Alexandria Engineering Journal 134 (2026) 122–134

Contents lists available at ScienceDirect

Alexandria Engineering Journal

journal homepage: www.elsevier.com/locate/aej

Original article

Autonomous charging system via manipulator-UGV docking using zero-shot 6-DoF pose estimation

Minkyu Jung, Andrew Jaeyong Choi *

School of Computing, Gachon University, 1342 Seongnam-daero, Sujeong-gu, Seongnam-si, 13120, Gyeonggi-do, Republic of Korea

ARTICLE INFO

Keywords:
Autonomous docking
6-doF pose estimation
Open-vocabulary object detection
Point cloud localization
Manipulator inverse kinematics
ROS-based integration

ABSTRACT

To ensure long-term field operation of Unmanned Ground Vehicles (UGVs) without human intervention, autonomous charging capability is essential. This paper presents a fully autonomous charging system that enables a UGV to perform path planning, align itself with the charging station, and a 6-DoF manipulator to locate and connect to the charging port without relying on external markers. By combining vision-language detection with geometric reasoning on RGB-D data, the system estimates a complete 6-DoF pose without relying on fiducial markers or task-specific retraining, thereby enhancing adaptability across different docking environments. Specifically, it employs the open-vocabulary object detector YOLO-World to identify the docking port in the RGB image and selects the corresponding 3D position from the depth map. To estimate orientation, the surrounding point cloud is clustered to extract the object shape, from which the without relying on fiducial markers or task-specific retraining, thereby enhancing adaptability across different docking environments. principal axis is computed, resulting in a fully generalizable and marker-free docking pipeline. This 6-DoF pose is used to compute a feasible manipulator configuration through inverse kinematics. The proposed pipeline is integrated into a ROS-based framework and enables precise docking in unstructured outdoor environments without manual supervision. Experimental results demonstrate the robustness and accuracy of the system under real-world conditions, achieving a docking success rate of 90% and an average total operation time of 34.4 s.

1. Introduction

The use of manipulators and unmanned ground vehicles (UGVs) is finding increasing use in military logistics, disaster relief, and industrial automation [1–3]. To remain in the field for hours or days without human intervention, such platforms must charge themselves autonomously. In this mission, the UGV navigates to a charging station, the arm locates the charging port of the vehicle, and the system completes the physical connection, all without external markers or teleoperation.

Early charging stations relied on planar fiducial markers such as AprilTag or ArUco to simplify pose estimation, yet these tags introduce three structural drawbacks: the charging port must follow a single, predefined geometry; the markers are easily obscured by dirt, glare, or mechanical damage; and every station requires periodic calibration and maintenance, particularly costly in outdoor environments. Point cloud registration methods such as CAD-to-ICP alignment [4] eliminate visual tags but inherit the same predefined geometry and demand dense, low-noise depth data.

Traditional image-class pair-based detectors have raised detection accuracy to near-real-time levels on desktop GPUs, but their closed vocabularies make them brittle in the field. Introducing a new port design still requires collecting and labeling hundreds of images, retraining or fine-tuning the network, and redeploying the weights, while any domain shift caused by dirt or wear can sharply erode performance. Even anchor-free YOLOv8 [5], although faster and more accurate, cannot escape this retraining cycle.

YOLO-World [6] represents a vision-language foundation model specifically redesigned for real-time object detection. Unlike conventional YOLO detectors that rely solely on visual supervision, mitigates this limitation by integrating a compact CLIP-based [7] text encoder with a YOLOv8 backbone to enable open-vocabulary recognition through image-text alignment. The CLIP text encoder is kept frozen without fine-tuning, while the RepVL-PAN module adaptively reparameterizes the vision-language features. Text prompts are encoded into 512-dimensional semantic embeddings and fused with visual representations via region-to-text contrastive learning, allowing the model to detect unseen object categories while maintaining the speed and efficiency required for robotic perception.

Using the two dimensional map generated by LiDAR-IMU SLAM, the UGV autonomously drives to the front of the charging station

* Corresponding author.
E-mail address: andrewjchoi@gachon.ac.kr (A.J. Choi).

https://doi.org/10.1016/j.aej.2025.12.015
Received 5 August 2025; Received in revised form 24 November 2025; Accepted 6 December 2025
Available online 9 December 2025

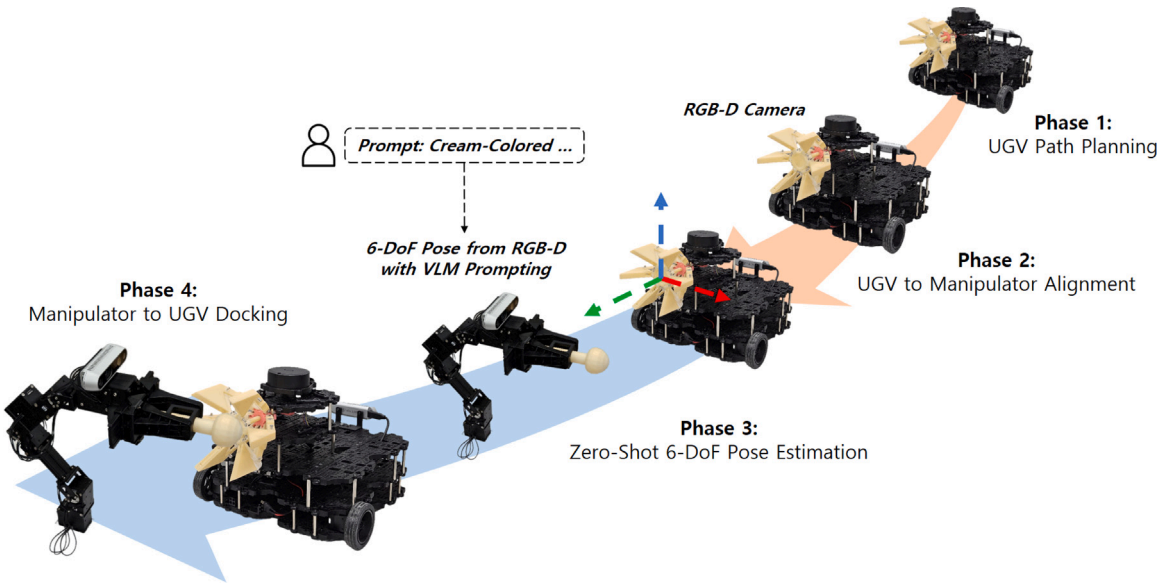


Fig. 1. Overall docking pipeline combining prompt-based object detection, 6-DoF pose estimation, and manipulator-UGV coordination.

while avoiding obstacles. Upon arrival, the vehicle relies on its onboard RGB-D camera to position itself inside the reachable workspace of the manipulator, ensuring stable and accurate docking. A RGB-D camera mounted on the station, together with the YOLO-World model, detects the docking port. A depth crop inside the bounding box produces a point cloud, and the point closest to the centroid defines a 3D goal. The manipulator's inverse kinematics controller then aligns the end-effector to this goal and executes the reach motion. Consequently, the full sequence shown in Fig. 1: UGV approach, vehicle positioning, port detection, manipulator control, and final docking concludes without human intervention even in cluttered environments.

This study makes the following main contributions:

(1) We propose a marker-free and retraining-free 6-DoF pose estimation framework built upon an open-vocabulary detection paradigm, enabling generalizable docking across diverse targets.
(2) We implement an integrated manipulator-UGV system that performs autonomous path planning, alignment, and docking without human intervention.
(3) We experimentally validate the proposed system on a real robotic platform, achieving a 90% docking success rate and an average total operation time of 34.4 s under real-world conditions.

2. Related work

In this section, we review existing autonomous charging systems and highlight the key distinctions between them and the proposed framework. Traditional autonomous charging systems generally rely on 3-DoF pose estimation, which focuses on position and orientation, restricting their ability to align accurately with the docking port. In contrast, the method presented in this study utilizes 6-DoF pose estimation, allowing for full spatial alignment in all degrees of freedom, resulting in more precise docking. Furthermore, while conventional systems typically depend on pre-trained models, our approach incorporates open-vocabulary object recognition to detect novel docking targets. This learning-based method enhances flexibility and adaptability to diverse environments.

The following subsections discuss various strategies for docking technology employed in autonomous charging systems.

2.1. Marker-based approaches

Early research on autonomous charging and docking relied heavily on visual fiducial markers. The introduction of two-dimensional black-and-white markers such as ARTag and AprilTag rapidly accelerated outdoor docking experiments for UGVs and UAVs [8,9]. Narváez et al. [10] proposed an autonomous docking system using a mobile manipulator to track, approach, and connect with a VTOL UAV based on fiducial marker detection. Progressively, arranging multiple tags on grid-style marker boards extended the detection range, and retro-reflective or optical fiber variants were developed to maintain robustness under extreme lighting conditions, including deep-sea and space environments [11,12]. Most recently, subpixel edge fitting techniques that minimize corner detection error [13], together with iterative visual servoing that predicts trajectories by tracking only the centroid marker [14], show centimeter-level precision with the same set of tags. Pivoňka et al. [15] applied monocular fiducial marker-based localization using a single monocular camera for autonomous docking of Autonomous Guided Vehicles (AGVs), estimating robot poses from marker positions. Bian et al. [16] introduced FiMa-Reader, a cost-effective fiducial marker system combining ArUco and Data Matrix markers in a hybrid approach with an IR camera-based reader for robust marker detection under varying lighting conditions. But, the problem of pre-defined geometry, contamination, and regular correction has not been fundamentally solved, so the non-marker family has recently been studied.

2.2. Deep learning-based approaches

Deep learning-based detection methods in robotic docking can be broadly divided into conventional vision-based detectors, which rely on fixed label sets for supervised training (see Section 2.2.1), and vision-language-based detectors, which leverage joint image-text embeddings to recognize unseen categories through cross-modal alignment (see Section 2.2.2).

2.2.1. Conventional vision-based detectors

Since the deep learning boom, research has shifted from marker-based localization to direct object detection. The introduction of R-CNN [17] enabled deep learning-based object detection, and Faster R-CNN [18] accelerated the detection pipeline. The YOLO detector series [5,19] further optimized real-time detection performance by

balancing speed and accuracy, making it widely adopted in robotics research [20–22]. There are docking studies [23–26] using these object detection models. Choi et al. [23,24] proposed a robust docking system for Unmanned Aerial Systems (UAS) by leveraging a YOLO-based object detection model. Yang et al. [25] proposed a 6-DoF refueling robotic arm positioning and docking method integrating YOLOv8 with the PnP algorithm for precise detection and pose estimation of refueling interfaces. However, this approach requires dedicated training for the docking port and does not integrate the entire refueling system including the UGV, limiting its applicability in full autonomous refueling scenarios. Xu et al. [26] developed a YOLOv4-based detection and improved RRT* planning system for efficient and collision-free robotic arm docking in complex environments. However, the system relies solely on an RGB camera without depth sensing, which limits precise docking and makes the system sensitive to environmental changes, potentially leading to inconsistent performance. Ren et al. [27] proposed a Dynamic Graph Transformer (DGT) that dynamically constructs local-to-global graph structures over 3D point clouds and applies self-attention to capture long-range dependencies, improving robustness against occlusion and sparse observations. However, since the model is primarily designed for LiDAR-based 3D detection, its computational cost and data format make it less practical for real-time robotic docking. Das et al. [28] developed a hybrid Vision Transformer (ViT)-Graph Convolutional Network (GCN) framework that jointly models global context and structural relations, demonstrating improved accuracy under low-contrast and cluttered scenes. While this hybrid design enhances generalization and interpretability, it introduces considerable architectural complexity and higher inference latency, posing challenges for deployment on resource-constrained robotic platforms. Yin et al. [29] combined graph neural networks with spatiotemporal Transformer attention for 3D video object detection, enabling temporal consistency and spatial reasoning across frames. This spatiotemporal modeling improves detection stability under motion and illumination variations, yet its heavy computational demand and sequence-based architecture make real-time integration difficult for continuous docking tasks.

2.2.2. Vision-language-based detectors

Vision-language-based detectors aim to generalize object recognition beyond fixed label sets by aligning image and text representations in a shared semantic space. CLIP learns a joint embedding in which a text prompt can retrieve visual concepts that were never seen during training. Building on CLIP, GLIP [30] extends this idea to sentence-level detection by predicting a bounding box for every noun phrase, and Grounding DINO [31] couples a large Transformer detector with a language encoder to ground arbitrary textual prompts in open-set images. Although these models achieve strong zero-shot accuracy, their hundred-million-parameter backbones and cross-modal decoders impose heavy computational and memory demands, making them unsuitable for real-time or embedded robotic perception. In contrast, YOLO-World inherits the vision-language alignment paradigm but employs a lightweight one-stage architecture with minimal cross-modal interaction, achieving high throughput and low inference latency. Due to its balance between semantic generalization and real-time efficiency, YOLO-World was selected as the core perception model in this study for real-time robotic docking tasks.

2.3. Reinforcement learning-based approaches

There are docking studies using reinforcement learning. These studies [32–34] focus on enhancing adaptability to dynamic environmental changes compared to conventional object detection-based docking systems. Reinforcement learning models learn optimal action strategies through reward-driven mechanisms, enabling dynamic obstacle avoidance and performance improvement through iterative training. Chu et al. [32] proposed a reinforcement learning-based docking control strategy for autonomous underwater vehicles by incorporating adaptive reward shaping to improve learning efficiency and docking performance. Weber et al. [33] proposed a reinforcement learning-based robot docking system that leverages neural vision for adaptive pose estimation and policy learning in dynamic environments. Muse et al. [34] developed a robot docking method that integrates omnidirectional visual perception with an actor-critic reinforcement learning framework to enable adaptive behavior in dynamic environments. However, such methods still require task-specific training for new target objects, resulting in high maintenance costs.

3. Method

In this section, we introduce the overall methodology of the proposed system, which is divided into three functional modules. Section 3.1 describes the UGV responsible for autonomous path planning. Section 3.2 provides a description of the docking port hardware. Section 3.3 explains the object recognition perception system that detects and localizes the docking port. Finally, Section 3.4 presents the manipulator module that executes the physical docking procedure.

3.1. UGV

The UGV employs a two-stage path planning approach for autonomous docking: a global path is generated using the A* algorithm [35] for collision-free navigation, and local adjustments are made in real-time with the Dynamic Window Approach (DWA) [36] to avoid dynamic obstacles and ensure smooth motion toward the manipulator. Once the UGV reaches the vicinity of the manipulator, it uses Algorithm 1 to align itself with the docking direction, computing the heading angle based on the detected target center.

Algorithm 1: Vision-based heading control toward target point

Input: Base frame $\mathcal{F}_{\text{base}}$
Target frame $\mathcal{F}_{\text{target}}$
Output: Velocity command (v, ω) at each stage

Initialize stage $s := 0$;
while *not shutdown* **do**
  if $s = 0$ **then**
    $(x, y) \leftarrow T(\mathcal{F}_{\text{base}} \rightarrow \mathcal{F}_{\text{target}})$;
    $\theta \leftarrow \text{atan2}(y, x)$;
    if $|\theta| < \epsilon$ **then**
      $s \leftarrow 1$;
    end
    $\omega \leftarrow \text{clip}(\theta, -\omega_{\max}, \omega_{\max})$;
    $(v, \omega) \leftarrow (0, \omega)$;
  else if $s = 1$ **then**
    $(x, y) \leftarrow T(\mathcal{F}_{\text{base}} \rightarrow \mathcal{F}_{\text{target}})$;
    if $x < d_{front}$ **then**
      $s \leftarrow 2$;
    end
    $\theta \leftarrow \text{atan2}(y, x)$;
    $(v, \omega) \leftarrow (v_{\text{fwd}}, \text{clip}(\theta, -\omega_{\max}, \omega_{\max}))$;
  else if $s = 2$ **then**
    $\phi \leftarrow T(\mathcal{F}_{\text{map}} \rightarrow \mathcal{F}_{\text{base}})$;
    if $|\phi - (\phi_0 + \pi)| < \epsilon$ **then**
      exit loop;
      break;
    end
    $\omega \leftarrow \text{clip}(\phi_{\text{target}} - \phi, -\omega_{\max}, \omega_{\max})$;
    $(v, \omega) \leftarrow (0, \omega)$;
  `Publish` (v, ω) `to /cmd_vel`;
end

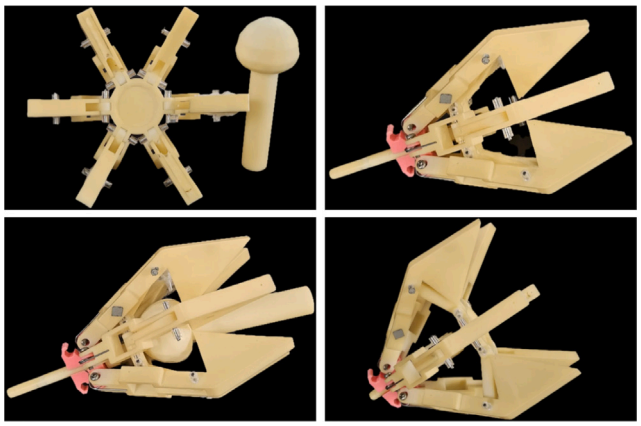


Fig. 2. Docking/release sequence showing the effect of spring on actuation.

3.2. Docking port

The docking port used in this system is derived from a probe-and-drogue mechanism design proposed by Choi et al. [23]. It features a recessed circular inlet with a diameter of 120 mm and is manufactured using high-resolution resin-based 3D printing as illustrated in Fig. 2.

Inside the port, a mechanical assembly enables passive switching between two discrete states: one in which the probe is secured and another in which it is released. This bistable behavior is achieved without continuous actuation by utilizing a torsional spring to support the locking phase and a pair of shape memory alloy (SMA) springs for releasing. The tapered inlet geometry and recessed profile guide the probe toward the locking interface, enabling smooth capture and reliable release through the bistable mechanism during the final stage of physical contact. In particular, the funnel-shaped lead-in surface and rounded entry boundary mitigate lateral misalignment during approach, allowing the probe to self-center even when positional or angular errors are present. Additionally, the internal locking latch is designed with mechanical compliance, allowing slight elastic deformation during insertion and ensuring firm engagement once the probe reaches the fully seated position. This geometry minimizes mechanical stress on both connector surfaces and provides robustness against small disturbances, ensuring repeatable docking and release cycles in real operation.

3.3. Object recognition

To detect the docking port, a YOLO-World model is used with open-vocabulary prompt. Unlike conventional object detectors that rely on closed-set classification and require retraining for new object types, YOLO-World allows open-vocabulary detection based on textual prompts. This flexibility is advantageous in outdoor autonomous systems, where target designs such as docking ports may vary across deployments and environments.

YOLO-World optimizes a total loss that combines four components: box regression, objectness, classification, and vision-language alignment. The total loss function is expressed as Eq. (1):

$$\mathcal{L}_{\text{total}} = \lambda_{\text{box}} \cdot \mathcal{L}_{\text{box}} + \lambda_{\text{obj}} \cdot \mathcal{L}_{\text{obj}} + \lambda_{\text{cls}} \cdot \mathcal{L}_{\text{cls}} + \lambda_{\text{align}} \cdot \mathcal{L}_{\text{align}} \tag{1}$$

Each loss term is scaled by a corresponding weight coefficient λ, which controls its contribution to the total objective during optimization. As shown in Eq. (2), the box regression loss is defined using the Complete IoU (CIoU) metric, which incorporates the overlap area, center distance, and aspect ratio between the predicted box b^{pred} and the ground truth box b^{gt}:

$$\mathcal{L}_{\text{box}} = 1 - \text{CIoU}(b^{\text{pred}}, b^{\text{gt}}) \tag{2}$$

Table 1
Descriptions of terms used in YOLO-World loss functions.

Symbol	Meaning
λ_{box}	Weight for box regression loss
λ_{obj}	Weight for objectness loss
λ_{cls}	Weight for classification loss
λ_{align}	Weight for alignment loss
b^{pred}	Predicted bounding box
b^{gt}	Ground truth bounding box
p_{obj}	Predicted objectness probability
y_{ij}	Ground truth binary label (1 if match, else 0)
N	Batch size
ϕ_i	Visual feature embedding from detector
t_i	Ground truth text embedding corresponding to ϕ_i
t_j	Non-matching text embedding in the batch
τ	Temperature parameter for softmax

A higher CIoU indicates better alignment, and the loss is defined as $1 - \text{CIoU}$ to minimize poorly aligned predictions. The objectness loss, defined as in Eq. (3), uses binary cross-entropy between the predicted objectness score p_{obj} and a target indicator function:

$$\mathcal{L}_{\text{obj}} = \text{BCE}(p_{\text{obj}}, \mathbb{1}_{\text{IoU}>0.5}) \tag{3}$$

where the indicator function is defined inline as $\mathbb{1}_{\text{IoU}>0.5} = 1$ if IoU $>$ 0.5, else 0. This encourages the model to assign high confidence only to bounding boxes that sufficiently overlap with ground truth.

As shown in Eq. (4), the classification loss in open-vocabulary settings is computed using the cosine similarity between visual features ϕ_i and text embeddings t_j, passed through a sigmoid activation $\sigma(x) = 1/(1 + \exp(-x))$:

$$\mathcal{L}_{\text{cls}} = \text{BCE}(\sigma(\phi_i^{\top} \cdot t_j), y_{ij}) \tag{4}$$

The output is compared against the ground truth label $y_{ij} \in \{0, 1\}$ using binary cross-entropy. The alignment loss, as shown in Eq. (5), encourages the visual embedding ϕ_i to be aligned with the corresponding text embedding t_i via contrastive learning:

$$\mathcal{L}_{\text{align}} = -\frac{1}{N} \sum_{i=1}^{N} \log \frac{\exp(\phi_i^{\top} t_i / \tau)}{\sum_{j=1}^{N} \exp(\phi_i^{\top} t_j / \tau)} \tag{5}$$

The temperature parameter τ scales the softmax distribution and stabilizes optimization. During inference, YOLO-World computes the class score between image region k and text prompt j as shown in Eq. (6):

$$s_{k,j} = \alpha \cdot \|e_k\| \cdot \|w_j\|^{\top} + \beta \tag{6}$$

Here, e_k and w_j denote the visual and textual embeddings, respectively, and α, β are learnable scalar parameters. This formulation enables the model to measure semantic correspondence through a calibrated inner product, allowing open-vocabulary detection without relying on fixed category classifiers. The inference pipeline that realizes this computation is illustrated in Fig. 4, where image features and text embeddings are projected into a shared space and compared to produce region-level similarity scores (see Table 1).

To enable precise docking to a 6-DoF pose, it is necessary to estimate a reliable 3D reference point corresponding to the detected object. Since YOLO-World provides only a 2D bounding box on the image plane, a center-guided point cloud localization is used to locate the closest 3D point in the point cloud that best represents the center of the target object. Algorithm 2 identifies the 3D point in the point cloud whose 2D projection lies closest to the center of a bounding box. This point is selected as the docking target position and is also used by the UGV to align itself with the charging station. $\bar{x}, \bar{y}, \bar{z}$ are the coordinates of the closest 3D point. w and h denote the image width and height. δ is the minimum 2D distance observed during the search. f_{proj} and f_{dist} are defined as follows:

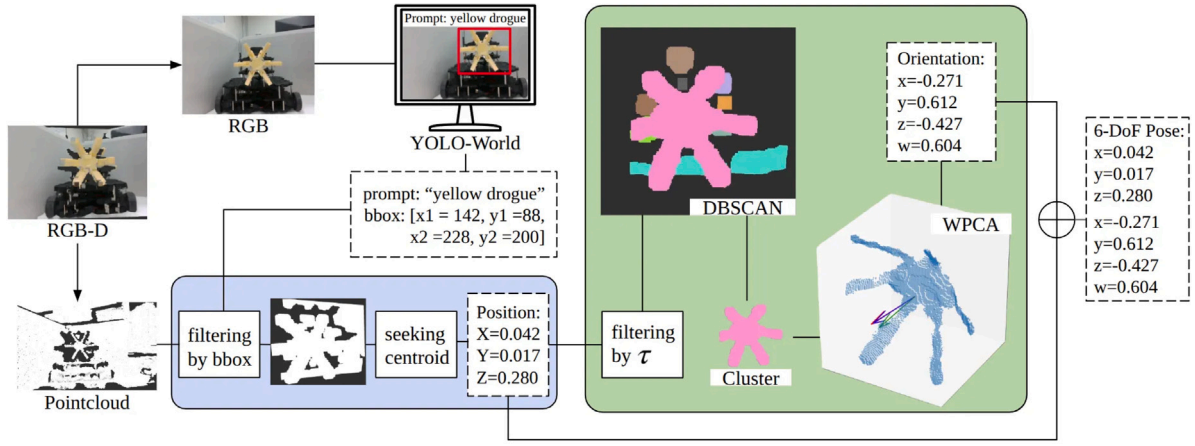


Fig. 3. Full pipeline of object recognition. The process includes object detection, point cloud clustering, and orientation estimation. The blue box corresponds to Algorithm 2, while the green box represents Algorithm 3.

Algorithm 2: Find the closest point to the center of the bounding box

Input: Point cloud $\mathcal{P}$
Bounding box $\mathcal{B}$
Camera parameters fx, fy, cx, cy
Output: Closest point coordinates $(\bar{x}, \bar{y}, \bar{z})$

Initialize $\tilde{\mathcal{P}}$ as empty list;
Initialize $i = -1, \bar{x} = 0, \bar{y} = 0, \bar{z} = 0$;
Initialize $\delta = \infty$;
for *each point* p *in* $\mathcal{P}$ **do**
  $(u, v) = f_{\text{proj}}(p, fx, fy, cx, cy)$;
  if $u \geq 0$ *and* $u <$ w *and* $v \geq 0$ *and* $v <$ h *and* $p.z > 0$ **then**
    if p *is inside* $\mathcal{B}$ **then**
      Add p to $\tilde{\mathcal{P}}$;
      $\hat{u} = (\mathcal{B}.x_{min} + \mathcal{B}.x_{max})/2$;
      $\hat{v} = (\mathcal{B}.y_{min} + \mathcal{B}.y_{max})/2$;
      $d = f_{\text{dist}}(u, v, \hat{u}, \hat{v})$;
      if $i == -1$ *or* $d < \delta$ **then**
        Update $i, \bar{x}, \bar{y}, \bar{z}, \delta$;
      end
    end
  end
end
Return $(\bar{x}, \bar{y}, \bar{z})$;

- f_{proj}: projects a 3D point to 2D image coordinates using camera intrinsics.
- f_{dist}: computes the 2D Euclidean distance to the bounding box center.

The orientation estimation process is detailed in Algorithm 3. Together with Algorithm 2, it forms the complete 6-DoF pose estimation pipeline that integrates position and orientation into a unified framework. Prior to clustering, a depth-based cropping is applied to the filtered point cloud $\mathcal{P}'$ using a threshold parameter τ, which excludes background points located beyond the expected object boundary. τ is experimentally set to 300 mm. The algorithm then applies DBSCAN [37] to the remaining points and selects the largest cluster $\mathcal{C}$ based on point density, where a point $p \in \mathcal{P}'$ is defined as the core point as shown in Eq. (7):

$$\mathcal{N}_\epsilon(p) := \{q \in \mathcal{P}' \mid \|p - q\|_2 \leq \epsilon\} \ \wedge \ |\mathcal{N}_\epsilon(p)| \geq k \tag{7}$$

Here, $\epsilon > 0$ is the neighborhood radius, and $k \in \mathbb{N}$ is the minimum number of neighboring points required to form a dense region. Weighted principal component analysis (WPCA) [38] is then applied to $\mathcal{C}$ to estimate the object's orientation, as formulated in Eq. (8). The three eigenvectors obtained from WPCA define the object's roll, pitch, and yaw orientations, forming a right-handed local frame. These vectors are physically valid only when the segmented point cloud has sufficient geometric asymmetry and point-density uniformity. In noisy conditions, random outliers can distort the covariance structure and cause unstable eigenvectors. To reduce this effect, spatial clustering is applied beforehand to remove spurious points and preserve the dominant geometry of the target. Each point p is assigned a weight $w(p)$ using a Gaussian kernel based on its distance to the unweighted centroid μ_0, as defined by Eq. (8):

$$w(p) = \exp\left(-\frac{\|p - \mu_0\|^2}{2\sigma^2}\right) \tag{8}$$

Here, $\sigma > 0$ is a tunable parameter that controls the spatial decay rate of the weighting function. The weighted covariance matrix is computed, and its eigenvectors are used to determine the principal axes. The z-axis is set as the smallest eigenvector and flipped if misaligned with the camera view. The final rotation matrix $\mathbf{R}$ is built via orthogonalization using the estimated orthonormal basis vectors $\vec{x}, \vec{y}, \vec{z}$, and explicitly expressed as Eq. (9):

$$\mathbf{R} = \begin{bmatrix} r_{00} & r_{01} & r_{02} \\ r_{10} & r_{11} & r_{12} \\ r_{20} & r_{21} & r_{22} \end{bmatrix} \tag{9}$$

where r_{ij} are the individual components derived from the basis vectors. This matrix is then converted into a quaternion $\mathbf{q} = (x, y, z, w)$, a compact and singularity-free representation of rotation suitable for robotic pose control.

The complete object recognition docking pipeline, which integrates the YOLO-World model along with Algorithm 2 and Algorithm 3, is illustrated in Fig. 3.

3.4. Manipulator

To evaluate whether a given pose is physically achievable by a 6-DoF manipulator, we first characterize its reachable space in terms of docking feasibility. Fig. 5(a) illustrates the manipulator's kinematic configuration, including its joints and links, which represent the physical structure of the robot. In Fig. 5(b), the reachable region is classified into stable, unstable, and unreachable zones, representing the area where the manipulator can operate effectively. This spatial classification serves as the basis for interpreting the inverse kinematics solutions, which are then used by the system to calculate the required joint configuration for stable docking. The manipulator utilizes this configuration to ensure the end-effector reaches the target position in a stable manner, facilitating a smooth and precise docking process.

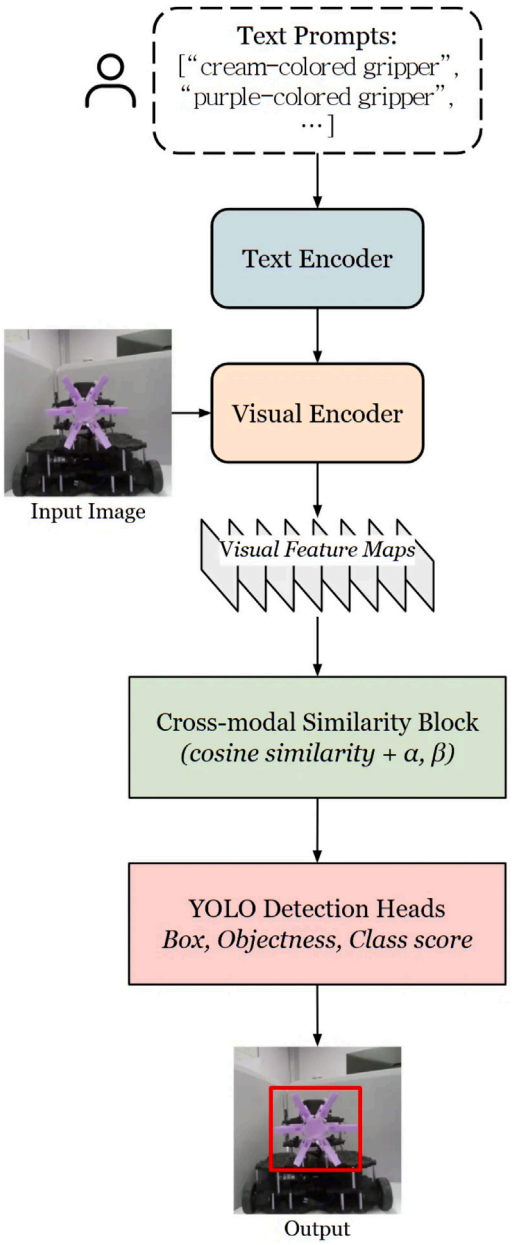


Fig. 4. Inference pipeline of YOLO-World.

Algorithm 3: Estimate orientation from clustered point cloud around the closest point

Input: Closest point $p_c = (\bar{x}, \bar{y}, \bar{z})$
Filtered point cloud $\tilde{\mathcal{P}}$
Cropping threshold τ
Output: Estimated orientation $\mathbf{q}$

Initialize filtered set $\mathcal{P}'$ as empty list;
for *each point* $p \in \tilde{\mathcal{P}}$ **do**
  if $\bar{z} \le p_c.z \le \bar{z} + \tau$ **then**
    Add p to $\mathcal{P}'$;
  end
end
Apply DBSCAN on $\mathcal{P}'$
  Select label $l^* = \arg\max_l \left|\{p \in \mathcal{P}' \mid \ell(p) = l\}\right|$;
  Extract cluster $\mathcal{C} = \{p \in \mathcal{P}' \mid \ell(p) = l^*\}$;
end
Apply WPCA to estimate orientation from $\mathcal{C}$
  Assign weight $w(p)$ to each point $p \in \mathcal{C}$ based on distance to centroid;
  Compute weighted centroid $\mu = \frac{1}{\sum w(p)} \sum w(p) \cdot p$;
  Compute weighted covariance matrix $\Sigma = \frac{1}{\sum w(p)} \sum w(p)(p-\mu)(p-\mu)^\top$;
  Compute eigenvectors $\vec{e}_1, \vec{e}_2, \vec{e}_3$ of Σ sorted by eigenvalues $\lambda_1 \ge \lambda_2 \ge \lambda_3$;
  Let $\vec{z} = \vec{e}_3$;
  if $\vec{z} \cdot \vec{z}_{cam} < 0$ **then**
    $\vec{z} \leftarrow -\vec{z}$;
  end
  $\vec{x} = \vec{e}_2 \times \vec{z}, \quad \vec{y} = \vec{z} \times \vec{x}$;
  Normalize $\vec{x}, \vec{y}, \vec{z}$ and form rotation matrix $\mathbf{R} = [\vec{x}\ \vec{y}\ \vec{z}]$;
end
Convert $\mathbf{R}$ to quaternion $\mathbf{q} = (x, y, z, w)$;
Return $\mathbf{q}$;

4. Experiments

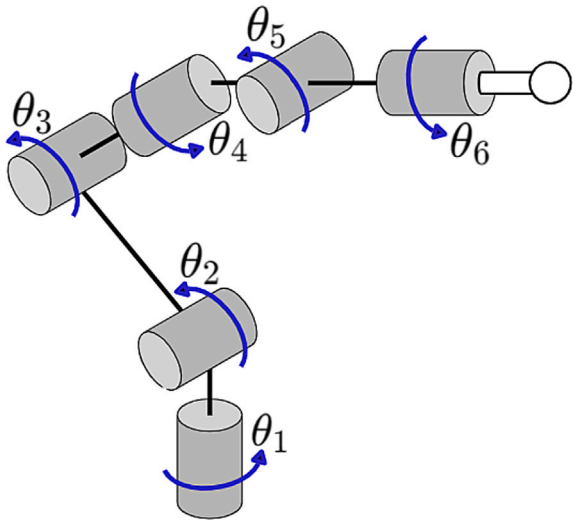
The proposed charging system was tested using a hardware setup consisting of a 6-DoF robotic arm (based on the OpenManipulator-X), a UGV platform (based on TurtleBot), an LDS-02 2D LiDAR, and an Intel RealSense D435i depth camera. Each robot platform is equipped with an NVIDIA Jetson Xavier NX for onboard processing. The manipulator and UGV use Dynamixel actuators, which are controlled by dedicated controllers: the U2D2 interface for the robotic arm, and the OpenCR board for the mobile base. Both the Jetson Xavier NX and the OpenCR are powered by COMS 18650 triple-cell battery packs: a 12 V 3 A (3000 mAh × 3) pack for the Xavier NX, and a 12 V 26 Ah (2600 mAh × 3) pack for the OpenCR. This separation ensures stable power delivery and prevents interference between compute and actuation modules. The object detection model and docking algorithms were executed on an external server equipped with an NVIDIA RTX 3080 GPU. The full hardware setup is illustrated in Fig. 6.

All experiments were conducted in a ROS Noetic environment. ROS (Robot Operating System) is an open-source robotics middleware that provides a standardized communication interface for distributed robot software components. It enables the design of modular systems by defining message-passing protocols for data exchange between processes, known as nodes. Each node typically encapsulates a specific functionality, such as sensor acquisition, motion control, or localization, and communicates with others via topics, services, or actions. ROS also includes a suite of development tools and libraries that facilitate real-time visualization, hardware abstraction, parameter management, and logging. The Noetic version, built on Python 3 and targeting Ubuntu 20.04, is the final long-term support (LTS) release of ROS 1, ensuring compatibility and stability for complex experimental setups.

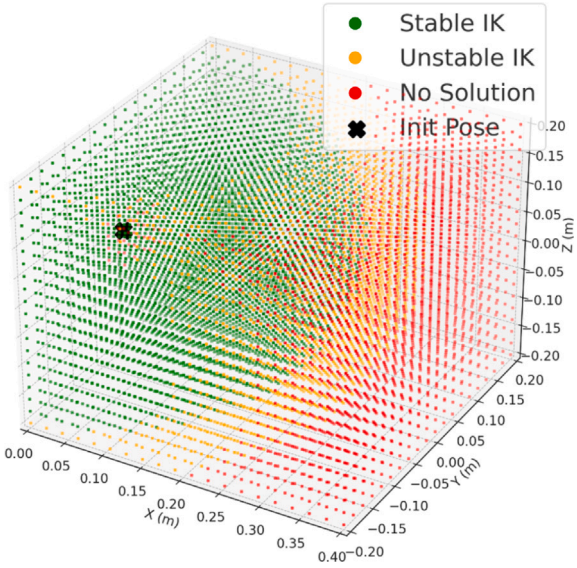
4.1. UGV path planning

As an initial step toward autonomous docking, we assessed the performance of path planning in a real-world indoor environment under standard LED lighting. The experiments were conducted on a map with a size of approximately 245 cm by 180 cm. A 2D SLAM (Simultaneous Localization and Mapping) algorithm was utilized to construct a map of the environment based on LiDAR data. SLAM enables simultaneous estimation of the robot's pose and mapping of its surroundings. The resulting occupancy grid map is presented in Fig. 8(b), while the actual layout of the experimental site is depicted in Fig. 8(a). Path planning was performed on the SLAM-generated map.

For path planning, we adopted a two-stage planning strategy: the A* algorithm was used for global path planning to determine a collision-free route to the docking area, and the Dynamic Window Approach



(a) 6-DoF manipulator.



(b) Reachable workspace of the 6-DoF manipulator.

Fig. 5. The reachable workspace (b) of the manipulator (a), where green points indicate stable configurations, orange points indicate unstable solutions, and red points denote unreachable targets. The init pose refers to the initial pose of the manipulator's end-effector.

(DWA) served as the local planner, adjusting velocities in real time to account for dynamic constraints and nearby obstacles. This strategy guided the robot toward a position where the UGV could perform alignment using Algorithm 1, enabling visual detection of the docking target. The shortest feasible trajectory generated by this planning strategy, which robots starting from various initial orientations successfully followed in the real environment, is visualized in Fig. 9. In a series of 20 trials, with 10 trials from each of two different starting directions, the robot successfully reached the alignment position in 100% of the trials, with an average distance of 55.2 ± 1.8 cm from the target and an angular error within 15°. Path planning was performed on the SLAM-generated map, with an average total time of 13.2 ± 3.7 s, indicating consistent real-time feasibility in the tested environment.

Once the robot reached the alignment position, it used vision-based object recognition, as shown in Fig. 7(a), to identify the charging station. Subsequently, it performed fine alignment using Algorithm 1 to ensure accurate docking orientation. The performance of this vision-based alignment is quantitatively evaluated over 20 trials, as summarized in Table 2.

Table 2
Quantitative results of vision-based alignment for manipulator to UGV docking.

Metric	Value
Position Error (cm)	1.7 ± 0.6
Orientation Error (°)	6.1 ± 3.7
Alignment Time (s)	19.3 ± 4.1

Table 3
Comparison of real-time performance among vision-language object detectors on the LVIS `minival` dataset in a zero-shot setting. In this study, the YOLO-World-S model was utilized, and its results were compared against other vision-language detectors of comparable scale. All values are reported based on official measurements on a single NVIDIA V100 GPU without TensorRT optimization.

Model	Backbone	Params	FPS ↑	AP ↑
GLIP-T [30]	Swin-T [39]	232M	0.12	24.9
Grounding DINO-T [31]	Swin-T [39]	230M	1.5	29.3
DetCLIP-T [40]	Swin-T [39]	155M	2.3	34.4
YOLO-World-S [6]	YOLOv8-S [5]	77M	74.1	26.2

4.2. Object recognition

To evaluate the proposed object recognition framework, we conducted a series of real-world experiments focusing on three key modules: (i) object detection using a vision-language detector, (ii) instance-level segmentation via point cloud clustering, and (iii) orientation estimation based on principal component analysis (PCA). The object detection component leverages YOLO-World, a prompt-driven zero-shot detector, to localize the docking target given natural language queries. Following detection, a clustering algorithm is applied to the filtered point cloud in order to isolate the target region. Finally, the 6-DoF pose estimation module combines the 3D localization and PCA-based orientation estimation described in Section 3.

Fig. 7 shows the target position and 6-DoF pose estimated during the real-world experiments. The green region represents the 2D projection of the 3D reachable workspace shown in Fig. 5(b). Fig. 7(a) illustrates the target position used for the UGV-to-manipulator approach, while Fig. 7(b) presents the 6-DoF target pose used by the manipulator to align with the UGV's docking port. Both frames were captured from the actual experimental environment.

The subsequent subsections provide quantitative and qualitative analyses for each module, including prompt robustness of the detector, comparative evaluation of clustering algorithms, and sensitivity analysis of orientation estimation under noise and occlusion.

4.2.1. Object detection

To generate effective textual prompts for zero-shot detection, we queried a language model (GPT-4) with a brief visual context (e.g., "describe the object in the center of the image") to obtain semantically rich phrases. For instance, it suggested the term "cream-colored gripper", which was then directly used as the input to YOLO-World. This process is illustrated in Fig. 11.

To assess the real-time capability of YOLO-World within robotic perception tasks, we conducted a comparative evaluation against leading vision-language detectors, namely GLIP-T, Grounding DINO-T, and DetCLIP-T, under an identical zero-shot evaluation protocol on the LVIS `minival` dataset. In the case of the conventional YOLOv8 detector, it performs strongly on categories seen during training but shows limited generalization to unseen objects. As shown in Table 3, YOLO-World attains markedly higher processing speed (74 FPS on

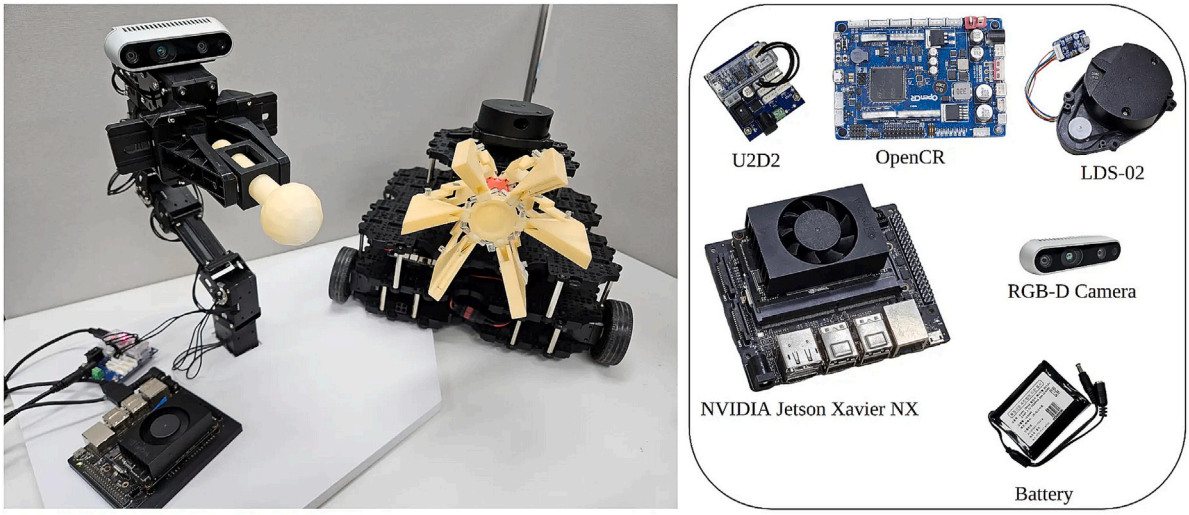


Fig. 6. Experimental hardware setup.

Table 4
Backbone ablation results of YOLO-World evaluated on the LVIS `minival` dataset in a zero-shot setting.

Model	Params	FPS ↑	AP ↑
YOLO-World-S [6]	77M	74.1	26.2
YOLO-World-M [6]	92M	58.1	31.0
YOLO-World-L [6]	110M	52.0	35.4

Table 5
Quantitative performance of clustering algorithms on the Iris dataset. Higher values of ARI, NMI, and Silhouette indicate better clustering quality, while lower runtime reflects higher computational efficiency.

Algorithm	ARI ↑	NMI ↑	Silhouette ↑	Time (ms) ↓
Agglomerative	0.61	0.67	0.44	2.60
DBSCAN	0.57	0.73	0.58	1.92
HDBSCAN	0.54	0.68	0.54	4.34

an NVIDIA V100) while preserving comparable or superior average precision (AP) to transformer-based counterparts. In addition to the cross-model comparison, we also examined the internal architecture of YOLO-World to analyze how the backbone, neck, and head modules respectively influence overall accuracy and speed, as summarized in Table 4. Additionally, the RepVL-PAN neck and the language-aligned detection head were observed to improve multimodal feature fusion and zero-shot prediction stability, respectively, contributing to the overall efficiency of the architecture [6]. These findings indicate that YOLO-World achieves an effective trade-off between open-vocabulary generalization and execution efficiency, demonstrating its practicality for real-time robotic docking applications.

To evaluate robustness across diverse detection environments, YOLO-World was tested using the prompt generated from the pipeline in Fig. 11. The same textual prompt was applied under four representative conditions that commonly degrade visual perception in robotics: standard indoor LED lighting (approximately 620 lux), a low-light environment (approximately 80 lux), partial occlusion caused by objects placed in front of the target, and scenes containing strong light reflections or noise. Illumination values were estimated at the target surface using a calibrated light-sensing module, and all measurements were taken at the same height and position as the object to ensure consistent acquisition. Because the language model captured both chromatic and geometric attributes during prompt generation, a single prompt remained sufficient for all conditions. As illustrated in Fig. 10, YOLO-World maintained correct localization of the target object across all tested scenarios, demonstrating stable detection performance despite substantial variation in environmental appearance.

4.2.2. Clustering algorithms

To evaluate the effectiveness of density-based clustering algorithms that do not require the number of clusters as a prior input, we compared three representative methods: Agglomerative Clustering [41], DBSCAN [37], and HDBSCAN [42]. These algorithms were applied to segment object boundaries from point cloud data. Table 5 summarizes the performance of each algorithm in terms of clustering accuracy (ARI and NMI, both ranging from 0 to 1, where higher values indicate better agreement with ground truth), intra-cluster compactness (Silhouette Score, ranging from −1 to 1, where higher values indicate tighter and more distinct clusters), and computational efficiency (Runtime).

As shown in Fig. 12, DBSCAN demonstrated the most stable and consistent clustering performance. In particular, it preserved well-defined cluster boundaries without over-segmentation or merging, even for objects with complex geometries. Considering both clustering quality and runtime efficiency, DBSCAN offered the best trade-off with minimal parameter tuning among the evaluated methods.

4.2.3. Orientation estimation

To quantitatively estimate object orientation, we conducted experiments using PCA-based algorithms to extract the principal axis from 3D point clouds. Each method computes a direction vector that is subsequently converted into a rotation matrix. The orientation error between the predicted rotation matrix R_{pred} and the ground truth matrix R_{gt} is computed as shown in Eq. (10):

$$\theta = \cos^{-1}\left(\frac{\text{Tr}(R_{\text{gt}}^{\text{T}} R_{\text{pred}}) - 1}{2}\right) \tag{10}$$

Here, $\text{Tr}(\cdot)$ denotes the matrix trace operator, and θ represents the minimal angle required to align the two rotations. The resulting value θ is converted from radians to degrees and subsequently reported. Fig. 13 visualizes the comparison of direction vectors obtained by PCA, WPCA, and the ground truth (GT) from various viewpoints. This includes challenging views as in Fig. 7(b), demonstrating that the 6-DoF pose can be robustly estimated even under perspective variation. The quantitative results are summarized in Table 6. This indicates that WPCA provides more consistent orientation estimation, even in the presence of noise and asymmetric point distributions.

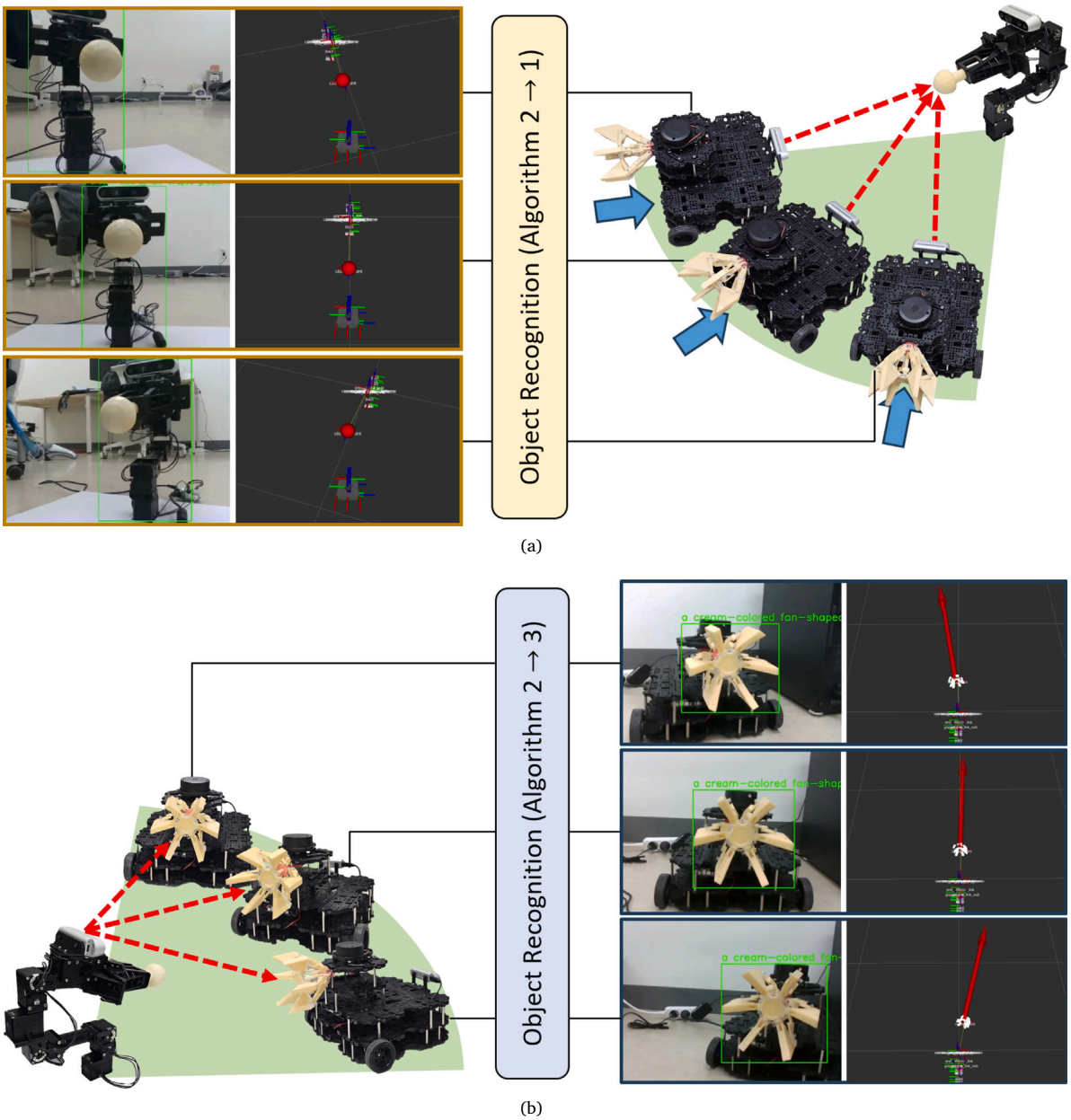


Fig. 7. Estimation results obtained from real experiments: (a) target position for UGV-to-manipulator approach and (b) 6-DoF target pose for manipulator-to-UGV docking. Both images are captured from RViz: the left shows the detection result from YOLO-World, and the right visualizes the goal pose as a TF frame.

Table 6
Average orientation error between predicted and ground truth directions. The lower the error, the better.

Algorithm	Error (°)
PCA	7.87
WPCA	**6.22**

4.3. Manipulator-UGV docking

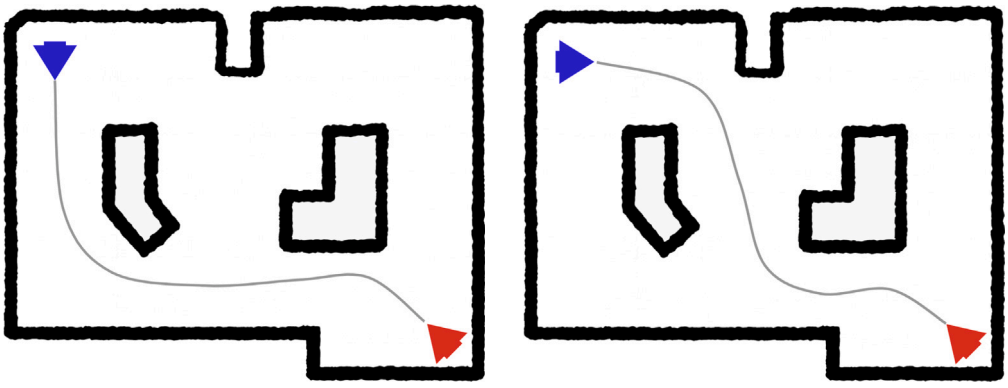
To validate the effectiveness of the proposed docking framework, we conducted a series of experiments using the real-world hardware configuration illustrated in Fig. 14. The experimental setup consists of a 6-DoF manipulator mounted on a fixed platform, while a UGV autonomously performs path planning and drives toward the charging station. The UGV is tasked with recognizing the target, aligning itself with the end-effector, and performing precise docking using both visual and geometric information. The manipulator executes inverse kinematics to reach the 6-DoF pose estimated by the object recognition system, enabling accurate spatial alignment with the incoming UGV.

In each trial, the UGV first executes zero-shot object recognition based on visual prompts, followed by a closed-loop alignment procedure that leverages zero-shot pose feedback to refine its position and orientation. This two-stage process ensures that the final mechanical engagement with the docking port is both accurate and repeatable. The system's performance was quantitatively assessed in terms of docking success rate and completion time, providing a reliable metric for evaluating docking effectiveness.

To assess the generalizability of the approach, the experiment was repeated 20 times under systematically varied initial positions and orientations of the UGV. These variations simulate realistic operational

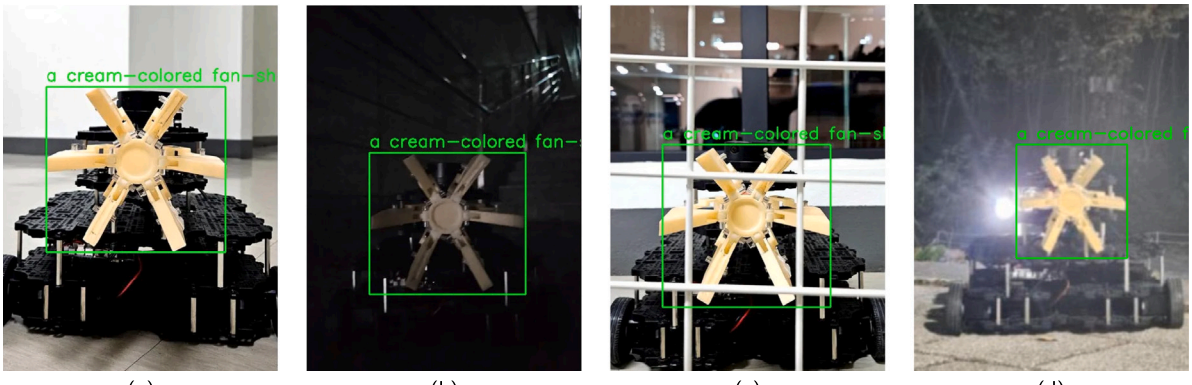
(a) (b)

Fig. 8. Comparison between (a) the real experimental environment and (b) the SLAM-generated map.



(a) (b)

Fig. 9. Effect of the initial orientation on path planning. Depending on the UGV's initial pose, the path planner generates different global paths as shown in (a) and (b). The gray line indicates the generated global path, the blue arrow denotes the UGV's heading direction, and the red arrow represents the manipulator's viewing direction.



(a) (b) (c) (d)

Fig. 10. Detection robustness of YOLO-World using the prompt generated in Fig. 11: ''a cream-colored fan-shaped gripper with six arms in the center of a black robot''. (a)–(d) represent four environmental conditions, indoor LED lighting, low-light environment, partial occlusion, and light reflection/noise with the corresponding detection results shown each input image.

uncertainty in field conditions. The resulting performance metrics, presented in Table 7, indicate that the proposed method consistently achieves successful docking with minimal variability, thereby demonstrating its robustness and adaptability in real-world scenarios. Here, the success rate is defined as the percentage of trials in which the UGV, starting from its initial pose, successfully completed the full docking procedure with the manipulator.

5. Conclusion

This study proposed an autonomous docking framework that integrates open-vocabulary object recognition, geometry-based segmentation, and 6-DoF pose estimation to achieve reliable alignment between a mobile UGV and a stationary manipulator. The method combines

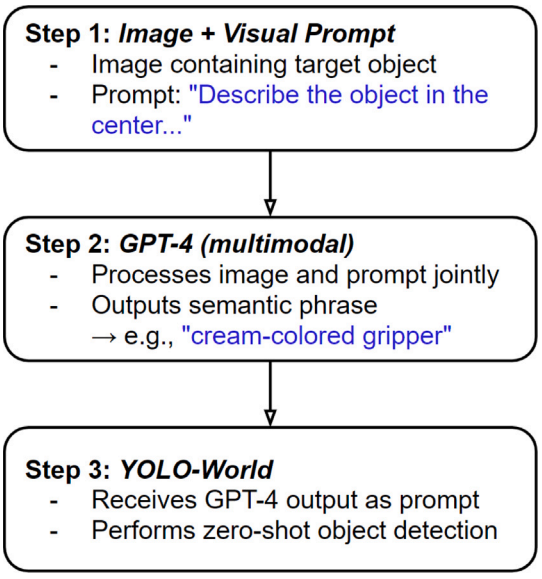


Fig. 11. Prompt generation pipeline using GPT-4 and YOLO-World for zero-shot object detection.

semantic perception using vision-language models with spatial reasoning via clustering and principal component analysis, enabling robust performance in unstructured and partially occluded environments.

Table 7
Execution time and success rate of the docking process. Docking time includes only the manipulator-UGV alignment phase, path planning time covers global and local planning, and total time includes both.

Metric	Value
Path Planning (s)	13.2 ± 3.7
UGV-manipulator alignment (s)	19.3 ± 4.1
Manipulator-UGV docking (s)	1.9 ± 0.2
Total Time (s)	34.4 ± 8.0
Success Rate (%)	90.0

Implemented on real robotic hardware within the ROS Noetic environment, the framework demonstrates the feasibility of precise docking using onboard sensing and computation, with minimal reliance on external resources for visual inference. Experiments with a TurtleBot-based UGV and a 6-DoF manipulator confirmed that the system successfully completes the docking task in an average of 34.4 s, achieving a 90% success rate.

By coupling semantic-driven perception with modular control, the system maintains both flexibility and reliability under real-world operating conditions. These results highlight its potential for applications such as autonomous maintenance, mobile recharging stations, and logistics systems in environments where prior infrastructure cannot be assumed.

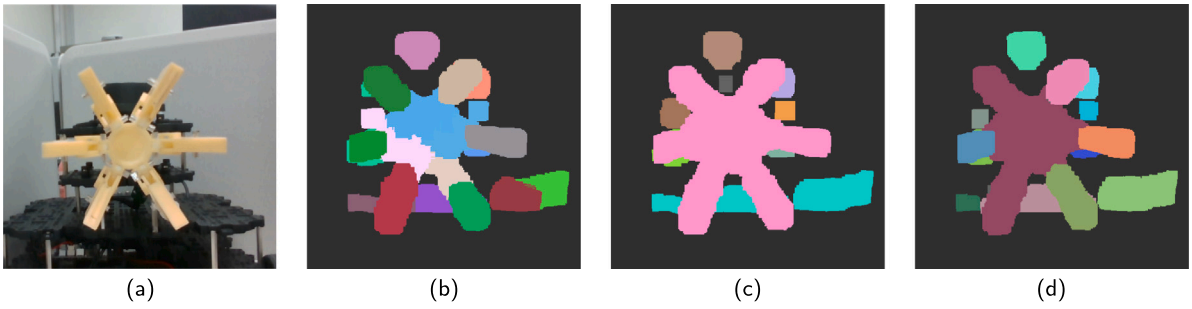


Fig. 12. Comparing clustering results of each algorithm: (a) input image, (b) Agglomerative clustering, (c) DBSCAN, and (d) HDBSCAN.

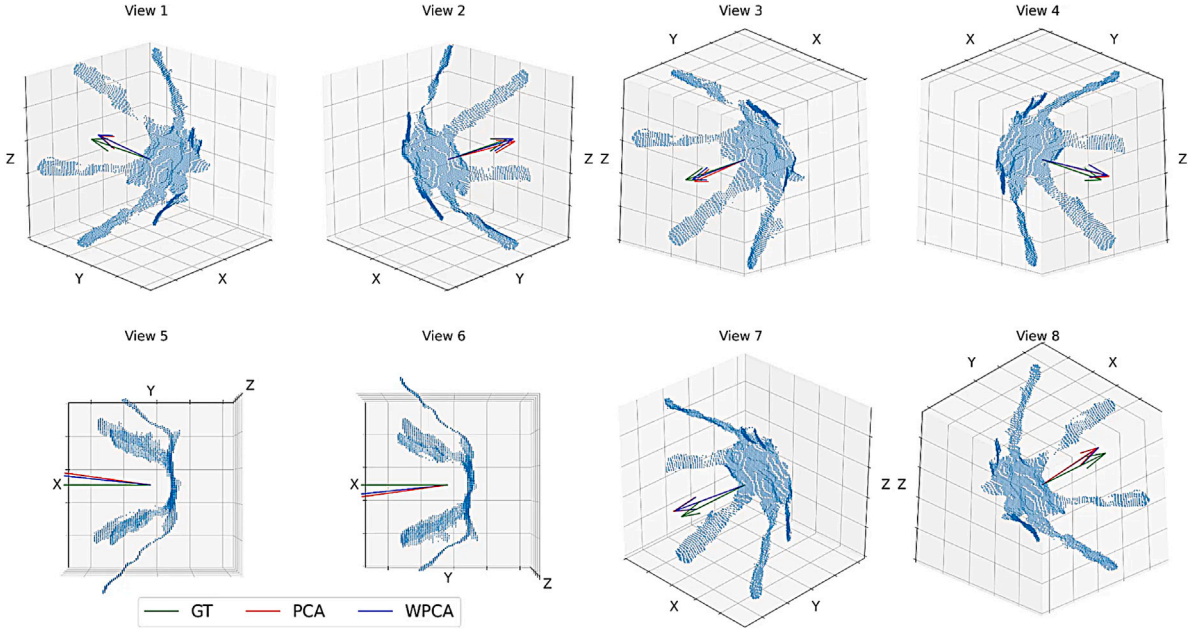


Fig. 13. Comparison of PCA and WPCA orientation vectors with ground truth.

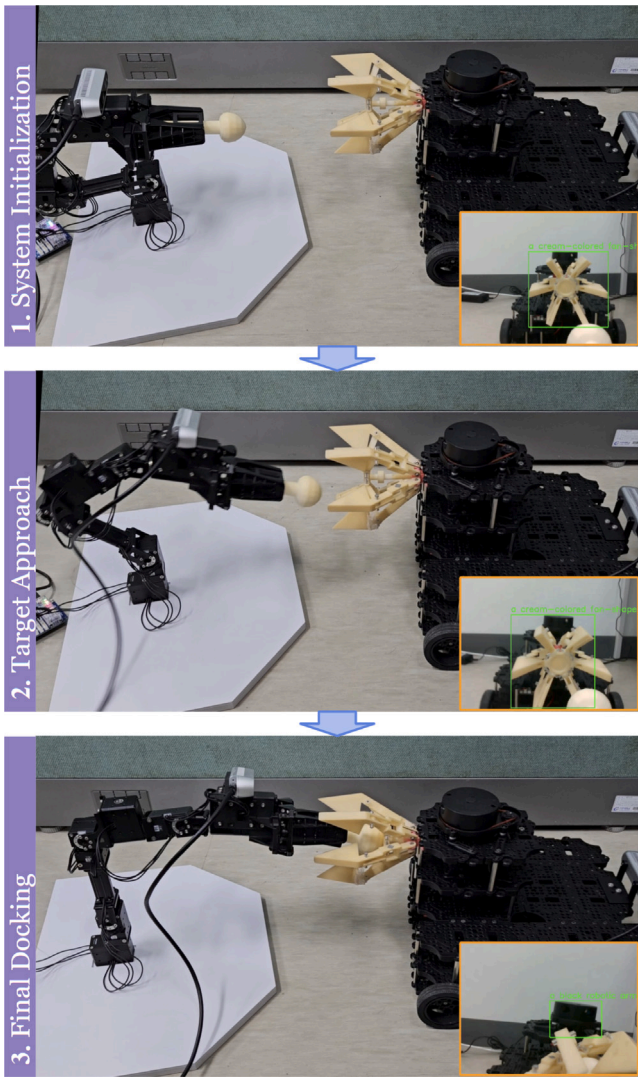


Fig. 14. Overall process of the docking experiment using real robot.

In the future, the system will be extended to support dynamic targets with non-static configurations, enable multi-robot coordination for collaborative docking, and improve alignment precision to support tighter tolerances in mechanical engagement. Furthermore, the docking pipeline will be expanded to handle tool-changing operations, allowing task reconfiguration without manual intervention.

CRediT authorship contribution statement

Minkyu Jung: Writing – original draft, Visualization, Validation, Software, Methodology, Formal analysis, Data curation, Conceptualization. **Andrew Jaeyong Choi:** Writing – review & editing, Resources, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Andrew Jaeyong Choi reports financial support was provided by Gachon University. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported by the Gachon University, Republic of Korea Research Fund in 2024 (GCU-202400520001) and by the Future Challenge Defense Technology Research and Development Program through the Agency for Defense Development (ADD), funded by the Defense Acquisition Program Administration (DAPA), Republic of Korea, in 2025 (No. 915134201).

References

[1] Y. Li, Y. Li, J. Nie, Z. Li, J. Li, J. Gao, Z. Fang, Navigation of the spraying robot in jujube orchard, Alex. Eng. J. 126 (2025) 320–340.
[2] K. Liu, J. Yu, Z. Huang, L. Liu, Y. Shi, Autonomous navigation system for greenhouse tomato picking robots based on laser SLAM, Alex. Eng. J. 100 (2024) 208–219.
[3] M.A. Alrowaily, O. Alruwaili, M. Alghamdi, M. Alshammeri, M. Alahmari, G. Abbas, Application of extreme machine learning for smart agricultural robots to reduce manoeuvering adaptability errors, Alex. Eng. J. 109 (2024) 655–668.
[4] P.J. Besl, N.D. McKay, A method for registration of 3-D shapes, IEEE Trans. Pattern Anal. Mach. Intell. 14 (2) (1992) 239–256.
[5] R. Varghese, M. Sambath, YOLOv8: A novel object detection algorithm with enhanced performance and robustness, in: 2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems, ADICS, 2024, pp. 1–6.
[6] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, Y. Shan, YOLO-world: Real-time open-vocabulary object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2024.
[7] A. Radford, J.W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, P. Sastry, Q.V. Le, M.S. Pittman, P.K. Krishnan, L.K. K., Learning transferable visual models from natural language supervision, in: Proceedings of the 38th International Conference on Machine Learning, Vol. 2021, ICML 2021, 2021, pp. 8748–8763.
[8] E. Narváez, A.A. Ravankar, A. Ravankar, Y. Kobayashi, T. Emaru, Vision based autonomous docking of VTOL UAV using a mobile robot manipulator, in: 2017 IEEE/SICE International Symposium on System Integration, SII, 2017, pp. 157–163.
[9] C. Cheng, X. Li, L. Xie, L. Li, A unmanned aerial vehicle (UAV)/unmanned ground vehicle (UGV) dynamic autonomous docking scheme in GPS-denied environments, Drones 7 (10) (2023) 613.
[10] E. Narváez, A.A. Ravankar, A. Ravankar, T. Emaru, Y. Kobayashi, Autonomous VTOL-UAV docking system for heterogeneous multirobot team, IEEE Trans. Instrum. Meas. 70 (2021) 1–18.
[11] K. Norman, D. Butterfield, J.G. Mangelson, AcTag: Opti-acoustic fiducial markers for underwater localization and mapping, in: 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, 2023, pp. 9955–9962.
[12] Z. Zhang, Y. Hu, G. Yu, J. Dai, DeepTag: A general framework for fiducial marker design and detection, IEEE Trans. Pattern Anal. Mach. Intell. 45 (3) (2023) 2931–2944.
[13] L. Šroba, R. Ravas, J. Grman, A new approach to obtain corner point coordinates in subpixel accuracy, DEStech Trans. Comput. Sci. Eng. (2016).
[14] P. Jiang, R. Unbehauen, Robot visual servoing with iterative learning control, IEEE Trans. Syst. Man, Cybern. - Part A: Syst. Humans 32 (2) (2002) 281–287.
[15] T. Pivoňka, R. Sell, H. Pikner, L. Přeučil, Fiducial marker-based monocular localization for autonomous docking, IFAC-PapersOnLine 56 (2) (2023) 2957–2962, 22nd IFAC World Congress.
[16] X. Bian, W. Chen, D. Ran, Z. Liang, X. Mei, FiMa-Reader: A cost-effective fiducial marker reader system for autonomous mobile robot docking in manufacturing environments, Appl. Sci. 13 (24) (2023) 13079.
[17] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 580–587.
[18] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, in: Advances in Neural Information Processing Systems, vol. 28, 2015.
[19] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2016, pp. 779–788.
[20] R. Ayachi, Y. Said, M. Afif, A. Alshammari, M. Hleili, A. Ben Abdelali, Assessing YOLO models for real-time object detection in urban environments for advanced driver-assistance systems (ADAS), Alex. Eng. J. 123 (2025) 530–549.
[21] Y. Song, Z. Chen, H. Yang, J. Liao, GS-LinYOLOv10: A drone-based model for real-time construction site safety monitoring, Alex. Eng. J. 120 (2025) 62–73.
[22] X. Chi, Z. Guo, F. Cheng, Dynamic obstacle avoidance model of autonomous driving with attention mechanism and temporal residual block, Alex. Eng. J. 105 (2024) 538–548.
[23] A.J. Choi, H.-H. Yang, J.-H. Han, Study on robust aerial docking mechanism with deep learning based drogue detection and docking, Mech. Syst. Signal Process. 154 (2021) 107579.

[24] A.J. Choi, J. Park, J.-H. Han, Automated aerial docking system using onboard vision-based deep learning, J. Aerosp. Inf. Syst. 19 (6) (2022) 421–436.
[25] M. Yang, J. Liu, Research on six-degree-of-freedom refueling robotic arm positioning and docking based on RGB-D visual guidance, Appl. Sci. 14 (11) (2024) 4904.
[26] C. Xu, H. Zhu, H. Zhu, J. Wang, Q. Zhao, Improved RRT∗ algorithm for automatic charging robot obstacle avoidance path planning in complex environments, Comput. Model. Eng. Sci. 137 (3) (2023) 2567–2591.
[27] S. Ren, X. Pan, W. Zhao, B. Nie, B. Han, Dynamic graph transformer for 3D object detection, Knowl.-Based Syst. 259 (2023) 110085.
[28] B. Das, H.A. Dagdogen, M.O. Kaya, R. Das, A novel hybrid model combining vision transformers and graph convolutional networks for monkeypox disease effective diagnosis, Inf. Fusion 117 (2025) 102858.
[29] J. Yin, J. Shen, X. Gao, D.J. Crandall, R. Yang, Graph neural network and spatiotemporal transformer attention for 3D video object detection from point clouds, IEEE Trans. Pattern Anal. Mach. Intell. 45 (8) (2023) 9822–9835.
[30] L.H. Li, P. Zhang, H. Zhang, J. Yang, C. Li, Y. Zhong, L. Wang, L. Yuan, L. Zhang, J. Hwang, K.W. Chang, J. Gao, Grounded language-image pretraining, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2022, pp. 14826–14835.
[31] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, L. Zhang, Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection, in: European Conference on Computer Vision, ECCV, 2024.
[32] S. Chu, M. Lin, D. Li, R. Lin, S. Xiao, Adaptive reward shaping based reinforcement learning for docking control of autonomous underwater vehicles, Ocean Eng. 318 (2025) 120139.
[33] C. Weber, S. Wermter, A. Zochios, Robot docking with neural vision and reinforcement, Knowl.-Based Syst. 17 (2) (2004) 165–172, AI 2003, the Twenty-third SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence.
[34] D. Muse, C. Weber, S. Wermter, Robot docking based on omnidirectional vision and reinforcement learning, Knowl.-Based Syst. 19 (5) (2006) 324–332, AI 2005 SI.
[35] P.E. Hart, N.J. Nilsson, B. Raphael, A formal basis for the heuristic determination of minimum cost paths, IEEE Trans. Syst. Sci. Cybern. 4 (2) (1968) 100–107.
[36] D. Fox, W. Burgard, S. Thrun, The dynamic window approach to collision avoidance, in: IEEE Robotics & Automation Magazine, vol. 4, 1997, pp. 23–33.
[37] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD, 1996, pp. 226–231.
[38] C. Zhang, Y. Shen, Weighted PCA for robust geodetic data analysis, J. Geod. 91 (7) (2017) 809–825.
[39] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin Transformer: Hierarchical vision transformer using shifted windows, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, ICCV, 2021, pp. 9992–10002.
[40] L. Yao, J. Han, X. Liang, D. Xu, W. Zhang, Z. Li, H. Xu, DetCLIPv2: Scalable open-vocabulary object detection pre-training via word-region alignment, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2023, pp. 23497–23506.
[41] J.H. Ward, Hierarchical grouping to optimize an objective function, J. Amer. Statist. Assoc. 58 (301) (1963) 236–244.
[42] R.J.G.B. Campello, D. Moulavi, J. Sander, Density-based clustering based on hierarchical density estimates, in: Proc. PAKDD, 2013, pp. 160–172.