dimensions. By capturing depth information, the camera helps the robot to accurately determine the position of the charging socket and adjust its movements accordingly.

Figure 2 shows the 3D model of the charging socket. The 3D model includes the exact geometric features and dimensions, ensuring the simulation reflects the real-world socket’s design, which, in this case, is based on the Type 2 EV charging socket. This model accurately portrays the interaction between the robot arm and the socket, enabling the development and testing of precise plugging and unplugging actions.

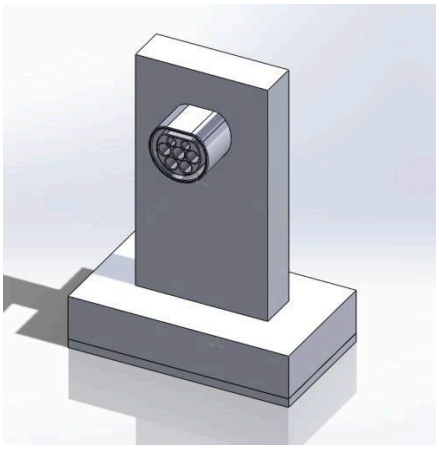


Figure 2: Type 2 EV charging socket

By integrating these models and components, the 4-DOF robot arm identifies the charging socket's location in pixel coordinates from the 2D image captured by the depth camera. These coordinates are then paired with depth information to obtain a 3D position within the camera's coordinate frame. This position is subsequently transformed into the robot base frame using TF transformation, based on the fixed spatial relationship between the camera and the robot arm. The resulting 3D coordinates are then used to estimate the robot arm's pose, allowing the system to determine the required joint configurations to accurately position the charging plug relative to the socket, thereby enabling the charging process.

Machine Learning-Based Object Detection Algorithm for Plugging and Unplugging

In selecting the most appropriate object detection algorithm, several factors were considered, including speed, accuracy, implementation complexity, and suitability for real-time processing. The two primary candidates evaluated were YOLO and SSD. YOLO is known for its speed, making it highly suitable for real-time applications. The algorithm employs a single neural network to predict bounding boxes and class probabilities in a single pass, significantly improving inference speed. This streamlined approach not only allows rapid processing but also simplifies the model architecture, making YOLO easier to train and deploy. YOLO can also efficiently and rapidly process full images. However, YOLO's performance can be slightly compromised when detecting smaller objects, where it may show lower accuracy compared to SSD.

SSD offers a more nuanced approach to object detection by using a series of smaller networks to detect objects at different scales and aspect ratios. This multiscale detection capability allows SSD to achieve higher accuracy, particularly for smaller objects. However, this complexity comes at the cost of increased computational resources and longer training times. SSD's multi-layer design, while enhancing accuracy, also makes the implementation more resource-intensive and less efficient than YOLO for real-time processing, particularly when dealing with larger objects. Considering the requirements of the research study, which focuses on real-time processing and high-speed detection of a charging socket, YOLO stands out as the most suitable algorithm. Its streamlined architecture ensures rapid inference and easier implementation, aligning well with the need for efficient and timely object detection[10-12]. As such, the latest version of YOLO used in this study is YOLOv8.

METHODOLOGY

Training the YOLO algorithm is a critical step in enabling it to accurately detect the charging socket. This process involves feeding the algorithm with annotated datasets and adjusting its parameters to optimise performance for the best result. The trained algorithm can then be used by the robot arm to detect charging sockets and carry out the charging process. The training setup for the YOLO algorithm involves configuring it to use the selected datasets. This includes setting hyperparameters such as the learning rate, batch size, and optimiser. The YOLO algorithm is initialised using the annotated datasets prepared earlier.

For training the YOLO algorithm, the YOLOv8n model is selected due to its lightweight architecture, which allows it to run on lower-end hardware. Three datasets are used, comprising 12 images of Type 2 EV sockets in total, including 3D models and real-life images annotated using CVAT in YOLO format. Due to the limited number of images, the same dataset was used for both training and validation. The model is trained up to 200 epochs using the default hyperparameters. The software tools employed for the development, training, and simulation of the YOLO-based object detection algorithm and robotic arm system are summarised in Table 1.

Table 1: Software tools

Category	Software
Detection Algorithm	YOLOv8n
Robot Framework	ROS2 Humble Hawkbill

Simulation	Gazebo-11
OS	WSL2 Ubuntu 22.04
Coding	Visual Studio Code

Image Dataset for Charging Socket Detection

A learning dataset is required to train the object detection algorithm. In this study, multiple images consisting of Type 2 EV sockets were compiled in order to be used as learning datasets. Because YOLO can be trained on a small dataset, three datasets with varying numbers of images were used to train the algorithm and assess its performance. Figure 3 shows the first- and second-category datasets for a Type 2 charging socket, based on a 3D model of the socket in SolidWorks. Figure 4 shows the second dataset, which includes some images of a real-life Type 2 charging socket. The third dataset includes all the images from the previous two datasets. After selecting the images and compiling the datasets,

(a) (b)

Figure 3: (a) Images of 3D Charging Socket on White Backgrounds (b) Images of 3D Charging Socket on Generic Backgrounds

Figure 4: Images of Real-Life Type 2 Charging Sockets

Proper annotation and labelling are applied to every image within the dataset. As shown in Figure 5, it is important to label a dataset correctly to ensure the accuracy and effectiveness of the object detection algorithm during training and testing phases [13]. Proper labelling allows the algorithm to learn and identify specific features as well as objects within the images.

Figure 5: Labeled Dataset

Virtual Environment for Simulation

A virtual environment for executing the charging process was developed by integrating various simulation components, including 3D models of the 4-DOF robot arm and the charging socket. Lighting conditions were also incorporated into the simulation environment, as they play a critical role in influencing visibility and object detection performance. The simulation environment illustrated in Figure 6 provides a controlled environment for testing different operating conditions, validating algorithm performance, and identifying potential system limitations, thereby improving the reliability and robustness of the proposed robotic charging system.

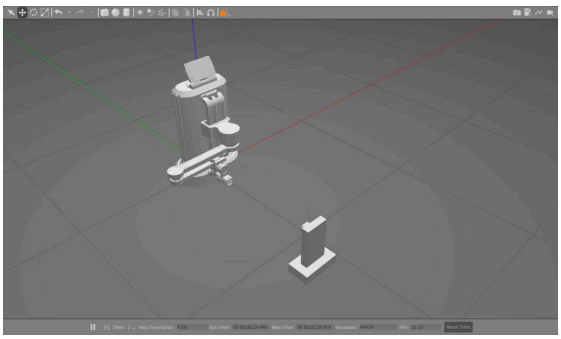


Figure 6: Virtual environment setup in Gazebo

Motion and Kinematic Development for Each Joint of the Robotic Charger

Before integrating with controllers, path-planning algorithms, and learning-based systems for testing and validation, the URDF package is utilised to model the robot's structure and simulate its kinematic behaviour within the ROS-based environment. It provides detailed descriptions of

the robot's links, joints, and physical properties, enabling accurate visualisation, motion planning, and collision-free simulation, as shown in Figure 7. It enables the simulation software to understand how the robot is constructed and how each component moves relative to one another, thus used for charging socket pose estimation and detection [14].

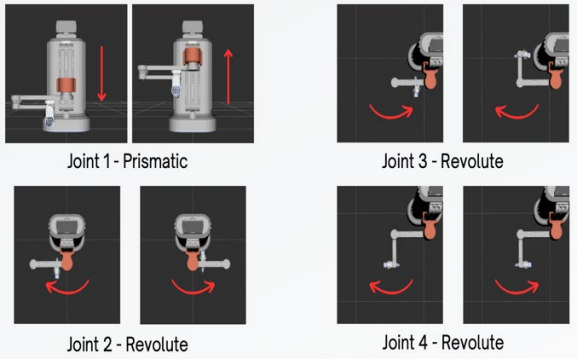


Figure 7: Detailed structure and kinematic behaviour

Results and Discussions

The programming for the motion planning of the 4-DOF robot arm, which is the MoveIt configuration package, resulted in the robot arm being able to plan the movements of its joints based on the end goal pose using the motion planner configurations and then executing the plan, moving the joints of the robot arm within the simulation environment. This enables the robot arm to plan and execute the motions required for the charging process. Figure 8 shows the planning and execution of movement in the simulation through the MoveIt configuration package. The performance of the 4-DOF robot arm is then evaluated by running 20 iterations across three simulation environments to collect data. The robot arm is in the same position, while the charging socket is 0.9m, 1m, or 1.1m away from the robot arm.

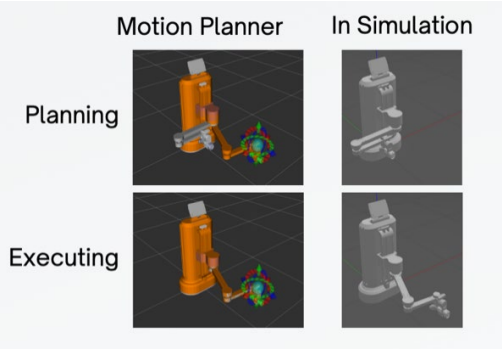


Figure 8: Configuration setup for the 4-DOF robotic arm in ROS

Confident Scores

The confidence scores of detected charging sockets in Figure 9 reflect the algorithm's certainty in its detections. The confidence score should be zero if there's nothing inside the box. However, if there is an object in the box, we want the confidence score to reflect how well the predicted box aligns with the actual object location [10]. High confidence scores indicate that the algorithm is not only detecting the charging socket but with a high degree of certainty, which is essential for ensuring reliable and accurate operation during the plug-in task of the charging process. The highest confidence score is obtained when the charging socket is placed 1.1m away from the robot arm, ranging from 0.85 to 0.87.

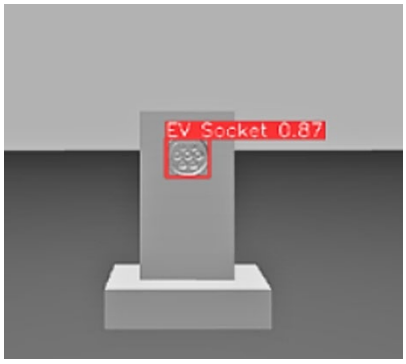


Figure 9: Charging socket detection

On the other hand, the lowest confidence score is recorded when the charging socket is placed 1m away, ranging from 0.73 to 0.75. These results indicate that the YOLO algorithm operates within an effective detection range, where increasing the distance does not necessarily improve the confidence score. This can be observed from the higher confidence score achieved at a distance of 0.9 m compared to 1.0 m as in Figure 10. At a distance of 0.9m, the average confidence score is 0.824, with a standard deviation of 0.010. Increase the distance to 1.0m, the average confidence score is 0.734, with a standard deviation of 0.011. Lastly, at a distance of 1.1m, the average confidence score is 0.874, with a standard deviation of 0.005. The standard deviations are very low across all distances, indicating high consistency in the model's performance at these ranges.

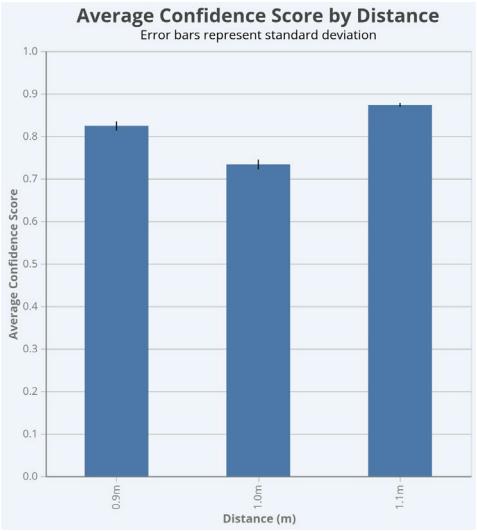


Figure 10: Model's confidence at different distances.

Accuracy of TF Transform

The accuracy of TF transforms is a metric that measures how precisely the robot arm's movements are mapped to the detected position of the charging socket. Accurate TF transforms are vital to ensuring the robot arm can align and insert the charging plug into the socket correctly, avoiding misalignment. The Comparison of Pixel Offset (Error) by Distance in Figure 11 reveals a clear trend. The detection accuracy improves significantly as the distance increases. The median pixel offset drops from approximately 1.83 at 0.9m to 0.63 at 1.0m, then to 0.04 at 1.1m. This suggests the system is most precise at a distance of 1.1 meters.

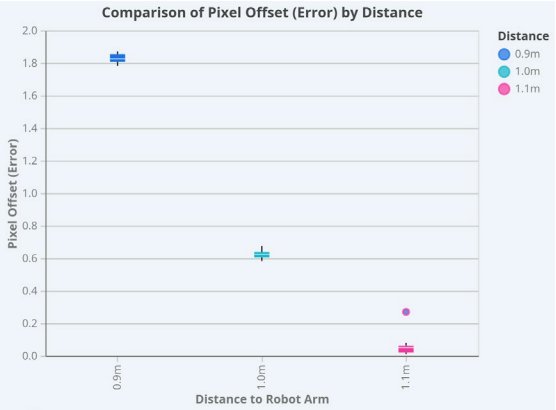


Figure 11: Offset of TF Transform against Iterations at Different Distances

Time Taken for Motion Planning

Additionally, the comparison of processing time versus distance in Figure 12 shows that processing time remains relatively consistent across all distances, typically ranging from 0.03 to 0.08 seconds. While there is slightly more variability at 1.1m (with some trials reaching nearly 0.1 seconds), the overall speed is comparable. The tight grouping in the box plots for 0.9m and 1.0m offsets indicates high consistency in the system's performance at those distances, whereas the 1.1m trials show a bit more spread in processing time, but the highest precision in positioning. These visualisations provide a comprehensive view of how distance affects both the precision and the speed of the robot arm's detection system.

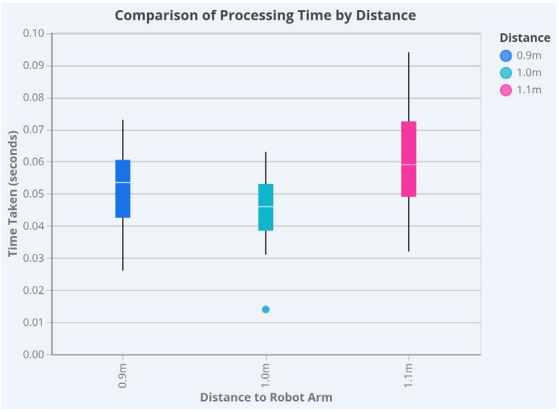


Figure 12: Time Taken for Motion Planning Against Iterations at Different Distances

Evaluation of Training and Validation Metrics

The algorithm is trained on three different datasets: the first dataset includes five synthetic images of a Type 2 charging socket based on a 3D model of the charging socket in SOLIDWORKS in front of various backgrounds, the second dataset is presented by seven real-life images of a Type 2 charging socket and the third dataset describes 12 images, merging the two previous datasets into a singular dataset. The performance of the learning-based algorithm is evaluated using various metrics, such as training and validation losses.

Training losses provide insight into how well the model is learning from the training data over epochs. There are three aspects of training losses: box loss, cls loss, and dfl loss. Figure 13 illustrates the box loss, which refers to the loss associated with the bounding box predictions during training. The analysis of train/box_loss reveals that synthetic images enable faster learning and lower localisation

errors due to their controlled environment. In contrast, real-life images present more challenges, leading to higher initial losses and greater variability. The combined dataset improves the model's ability to handle diverse scenarios, resulting in the lowest and most stable box loss.

Figure 13: Comparison of synthetic, real-life, and combined datasets during training.

Furthermore, Figure 14 illustrates the classification loss during training, which measures how well the predicted class labels match the actual class labels for each detected object. The cls_loss analysis highlights that synthetic images enable quicker learning and lower classification errors. Real-life images introduce variability, leading to greater fluctuations and higher losses. The combined dataset helps achieve a balance, resulting in smoother, lower classification loss and reflecting improved robustness in the model's ability to classify Type 2 sockets across different environments.

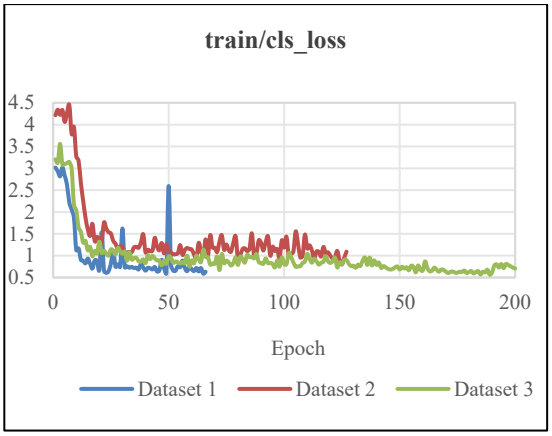


Figure 14: Comparison of classification loss for all datasets during training.

Figure 15 illustrates the distribution focal loss used during training to address class imbalance and improve the learning of difficult-to-classify samples. The dfl_loss analysis suggests that synthetic images provide a controlled environment where the model can quickly handle class imbalance. Real-life images introduce more variability, leading to higher fluctuations and losses. The combined dataset offers a balanced approach, resulting in smoother, lower dfl_loss and demonstrating the model's improved capability to manage class distributions effectively.

Another analysis focuses on validation losses, which measure the model's performance on unseen data not used during training. Similarly to training losses, there are three different aspects which are also box loss, cls loss and dfl loss. Figure 15 illustrates the validation loss for bounding box predictions, providing an evaluation of the model's performance on the validation set.

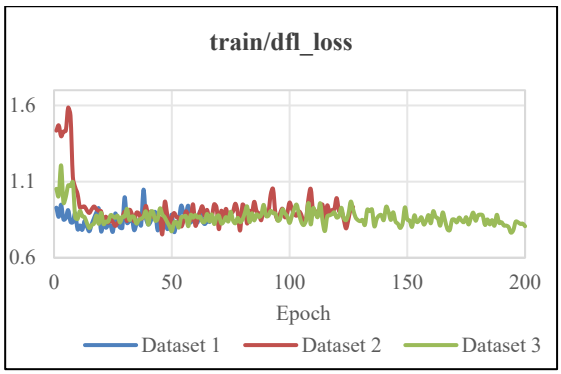


Figure 15: Comparison of focal loss distribution for all datasets during training.

The val/box_loss analysis in Figure 16 indicates that while the model generalises well on synthetic data, it faces challenges with real-life images, leading to higher losses. The combined dataset shows an improvement, though not as significant as in training losses, suggesting that real-life variability continues to present challenges for generalisation despite improved robustness.

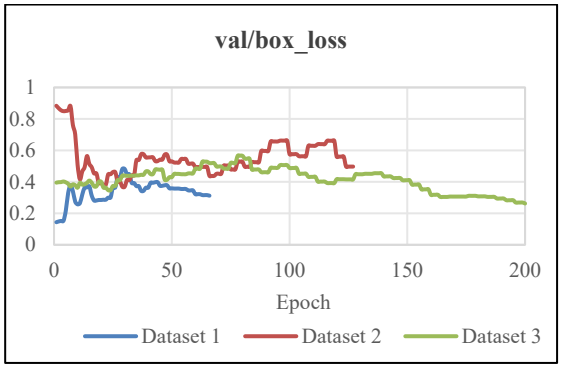


Figure 16: Comparison of validation loss related with the bounding box predictions

On the other hand, Figure 17 illustrates the classification loss during validation, evaluating how well the model predicts class labels on the validation dataset. The val/cls_loss analysis reveals that the model performs well on synthetic data but struggles with real-world images, resulting in higher losses. The combined dataset improves classification, although the complexity of real-life scenarios still poses challenges, which shows the need for more diverse and extensive real-life data for better generalisation [15].

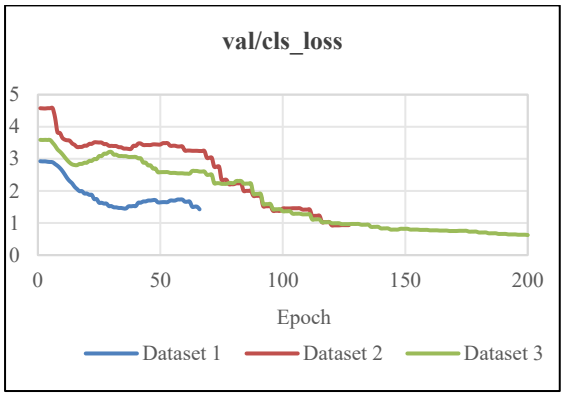


Figure 17: Comparison of validation loss related with the classification on the dataset

Moreover, the val/dfl_loss analysis shown in Figure 18 suggests that while the model handles class imbalance well on synthetic data, real-world images introduce greater variability, leading to higher losses. The combined dataset offers improved performance, though real-world complexity continues to challenge class-imbalance handling, requiring more diverse training data.

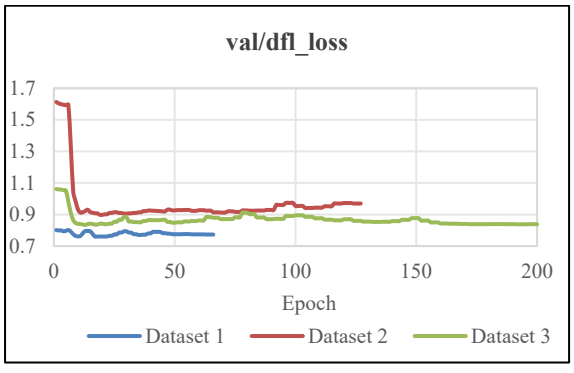


Figure 18: Comparison of validation loss related with the focal distribution in dataset

Precision and recall metrics are also key indicators of the model's accuracy and completeness in detecting the target objects. Precision measures the proportion of true positive detections among all positive detections made by the model, while recall measures the proportion of true positive detections among all actual positives. Figure 19 presents the precision, which measures the accuracy of the positive predictions. It is the ratio of true positive detections to the total number of positive detections (true positives + false positives). The precision analysis shows that synthetic images allow the model to achieve high accuracy quickly, with minimal false positives. Real-life images introduce greater variability, leading to initial fluctuations in precision. The combined dataset stabilises precision, resulting in consistently high accuracy, highlighting the need for diverse training data.

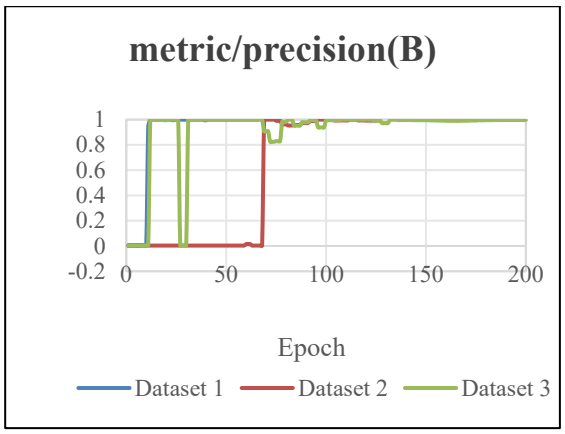


Figure 19: Precision performance comparison for synthetic, real-life, and combined datasets during model training.

Meanwhile, Figure 20 shows the recall metric analysis, which measures the ability of the model to find all relevant instances in the dataset. It is the ratio of true positive detections to the total number of actual positives (true positives + false negatives). The recall analysis reveals that the model quickly detects all instances in synthetic data, whereas real-life images pose greater challenges, leading to initial fluctuations. The combined dataset helps maintain high recall, demonstrating the model's improved ability to detect Type 2 sockets in deployment.

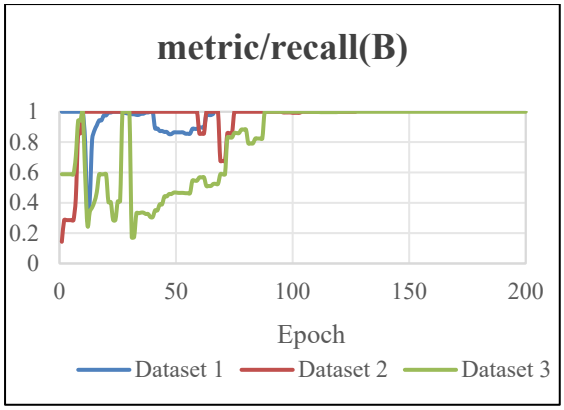


Figure 20: Recall performance comparison for synthetic, real-life, and combined datasets during model training.

Conclusion

This study aimed to develop a learning-based control algorithm for a 4-DOF robot arm to enhance the charging processes of Autonomous Electric Vehicles (AEVs). Motivated by the limitations of manual AEV charging methods, the project successfully integrated a 4-DOF robot arm model into Gazebo for simulation testing. The developed control algorithm enables the robot arm to identify and align the charging plug with the socket, reducing the reliance on manual operations. Evaluation metrics such as precision and recall demonstrated the algorithm's effectiveness across various datasets, validating its adaptability to different conditions.

Future considerations that can help improve the robot arm and algorithm include transitioning from simulation to real-world testing, recoding any failed packages, and expanding the dataset to increase its size and diversity for a more robust algorithm.

Acknowledgements

The authors thank Universiti Teknologi Malaysia (UTM) and Faculty of Mechanical Engineering for supporting this research.

References

[1] *Murat,A. A & Kiran, M.S. (2025).* A comprehensive review on YOLO versions for object detection. *Engineering Science and Technology, an International Journal, 70, 102161. https://doi.org/10.1016/j.jestch.2025.102161*

[2] *Ali, M. L., & Zhang, Z. (2024). The YOLO Framework: A Comprehensive Review of Evolution, Applications, and Benchmarks in Object Detection. Computers, 13(12), 336. https://doi.org/10.3390/computers13120336*

[3] *Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. and Berg, A. (2016). 'SSD: Single Shot MultiBox Detector'. Lecture Notes in Computer Science (Including Subseries lecture Notes in Artificial Intelligence and lecture Notes in Bioinformatics), 9905 LNCS, 21-37. https://doi.org/10.1007/978-3-319-46448-0_2.*

[4] *Zhang, Z., Fang, Z., Lian, H., Zhang, C. and Yang, G. (2021). 'Robot Peg-in-Hole Assembly System Based on Vision and Fuzzy Control', 33rd Chinese Control and Decision Conference (CCDC). https://doi.org/10.1109/CCDC52312.2021.9602060.*

[5] *Xing, X. and Chang, D.E. 'Deep Reinforcement Learning Based Robot Arm Manipulation with Efficient Training Data through Simulation', 19th International Conference on Control, Automation and Systems (ICCAS). https://doi.org/10.23919/ICCAS47443.2019.8971637.*

[6] *Shen, Y., Jia, Q., Wang, R., Huang, Z., and Chen, G. (2023). 'Learning-Based Visual Servoing for High-Precision Peg-in-Hole Assembly' Actuators. https://doi.org/10.3390/act12040144.*

[7] *Xu, S., Chen, K., Ou, Y., Wang Z. and Yang, C. 'A Learning-Based Object Tracking Strategy Using Visual Sensors and Intelligent Robot Arm', IEEE Transactions on Automation Science and Engineering. https://doi.org/10.1109/TASE.2022.3213730.*

[8] *Huang, Y., Zhang, X., Chen, X. and Ota, J. (2017). 'Vision-guided peg-in-hole assembly by Baxter robot', Advances in Mechanical Engineering. https://doi.org/10.1177/168781401774807*

[9] *Saputra, H.M., Rifansyah,R.Y., Baskoro, C. H. A. H. B.,Mohd Nor, N. S., Rijanto E. and Pahrurrozi,A.(2024) "Comparative Study of Machine Learning Models for Inverse Kinematic Prediction of a Flexible-Tube Wrist Mechanism in Robotic Charging Stations," IEEE International Conference on Smart Mechatronics (ICSMech), Yogyakarta, Indonesia, 2024, pp. 152-158, doi: 10.1109/ICSMech62936.2024.10812340.*

[10] *Güney, E., Bayılmış, C., Çakar, S., Erol, E., & Atmaca, Ö. (2024). Autonomous control of shore robotic charging systems based on computer vision. Expert Systems with Applications, 238, 122116.*

[11] *Jia, X., Tong, Y., Qiao, H., Li, M., Tong, J., & Liang, B. (2023). Fast and accurate object detector for autonomous driving based on improved YOLOv5. Scientific Reports, 13 (1), 1–13. https://doi.org/10.1038/s41598-023-36868-w*

[12] *Lu, Y., Qiu, Z., Liao, C., Zhou, Z., Li, T., & Wu, Z. (2022). A GIS Partial Discharge Defect Identification Method Based on YOLOv5. Applied Sciences (Switzerland), 12(16). https://doi.org/10.3390/app12168360.*

[13] *Zhu, H., Sun, C., Zheng, Q., & Zhao, Q. (2023). Deep Learning Based Automatic Charging Identification and Positioning Method for Electric Vehicle. Computer Modeling In Engineering & Sciences, 136(3), 3265–3283. https://doi.org/10.32604/cmes.2023.025777*

[14] *Chikurtev, D. (2020, October). Mobile robot simulation and navigation in ROS and Gazebo. In 2020 International Conference Automatics and Informatics (ICAI) (pp. 1-6). IEEE.*

[15] *Tang, G., Liu, S., Fujino, I., Claramunt, C., Wang, Y., & Men, S. (2020). H-yolo: A singleshot ship detection approach based on region of interest preselected network. Remote Sensing, 12(24), 1–18. https://doi.org/10.3390/rs12244192*