Stable Vision-Based Robot Kinematic Control with Deep Learning-Based Oriented Object Detector

Sitan Li, Chao Liu, Koji Matsuno, Chien Chern Cheah

Stable Vision-Based Robot Kinematic Control with Deep Learning-Based Oriented Object Detector

Sitan Li, Chao Liu, Koji Matsuno and Chien Chern Cheah

***Abstract*— Recent advances in machine learning and deep learning have significantly enhanced robot control by improving object detection and visual feature extraction. However, ensuring theoretical guarantees of stability and convergence in learning-enabled control systems remains a major challenge. In this paper, we propose a vision-based control framework that integrates a deep learning oriented-object detector with a Lyapunov-stable servo control law. The proposed method ensures provably stable convergence of the robot end-effector or its grasped object's pose to a desired camera image region for both eye-in-hand and eye-to-hand configurations. Unlike existing deep learning based visual servoing methods, which either lack formal stability guarantees or ignore object orientation control, our approach incorporates object orientation into the control loop through a region-based method using quaternion representation and formally guarantees stability. We validated our framework on a 6-DoF UR5e manipulator performing cup insertion and centering tasks, demonstrating accurate and stable control in both camera setups.**

Index Terms— Robot control, kinematic control, vision-based control, Lyapunov stability, asymptotic convergence.

I. Introduction

Vision-based robot control has become a cornerstone in modern robotics, enabling robust and flexible interactions across a wide range of visually guided tasks. Vision-based robot control strategies are broadly categorized into joint-space control, where visual information is converted into joint reference signals through inverse kinematics from image space [1], and task-space control methods, which directly regulate end-effector motion in the visual coordinate space [2]. By operating directly in a space aligned with task objectives, task-space control enables intuitive, error-driven regulation and offers significant advantages in emerging applications involving complex environments and calibration or modeling uncertainties [3].

Recent advances in deep learning have significantly improved vision-based robot control by improving object detection, segmentation, and general visual perception [4], [5]. These advances open new possibilities for using high-level semantic visual features in real-time closed-loop control tasks. Despite the latest developments, existing deep learning based visual feedback control approaches either (i) lack theoretical stability guarantees or (ii) ignore full 6-DOF pose control, especially orientation of unknown object. Classical vision-based controllers [6]–[10] can guarantee position or pose control stability even under model uncertainties but rely on simple hand-crafted visual features or reference markers. Recent learning-based approaches [11]–[17] have proposed using deep networks to directly learn visual policies via end-to-end training or imitation learning, bypassing model-based control. However, such methods lack formal guarantees on stability and safety, which are critical in robotic systems. Although orientation control is essential in many manipulation tasks, existing deep learning based visual servoing frameworks rarely incorporate orientation feedback into a closed-loop control system with theoretical guarantees. By using a learned interaction matrix, Tokuda et al. [17] proposed a CNN-based eye-to-hand manipulator control scheme with consideration of orientation, but again stability analysis was not provided. Guo et al. [18] first combined CNN-based visual features with Lyapunov control, but their method handled only object position without orientation and only considered eye-in-hand configuration. This underscores a major gap: the lack of learning-based visual servo control frameworks that achieve full 6-DOF pose alignment, including orientation control, with formal stability guarantees.

In this paper, we bridge this gap by proposing a vision-based control scheme that integrates a deep learning-based oriented object detector with a Lyapunov-stable kinematic controller. Our approach operates on oriented bounding box information without requiring full 3D reconstruction or special markers and provides a Lyapunov stability proof for the closed-loop system. This integration allows for controlling both the position and rotation of an object in the camera field of view (FOV), a capability that prior deep learning-based servo methods did not demonstrate. To our knowledge, this is the first vision-based control approach to achieve provable stability while incorporating orientation feedback from a deep learning-based detector. Furthermore, by handling both eye-in-hand and eye-to-hand scenarios in one framework, we show the method's flexibility comparing to most prior works that can address only one configuration.

We validate the proposed method on a 6-DoF UR5e manipulator performing real-world object insertion and centering tasks. The experimental results demonstrate stable

Manuscript received: August 7, 2025; Revised: November 23, 2025; Accepted: January 3, 2026. This paper was recommended for publication by Editor Cosimo Della Santina upon evaluation of the Associate Editor and Reviewers comments. This work was supported in part by the Ministry of Education, Singapore, through the Academic Research Fund Tier 1, under Project Grant RG65/22, and in part by the Pole Universitaire d'Innovation de Montpellier, University of Montpellier, under the "Companies and Campus" project INTELLISHARE. *Corresponding author: Chao Liu.*

Sitan Li, Koji Matsuno and Chien Chern Cheah are with the School of Electrical and Electronic Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798 (sitan001@e.ntu.edu.sg; KOJI002@e.ntu.edu.sg; ecccheah@ntu.edu.sg).

Chao Liu is with the Department of Robotics, LIRMM, CNRS-University of Montpellier, 34095 Montpellier, France (liu@lirmm.fr).

Digital Object Identifier (DOI): see top of this page.

and accurate pose control performance in both camera configurations. Notably, the proposed method outperforms a classical image-processing baseline and maintains robust performance even in unstructured, real-world scenarios. The main contributions of this work are summarized as follows:

- We develop a unified vision-based robot control framework that integrates a deep learning-based rotated object detector with a rigorously derived Lyapunov-based kinematic visual servoing control law, ensuring provable convergence of the object's image-plane pose (position and orientation) to a desired target. Importantly, this framework is validated in both eye-in-hand and eye-to-hand camera setups, showing its versatility and practical applicability across different visual setups.
- We introduce a quaternion-based orientation error defined over a tolerance region rather than a fixed angle, providing flexibility in achieving orientation alignment. This extends prior region-reaching concepts to the orientation domain, allowing the controller to treat an orientation range as control goal which can be advantageous in tasks with allowable orientation error.
- The method does not require known 3D target pose or a predefined desired image, in contrast to many prior visual servoing approaches. The robot can align the object based solely on the live detector output, which improves practicality in unknown or dynamic settings.

II. Deep Learning-Based Stable Control Method

In vision-based robotic manipulation tasks, visual feedback plays a crucial role in achieving robust control of the end-effector relative to a target object. In this work, a deep learning-based oriented object detector is utilized to detect the target object and provide key information using bounding box of the object of interest. Both eye-in-hand and eye-to-hand configurations are formulated and treated under the same framework considering two typical control tasks: move the grasped object by robot to a desired position with specified bounding box dimension and orientation in the camera FOV for eye-to-hand setup; and control the robot to position the object at a desired position with specified dimension and orientation in the camera FOV for eye-in-hand configuration.

While learning-based object detectors offer high accuracy and generalization capabilities, their outputs may exhibit temporal inconsistencies due to sensor noise or model confidence fluctuations. To ensure reliable integration into control tasks, these uncertainties can be effectively mitigated through a combination of temporal filtering, such as Kalman filtering, robust tracking algorithms like DeepSORT [19] or ByteTrack [20], and post-processing heuristics including bounding box smoothing and confidence thresholding [21]. These methods help stabilize detection outputs over time, reduce chattering effects, and maintain consistency in object localization, which is essential for preventing high-frequency detector noise from propagating to the robot's joint velocity commands and compromising system stability.

An illustration of an oriented bounding box generated by the learning-based object detector in the camera FOV is shown in Fig. 1. The u-v axis of the FOV, the center of the bounding box (u_3, v_3), the middle point (u_1, v_1) of the width w and the middle point (u_2, v_2) of the height h are indicated in Fig. 1.

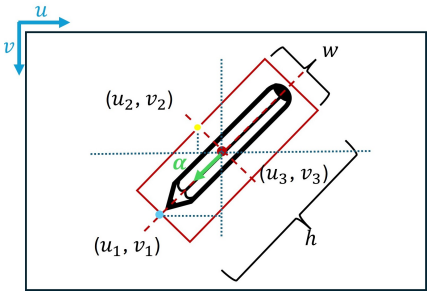

Fig. 1: Rotated object in the camera FOV

Consider a vector in the image plane in the direction from the center point of the bounding box $[u_3, v_3]$ to the point $[u_1, v_1]$ as shown in Fig. 1, a unit vector α representing the orientation of the object in 2D camera frame can be expressed as:

$$\boldsymbol{\alpha} = \begin{bmatrix} \frac{u_1 - u_3}{\sqrt{(u_1-u_3)^2+(v_1-v_3)^2}} \\ \frac{v_1 - v_3}{\sqrt{(u_1-u_3)^2+(v_1-v_3)^2}} \\ 0 \end{bmatrix} \tag{1}$$

By using a rotation matrix $\mathbf{R}$ from the camera frame to robot base frame, its corresponding vector in 3D robot frame is given by:

$$\boldsymbol{\alpha}_{\mathrm{r}} = \mathbf{R}\boldsymbol{\alpha} \tag{2}$$

Let $\boldsymbol{\alpha}_n$ be a unit vector in robot frame that indicates the rotation axis of the robot to orientate the object or camera, $\boldsymbol{\alpha}_r$ is then projected to $\boldsymbol{\alpha}_{[r,p]}$, the component orthogonal to the robot's rotation axis $\boldsymbol{\alpha}_n$, to decouple the control:

$$\boldsymbol{\alpha}_{[\mathrm{r,\ p}]} = \frac{\boldsymbol{\alpha}_{\mathrm{r}} - (\boldsymbol{\alpha}_{\mathrm{r}} \cdot \boldsymbol{\alpha}_n)\, \boldsymbol{\alpha}_n}{\left\| \boldsymbol{\alpha}_{\mathrm{r}} - (\boldsymbol{\alpha}_{\mathrm{r}} \cdot \boldsymbol{\alpha}_n)\, \boldsymbol{\alpha}_n \right\|} \tag{3}$$

Then an intermediate rotation matrix $\mathbf{R}_o$ is obtained from the unit vectors $\boldsymbol{\alpha}_n, \boldsymbol{\alpha}_{[\mathrm{r,\ p}]}$ and $\boldsymbol{\alpha}_n \times \boldsymbol{\alpha}_{[\mathrm{r,\ p}]}$, which serves as a 3D representation of the current visible orientation of the object or camera. For every unique intermediate rotation matrix $\mathbf{R}_o$, there exists a corresponding quaternion $\boldsymbol{q}$ that represents the exact same physical orientation, and thus the role of $\mathbf{R}_o$ is to leverage the numerical superiority of the quaternion $\boldsymbol{q}$ as the final orientation state.

In this work, quaternions are adopted for the orientation representation due to their computational efficiency, numerical stability, and ability to avoid singularities (such as gimbal lock) that plague Euler angle representations, making them ideal for the real-time control loop [22], [23]. Following the quaternion definition, a quaternion vector $\boldsymbol{q} = [q_0, \boldsymbol{q}_v^T]^T$ that represents the orientation of the robot end-effecter is then obtained from the rotation matrix $\mathbf{R}_o$, where q_0 is the scalar part of the quaternion, and $\boldsymbol{q}_v = [q_1, q_2, q_3]^T$ is the vector part [24].

Next, we define a task variable vector $\boldsymbol{\gamma}_{pos}$ for control as:

$$\boldsymbol{\gamma}_{pos} = [u, v, h, w]^T \tag{4}$$

This vector is the set of image-plane features that the robot must control, where u and v specify the bounding box's center coordinate in the FOV, whereas w and h indicate the bounding box's dimensions, specifically its width and height. In this formulation, changes in object scale (h, w) are used as indirect cues for motion along the camera's principal axis, without requiring explicit depth estimation. Equation (4) can be further written as:

$$\boldsymbol{\gamma}_{pos} = \begin{bmatrix} u \\ v \\ h \\ w \end{bmatrix} = \begin{bmatrix} u_3 \\ v_3 \\ 2\sqrt{\Delta u_1{}^2 + \Delta v_1{}^2} \\ 2\sqrt{\Delta u_2{}^2 + \Delta v_2{}^2} \end{bmatrix} \tag{5}$$

where $\Delta u_1 = u_3 - u_1, \Delta v_1 = v_1 - v_3, \Delta u_2 = u_3 - u_2, \Delta v_2 = v_3 - v_2$, $u_1, v_1, u_2, v_2, u_3, v_3$ are the pixel coordinates of the image feature points as shown in Fig 1. Here, Δu_i and Δv_i denote signed pixel differences consistent with the camera coordinate orientation but not absolute values.

Differentiating $\boldsymbol{\gamma}_{pos}$ with respect to time, we have

$$\dot{\boldsymbol{\gamma}}_{pos} = \begin{bmatrix} \dot{u_3} \\ \dot{v_3} \\ \frac{2\Delta u_1 \Delta \dot{u}_1 + 2\Delta v_1 \Delta \dot{v}_1}{\sqrt{\Delta u_1{}^2 + \Delta u_1{}^2}} \\ \frac{2\Delta u_2 \Delta \dot{u}_2 + 2\Delta v_2 \Delta \dot{v}_2}{\sqrt{\Delta u_2{}^2 + \Delta u_2{}^2}} \end{bmatrix} = \mathbf{Q}\dot{\boldsymbol{p}} \tag{6}$$

where

$$\mathbf{Q} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -q_{w1} & q_{w2} & 0 & 0 & q_{w1} & -q_{w2} \\ 0 & 0 & -q_{h1} & -q_{h2} & q_{h1} & q_{h2} \end{bmatrix} \tag{7}$$

and $q_{w1} = \frac{2\Delta u_1}{\sqrt{\Delta u_1{}^2 + \Delta v_1{}^2}}, q_{w2} = \frac{2\Delta v_1}{\sqrt{\Delta u_1{}^2 + \Delta v_1{}^2}}, q_{h1} = \frac{2\Delta u_2}{\sqrt{\Delta u_2{}^2 + \Delta v_2{}^2}}, q_{h2} = \frac{2\Delta v_2}{\sqrt{\Delta u_2{}^2 + \Delta v_2{}^2}}$, $\dot{\boldsymbol{p}} = [\dot{u}_1, \dot{v}_1, \dot{u}_2, \dot{v}_2, \dot{u}_3, \dot{v}_3]^T$.

For velocity $[\dot{u}_i, \dot{v}_i]^T$ $(i = 1, 2, 3)$ of each feature point in the image, it has the following relationship:

$$\begin{bmatrix} \dot{u_i} \\ \dot{v_i} \end{bmatrix} = \mathcal{J}_{ci}\dot{\boldsymbol{r}} \tag{8}$$

where $\dot{\boldsymbol{r}}$ represents the velocity of the robot end-effector in Cartesian space and $\mathcal{J}_{ci}$ is the image Jacobian that maps $\dot{\boldsymbol{r}}$ to the image velocity. The term $\mathcal{J}_{ci}$ is crucial in visual servoing applications as it encapsulates the mapping from the robot's motion in the world frame to the corresponding feature point movement in the image plane.

The relationship between the motion of a robotic manipulator's joints and the resulting end-effector velocity in task space is fundamental to robotic control and motion planning. Specifically, the corresponding Cartesian space linear velocity $\dot{\boldsymbol{r}}$ and angular velocity $\boldsymbol{\omega}$ are functions of the manipulator's joint velocities $\dot{\boldsymbol{\theta}}$. This relationship can be expressed using the manipulator Jacobian matrix, which encapsulates the kinematic mapping from joint space to Cartesian space:

$$\begin{bmatrix} \dot{\boldsymbol{r}} \\ \boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} \mathcal{J}_r(\boldsymbol{\theta}) \\ \mathcal{J}_\omega(\boldsymbol{\theta}) \end{bmatrix} \dot{\boldsymbol{\theta}} \tag{9}$$

where $\boldsymbol{\theta}$ is a vector of joint angles, $\mathcal{J}_r(\boldsymbol{\theta})$ is the translational Jacobian and $\mathcal{J}_\omega(\boldsymbol{\theta})$ is the rotational Jacobian.

Substitute equation (6) and (8) into (9), we have:

$$\begin{bmatrix} \dot{\boldsymbol{\gamma}}_{pos} \\ \boldsymbol{\omega} \end{bmatrix} = \mathcal{J}\dot{\boldsymbol{\theta}} \tag{10}$$

where

$$\mathcal{J} = \begin{bmatrix} \mathbf{Q}\mathcal{J}_C & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathcal{J}_r(\boldsymbol{\theta}) \\ \mathcal{J}_w(\boldsymbol{\theta}) \end{bmatrix} \tag{11}$$

and $\mathcal{J}_C = [\mathcal{J}_{c1}^T, \mathcal{J}_{c2}^T, \mathcal{J}_{c3}^T]^T$.

The desired orientation $\boldsymbol{q}_d$ is determined by a desired constant rotation matrix $\mathbf{R}_{od}$, which can be calculated using (2), (3) given a desired constant unit vector $\boldsymbol{\alpha}_d$ and the desired unit vector $\boldsymbol{\alpha}_{nd}$. $\boldsymbol{\alpha}_d$ is a vector representing the desired orientation of the object in image plane and $\boldsymbol{\alpha}_{nd}$ is perpendicular to the image plane. Let the quaternion error $\boldsymbol{e}_q = [\boldsymbol{e}_0, \boldsymbol{e}_v^T]^T = \boldsymbol{q}\boldsymbol{q}_d^*$, which quantifies the angular mismatch between the current orientation q and the desired orientation q_d, and $\boldsymbol{q}_d^* = [q_{0d}, -\boldsymbol{q}_{vd}^T]^T$, $\boldsymbol{e}_0$ and q_{0d} are the scalar parts of $\boldsymbol{e}_q$ and $\boldsymbol{q}_d^*$ respectively, $\boldsymbol{e}_v$ and $\boldsymbol{q}_{vd}$ are the vector parts of $\boldsymbol{e}_q$ and $\boldsymbol{q}_d^*$ respectively.

The derivative of the quaternion error is given as [25]:

$$\begin{bmatrix} \dot{\boldsymbol{e}}_0 \\ \dot{\boldsymbol{e}}_v \end{bmatrix} = \frac{1}{2} \begin{bmatrix} -\boldsymbol{e}_v^T \\ e_0\mathbf{I} - \boldsymbol{e}_v^\times \end{bmatrix} \boldsymbol{\omega} \tag{12}$$

where $\boldsymbol{e}_v = [e_{v_1}, e_{v_2}, e_{v_3}]^T$ and

$$\boldsymbol{e}_v^\times = \begin{bmatrix} 0 & -e_{v_3} & e_{v_2} \\ e_{v_3} & 0 & -e_{v_1} \\ -e_{v_2} & e_{v_1} & 0 \end{bmatrix} \tag{13}$$

The objective functions for the control task are formulated as described below:

$$\begin{aligned} f_u(\Delta\boldsymbol{\gamma}_{pos_1}) &= (u - u_d)^2 - b_u^2 \leq 0 \\ f_v(\Delta\boldsymbol{\gamma}_{pos_2}) &= (v - v_d)^2 - b_v^2 \leq 0 \\ f_h(\Delta\boldsymbol{\gamma}_{pos_3}) &= (h - h_d)^2 - b_h^2 \leq 0 \\ f_w(\Delta\boldsymbol{\gamma}_{pos_4}) &= (w - w_d)^2 - b_w^2 \leq 0 \\ f_o(\boldsymbol{e}_v) &= \|\boldsymbol{e}_v\|^2 - b_{e_v}^2 \leq 0 \end{aligned} \tag{14}$$

where u_d and v_d are the constant horizontal and vertical coordinates of the desired bounding box center, h_d and w_d are the desired constant height and width of the bounding box, b_u, b_v, b_h, b_w and b_{e_v} are the thresholds. The functions f_u and f_v represent the desired region as a rectangular area centered at (u_d, v_d), with dimensions $2b_u$ and $2b_v$ for its width and height respectively. If $b_u = b_v = 0$, this region collapses into a single point located at (u_d, v_d). Likewise, the desired range for h is specified using f_h, while f_w defines the desired range for w, f_o determines the desired range for the quaternion error. The parameters u_d, v_d, h_d, and w_d are predetermined values that can be customized by the user as required.

A potential energy P_{pos} is designed to provide a flexible and practical method for controlling the object within the camera frame. This function allows precise adjustment to ensure the object is displayed at the desired location and with the appropriate dimensions or scales, enabling better

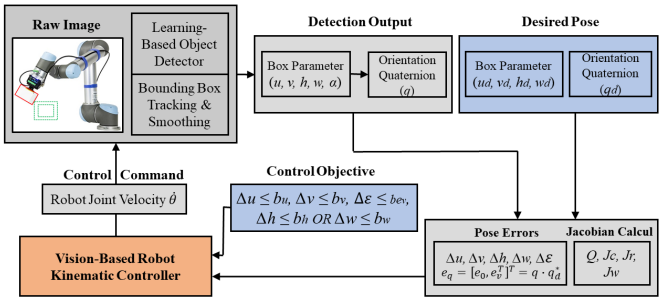

Fig. 2: Overall control diagram of the robot control system

visualization and alignment. The potential energy, denoted as P_{pos}, is expressed as follows:

$$P_{pos}(\Delta\boldsymbol{\gamma}_{pos}) = \tfrac{1}{6}k_u\left[max(0, f_u(\Delta\gamma_{pos_1}))\right]^3 + \tfrac{1}{6}k_v\left[max(0, f_v(\Delta\gamma_{pos_2}))\right]^3 + \tfrac{1}{6}k_h\left[max(0, f_h(\Delta\gamma_{pos_3}))\right]^3 \cdot k_w\left[max(0, f_w(\Delta\gamma_{pos_4}))\right]^3 \tag{15}$$

where k_u, k_v, k_h and k_w are positive constants. The potential energy function reduces to zero when:

$$\begin{aligned} &u_d - b_u \le u \le u_d + b_u \\ &v_d - b_u \le v \le v_d + b_v \\ &w_d - b_w \le w \le w_d + b_w \text{ or } h_d - b_h \le h \le h_d + b_h \end{aligned} \tag{16}$$

The gradient of potential function $P_{pos}(\Delta\boldsymbol{\gamma}_{pos})$ is represented by the variable $\Delta\boldsymbol{\varepsilon}$ and is defined as follows, which serves as the error gradient driving the robot's motion toward the desired target region.

$$\Delta\boldsymbol{\varepsilon} = \left[\frac{\partial P_{pos}(\Delta\boldsymbol{\gamma}_{pos})}{\partial\Delta\boldsymbol{\gamma}_{pos}}\right]^T = \begin{bmatrix}\Delta\varepsilon_1 & \Delta\varepsilon_2 & \Delta\varepsilon_3 & \Delta\varepsilon_4\end{bmatrix}^T = \left[\frac{\partial P_{pos}(\Delta\boldsymbol{\gamma}_{pos})}{\partial\Delta\gamma_{pos_1}} \;\; \frac{\partial P_{pos}(\Delta\boldsymbol{\gamma}_{pos})}{\partial\Delta\gamma_{pos_2}} \;\; \frac{\partial P_{pos}(\Delta\boldsymbol{\gamma}_{pos})}{\partial\Delta\gamma_{pos_3}} \;\; \frac{\partial P_{pos}(\Delta\boldsymbol{\gamma}_{pos})}{\partial\Delta\gamma_{pos_4}}\right]^T \tag{17}$$

where

$$\frac{\partial P_{pos}(\Delta\boldsymbol{\gamma}_{pos})}{\partial\Delta\gamma_{pos_1}} = k_u\left[\max(0, f_u(\Delta\gamma_{pos_1}))\right]^2(u - u_d) \tag{18}$$

$$\frac{\partial P_{pos}(\Delta\boldsymbol{\gamma}_{pos})}{\partial\Delta\gamma_{pos_2}} = k_v\left[\max(0, f_v(\Delta\gamma_{pos_2}))\right]^2(v - v_d) \tag{19}$$

$$\frac{\partial P_{pos}(\Delta\boldsymbol{\gamma}_{pos})}{\partial\Delta\gamma_{pos_3}} = k_h\left[\max(0, f_h(\Delta\gamma_{pos_3}))\right]^2(h - h_d) \cdot k_w\left[\max(0, f_w(\Delta\gamma_{pos_4}))\right]^3 \tag{20}$$

$$\frac{\partial P_{pos}(\Delta\boldsymbol{\gamma}_{pos})}{\partial\Delta\gamma_{pos_4}} = k_w\left[\max(0, f_w(\Delta\gamma_{pos_4}))\right]^2(w - w_d) \cdot k_h\left[\max(0, f_h(\Delta\gamma_{pos_3}))\right]^3 \tag{21}$$

Building upon the above development, we now propose the control input for the robot joint velocity as:

$$\dot{\boldsymbol{\theta}} = -k\begin{bmatrix}\mathcal{J}_r(\boldsymbol{\theta}) \\ \mathcal{J}_w(\boldsymbol{\theta})\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{Q}\mathcal{J}_C & 0 \\ 0 & 1\end{bmatrix}^T\begin{bmatrix}\Delta\boldsymbol{\varepsilon} \\ k_o(max(0, f_o(\boldsymbol{e}_v)))^2\boldsymbol{e}_v\end{bmatrix} \tag{22}$$

where k and k_o are positive constants. The overall structure of the control diagram with the proposed controller is shown in Fig. 2.

A Lyapunov-like function candidate V is defined as follows:

$$V = P_{pos}(\Delta\boldsymbol{\gamma}_{pos}) + (1 - \boldsymbol{e}_0)^2 + \boldsymbol{e}_v^T\boldsymbol{e}_v \tag{23}$$

Differentiating eqn. (23) with respect to time and substituting (12) and eqn. (17) into it, we obtain

$$\begin{aligned}\dot{V} &= \frac{\partial P_{pos}(\Delta\boldsymbol{\gamma}_{pos})}{\partial\Delta\boldsymbol{\gamma}_{pos}}\dot{\boldsymbol{\gamma}}_{pos} - 2\dot{\boldsymbol{e}}_0 + 2\boldsymbol{e}_0\dot{\boldsymbol{e}}_0 + 2\boldsymbol{e}_v^T\dot{\boldsymbol{e}}_v \\ &= \Delta\varepsilon^T\dot{\boldsymbol{\gamma}}_{pos} + \boldsymbol{e}_v^T\boldsymbol{w}\end{aligned} \tag{24}$$

Substituting (22) into (10), we have:

$$\begin{bmatrix}\dot{\boldsymbol{\gamma}}_{pos} \\ \boldsymbol{\omega}\end{bmatrix} = -k\begin{bmatrix}\mathbf{Q}\mathcal{J}_C(\mathbf{Q}\mathcal{J}_C)^T & 0 \\ 0 & I\end{bmatrix}\begin{bmatrix}\Delta\varepsilon \\ k_o(max(0, f_o(\boldsymbol{e}_v)))^2\boldsymbol{e}_v\end{bmatrix} \tag{25}$$

From (24) and (25), we have:

$$\begin{aligned}\dot{V} &= \begin{bmatrix}\Delta\varepsilon^T & \boldsymbol{e}_v^T\end{bmatrix}\begin{bmatrix}\dot{\boldsymbol{\gamma}}_{pos} \\ \boldsymbol{\omega}\end{bmatrix} \\ &= -k\Delta\boldsymbol{\varepsilon}^T\mathbf{Q}\mathcal{J}_C(\mathbf{Q}\mathcal{J}_C)^T\Delta\boldsymbol{\varepsilon} - k\cdot k_o(max(0, f_o(\boldsymbol{e}_v)))^2\boldsymbol{e}_v^T\boldsymbol{e}_v \le 0\end{aligned} \tag{26}$$

Theorem: Considering the kinematic system described in (10), with the joint velocity control input defined in (17) and (22), we have $\max(0, f_u(\Delta\boldsymbol{\gamma}_{pos_1})) \to 0$, $\max(0, f_v(\Delta\boldsymbol{\gamma}_{pos_2})) \to 0$, $\max(0, f_o(\boldsymbol{e}_v)) \to 0$, and $\max(0, f_h(\Delta\boldsymbol{\gamma}_{pos_3})) \to 0$ or $\max(0, f_w(\Delta\boldsymbol{\gamma}_{pos_4})) \to 0$ when $t \to \infty$.

Proof: Since $V \ge 0$ and $\dot{V} \le 0$, V is bounded, which ensures the boundedness of P_{pos} and $\boldsymbol{e}_v$. Therefore, from (14) and (17)-(21), $\boldsymbol{\gamma}_{pos}$ and $\Delta\boldsymbol{\varepsilon}$ are also bounded. $\mathbf{Q}$ in (7) is a function of the object dimensions in image space and hence it is bounded, and the image Jacobian $\mathcal{J}_C$ is a function of the camera's intrinsic parameters and extrinsic parameters, which are fixed and finite and thus the image Jacobian $\mathcal{J}_C$ is also bounded. Therefore, $(\mathbf{Q}\mathcal{J}_C)^T\Delta\boldsymbol{\varepsilon}$ is bounded. From (26), it can be concluded that $(\mathbf{Q}\mathcal{J}_C)^T\Delta\boldsymbol{\varepsilon} \in L_2(0, +\infty)$. Since the manipulator Jacobian contains only robot and object physical parameters and sin, cos functions of the robot's joint angles, the Jacobian $\mathcal{J}_r$ and $\mathcal{J}_\omega$ remains bounded over the robot's workspace. Therefore $\mathcal{J}$ in (10) are bounded. As $(\mathbf{Q}\mathcal{J}_C)^T\Delta\boldsymbol{\varepsilon} \in L_2(0, +\infty)$ and $\mathcal{J}$ are bounded, $\dot{\boldsymbol{\theta}}$ in (22) is therefore bounded in a finite task space where the manipulator Jacobian is non-singular and it can be hence inferred that $\dot{\boldsymbol{p}}$, $\dot{\boldsymbol{r}}$ and $\boldsymbol{\omega}$ are bounded. The boundness of $\boldsymbol{\omega}$ ensures the boundness of $\dot{\boldsymbol{e}}_v$, the boundness of $\dot{\boldsymbol{p}}$ ensures the boundness of $\dot{\mathbf{Q}}$ and $\Delta\dot{\boldsymbol{\varepsilon}}$, and the boundness of $\dot{\boldsymbol{r}}$ ensures the boundness of $\dot{\mathcal{J}}_C$, which ensures the boundness of the derivative of $(\mathbf{Q}\mathcal{J}_C)^T\Delta\boldsymbol{\varepsilon}$. Therefore, $(\mathbf{Q}\mathcal{J}_C)^T\Delta\boldsymbol{\varepsilon}$ and $k_o(max(0, f_o(\boldsymbol{e}_v)))^2\boldsymbol{e}_v$ are uniformly continuous. According to the Lemma C1 in its Appendix C from [26], $(\mathbf{Q}\mathcal{J}_C)^T\Delta\boldsymbol{\varepsilon} \to 0$, $k_o(max(0, f_o(\boldsymbol{e}_v)))^2\boldsymbol{e}_v \to 0$ as $t \to \infty$. Hence $max(0, f_o(\boldsymbol{e}_v)) \to 0$.

Next, $(\mathbf{Q}\mathcal{J}_C)^T\Delta\boldsymbol{\varepsilon}$ can be expressed as:

$$(\mathbf{Q}\mathcal{J}_C)^T\Delta\boldsymbol{\varepsilon} = \boldsymbol{j}_{I1}\Delta\varepsilon_1 + \boldsymbol{j}_{I2}\Delta\varepsilon_2 + \boldsymbol{j}_{I3}\Delta\varepsilon_3 + \boldsymbol{j}_{I4}\Delta\varepsilon_4 \tag{27}$$

where $\boldsymbol{j}_{Ii}, i = 1, 2, 3, 4$ are the column vectors of $(\mathbf{Q}\mathcal{J}_C)^T$. Substitute (20) and (21) into (27), we have:

$$(\mathbf{Q}\mathcal{J}_C)^T \Delta\boldsymbol{\varepsilon} = \mathcal{J}_I \begin{bmatrix} \Delta\varepsilon_1 \\ \Delta\varepsilon_2 \\ \Delta\bar{\varepsilon}_3 \end{bmatrix} \tag{28}$$

where $\mathcal{J}_I = [\boldsymbol{j}_{I1}\ \boldsymbol{j}_{I2}\ \bar{\boldsymbol{j}}_{I3}]$, and

$$\begin{aligned} \bar{\boldsymbol{j}}_{I3} = \boldsymbol{j}_{I3} \cdot (h - h_d) \cdot \max(0, f_w(\Delta\boldsymbol{\gamma}_{pos_4})) \\ + \boldsymbol{j}_{I4} \cdot (w - w_d) \cdot \max(0, f_h(\Delta\boldsymbol{\gamma}_{pos_3})) \end{aligned} \tag{29}$$

$$\Delta\bar{\varepsilon}_3 = k_w \left[\max(0, f_w(\Delta\boldsymbol{\gamma}_{pos_4}))\right]^2 \cdot k_h \left[\max(0, f_h(\Delta\boldsymbol{\gamma}_{pos_3}))\right]^2 \tag{30}$$

If matrix $\mathcal{J}_I$ is full rank, i.e. $\boldsymbol{j}_{I1}, \boldsymbol{j}_{I2}$ and $\bar{\boldsymbol{j}}_{I3}$ are linearly independent, then $(\mathbf{Q}\mathcal{J}_C)^T\Delta\varepsilon \to 0$ implies that $\Delta\varepsilon_1 \to 0$, $\Delta\varepsilon_2 \to 0$, $\Delta\bar{\varepsilon}_3 \to 0$, which means $\max(0, f_u(\Delta\boldsymbol{\gamma}_{pos_1})) \to 0$ and $\max(0, f_v(\Delta\boldsymbol{\gamma}_{pos_2})) \to 0$, $\max(0, f_h(\Delta\boldsymbol{\gamma}_{pos_3})) \to 0$ or $\max(0, f_w(\Delta\boldsymbol{\gamma}_{pos_4})) \to 0$. If both $\max(0, f_h(\Delta\boldsymbol{\gamma}_{pos_3})) = \max(0, f_w(\Delta\boldsymbol{\gamma}_{pos_4})) = 0$, then $\bar{\boldsymbol{j}}_{I3} = \Delta\bar{\varepsilon}_3 = 0$, and therefore $\Delta\varepsilon_1 = \Delta\varepsilon_2 = 0$ since $\boldsymbol{j}_{I1}$ and $\boldsymbol{j}_{I2}$ are linearly independent. □

Remark: The full-rank assumption for $\mathcal{J}_I$ holds provided that the object is not positioned in a visual singular configuration, such as when the chosen image features (the bounding box points) are collinear or poorly distributed with respect to the camera's optical axis, and that the robot's end-effector is not commanded to a position near a kinematic singularity where the robot's Jacobian becomes rank-deficient. These conditions ensure a non-singular transformation for the control input.

III. Experiments

A series of experiments were designed to demonstrate the performance of the proposed vision-based control approach for both eye-in-hand and eye-to-hand configurations. The different configurations were considered to verify the versatility and robustness of the proposed method. The experiments were conducted using an industrial collaborative robot UR5e with 6 degrees of freedom. The robot's workspace for all experiments was constrained to a continuous, non-singular region of the UR5e's physical limits, ensuring the manipulator Jacobian remained non-singular throughout the task duration. The camera used is the Intel D435I camera with a frame rate at 30 FPS and a frame resolution of 640*480. YOLOv11-OBB (You Only Look Once with Oriented Bounding Boxes) was utilized in the experiment for the object detection. The pre-trained model based on the DOTAv1 dataset was then further trained for 300 epochs on a dataset of 1,000 manually collected and labelled images, divided into training, validation and test sets in a 2:1:1 ratio. The bounding box parameters generated by the trained model were smoothened using ByteTrack [20] online and then used in the controller. The experimental background is designed with cluttered visual scenes that closely reflect realistic manipulation environments as seen in Fig 3. The YOLOv11-OBB generates the detection result for each captured camera image with 13 ms of processing time on our experiment computer, which is sufficient to keep up with the camera and control command calculation.

A. Eye-to-Hand Configuration

In the eye-to-hand configuration, a fixed camera was used to observe the object in a finite workspace. The goal is to move the object grasped by the robot manipulator to a specific position with specific orientation in the image frame to accomplish object manipulation tasks requiring spatial alignment.

Two tasks have been designed. The first task involved inserting one paper cup into another, which required fine control of both three-dimensional position and orientation to ensure successful insertion.

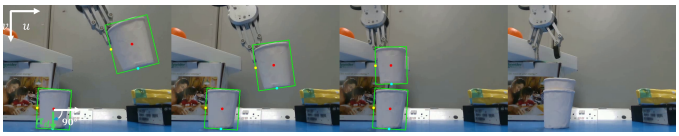

Fig. 3: Image frame sequences in the eye-to-hand configuration based cup insertion task.

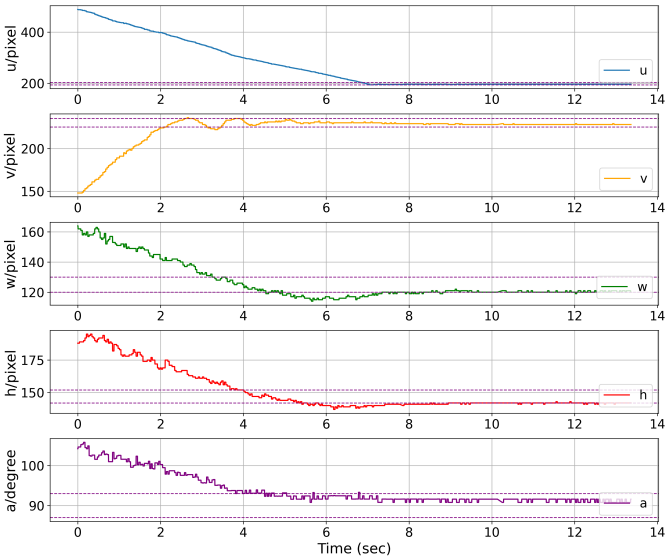

Fig. 4: Plots of task variables, u, v, h, w, α over time for eye-to-hand cup insertion task

In this experiment, the robot end-effector held a paper cup, while another cup was placed on the table as shown in the first frame of Fig 3. The stationary cup was initially positioned behind the grasped cup to demonstrate the feasibility of 3D object manipulation using a single camera with the proposed method. In this experiment, the bounding box parameters of the static cup on the table were given by the detection algorithm as (u_s, v_s, h_s, w_s) to realize the cup placement task. The desired center coordinate of the grasped cup should be positioned directly above the desired center coordinate of the static cup as $(u_d, v_d) = (u_s, v_s - 250)$. The desired width and height of the grasped cup should be the same as the static cup as $(h_d, w_d) = (h_s, w_s)$, ensuring proper alignment for the grasped cup to be placed into the static one. The desired bounding box parameters (u_d, v_d, h_d, w_d)

TABLE I: Summary of Experiment Parameters

Experiment	**Desired Bounding Box Parameters** $[u_d, v_d, h_d, w_d]$	**Tolerance Region** $[b_u, b_v, b_h, b_w]$	**Desired angle** α_d	**Tolerance Region** b_α	**Gains** $[k_u, k_v, k_h, k_w, k_o, k]$
Insertion Task Eye-to-hand	[198, 230, 147, 125]	[5, 5, 10, 10]	90	5	$[5\times10^{-3}, 5\times10^{-3}, 10^{-6}, 10^{-6}, 10^{3}, 1]$
Centering Task Eye-to-hand	[320, 240, 300, 220]	[5, 5, 10, 10]	90	5	
Centering Task Eye-in-hand	[320,240, 440, 280]	[5, 5, 10, 10]	90	5	

are determined as shown in the first row of Table I. The desired vector $\boldsymbol{\alpha}_d$ that decides the desired quaternion $\boldsymbol{q}_d$ was parallel to the v axis of the camera FOV, as indicated in the first frame of Fig 3. To clearly define the unit vector $\boldsymbol{\alpha}_d$, we consider angle α_d in the image plane to represent the direction of $\boldsymbol{\alpha}_d$. Let u-axis of the image coordinate system represent $\alpha_d = 0$ degrees and α_d increase in the clockwise direction. Thus, an angle of $\alpha_d = 90$ degrees represents the desired unit vector $\boldsymbol{\alpha}_d$ in Fig 3. The desired α_d and tolerance region is shown in Table I. Similarly, we use angle α to represent the direction of current unit vector $\boldsymbol{\alpha}$. The gains $[k_u, k_v, k_h, k_w, k_o, k]$ in (22) are set as shown in Table I. To prevent the robot's velocity from reaching its physical limits at the onset of motion, the region error $\Delta\boldsymbol{\varepsilon}$ in velocity command (22) is saturated within the range of [-0.001, 0.001], which is a common technique employed in practical implementations of task-space control. As $\Delta\boldsymbol{\varepsilon}$ in (22) is derived from image coordinates expressed in pixels, it can attain relatively large magnitudes. Consequently, the control gains were carefully tuned to small values to ensure that the resulting joint velocities remain within the prescribed operational limits of the robot.

The plots showing how the task variables changed over time are presented in Fig. 4. It is evident that the coordinate u, v, and the width w and height h of the bounding box center converged within their respective target ranges, ensuring that the target center was properly positioned and aligned above the cup on the table. Additionally, the orientation variable α fell within its desired range, as shown in the last subfigure in Fig. 4, so that the end-effector held the cup in the same orientation as the cup on the table. These results demonstrate that all task variables u, v, h, w, and α achieved convergence within their respective ranges so that the robot end-effector moved the cup directly above the cup on the table as shown in the third frame of Fig 3. The following opening of the end-effector allowed the cup in hand to drop into the one on the table and thus successfully realized the cup placement task.

The second task carried out in the eye-to-hand configuration is to move the cup to the center of the image frame to generate a better view or display of the object. The cup was initially placed near the boundary of the camera FOV as shown in the first frame of Fig 5. The desired bounding box parameters, tolerance region and gains are determined as shown in the second row of Table I. The values of h_d and w_d are chosen to define a desired area within the FOV that can best observe the object. The desired area is constant

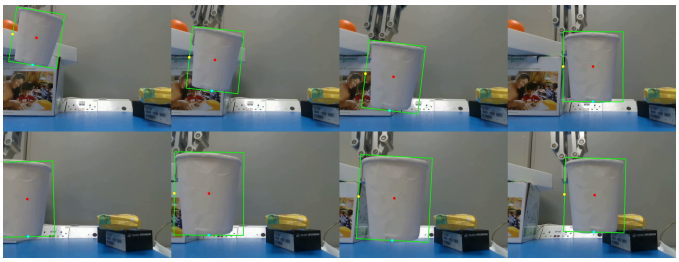

Fig. 5: Image frame sequences in the eye-to-hand configuration based cup centering task.

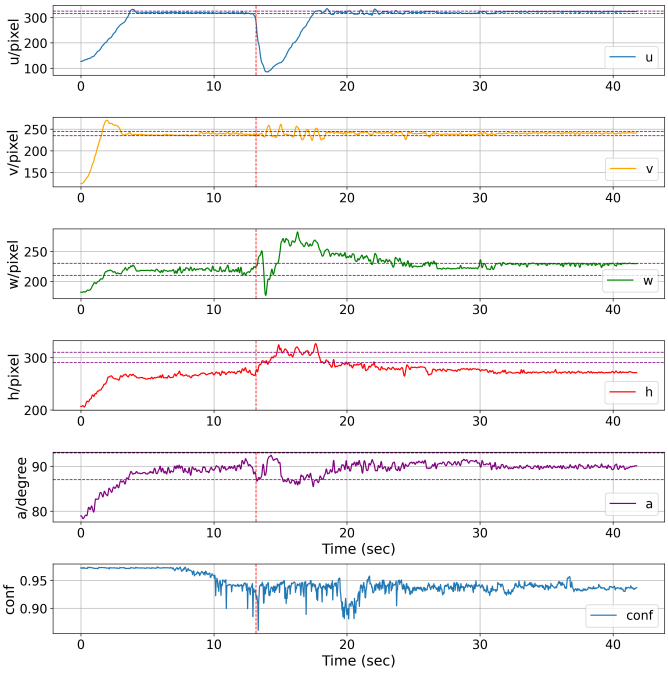

Image frame sequences ...

Fig. 6: Fig. 6: Plots of task variables, u, v, h, w, α, and bounding box confidence level over time for eye-to-hand cup centering task

and determined solely by the size of the FOV, irrespective of the object's size or aspect ratio. The tolerance parameters are accordingly chosen to allow sufficient margins to adjust object orientation as needed.

The plots showing how the task variables changed over time are presented in Fig. 6. The task variables u, v, and α all converged to their respective desired ranges, ensuring that the cup was centered and held upright in the image frame as shown in Fig. 5. Although the task variable w eventually settled beyond its desired range, the controller succeeded to control the other task variable h to fall within its target range,

which is consistent with the Theorem that at least one of h and w is guaranteed to converge to its desired range.

Additionally, as shown in the second row in Fig. 5, after the task variables converged to their desired values, camera position was manually adjusted which caused the cup to move in the image and become partially outside the FOV. The temporary excursion outside the FOV led to a drop in the bounding box's confidence level to 86% as indicated by the red dotted line in the sixth subfigure of Fig 6. Despite these perception interruptions, the proposed controller remained stable and was still able to drive the robot toward the desired target position. The confidence level eventually rises up from 86% to 93.7%, demonstrating that achieving a better view results in a more reliable bounding box and hence more precise object detection. Also, the controller successfully maintained object tracking, with the task variables u, v, w and α converging again to their desired ranges.

B. Eye-in-Hand Configuration

In the eye-in-hand setup, the camera was mounted on the end-effector of the robot. The camera can start from any initial position and orientation, but the object must be inside the FOV. The goal is to control the robot's end-effector to position the target object at the center of the image frame, with the desired orientation and a predefined appearance dimension in the camera FOV. To illustrate the controller's ability to maintain its tracking performance and robustness against variations, the object was also manually repositioned to a different pose when the task variables converged to their desired values.

The desired bounding box parameters, tolerance region and gains are determined as shown in the third row of Table I. The plots showing how the task variables changed over time are presented in Fig. 8. The task variables u, v, w and α all converged to their respective desired ranges, ensuring that the cup was centered in the image frame with the correct orientation as shown in the first row of Fig 7. The cup was then manually repositioned as shown in the second row of Fig. 7. The reposition of cup caused sudden changes of the task variables from the moment indicated by the red dotted line as shown in Fig. 8. Despite the sudden changes in task variables, the controller successfully maintained object tracking, with the task variables u, v, w and α converging again to their desired ranges.

The experimental results demonstrate the versatility and robustness of the proposed vision-based control approach

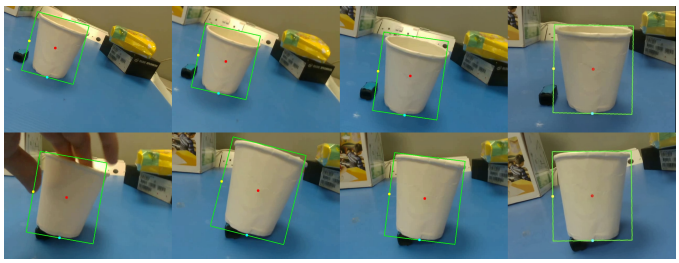

Fig. 7: Image frame sequences in the eye-in-hand configuration based cup centering task.

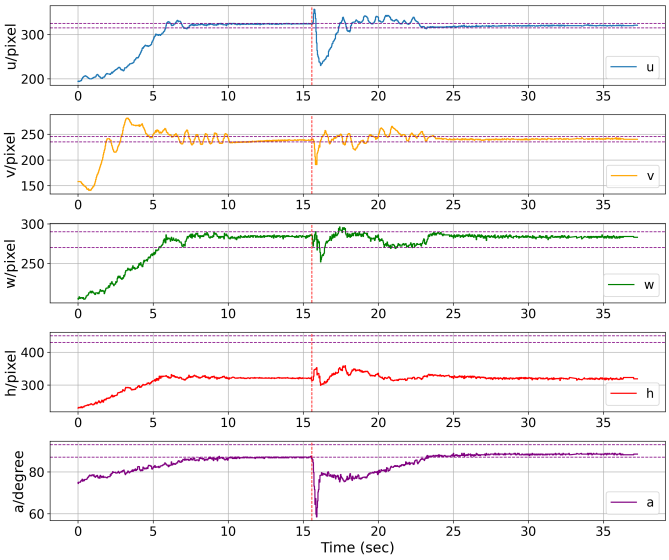

Fig. 8: Plots of task variables, u, v, h, w, α over time for eye-in-hand cup centering task

for both eye-in-hand and eye-to-hand configurations. The algorithm consistently achieved precise control of the task variables, including bounding box position, dimensions, and orientation, within the predefined tolerance ranges. In the eye-to-hand setup, it successfully handled the cup insertion and centering task while improving detection confidence through optimal object positioning. In the eye-in-hand setup, the controller exhibited its accurate tracking performance and robustness against external disturbances. These results highlight the efficiency of the proposed method in achieving precise, real-time control while ensuring stability and adaptability across different configurations, and demonstrate its feasibility for practical robotic applications.

C. Baseline Comparison

To validate our method, we compare it with a classical image-processing baseline on the centering task in Fig. 5. Since traditional segmentation-based approaches do not extract semantic bounding-box parameters, the baseline regulates only the object's image-plane center and orientation, excluding scale.

White regions are extracted via color thresholding, and candidate regions are selected based on area and aspect ratio. The region's centroid and orientation (via a fitted contour line) serve as visual features, implemented using OpenCV and smoothed before control. The baseline performs well under ideal conditions with a uniform black background (Fig. 9(a)), achieving low position error (Fig. 10). However, its accuracy degrades with moderate texture (Fig. 9(b)) and fails in cluttered scenes (Fig. 9(c)), due to its sensitivity to lighting and background variations.

In contrast, the YOLOv11-OBB detector maintains accurate estimates across diverse environments, showing robustness to background clutter and illumination changes. While the baseline is effective only under simplified conditions, our method achieves reliable performance in realistic, unstructured settings.

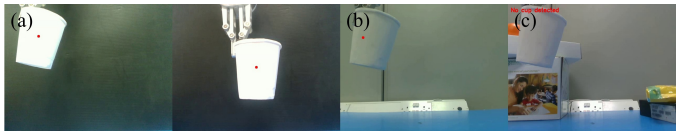

Fig. 9: (a) Image frame sequences in the eye-to-hand configuration based cup centering task (baseline) (b) Large center-offset detection error (c) Fail to detect in cluttered background

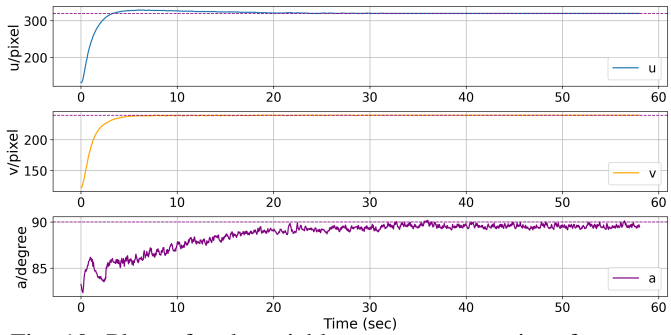

Fig. 10: Plots of task variables, u, v, α over time for eye-to-hand cup centering task (baseline)

IV. Conclusion

In this work, we developed a vision-based robotic control approach that successfully integrates an oriented object detector with a Lyapunov-stable controller. The proposed method enables stable positioning and orientation alignment of a grasped object using only visual input, and operates effectively in both eye-in-hand and eye-to-hand camera configurations. This work bridges the gap between modern deep learning perception and classical control theory in robotics, contributing to more reliable vision-based manipulation. The approach was validated experimentally on a 6-DoF UR5e manipulator performing insertion and centering tasks using a cylindrical cup object. Comparative results with a classical baseline further confirm that the proposed approach outperforms the traditional image-processing method, particularly in complex scenes with background clutter. Furthermore, the region-based nature of our control design guarantees the convergence for the bounding box parameters u, v, e_v, and h or w to their respective desired regions provides the necessary flexibility and robustness to handle highly view-variant objects, which is an important direction for future experimental work. Since the current formulation controls only the in-plane orientation and cannot distinguish visually symmetric configurations, future work will also extend the framework to full 6-DoF pose regulation by incorporating depth or key-point cues.

References

[1] H. H. Fakhry and W. J. Wilson, "Modified resolved acceleration controller for position-based visual servoing," *Math. Comput. Model.*, vol. 24, no. 5/6, pp. 1–9, 1996.

[2] L. Weiss, A. Sanderson and C. Neuman, "Dynamic sensor-based control of robots with visual feedback," *IEEE Journal Rob. Auto.*, vol. 3, no. 5, pp. 404-417, 1987.

[3] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, "Motion control," *Robotics: Modelling, Planning and Control*, Springer, 2009.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pp. 779–788, Las Vegas, Nevada, USA, 2016.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, 2017.

[6] C. C. Cheah, M. Hirano, S. Kawamura, and S. Arimoto, "Approximate Jacobian control for robots with uncertain kinematics and dynamics," *IEEE Trans. Robot. Autom.*, vol. 19, no. 4, pp. 692–702, Aug. 2003.

[7] C. C. Cheah, C. Liu, and J. J. E. Slotine, "Adaptive tracking control for robots with unknown kinematic and dynamic properties," *Int. J. Robot. Res.*, vol. 25, no. 3, pp. 283–296, 2006.

[8] Y. H. Liu, H. Wang, C. Wang, and K. K. Lam, "Uncalibrated visual servoing of robots using a depth-independent interaction matrix," *IEEE Trans. Robot.*, vol. 22, no. 4, pp. 804–817, Aug. 2006.

[9] C. Cai, E. Dean-León, D. Mendoza, N. Somani and A. Knoll, "Uncalibrated 3D stereo image-based dynamic visual servoing for robot manipulators," in Proc. *IEEE/RSJ Int. Conf. Intelligent Robots & Systems*, pp. 63-70, Tokyo, Japan, 2013.

[10] C. C. Cheah, D. Q. Wang, and Y. C. Sun, "Region-reaching control of robots," *IEEE Trans. Robot.*, vol. 23, no. 6, pp. 1260–1264, 2007.

[11] K. Ahlin, B. Joffe, A. Hu, G. McMurray and N. Sadegh, "Autonomous leaf picking using deep learning and visual-servoing," *IFAC PapersOnLine* 49.16, 177-183, 2016.

[12] H. Cheng, J. Xin, Y. M. Yao, Y. M. Zhang and D. Liu, "Deep Learning for Manipulator Visual Positioning," in *IEEE Int. Conf. CYBER Tech. Autom., Control, and Intell. Syst*, pp. 373-378, Tianjin, China, 2018.

[13] V. Harish, "DFVS: Deep flow guided scene agnostic image based visual servoing," *IEEE Int. Conf. Robot. Autom.*, pp. 9000-9006, 2020.

[14] N. Adrian, V. T. Do and Q. C. Pham, "DFBVS: Deep Feature-Based Visual Servo," in *IEEE 18th Int. Conf. Autom. Sci. Eng.*, pp. 1783-1789, Mexico City, Mexico, 2022.

[15] A. Saxena, H. Pandya, G. Kumar, A. Gaud and K. M. Krishna, "Exploring convolutional networks for end-to-end visual servoing," in *IEEE Int. Conf. Rob. & Auto.*, pp. 3817-3823, Singapore, 2017.

[16] Z. Hong, S. Bian, P. Xiong and Z. Li, "Vision-Locomotion Coordination Control for a Powered Lower-Limb Prosthesis Using Fuzzy-Based Dynamic Movement Primitives," *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 2, pp. 1188-1200, April 2024.

[17] F. Tokuda, S. Arai and K. Kosuge, "Convolutional Neural Network-Based Visual Servoing for Eye-to-Hand Manipulator," *IEEE Access*, vol. 9, pp. 91820-91835, 2021.

[18] J. Guo, H. T. Nguyen, C. Liu and C. C. Cheah, "Convolutional Neural Network-Based Robot Control for an Eye-in-hand Camera", *IEEE Trans. Syst. Man Cybern.: Syst.*, vol. 53, no. 8, pp. 4764-4775, 2023.

[19] N. Wojke, A. Bewley, and D. Paulus, "Simple Online and Realtime Tracking with a Deep Association Metric," in *Proc. IEEE Int. Conf. Image Process*, pp. 3645-3649, 2017.

[20] Y. Zhang1, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, "Bytetrack: Multi-object tracking by associating every detection box". *Eu. Conf. Comp. Vis.*, pp. 1-21, Switzerland, 2022.

[21] P. Dai, R. Weng, W. Choi, C. Zhang, Z. He, and W. Ding, "Learning a Proposal Classifier for Multiple Object Tracking", in *EEE/CVF Conf. Comput. Vis. Pattern Recognit.* , pp. 2443–2452, 2021.

[22] J. S. Yuan, "Closed-loop manipulator control using quaternion feedback," *IEEE Journal Rob. Auto.*, vol. 4, no. 4, pp. 434-440, 1988.

[23] J. Funda, R. H. Taylor, and R. P. Paul, "On homogeneous transforms, quaternions, and computational efficiency", *IEEE Trans. Robot. Autom.*, pp.382-388, 1990.

[24] G. Hu, N. Gans, W. Dixon, "Quaternion-based visual servo control in the presence of camera calibration error", *Int. J. Robust Nonlinear Control*, Vol. 20, No. 5, pp. 489-503, 2010.

[25] D. Braganza, W. Dixon, D. Dawson and B. Xian, "Tracking control for robot manipulators with kinematic and dynamic uncertainty", in *IEEE Conf. Decision Control*, pp. 5293-5297, Seville, Spain, 2005.

[26] S. Arimoto, *Control Theory of Nonlinear Mechanical Systems: A Passivity-Based and Circuit-Theoretic Approach*, Oxford, U.K., Claremdon Press, 1996.

[27] J. Bento, T. Paixão, and A. B. Alvarez, "Performance Evaluation of YOLOv8, YOLOv9, YOLOv10, and YOLOv11 for Stamp Detection in Scanned Documents" *Appl. Sci.* 15, no. 6: 3154, 2025.

[28] R. Kishor, "Performance Benchmarking of YOLOv11 Variants for Real-Time Delivery Vehicle Detection: A Study on Accuracy, Speed, and Computational Trade-Offs". *Asian J. Res. Comput. Sci.*, no. 12, pp:108-122, 2024.