IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

ViKi-HyCo: A Hybrid-Control approach for complex car-like maneuvers

EDISON P. VELASCO-SÁNCHEZ, MIGUEL ÁNGEL MUÑOZ-BAÑÓN, FRANCISCO A. CANDELAS, SANTIAGO T. PUENTE, FERNANDO TORRES, (Senior Member, IEEE)
Automatics, Robotics, and Computer Vision Group (AUROVA), University of Alicante, 03690 Alicante, Spain.

Corresponding author: Edison P. Velasco-Sánchez (e-mail: edison.velasco@ua.es).

This work was supported in part by the Spanish Government through the research project PID2021-122685OB-l00, the grants for Training of Research Staff PRE2019-088069, from the Government of Spain, and ACIF/2019/088 from the Valencian Community Government and the European Regional Development Fund.

ABSTRACT While Visual Servoing is deeply studied to perform simple maneuvers, the complex cases where the target is far out of the camera field of view (FOV) during the maneuver are not common in the literature. For this reason, in this paper, we present ViKi-HyCo (Visual Servoing and Kinematic Hybrid-Controller). This approach generates the necessary maneuvers for the complex positioning of a non-holonomic mobile robot in outdoor environments. In this method, we use LiDAR-camera fusion for automatic target calculation. The multi-modal nature of our target representation allows us to hybridize the visual servoing with a kinematic controller. In this way, we can perform complex maneuvers even when the target is far away from the camera's FOV. The automatic target calculation is performed through object localization for outdoor environments, which estimates the spatial location of a target point for the kinematic controller and allows the dynamic calculation of a desired bounding box of the detected object for the visual servoing controller. The proposed approach does not require an object-tracking algorithm and can be applied to any visually tracking robotic task where its kinematic model is known. ViKi-HyCo has an error of $\mathbf{0.0428 \pm 0.0467}$ m in the X-axis and $\mathbf{0.0515 \pm 0.0323}$ m in the Y-axis at the end of a complete positioning task.

INDEX TERMS autonomous robots, domestic waste localization, hybrid-control, outdoor environment, sensor fusion, visual servoing.

I. INTRODUCTION

MOBILE robotics is increasingly challenging precise positioning in unstructured environments, which requires the performance of maneuvers for the robot's motion to achieve an accurate position concerning a specific target in the environment. This task is usually addressed by a visual servoing controller, which is one of the most studied methods in mobile robotics [1], [2], where the control actions are obtained by calculating the errors between a target detected in the image plane and its desired position.

Object-tracking algorithms are commonly employed when a target is lost or goes out of the image. However, they are typically applicable in relatively straightforward maneuvers, such as positioning in front of a target. The complexity of these maneuvers can escalate based on the specific robotics application. For instance, consider a waste collection scenario using an Ackermann vehicle equipped with a robotic arm positioned at its rear. In this case, the vehicle needs to approach the object (waste) in the front direction and execute precise maneuvers to position itself at the rear for trash pickup using the robotic arm. These complex maneuvers result in the inability to use only traditional controllers. Therefore, combined robotic algorithms and controller approaches have been employed for task execution [3]–[7]. In this way, this study researches hybridized controllers that perform complex car-like maneuvers, as in the above example, by exploiting the multi-modality of the LiDAR-camera fusion [8], [9].

The complexity of algorithms and controllers usually depends on the sensors used. Robotic systems for precise positioning integrate different types of sensors to recognize their environment and plan the movement toward a target point. Monocular cameras, LiDAR, and RGB-D are the most commonly used environmental perception sensors in mobile robotics [10]. Despite the low cost and easy integration of monocular cameras, systems using only these sensors present

several challenges when trying to recover the metric scale of the environment, in this way the aim is to unify several sensors [11]. To improve visual servoing systems, the authors in [12] conclude that integration with multiple modules can increase the accuracy and functions of visual systems in robots. Consequently, the integration of sensor fusion and controller management can perform tasks with improved performance.

Visual servoing controllers rely primarily on predefined visual features extracted from an object or scene. Therefore, this type of controller can be abruptly interrupted if it loses these features when the identified object or scene is hidden or leaves the camera's field of view (FOV). Strategies have been developed to mitigate these drawbacks, including the utilization of homography between the current frame and the keyframe [13] or employing a virtual target-guided fast scanning random tree (RRT) [14]. However, when performing a complex maneuver or the exchange of the image source for the visual servoing controller is not possible. In addition, another drawbacks of visual servoing controllers is the constant need for the visual characteristics of a target point and prior knowledge of the visual features associated with this desired point. This is why several visual servoing controller techniques rely on a known visual marker [15]–[18], and by means of image calibration and marker identification techniques, the position of a target point is known to then establish the best technique of movement towards this point. These investigations focus more on the controller development aspects and leave aside feature extraction and object detection. Therefore, these works have the disadvantage of not operating properly in unstructured environments where there are no predetermined visual markers. For this reason, several robotic motion techniques are based on learning visual servoing that identifies generic characteristic patterns [19]–[22].

On the other hand, visual servoing controllers based on object recognition and tracking have been developed, where the feature patterns of the targets are obtained by detecting objects with Neural Network (NN). In [23] and [24], depth estimation of objects is performed by designing a Recurrent Neural Network (DBox) using a generalized representation of bounding boxes and the motion of a camera. Recently, object detection NN based on the YOLO (You Only Look Once) algorithm have been applied for mobile robot visual navigation [25]–[28]. In [29] is introduced MGBM-YOLO, an application for visual servoing controller with object detection NN, where the authors propose two YOLOv3 models that are applied to the robotic grasping system of bolster spring based on image-based visual servoing. The MGBM-YOLO visual servoing controller presents a depth estimator depending on the actual area and the desired area of the object's bounding box, therefore, the system must know the object's dimensions beforehand.

This paper presents ViKi-HyCo, a Hybird-Control approach that combines a visual servoing and a kinematic controller for complex maneuvers and positioning of a car-like robot for waste collection. Using a YOLOv5 NN object detection as a feature detector for the visual servoing controller, the method works in outdoor environments and does not rely on visual markers. In addition, the system includes a spatial target point estimation algorithm with an RGB-D camera for near sensing tasks or LiDAR-camera fusion for distant detection tasks that are used for a kinematic controller. Our method continuously calculates the four vertices of the desired bounding box of the detected object, so unlike other visual servoing control methods, our algorithm has no prior knowledge of the object dimensions, allowing it to adapt to any object that the NN detects. First, the target point is detected with an object detection NN, and four characteristic points are determined by the detected bounding box. Next, the desired points are calculated using the current object's depth and the desired final depth. The visual servoing controller calculates the camera velocities, which are transformed into robot velocities utilizing the robot's kinematics model. Then, the algorithm determines which controller to use (visual servoing or kinematic) for the robotic system's maneuvers. This algorithm depends on the detection of the object, so the visual servoing controller is used when an object is detected; otherwise, a kinematic controller is used when the target is lost due to image occlusions, failures of the neural object detection network, or if the robot's displacement leaves the detected object outside the camera FOV. Thus, implementing a Hybird-Control solves the problems of a visual servoing controller when performing complex maneuvers by hybridizing with a kinematic controller.

The main contributions of the paper are the following:

- A Hybird-Control approach based on a visual servoing and a kinematic controller for complex maneuvers on car-like robots in outdoor environments by exploiting the multi-modality of the LiDAR-camera fusion.
- The dynamic calculation of the desired bounding box of unknown objects for the visual servoing controller, based on the object bounding box detected by a YOLOv5 NN and the distance estimated by LiDAR-camera fusion for long distances or by an RGB-D camera for short distances.
- Tests and comparison with another system using a visual servoing controller with object detection with a YOLO NN [29].

The rest of the paper is organized as follows: In Section II, we present an overview of complete proposed controller architecture with the subsections: *Robot and Camera Kinematic Model*, the *LiDAR-Camera Fusion*, *Object Localization* and the *Desired Features Estimation* of the detected object. Next, in Section III, we describe the experimental results of the proposed method using our own real robot BLUE [30], where we compared our method with only the visual servoing controller for forward and backward positioning, and with an algorithm of state-of-the-art. Also, this section shows the run-time of our method. Finally, in Section IV, we present the main conclusions obtained from this work and possible

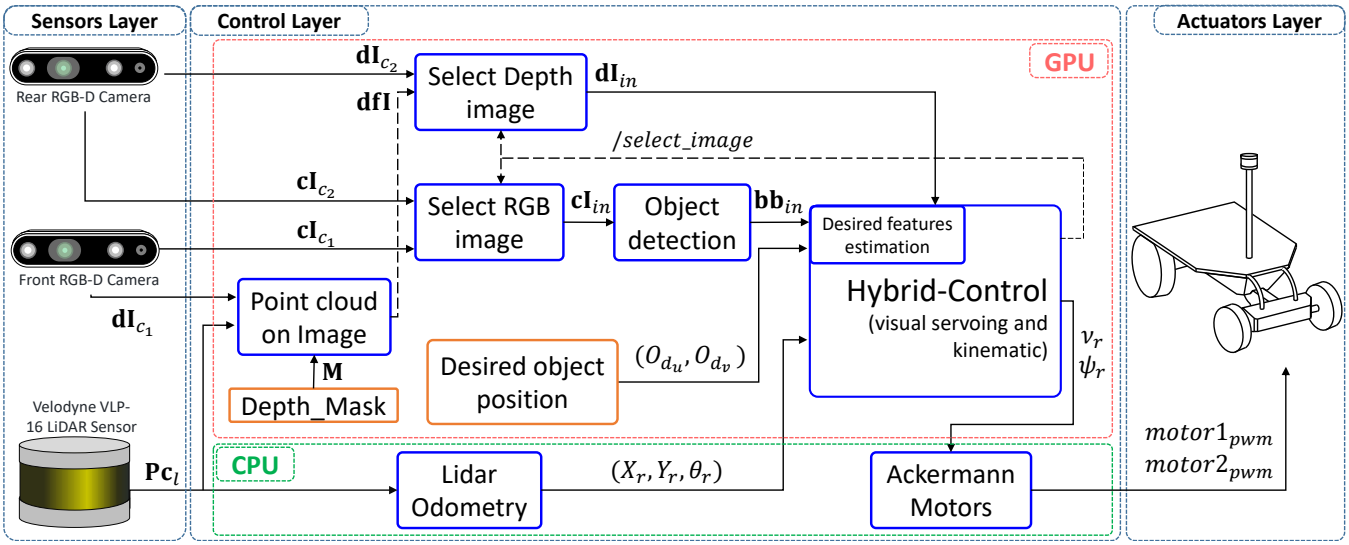


FIGURE 1. Complete ViKi-HyCo pipeline divided into three layers: **Sensors**: the layer acquiring data from the environment. **Control**: the sub-processes Point cloud on Image, select depth and RGB image, and Object detection take the data from the sensors layer and process them for the Visual Servoing and Kinematic controllers sub-process. The orange boxes are preset parameters, these are the mask for LiDAR and depth camera fusion, and the desired positions of the object in the image plane. The blue boxes are the sub-process in our approach and are detailed in the article. Finally, the LiDAR-based odometry sub-processes for the robot position and the Ackermann Motors sub-process that converts the controller output velocities into velocities to the motors for the **Actuators** layer.

future works.

II. PROPOSED CONTROLLER ARCHITECTURE

In this section we define the architecture of the proposed approach, where we start from the control laws of the visual and kinematic servoing controllers. Then, the LiDAR-Camera fusion for the localization of distant objects and the use of an RGB-D camera for close objects are detailed. Next, we describe how to estimate a desired bounding box based on detections by a YOLOv5 NN and object localization. Finally, we define the control law of our ViKi-HyCo hybrid control, which continuously calculates the robot motion velocities and decides which controller to use (kinematic or visual servoing). This approach allows the robot to use a visual servoing controller without interruptions due to occlusions in the image or loss of the detected object, including the change of data input sources for the visual servoing controller, thus achieving complex positioning maneuvers.

Fig.I shows the structure of the proposed system and the sub-processes needed to implement the ViKi-HyCo algorithm. The structure of the system is divided into three layers: Sensors, Control and Actuator layer. The sensor layer takes 2D data from the environment in RGB images, and 3D data from point clouds and depth images of the D channel of the cameras. The Control layer includes the LiDAR odometry process and the ViKi-HyCo controller, which has the following sub-processes: projection of point cloud on image, object detection, visual servoing and kinematic controller. In addition, ViKi-HyCo has a sub-process that depends on object detection and decides the source of information for the visual servoing and kinematic controller calculations. Finally, the velocities are sent from the controller to the robot motors in

FIGURE 2. BLUE: roBot for Localization in Unstructured Environments

Actuators layer. To test our approach, the ViKi-HyCo method has been employed in our research platform BLUE shown in the Fig. 2, which is a non-holonomic robot developed for autonomous navigation in unstructured outdoor environments. The Fig. 3 shows the location of each of the sensors used in this work. The origin position of each robot sensor is defined by the letter $\mathbf{T}$, where the upper index has the name of the sensor, e.g. $\mathbf{T}^{c_1}$ is the base point of front camera. In addition, the robot's transformation matrices are denoted by the letter $\mathbf{T}$, where a upper index and a lower index indicate the origin and the destination point respectively of the transformation matrix. Thus, the transformation matrix between the robot's base and the front camera is denoted as ${}^{r}\mathbf{T}_{c_1}$.

A. ROBOT KINEMATIC MODEL

The robotic kinematic model of a non-holonomic robot is widely known in the literature and developed in different

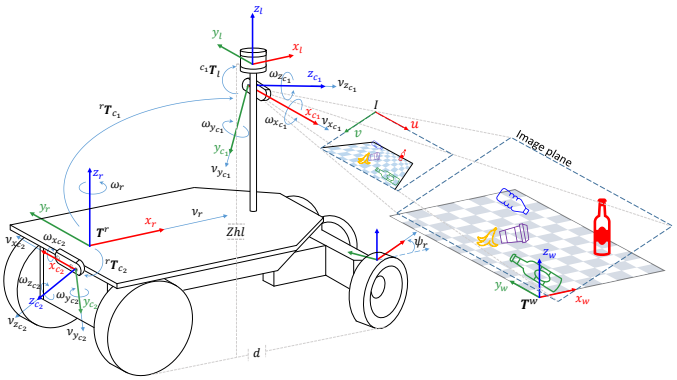


FIGURE 3. Sensor transformation systems onboard the BLUE robot. The transformation is determined by the letter $\mathbf{T}$ where the upper index is the origin and the lower index is the destination ${}^{origin}\mathbf{T}_{destination}$.

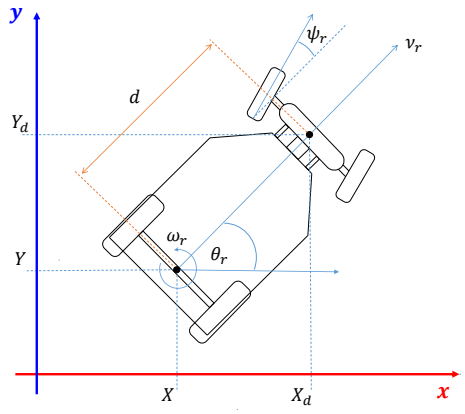


FIGURE 4. Kinematic model of BLUE robot.

applications [31](in chapter 16.5). In order to generate the control law we use the kinematic model of a car-like robot as shown in Fig 4, where the physical characteristics of the robot are determined. Thus, we have the kinematic modeling of the robot as shown in (1):

$$\dot{\mathbf{h}} = \begin{bmatrix} \dot{X} \\ \dot{Y} \end{bmatrix} = \begin{bmatrix} \cos\theta_r & -d\sin\theta_r \\ \sin\theta_r & d\cos\theta_r \end{bmatrix} \cdot \begin{bmatrix} \nu_r \\ \omega_r \end{bmatrix} \tag{1}$$

$$\mathbf{J}_b = \begin{bmatrix} \cos\theta_r & -d\sin\theta_r \\ \sin\theta_r & d\cos\theta_r \end{bmatrix} \tag{2}$$

$$\dot{\mathbf{h}} = \mathbf{J}_b \cdot \mathbf{V}_r \tag{3}$$

Where the velocities necessary to reach a target point are calculated through the position error $\dot{\mathbf{h}} = \mathbf{h} - \mathbf{h_d}$. Where $\mathbf{h}$ is the current $[X;Y]$ robot's position and $\mathbf{h_d}$ is the desired $[X_d;Y_d]$ position. Beginning from the Blue robot's kinematics, the Jacobian matrix $\mathbf{J}_b$ (2) and the velocities $\mathbf{V_r} = [\nu_r;\omega_r]$ are determined. Then feedback law is in (4).

$$\mathbf{V}_r = \mathbf{J}_b^{-1} \cdot (\mathbf{k}_1 \cdot \tanh(\dot{\mathbf{h}})) \tag{4}$$

The constant $\mathbf{k}_1$ is the positive definite matrices with the controller gains. In addition, the steering angle of the robot is determined in (5), where d is the distance between the front and rear wheels.

$$\psi_r = \arctan\left(\frac{d\cdot\omega_r}{\nu_r}\right) \tag{5}$$

The controller velocities are saturated with a tangential function of the error $\dot{\mathbf{h}}$, thus limiting the velocities to the mechanical characteristics of the robot in the ranges of linear velocity $-0.5 < \nu_r < 0.5$ m/s and a steering angle $-0.44 < \psi_r < 0.44$ rad.

The θ_r orientation and the displacement of the robot in the X, Y axis, shown in (1), are determined by a 6DOF LiDAR-based odometry system. This LiDAR odometry system is based on the F-LOAM method [32] and adapted to our BLUE robot [33]. This algorithm minimizes the offset position between a measured point cloud $\mathbf{P_c}$ and a local point cloud map. The positioning of the robot to a desired point is controlled by the LiDAR-based odometry system and the robotic kinematic model.

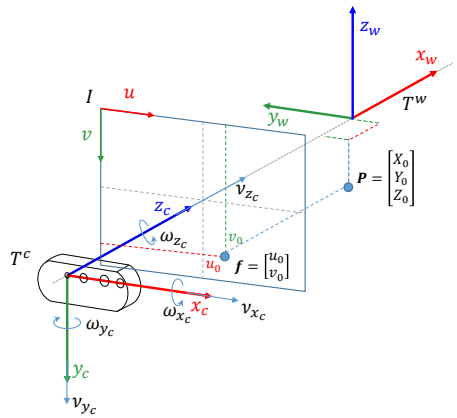


FIGURE 5. Model of camera system. The point $\mathbf{P}$ corresponds in the image plane to $\mathbf{f}$.

B. CAMERA KINEMATIC MODEL

Fig. 5 shows the kinematic model of the camera we use [34]. The point $\mathbf{P} = [X_0, Y_0, Z_0]$ corresponds in the image plane to $\mathbf{f} = [u_0, v_0]$. Thus, the displacement of a feature point in the image is described by (6).

$$\dot{\mathbf{f_0}} = \mathbf{L}_{f_0} \cdot \mathbf{V}_c \tag{6}$$

Where, $\mathbf{L}_{f_0}$ is the Jacobian matrix and $\mathbf{V}_c$ are the linear and angular velocities $[\nu_{x_c}\ \nu_{y_c}\ \nu_{z_c}\ \omega_{x_c}\ \omega_{y_c}\ \omega_{z_c}]^T$ in each axis-camera . Thus, $\mathbf{L}_{f_0}$ is defined in (7), where l represents the camera focal length.

$$\mathbf{L}_{f_0} = \begin{bmatrix} -\frac{l}{Z_0} & 0 & \frac{u_0}{Z_0} & \frac{u_0 v_0}{l} & -(l + \frac{{u_0}^2}{l}) & v_0 \\ 0 & -\frac{l}{Z_0} & \frac{v_0}{Z_0} & l + \frac{{v_0}^2}{l} & -\frac{u_0 v_0}{l} & -u_0 \end{bmatrix} \tag{7}$$

Therefore, for $\mathbf{f}$th feature points the camera kinematic model is represented in (8).

$$\dot{\mathbf{f}} = \mathbf{L}_f \cdot \mathbf{V}_c \tag{8}$$

Where the camera Jacobian matrix for n features points $\mathbf{L}_f$ is detailed in (9).

$$\mathbf{L}_f = \begin{bmatrix} -\frac{l}{Z_0} & 0 & \frac{u_0}{Z_0} & \frac{u_0 v_0}{l} & -(l+\frac{{u_0}^2}{l}) & v_0 \\ 0 & -\frac{l}{Z_0} & \frac{v_0}{Z_0} & l+\frac{{v_0}^2}{l} & -\frac{u_0 v_0}{l} & -u_0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -\frac{l}{Z_n} & 0 & \frac{u_n}{Z_n} & \frac{u_n v_n}{l} & -(l+\frac{{u_n}^2}{l}) & v_n \\ 0 & -\frac{l}{Z_n} & \frac{v_n}{Z_n} & l+\frac{{v_n}^2}{l} & -\frac{u_n v_n}{l} & -u_n \end{bmatrix} \tag{9}$$

The control law for a visual servoing controller is shown in equation (10), where the feature error $\dot{\mathbf{f}} = (\mathbf{f} - \mathbf{f}_d)$, is the displacement of the current $\mathbf{f}$ and desired $\mathbf{f}_d$ features, and the pseudo inverse of the image Jacobian is $(\mathbf{L}_f)^\dagger$. In addition, we have λ as the positive definite matrix of the control gains.

$$\mathbf{V_c} = -\lambda(\mathbf{L}_f)^\dagger \cdot \dot{\mathbf{f}} \tag{10}$$

The control law expressed in equation (10), calculates the motion velocities of the camera to decrease the error between actual and desired visual features, however, to convert these velocities to motion velocities of the robot, it is necessary the physical characteristics of the robot by applying its kinematic model. This will be addressed in the hybrid controller section.

C. LIDAR AND FRONTAL CAMERA FUSION

The fusion of several optical sensors is nowadays clearly helpful for autonomous navigation applications, to identify distances to target points or for mapping [8]. The use of LiDAR sensor and RGB cameras requires a first calibration step that has been widely studied in the literature [35], [36]. For this purpose, there are tools that allow a calibration of the LiDAR sensor and an RGB camera. We calibrate our LiDAR sensors and camera using an application developed in ROS [36]. This method finds a rigid body transformation to determine the extrinsic parameters of a LiDAR and a camera using 3D-3D point correspondences. In this way, we obtain the transformation matrix ${}^{c_1}\mathbf{T}_l = ({}^{c_1}\mathbf{R}_l,\ {}^{c_1}\mathbf{t}_l)$. This transformation matrix with the intrinsic camera parameter matrix $\mathbf{M}_{c_1}$, converts point cloud data $\mathbf{Pc}_l = [X_l, Y_l, Z_l]$ into image plane projections at points $[\mathbf{u}_l, \mathbf{v}_l]$, as shown in (11).

$$\begin{bmatrix} u \\ v \\ e \end{bmatrix} = \mathbf{M}_{c_1} \begin{bmatrix} {}^{c_1}\mathbf{R}_l & {}^{c_1}\mathbf{t}_l \\ \mathbf{0} & 1 \end{bmatrix} \cdot \begin{bmatrix} X_l \\ Y_l \\ Z_l \\ 1 \end{bmatrix}$$
$$\mathbf{u}_l = \vec{u} \odot (\vec{e})^{-1}$$
$$\mathbf{v}_l = \vec{v} \odot (\vec{e})^{-1} \tag{11}$$
$$\mathbf{u}_l := \{u_l : u_l \in \mathbb{Z}^+, 0 < u_l < W_{img}\}$$
$$\mathbf{v}_l := \{v_l : v_l \in \mathbb{Z}^+, 0 < v_l < H_{img}\}$$

Where W_{img} and H_{img} are the width and height of the image plane. Hence, we generate the depth image $\mathbf{dI}_l$ with the depth modulus of each $\mathbb{R}^3$ coordinate of the $\mathbf{Pc}_l$ point cloud with $[\mathbf{u}_l, \mathbf{v}_l]$ coordinates of the image plane as shown in (12).

$$\mathbf{dI}_{l(u_l, v_l)} = \|X_l + Y_l + Z_l\|_2 \tag{12}$$

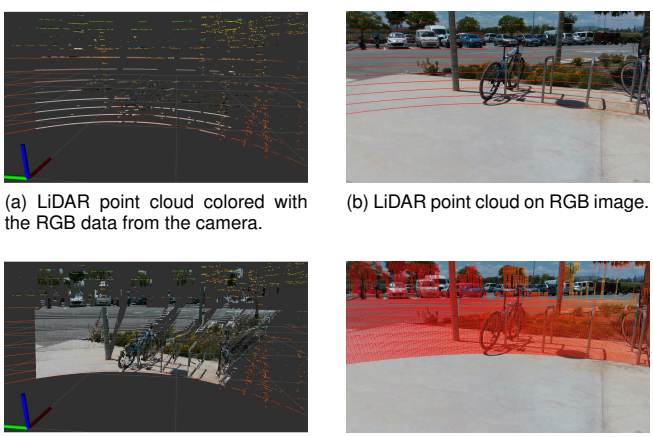
LiDAR sensors are not able to obtain depth information of small objects because they generate a point cloud in layers and this is not as dense as a depth camera. In our case, we use a 16-layer lidar sensor (see Fig. 6a). Therefore, we augmented the point cloud data with a linear interpolation between each laser beam of the LiDAR sensor. In this way, we generate virtual point cloud data by interpolating the real point cloud $\mathbf{Pc}_l \in \mathbb{R}^3$ into a $\mathbb{R}^2$ range image and using 2D linear interpolation [37]. These data are converted back to $\mathbb{R}^3$ points, obtaining a new interpolated point cloud as shown in Fig. 6c.

As shown in the Fig. 6b and Fig. 6d, the RGB image $\mathbf{cI}_{c1}$ has an area where the LiDAR cannot be projected, this is due to the location of the LiDAR sensor with respect to the front camera, so it is not possible to know the distance information of this area (called LiDAR blind spots). The camera has a depth channel that generates a point cloud working at a maximum distance range of five meters. Therefore, we combine the depth image $\mathbf{dI}_{c_1}$ with the depth data of the LiDAR point cloud already calibrated and interpolated. In this way, the depth channel of the RGB-D camera is used to know the distance of close objects, and the LiDAR-camera fusion is used for distant objects.

To combine both images, we use a mask that removes the depth information from $\mathbf{dI}_{c_1}$ where the LiDAR points are projected. We define the mask as all points with coordinates X, Y and with height of the LiDAR sensor to the ground plane $Zhl = -110\,\text{cm}$ that are within a radius $R = 410.52\,\text{cm}$. In this way, by means of the equation (11) we generate a 2d projection in the image plane of the LiDAR blind spots on the ground. To generate the pixel coordinates of the mask, the values of $-411\,\text{cm} \leq X \leq 411\,\text{cm}$ and $-411\,\text{cm} \leq Y \leq 411\,\text{cm}$ in (13) are sampled at 0.1 cm.

$$\mathbf{u}_m, \mathbf{v}_m := \left\{ \begin{matrix} u_m, v_m : (u, v) \in \mathbb{Z}^+, \\ \forall (X, Y) : X^2 + Y^2 + Zhl = R^2 \end{matrix} \right\} \tag{13}$$

We define the mask $\mathbf{M}$ (14) (Fig. 7b) as a matrix of size $W_{img} \times H_{img}$, where the pixels with coordinates $(\mathbf{u}_m, \mathbf{v}_m)$



(a) LiDAR point cloud colored with the RGB data from the camera.

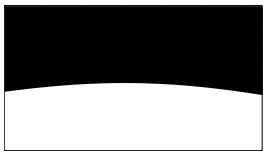
(b) LiDAR point cloud on RGB image.

(c) LiDAR interpolated point cloud colored with camera RGB data.

(d) LiDAR interpolated point cloud on RGB image

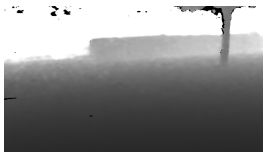
FIGURE 6. LiDAR and fusion camera on the point clouds and in the image plane.

(a) LiDAR interpolated on the RGB image.

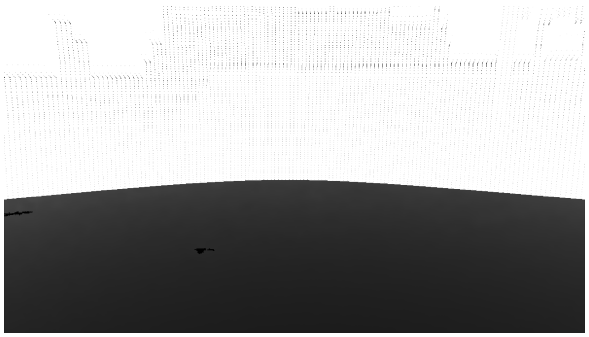


(b) LiDAR blind spot mask.

(c) LiDAR depth points projected on the image plane.



(d) Front camera depth channel image.



(e) Depth fusion of **(c)** and **(d)** depths applied to the mask of **(b)**

FIGURE 7. Fusion of LiDAR at depths in the image plane and the camera depth channel. **Note**. Figures (c) and (e) show the colors inverted in the projection area of the LiDAR points for a better visual representation.

(13) are 1 and the otherwise are 0. Finally, we obtain the fused depth image $\mathbf{dfI}$ (15) combining the LiDAR depth image $\mathbf{dI}_l$ (Fig.7c) and the front camera depth image $\mathbf{dI}_{c_1}$ (Fig.7d) filtered by the element-to-element product $\odot$ with the mask $\mathbf{M}$.

$$\mathbf{M} = \begin{cases} 1 & \forall (u_m, v_m) \in \mathbb{Z}^+ \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

$$\mathbf{dfI} = \mathbf{dI}_l + (\mathbf{M} \odot \mathbf{dI}_{c_1}) \tag{15}$$

The Fig. 7e shows the depth information of the robot's front stage and is used for the forward visual servoing control. For the case of reverse robot positioning, the visual servoing controller uses the depth image $\mathbf{dI}_{c_2}$ from the rear camera, because it has a FOV of four meters in the floor plane, which is sufficient for accurate depth measurements to obtain the object's spatial coordinates. Then, we define $\mathbf{dI}_{in}$ as the depth image selected between $\mathbf{dfI}$ for forward visual servoing controller and $\mathbf{dI}_{c_2}$ for rearward visual servoing control.

D. OBJECT LOCALIZATION

Locating a positioning target point is possible with the depth information of robot's environment. We define as target points the objects detected with a YOLOv5 object detection NN. In particular, we start from the work developed in [38],

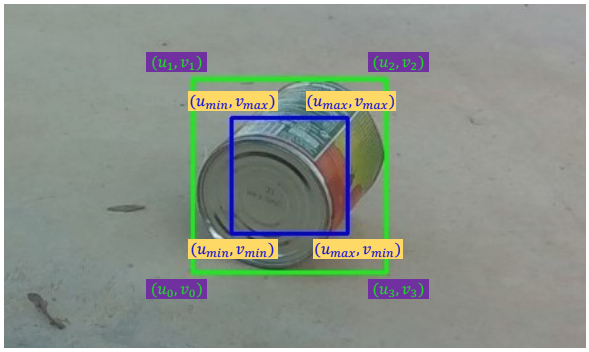


FIGURE 8. The green box is the bounding box of the object detected with the YOLOv5 NN. The blue box is the 40% reduction of the original bounding box.

where objects are detected and localized with respect to our research platform BLUE with the camera depth information and YOLOv5 objects detection in outdoor environments. The results of objects detection were a mAP@.5 around 0.99 and for a mAP@.95 over 0.84 with an average error smaller than 0.25 m.

On the contrary to method in [38], where the camera-object distance is the value of the center of the bounding box detected in a depth image, on the basis of research [9], we define the camera-object distance with the average value in the bounding box detected in the depth image $\mathbf{dI}_{in}$.

Due to irregularities of the detected object some points inside the bounding box may not belong to the object (see Fig. 8). For this reason, to avoid measurement errors, a new bounding box $\mathbf{bb}'$ is generated that is 40% smaller than the original bounding box $\mathbf{bb}$, where the new bounding box coordinates are defined in the equation (16) . Thus, the camera-object distance d_o is defined as the average of the values inside the new bounding box that are different from 0 (17).

$$\begin{aligned} \mathbf{bb} &= \begin{Bmatrix} (u_1, v_1) & (u_2, v_2) \\ (u_0, v_0) & (u_3, v_3) \end{Bmatrix} \\ u_{min} &= \lfloor 0.4\,(0.5 \times (u_2 - u_0)) + u_0 \rceil \\ u_{max} &= \lfloor 0.4\,(0.5 \times (u_0 - u_2)) + u_2 \rceil \\ v_{min} &= \lfloor 0.4\,(0.5 \times (v_2 - v_0)) + v_0 \rceil \\ v_{max} &= \lfloor 0.4\,(0.5 \times (v_0 - v_2)) + v_2 \rceil \\ \mathbf{bb}' &= \begin{Bmatrix} (u_{min}, v_{max}) & (u_{max}, v_{max}) \\ (u_{min}, v_{min}) & (u_{max}, v_{min}) \end{Bmatrix} \end{aligned} \tag{16}$$

$$d_o = \frac{1}{n} \sum_{v=v_{min}}^{v_{max}} \sum_{u=u_{min}}^{u_{max}} [\mathbf{dI}_{in}(u, v) \neq 0] \tag{17}$$

Therefore, the position of an object is defined as $\mathbf{P}_o \in \mathbb{R}^3$ (18), where O_{o_u} and O_{o_v} are the coordinates in the image plane of the center of the detected object's bounding box, and the values of (u_c, v_c) and (l_u, l_v) are the camera's optical center and focal lengths respectively.

$$\mathbf{P_o} = \begin{cases} Z_o = d_o \\ X_o = \dfrac{(O_{o_u} - u_c)d_o}{l_u} \\ Y_o = \dfrac{(O_{o_v} - v_c)d_o}{l_v} \end{cases} \tag{18}$$

E. DESIRED FEATURES ESTIMATION

The control law of a visual servoing controller, shown in section II-B in equation (10), calculates the camera velocities $\mathbf{V}_c$ to minimize the error $\dot{\mathbf{f}}$ of desired visual characteristics and current characteristics with the pseudo inverse image Jacobian $(\mathbf{L}_f)^\dagger$. As detailed in [31] in chapter 11.2.3, at least three features are necessary in the image, where these cannot be col-linear, so there are no singularities in the image Jacobian matrix (7). We use the four corners of the object bounding box detected as current features $\mathbf{f}$ at each iteration and obtain the desired features $\mathbf{f}_d$ from the desired center coordinates (O_{d_u}, O_{d_v}) in the plane image, the width w_o and height h_o of the current object bounding box, the variable k, which is the ratio of the current camera-object depth Z_o (17) and the value Z_d defined as the depth at each iteration in image $\mathbf{dI}_{in}$ at the coordinates (O_{d_u}, O_{d_v}). In this way, we calculate the coordinates $\mathbf{f}_d = \{u_d, v_d\}$ of each desired feature in (19), where the ratio is defined as $k = Z_o \times Z_d^{-1}$.

$$\begin{aligned} u_d = O_{d_u} \pm \tfrac{w_d}{2} \quad , \quad v_d = O_{d_v} \pm \tfrac{h_d}{2} \\ w_d = w_o \times k \quad , \quad h_d = h_o \times k \end{aligned} \tag{19}$$

In this work, the implemented image Jacobian (7) uses the same distance Z_o shown in (17) for each feature $\mathbf{f}$. Thus, the velocities $\mathbf{V}_c$ (10) calculated by the visual servoing control law are the camera velocities to minimize the positioning of $\mathbf{f}$ to $\mathbf{f}_d$.

F. VIKI-HYCO CONTROL LAW

Visual servoing controllers are widely used when working with image features, generally used in structured environments where there are markers on the stage or on the target objects of the robotic system [15]. There has been little work in outdoor environments with varied lighting and where object features are not geometric features or visual markers [19], [20], [29]. The main problem with visual servoing controllers is that they require the object features in each frame to compute the controller output velocities. This loss of visual features of detected objects may be due to robot maneuvers, camera-object occlusions or the detected object being outside the FOV of the camera. In this work, we propose a control law that combines the visual servoing controller with the robot kinematic controller, where, the kinematic controller is triggered when the visual servoing loses the object features. This avoids the problems of the visual servo controller when the visual characteristics of the object are lost and allows positioning maneuvers for the robot even if the object is not detected.

For this, we first convert the axis-camera velocities $\mathbf{V}_c$ calculated in section II-B, to $\mathbf{V}_r$ robot velocities through equation (20).

$$\dot{\mathbf{f}} = \mathbf{L}_f \cdot \mathbf{T}_r^c \cdot \mathbf{J}_r \cdot \mathbf{V}_r \tag{20}$$

To calculate $\mathbf{V}_r$, the BLUE robot Jacobian $\mathbf{J}_b$ (2) must be defined as a matrix of size 6x6. Therefore, we define a new Jacobian $\mathbf{J}_r$ in (21).

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \mathbf{1} \end{bmatrix} = \underbrace{\begin{bmatrix} \cos\theta_r & -d\sin\theta_r & \mathbf{0} \\ \sin\theta_r & d\cos\theta_r & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}}_{\mathbf{J}_r} \cdot \begin{bmatrix} v_r \\ \omega_r \\ \mathbf{1} \end{bmatrix} \tag{21}$$

Where $\mathbf{T}_r^c$ (22) is the robot-camera transformation matrix. This matrix uses the rotation matrix ${}^r\mathbf{R}_c$ and the skew-symmetric matrix of the translation matrix ${}^r\mathbf{t}_c$ (23).

$$\mathbf{T}_r^c = \begin{bmatrix} {}^r\mathbf{R}_c & [{}^r\mathbf{t}_c]_\times {}^r\mathbf{R}_c \\ \mathbf{0} & {}^r\mathbf{R}_c \end{bmatrix} \tag{22}$$

$$[{}^r\mathbf{t}_c] = \begin{bmatrix} 0 & -t_Z & t_Y \\ t_Z & 0 & -t_X \\ -t_Y & t_X & 0 \end{bmatrix} \tag{23}$$

Therefore, and with $\dot{\mathbf{f}} = (\mathbf{f} - \mathbf{f}_d)$, the visual servoing controller law for a car-like mobile robot is described in equation (24).

$$\mathbf{V}_r = -\lambda(\mathbf{L}_f \cdot \mathbf{T}_r^c \cdot \mathbf{J}_r)^\dagger \cdot (\mathbf{f} - \mathbf{f}_d) \tag{24}$$

The matrix transformation $({}^r\mathbf{R}_c, {}^r\mathbf{t}_c)$ is selected according to the type of positioning, choosing between $({}^r\mathbf{R}_{c_1}, {}^r\mathbf{t}_{c_1})$ corresponding to the front camera-robot transform for forward positioning and $({}^r\mathbf{R}_{c_2}, {}^r\mathbf{t}_{c_2})$ corresponding to the rear camera-robot transform for backward positioning.

Thus, we define the control law of the ViKi-HyCo method in equation (25), where the controller depends on the variable c which is 1 when there is object detection and 0 when there is no object detection, so we can calculate the robot output velocities $\mathbf{V} = [\nu_r, \omega_r]^T$. Finally the velocities for our robot are the linear velocity ν_r and the steering angle ψ_r defined in section II-A in equation (5).

$$\begin{aligned} \mathbf{V} = (1-c)\,\{-\lambda(\mathbf{L}_f \cdot\ \mathbf{T}_r^c \cdot \mathbf{J}_r)^\dagger \cdot (\mathbf{f} - \mathbf{f}_d)\} + \\ c\,\{\mathbf{J}_b^{-1} \cdot (\mathbf{k}_1 \cdot \tanh(\dot{\mathbf{h}}))\} \end{aligned} \tag{25}$$

In order to reduce abrupt velocity changes with the control law in (25), the velocity that the robot uses is calculated with the velocities of a previous frame $(n-1)$ and a current frame n, as defined in (26).

$$\mathbf{V}_n = (1 - (\mathbf{V}_n - \mathbf{V}_{n-1})) \odot \mathbf{V}_n \tag{26}$$

To enable the implemented method to calculate the robot's positioning velocity, at least one detection of the object must be made at the first instant, so that the target point for the kinematic controller can be determined. Once this first detection has been made, the controller to be used is determined according to the number of detections of the object.

III. EVALUATION AND RESULTS

In this section we validated the robustness and accuracy of our ViKi-HyCo method by performing several experiments comparing the advantages of using a hybrid-control combining a visual servoing controller and a kinematic controller for the positioning of our research platform BLUE. We divided the section into four groups of experiments. *Experiments with only visual servoing controller*, where we evaluate the problems of having only visual servoing controller on the mobile robot when it performs positioning maneuvers and loses detection of its target point. *Experiments with the ViKi-HyCo method*, where we evaluated our hybrid controller approach by allowing the visual servoing controller to work in conjunction with a kinematic controller for complex positioning maneuvers of the mobile robot. *Comparative Experiments*, where we evaluate the approach of continuous calculation of a desired bounding box for unknown objects in a visual servoing controller with another controller from the literature where its desired bounding box is given by the physical characteristics of a known object. Finally, in the *Robot Placement Experiments*, using our Viki-Hyco hybrid controller, we detail the positioning maneuvers of the mobile robot that the hybrid-control generates and that are necessary to position the robot towards a detected object in a desired zone of the robot. For this experiment, it is necessary for the positioning of the robot maneuvers where the data input is switched from a front to a rear camera for the visual servoing controller, thus, the use of a visual object tracker is not possible.

A. EXPERIMENTAL SETUP

We evaluated the approach ViKi-HyCo in our research platform BLUE. It has a VLP-16 LiDAR sensor, two Intel ® RealSense ™ D435i RGB-D cameras, and an MSI onboard computer with a 2.60 GHz 6-core processor with 16 GB of RAM running Ubuntu 18.04 and an NVIDIA GTX 1660 graphics card with 6.0 GB of video memory. In the experiments, we limited the velocities to the mechanical characteristics of the robot BLUE in the ranges of linear velocity $-0.5 < \nu_r < 0.5$ m/s and a steering angle $-0.44 < \psi_r < 0.44$ rad. We compare the positioning errors of a forward and backward positioning when using ViKi-HyCo regarding a only the visual servoing controller. Also, we analyze the results of the robot positioning task to an object, where forward and backward positioning are used together. In addition, we compare our approach of continuous calculation of a desired bounding box for unknown objects with the method MGBM-YOLO proposed in [29], where these authors use a visual servoing controller with the bounding box features of an object detected by a YOLOv3 NN as well as a Jacobian image matrix, instead of using camera-object depths, to estimate the depth values with the area of the desired object and the area of the current object. For the positioning experiments, we consider the maneuvers of positioning the robot towards a detected object in a desired area of interest of the robot, where we consider this area of interest as the robot's manipulation zone for future object grasping tasks. Finally, we compare the results of position error, translation point evolution, camera-axis velocities, and calculated robot velocities of all proposed experiments.

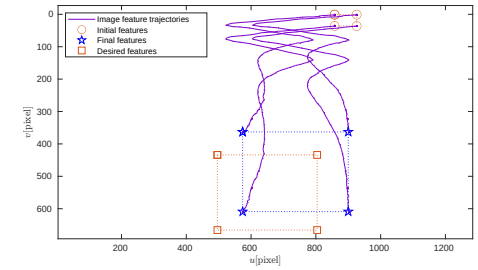
The experiments performed consider solved planning and autonomous global and local navigation tasks of our BLUE robot with the method presented in [39]. Therefore, we evaluated ViKi-HyCo in an unstructured environment. Besides, we use only one object per experiment of the categories: can, plastic, glass or carton. As a matter of fact, we do not evaluate our controller with several objects in scene, because the ViKi-HyCo method actually does not include an object tracker. We neither classify the objects, because we consider them as a target point regardless of the object class. The performance of the NN in detecting objects is not evaluated. This has already been analyzed in [38] and mentioned in the Section II-D. For this reason, our scenarios are not challenging for object detection by YOLOv5. However, experimentation shows that a NN may eventually lose object detection in a simple scenario. In addition, our hybrid-control system does not have a visual object tracker, because this would not be possible when performing the complete positioning maneuver where it is necessary to change the image input source of the visual servoing controller.

The ground truth was established as the physical measurements taken from the robot's base point $\mathbf{T^r}$ to the object location point. We consider for forward visual servoing positioning the RGB image from the front camera and the interpolated LiDAR to 112 virtual layers (II-C), and for backward positioning we use the depth image and the RGB image from the rear camera. We set the two cameras at 15 fps (frames per second) with 1280x720 resolution ($W_{img} = 1280$, $H_{img} = 720$). The gains in (25) were established prior experimentation, where $\lambda = [0.85, 0.3, 1.0, 1.0, 1.0]$ to the front camera and $\lambda = [0.85, 1.05, 1.0, 1.0, 1.0]$ to the rear camera. The gains of the forward kinematic controller is $\mathbf{k}_1 = [2.0, 1.0]$ and the backward kinematic controller is $\mathbf{k}_1 = [1.0, 2.0]$. Additionally, according to the metrics for determining the error of a positioning system presented in [40], we evaluated the results of our method using the Mean Squared Error (MSE). Finally, each iteration shown in the result plots represents the run-time of the method and is equivalent to an average of 44 ms each iteration.

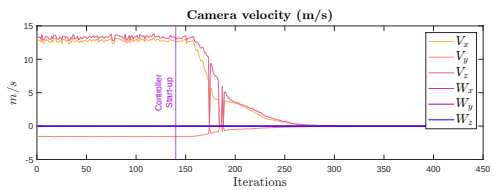
B. EXPERIMENTS WITH THE CLASSIC VISUAL SERVOING CONTROLLER

We performed 40 experiments with four different objects in two outdoor environments. 20 experiments with forward positioning and 20 with backward positioning, applying in both cases the visual servoing controller, calculating the robot velocity V_r by means of the equation (24).

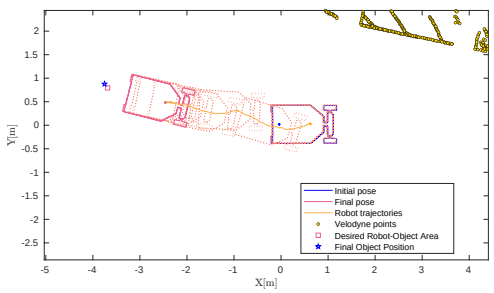
In Fig. 9c shows an experiment of the positioning path of the robot towards the object with the visual servoing controller backward. When using a visual servoing controller we have a positioning error of 0.0645 m in X-axis and 0.0832 m in Y-axis (see Fig. 9e). In the image plane (Fig. 9a), the



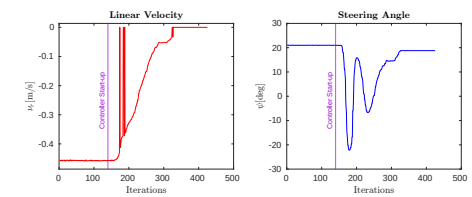
(a) Camera path



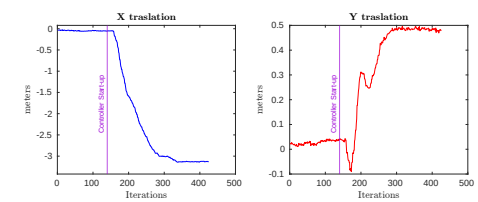
(b) Camera velocity



(c) Robot path

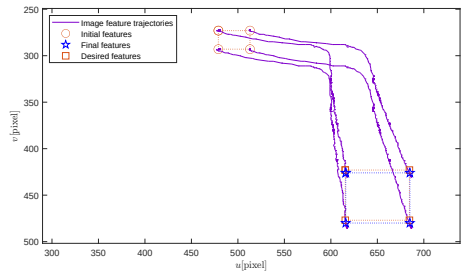


(d) Lineal velocity and steering angle

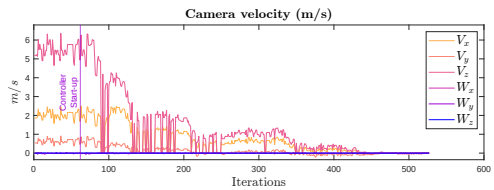


(e) X and Y robot translation

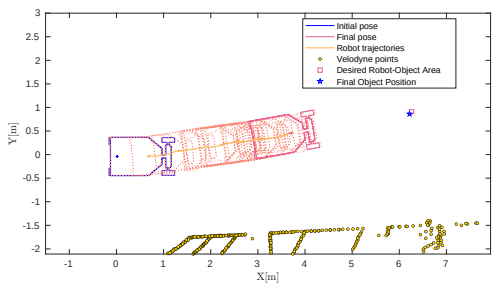
FIGURE 9. Backward positioning of BLUE robot towards an object detected using only the visual servoing controller.



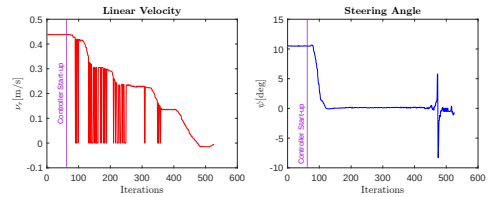
(a) Camera path



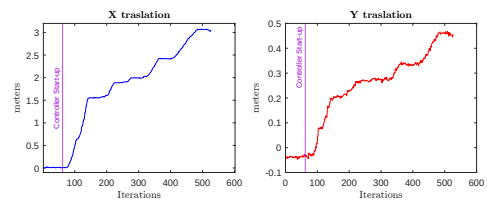
(b) Camera velocity



(c) Robot path



(d) Lineal velocity and steering angle

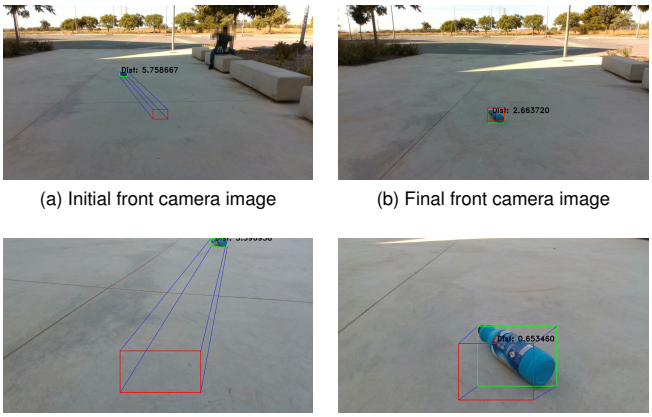


(e) X and Y robot translation

FIGURE 10. Forward positioning of BLUE robot towards an object detected using only visual servoing controller. Missed object detections cause the robot to stop abruptly.

camera motion fails to converge completely to the desired characteristics, this is because the YOLOv5 NN detector loses object detections in some frames, consequently the linear and angular velocities are 0 m/s, stopping the robot's motion. The velocities change to 0 m/s because no object is detected in the image plane. As a result, there are no features $\mathbf{f}$ for the visual servo controller, and through the desired feature $\mathbf{f}_d$ (19) and the visual servo control law (24), the velocities calculated by the controller become 0 m/s (see Fig.9b). The above is shown in Fig. 9d, where the calculated linear velocity is 0 m/s in some iterations of the experiment (near iteration number 200).

Fig. 10 shows the forward positioning of the robot with the visual servoing controller. In this experiment, it is seen in more detail that the calculated camera velocities (see Fig. 10b) and linear velocities of the mobile robot (see Fig. 10d) are equal to 0 m/s. Likewise, as in the backward position-



(a) Initial front camera image (b) Final front camera image

(c) Final rear camera image (d) Final rear camera image

FIGURE 11. Initial and final images from the cameras when using a classic visual servoing controller in the forward and backward positioning of the BLUE robot.

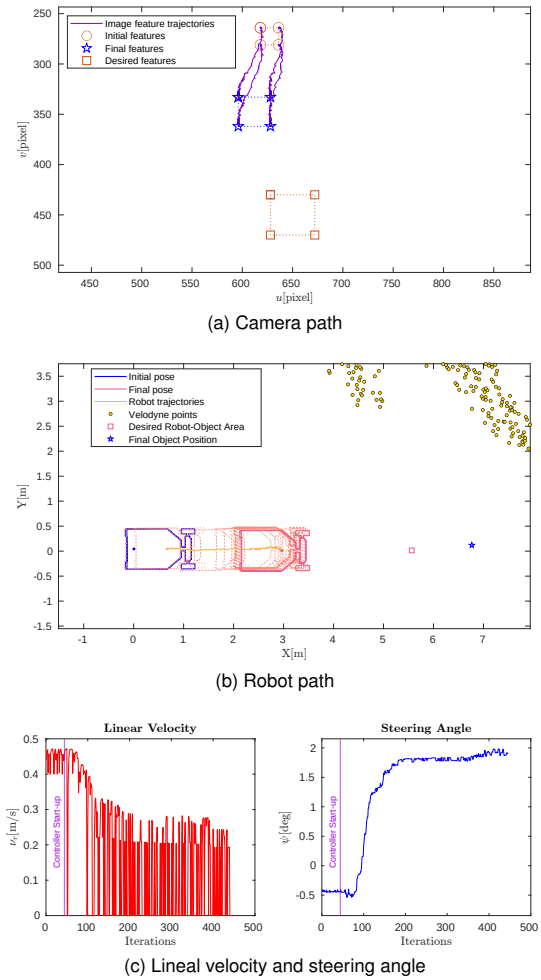
ing with the visual servoing controller, the abrupt velocity changes are caused when the object detections are missed. Despite the fact that the robot converges to the desired position in the image plane (see Fig. 10a) and approaches the desired position in the navigation plane (see Fig. 10c), the missed object detections cause the robot stopped, generating a discontinuous trajectory, as seen in the trajectory in X and Y (see Fig. 10e). This experiment has an error of 0.0416 in X-axis and 0.0466 in Y-axis.

The initial and final image of the front and rear cameras when using only the visual servoing controller are shown in Fig. 11.

During several experiments performed in forward positioning with only the visual servoing controller, object detections by YOLOv5 were lost and the robot stopped before the desired position. When the robot stops, the desired and actual bounding boxes do not converge, as shown in Fig. 12a. These lost detections are caused by the robot's positioning movements, causing the object to move in the image plane. The robot does not move to the desired position due to the abrupt changes of the linear velocities that stop it (see Fig. 12c). Thus there are positioning errors of 1.202 m in the X-axis and 0.105 in the Y-axis (see Fig. 12b).

C. EXPERIMENTS WITH VIKI-HYCO METHOD

Like in the previous experimentation, we performed 40 tests of our ViKi-HyCo algorithm, which are: 20 for forward positioning and 20 for backward positioning with four different objects. As shown in Fig. 13a, our ViKi-HyCo method approaches the desired bounding box to the current one in the image plane in backward positioning. Moreover, unlike the visual servoing controller (see Section. III-B), ViKi-HyCo has no discontinuities when it misses the object detections. Since, immediately after missing a detection, the kinematic controller of the robot is activated. In this way, although object detections are lost, the hybrid controller allows the robot to perform smooth and uninterrupted maneuvers. Consequently, as shown in Fig. 13b and Fig. 13d, the camera and mobile robot velocities does not decrease to 0 m/s and is continuous with lower perturbations. Thus, the Fig. 13c shows the robot's path of backward positioning towards the desired object with positioning errors of 0.0271 m in the X-axis and 0.0336 m in the Y-axis. Additionally, Fig. 13e shows that the trajectory has no discontinuities.



(a) Camera path

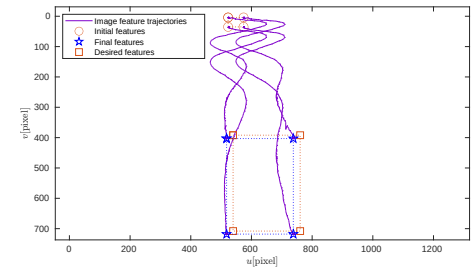
(b) Robot path

(c) Lineal velocity and steering angle

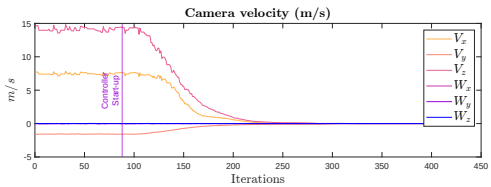
FIGURE 12. Errors on forward positioning of BLUE robot towards an object detected by YOLOv5 using the visual servoing controller. Missed object detections cause the robot to stop abruptly.

Also, in forward positioning with the ViKi-HyCo algorithm, Fig. 14a shows that the controller converges to the desired bounding box in the plane image, and the Fig 14c shows that the robot moves to the desired point with positioning errors of 0.0256 m in the X-axis and 0.0469 m in the Y-axis. Similarly to the previous experiment, the camera and mobile robot velocities does not decrease to 0 m/s (see Fig. 14b and Fig. 14d). The X and Y translations (see Fig.14e) shows that the robot had a smooth positioning without stopping.

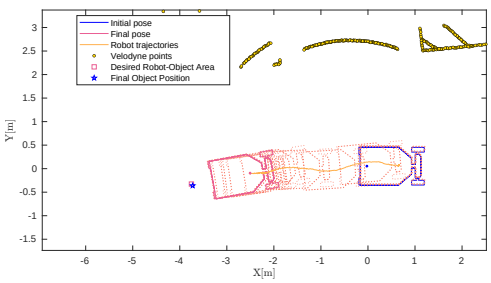
The initial and final image of the front and rear cameras with The ViKi-HyCo method are shown in Fig. 15.



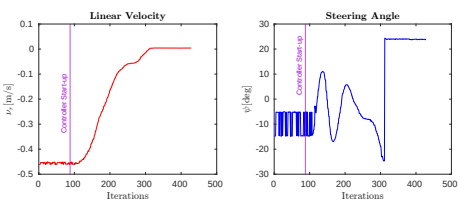
(a) Camera path



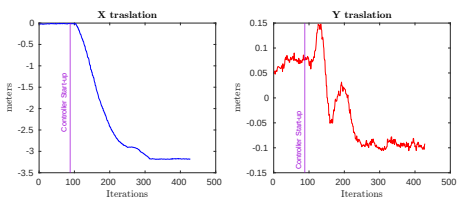
(b) Camera velocity



(c) Robot path

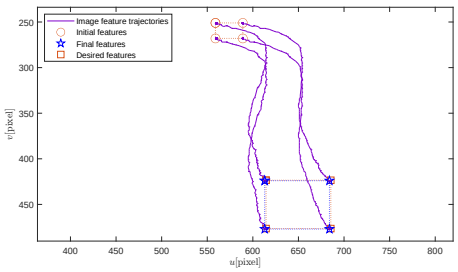


(d) Lineal velocity and steering angle

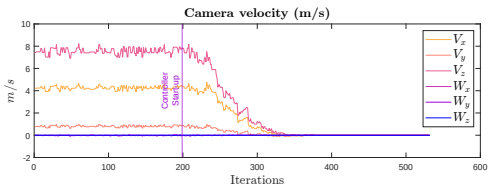


(e) X and Y robot translation

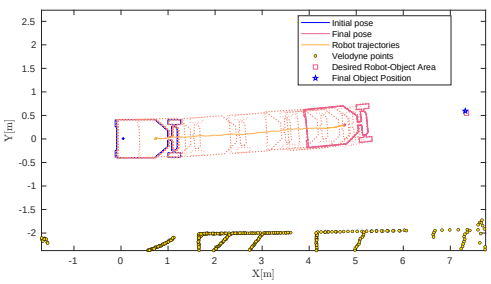
FIGURE 13. Backward positioning of BLUE robot towards an object detected by YOLOv5 using the ViKi-HyCo method.



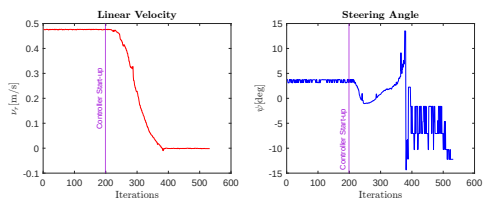
(a) Camera path



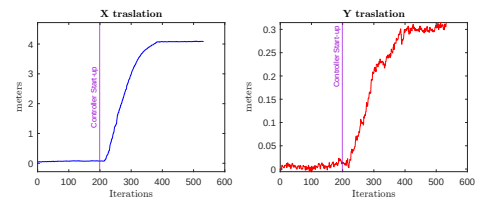
(b) Camera velocity



(c) Robot path



(d) Lineal velocity and steering angle

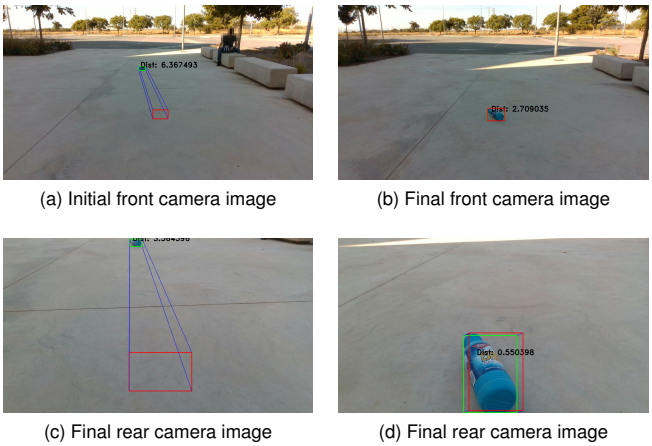


(e) X and Y robot translation

FIGURE 14. Forward positioning of BLUE robot towards an object detected by YOLOv5 using the ViKi-HyCo method.

D. COMPARATIVE EXPERIMENTS

The MGBM-YOLO method proposed in [29], uses a visual servoing controller with YOLOv3 detections and modifies the Jacobian's image. Instead of using the camera-object depth, it works with the desired and current bounding box of the detected object. MGBM-YOLO method has the final bounding box because it knows the dimensions and shape of the object projected on the image plane, in this case springs. In this way, in their application, the velocities calculated by the controller converge and reduce the positioning errors. It is worth mentioning that, although MGBM-YOLO is developed for spring grasping with an image-based robotic arm, it uses a methodology similar to ours. The visual servoing controller law is similar to ours, since it uses the vertices of the bounding box of the YOLOv3 NN. The main difference in our proposal is that in the visual servoing controller, we apply the desired bounding box update to facilitate the match the final shape. That is why, in this experimentation, we compare



(a) Initial front camera image (b) Final front camera image

(c) Final rear camera image (d) Final rear camera image

FIGURE 15. Initial and final images from the cameras when using ViKi-HyCo in the forward and backward positioning of the BLUE robot.

the behavior of the visual servoing controller proposed in MGBM-YOLO and part of our proposal which is the updating of the bounding box in the visual servoing controller. To perform this experiment, we generated a desired object's bounding box with the dimensions of a bottle as if the robot's camera placed it in its final position. The disadvantage of the MGBM-YOLO method is that the shape and dimensions of the detected object must be known in advance to generate a desired bounding box. Therefore, if the shape of the actual bounding box of the object does not match the final shape, the velocities of the controller will not converge.

Fig. 16a shows that the visual servoing controller based on the MGBM-YOLO method, fails when navigate to the desired point, because the desired object's bounding box is static and is not constantly updated to the shape of the current object's bounding box. Hence, as shown in Fig. 16e, the calculated linear velocity and steering angle of the controller oscillate as they approach the desired point and do not converge to static value. Then, as shown in Fig. 16b and 16g, the robot does not find a stabilization point. The stabilization point is only reached if the controller velocities converge to zero by making the bounding boxes match in size and shape.

We performed the same experiment with the ViKi-HyCo method, as shown in Fig. 16c, the desired and detected bounding boxes converge, thus, the robot positioning velocities (see Fig. 16f) have a stabilization point to finish the trajectory, so the robot does not present oscillations to reach the target point, as shown in Fig. 16d and 16h. The visual servoing controller of the MGBM-YOLO method does not converge in the image plane, with the result that the calculated velocities for the camera never reach 0m/s. On the other hand, in the visual servoing controller of ViKi-HyCo, the velocities are stabilized during robot positioning, as the actual and desired features converge on the image plane. Fig. 17 shows the camera velocities provided by the visual servoing controller of the MGBM-YOLO and ViKi-HyCo methods in the comparative experiment.

E. ROBOT PLACEMENT EXPERIMENTS

In this section, we evaluate the results of the complete placement task of the BLUE robot, which consists of implementing experiments in III-C in a combined positioning process. In this way, we analyze the complete placement task of the robot to reverse towards a target object. We consider this as the manipulation zone, because the robot must position itself in this way to be able to manipulate objects with the onboard robotic arm. To analyze the results of these experiments, we performed a total of 12 experiments with different objects. It is important to mention that the experiment is initialized when there is at least one detection of the object by the YOLOv5 NN with the front camera image. In this way, the object is located with respect to the robot. In these experiments, the full ViKi-HyCo method implemented is divided into 3 states, which are the following:

State 1. ViKi-HyCo forward positioning The method used in experimentation III-C. (see Fig.14).

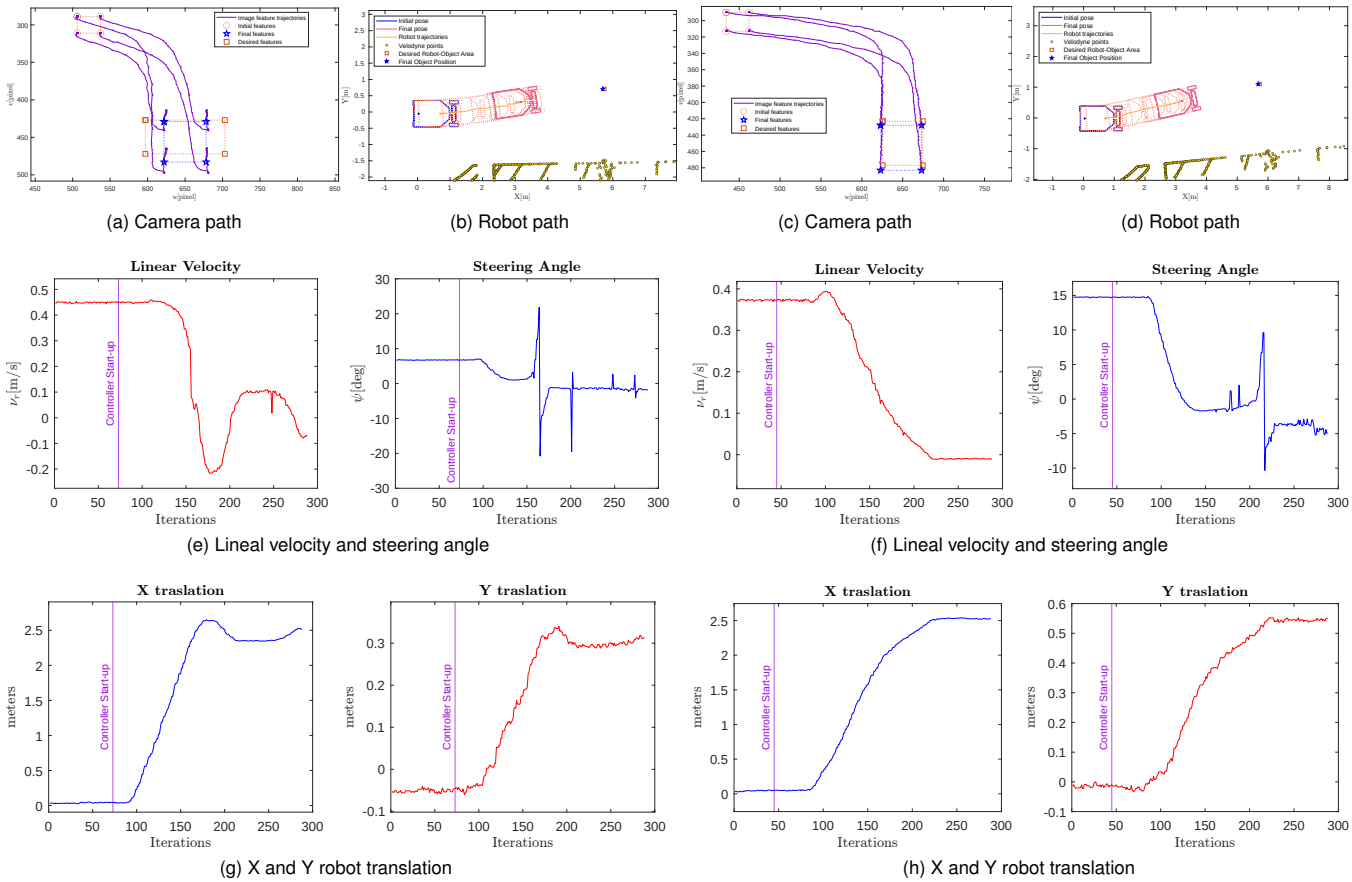
State 2. Kinematic control Since no camera focuses on the desired object while the robot rotates, a kinematic controller is used for robot rotation. This is used after positioning the object at the desired point in front of the robot with the ViKi-HyCo forward positioning.

State 3. ViKi-HyCo backward positioning The method used in experimentation III-C. (see Fig.13).

The robot carries out the transition from one state to another of the experiment automatically. To do this, the transition are validated with the value of the features error of the visual servoing controller or the position error of the kinematic control. In the case of ViKi-HyCo controllers for forward and backward positioning, the $\dot{\mathbf{f}}$ feature error is evaluated, which must be in range $-2 < \dot{\mathbf{f}} < 2$ pixels to establish that the positioning has been successful. For kinematic control, the change of state is given with a positioning error within the range $-0.01 < \dot{\mathbf{h}} < 0.01$ m. Finally, our method prioritizes the detections with the front camera, in this way, the process begins with the front camera detection of the object.

Fig. 18 shows the results of the complete positioning experiment using the ViKi-HyCo controller. The path in Fig. 18a shows that the robot is positioned in reverse towards the object, locating it in the desired area of the experiment. The velocities of the front and rear camera are shown in Fig.18b. In Step 1, the velocities of the front camera are calculated with the ViKi-HyCo controller. In Step 2, the velocities of both cameras are 0 m/s, since only the kinematic controller is used and the object is outside the camera FOV. Finally, in Step 3, the velocities shown correspond to the rear camera, calculated by the ViKi-HyCo controller.

Fig. 18e, 18f, 18c and 18d shows respectively the X translation, Y translations, the linear velocity and the steering angle in each of the three states of the robot's trajectory from the initial point to the reverse positioning towards the desired object. Finally, Fig. 19a and 19b show the initial and final

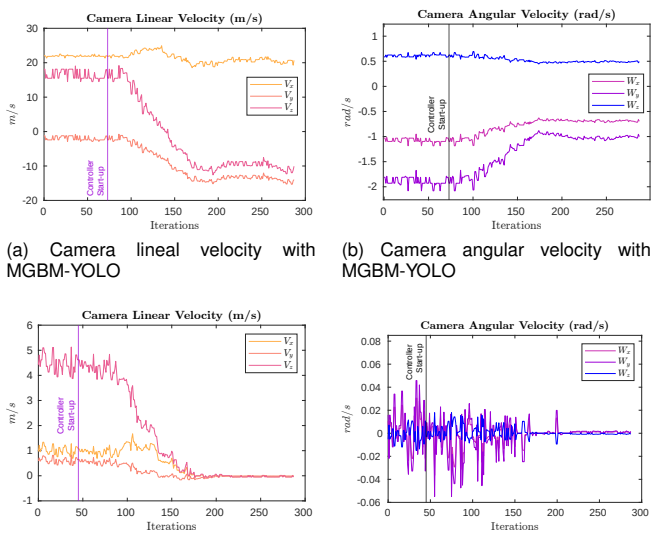


(a) Camera path (b) Robot path (c) Camera path (d) Robot path

(e) Lineal velocity and steering angle (f) Lineal velocity and steering angle

(g) X and Y robot translation (h) X and Y robot translation

FIGURE 16. **(a), (b), (e)** and **(g)** results of forward positioning of the BLUE robot implemented the MGBM-YOLO method, where the Jacobian's image is calculated with the areas of the current and desired object's bounding box. **(c), (d), (f)** and **(h)** results of forward positioning of the BLUE robot implemented the ViKi-HyCo method. In the MGBM-YOLO method, the robot does not arrive at a static position because the actual and desired features in the image plane do not match. On the other hand, with the ViKi-HyCo method, the robot does arrive at a static position, since the actual and desired features eventually coincide.



(a) Camera lineal velocity with MGBM-YOLO (b) Camera angular velocity with MGBM-YOLO

(c) Camera lineal velocity with ViKi-HyCo (d) Camera angular velocity with ViKi-HyCo

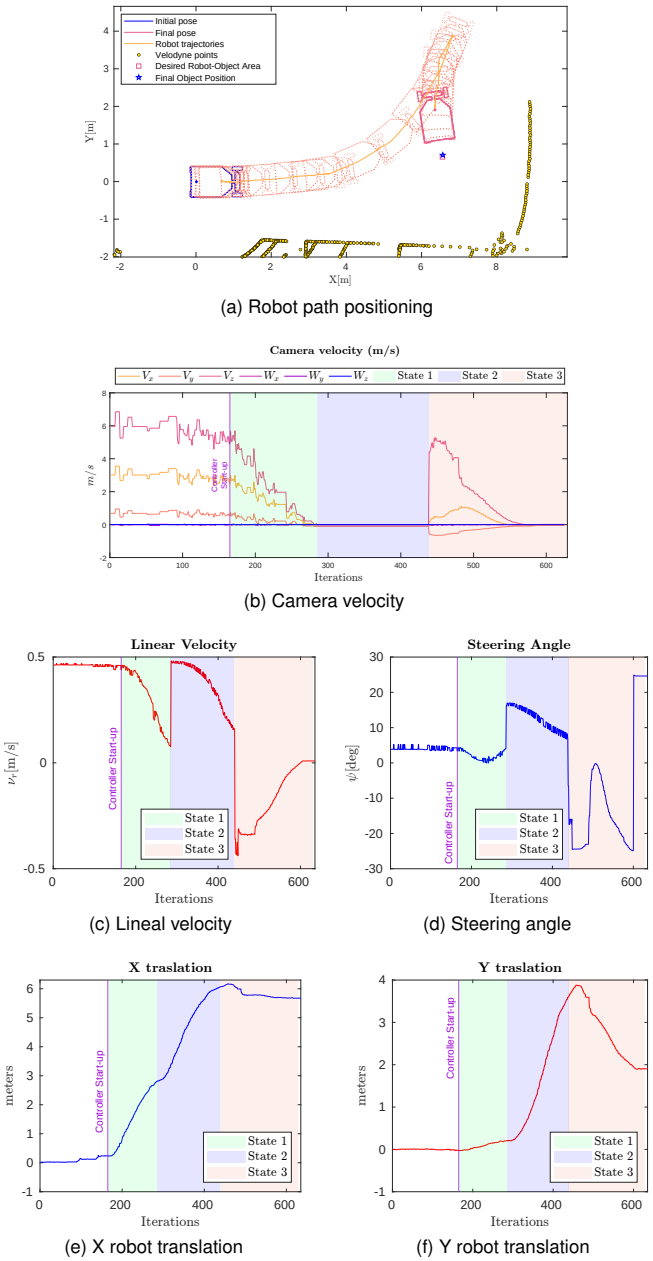
FIGURE 17. Camera velocities calculated by visual servoing controllers using the MGBM-YOLO and ViKi-HyCo methods.

pose of the robot's trajectory, and the Fig. 19c with Fig. 19d show the initial and final frame of frontal and rear camera respectively.

F. RESULTS

In the following section, we show the statistical results of 94 experiments of sections III-B, III-C,III-D and III-E. Fig.20a shows the reverse positioning errors when the robot arrives at a desired position. It can be seen that, when using a kinematic controller combined with a visual servoing controller (ViKi-HyCo), the translations on the x-axis show a minor improvement as opposed to when only a visual servoing controller is used. This small difference is due to the fact that in this positioning some object detection frames were lost, since the camera focuses on the ground, having the object in direct line of sight and at close range. Our backward positioning results have an error of 0.0375 ± 0.0334 m for the X axis and 0.0452 ± 0.0601 m for the Y axis.

For the case of the forward positioning experiments with the visual servoing and ViKi-HyCo controllers, Fig. 20b shows that our method has a considerable decrease of the error in the X-axis, this is due to the fact that when the visual



(a) Robot path positioning

(b) Camera velocity

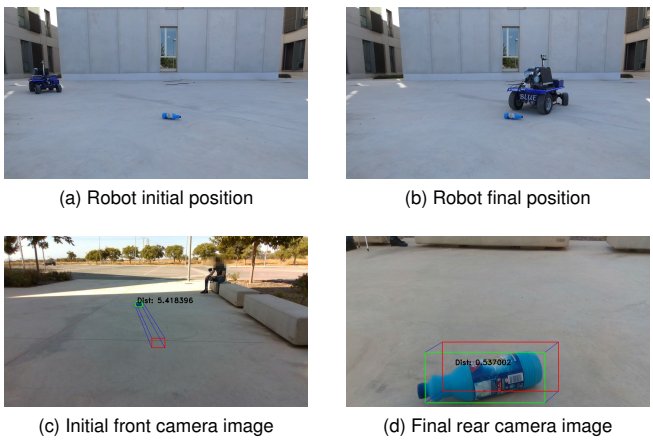
(c) Lineal velocity

(d) Steering angle

(e) X robot translation

(f) Y robot translation

FIGURE 18. Results of the complete positioning experiment using the ViKi-HyCo method. The colored zones represent the state of the experiment. State 1 (green) forward positioning. State 2 (blue) Robot rotation with kinematic controller. State 3 (red) backward positioning.



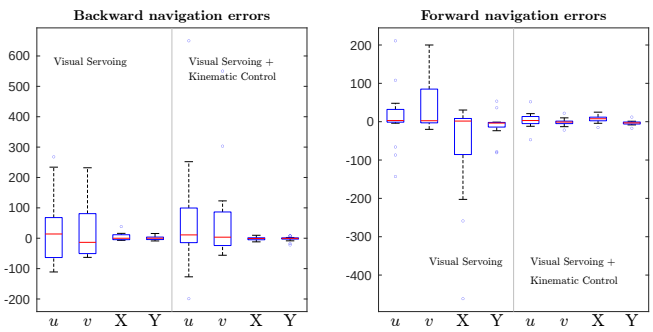
(a) Robot initial position

(b) Robot final position

(c) Initial front camera image

(d) Final rear camera image

FIGURE 19. Initial and final position of the robot in the complete positioning experiment and images from the front and rear cameras.



(a) Results of backward positioning

(b) Results of forward positioning

FIGURE 20. Comparative results of forward and backward positioning experiments using the classic visual servoing controller versus the ViKi-HyCo method. The plane image errors u and v are in pixels and positioning errors X and Y are in centimeters. In each box, the red center line indicates the median, and the lower and upper ends of the box indicate the 25th and 75th percentiles, respectively. The dashed line extends to the most extreme data points, and outliers are individually represented by the blue symbol "o".

servoing controller does not have feature points, obtained from the object detections, the kinematic controller is used. Additionally, switching from the visual servoing controller to a kinematic controller helps to recover object detections with YOLOv5. Hence, the robot stops are avoided when it performs position maneuvers. In this way, the u,v pixel errors also decrease. The positioning errors in the forward positioning of our method are 0.0911 ± 0.0631 m in the X-axis and 0.0457 ± 0.0437 m in the Y-axis.

When comparing ViKi-HyCo against MBGB-YOLO, the results in section III-D show that our method does not depend on the physical characteristics of the target object of the controller, since it is not necessary to know in advance the dimensions of the desired bounding box. Our system constantly updates the characteristics of the bounding box depending on the current distance to the object and the position in the image plane where we want to place the object. Although MBGB-YOLO positioning errors approach zero, they do not converge because the calculated camera velocities do not decrease due to errors in the image plane. ViKi-HyCo converges to the desired point by constantly updating the desired bounding box.

Finally, by using both positioning experiments in a combined task, the robot positioning error with reference to the target point resulting in an error of 0.0428 ± 0.0467 m in the X-axis and 0.0515 ± 0.0313 m in the Y-axis at the end of the task. In this way, we demonstrate that the proposed method generates the velocities required for the BLUE robot's positioning, in order to position it towards an object that is in the ground plane. These positioning errors are adequate for our

TABLE 1. Total run-time of the process in each iteration

Process	Average time (ms)
Object detection (YOLOv5):	12
Interpolated Point cloud and LiDAR-Camera fusion:	30
Visual Servoing and Kinematic controller:	2
Total	44

experimentation since the task has successfully positioned the BLUE robot with respect to the object in the manipulation zone.

G. VIKI-HYCO RUN-TIME

To validate the execution time of each iteration of the complete process, the times of each of the sub-processes has been measured, and they are indicated in Table 1. The total time of Object detection, Interpolated point cloud, LiDAR-Camera fusion, Visual Servoing and Kinematic controller, result in a time of less than 45 ms. This is guaranteed for real-time in our application, because the camera works at 15 fps and the odometry has a frequency of 20 Hz. It is worth mentioning that the Visual Servoing and Kinematic Controller execution time of 2 ms is the control law calculation time of equation (25), together with equation (26), which prevents abrupt velocity changes in the controller output. In addition, the transition between controllers is despicable, since it is incorporated in the same control law.

IV. CONCLUSION

In this paper, we present a hybrid-control system combining a visual servoing and a kinematic controller for the positioning maneuvers of a non-holonomic robot. As features of the visual servoing controller, we use the detections of a YOLOv5 NN, which detects and generates the current and desired bounding box of domestic waste in outdoor environments. We demonstrate that the union of both controllers is possible by performing several experiments of forward and backward positioning and a combined task in outdoor environments with depth images that can come from RGB-D cameras or from point clouds generated by a LiDAR sensor. Furthermore, our approach demonstrates that it does not need a visual tracker algorithm for object tracking when the camera is in motion or when the image input source is switched, this is achieved with the kinematic controller that locates the detected object in a desired position. Our method achieves a run-time of less than 45 ms, which we consider real-time for our robotic applications. In addition, the method has a small positioning error considered for the applications for which the BLUE robot was designed. ViKi-HyCo allows positioning control for any type of robot as long as the kinematic model of the robot is known and the appropriate sensing sensors are available.

In future projects, we will use a cost mapping system to generate a safe route to the object if there is an obstacle in the scenario that prevents proper navigation [41]. In addition, having solved the positioning to domestic waste in our BLUE robot, we will design a method for handling this domestic waste in outdoor environments with a robotic arm on board the BLUE robot.

REFERENCES

[1] Y. Huang and J. Su, "Visual servoing of nonholonomic mobile robots: A review and a novel perspective," IEEE Access, vol. 7, pp. 134 968–134 977, 2019.

[2] A. A. Nazari, K. Zareinia, and F. Janabi-Sharifi, "Visual servoing of continuum robots: Methods, challenges, and prospects," The International Journal of Medical Robotics and Computer Assisted Surgery, vol. 18, no. 3, p. e2384, 2022.

[3] S. Zhang, Y. Chen, S. Chen, and N. Zheng, "Hybrid a-based curvature continuous path planning in complex dynamic environments," in 2019 IEEE Intelligent Transportation Systems Conference (ITSC). IEEE, 2019, pp. 1468–1474.

[4] A. AbuBaker, Y. Ghadi, A. A. Baker, and Y. Ghadi, "Mobile robot controller using novel hybrid system," International Journal of Electrical and Computer Engineering (IJECE), vol. 10, no. 1, pp. 1027–1034, 2020.

[5] T.-W. Zhang, G.-H. Xu, X.-S. Zhan, and T. Han, "A new hybrid algorithm for path planning of mobile robot," The Journal of Supercomputing, vol. 78, no. 3, pp. 4158–4181, 2022.

[6] S. Mondal, R. Ray, S. Reddy, and S. Nandy, "Intelligent controller for non-holonomic wheeled mobile robot: A fuzzy path following combination," Mathematics and Computers in Simulation, vol. 193, pp. 533–555, 2022.

[7] Z. Fu, A. Kumar, A. Agarwal, H. Qi, J. Malik, and D. Pathak, "Coupling vision and proprioception for navigation of legged robots," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 17 273–17 283.

[8] M. Á. Muñoz-Bañón, F. A. Candelas, and F. Torres, "Targetless camera-lidar calibration in unstructured environments," IEEE Access, vol. 8, pp. 143 692–143 705, 2020.

[9] I. d. L. Páez-Ubieta, E. Velasco-Sánchez, S. T. Puente, and F. A. Candelas, "Detection and depth estimation for domestic waste in outdoor environments by sensors fusion," IFAC-PapersOnLine, vol. 56, no. 2, pp. 9276–9281, 2023.

[10] X. Xiao, B. Liu, G. Warnell, and P. Stone, "Motion planning and control for mobile robot navigation using machine learning: a survey," Autonomous Robots, pp. 1–29, 2022.

[11] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," IEEE Transactions on Robotics, vol. 34, no. 4, pp. 1004–1020, 2018.

[12] C. Li, B. Li, R. Wang, and X. Zhang, "A survey on visual servoing for wheeled mobile robots," International Journal of Intelligent Robotics and Applications, vol. 5, no. 2, pp. 203–218, 2021.

[13] B. Jia and S. Liu, "Switched visual servo control of nonholonomic mobile robots with field-of-view constraints based on homography," Control Theory and Technology, vol. 13, no. 4, pp. 311–320, 2015.

[14] R. Wang, X. Zhang, Y. Fang, and B. Li, "Virtual-goal-guided rrt for visual servoing of mobile robots with fov constraint," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 52, no. 4, pp. 2073–2083, 2021.

[15] N. Tian, A. K. Tanwani, J. Chen, M. Ma, R. Zhang, B. Huang, K. Goldberg, and S. Sojoudi, "A fog robotic system for dynamic visual servoing," in 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019, pp. 1982–1988.

[16] C. Molnár, T. D. Nagy, R. N. Elek, and T. Haidegger, "Visual servoing-based camera control for the da vinci surgical system," in 2020 IEEE 18th International Symposium on Intelligent Systems and Informatics (SISY). IEEE, 2020, pp. 107–112.

[17] P. Arora and C. Papachristos, "Mobile manipulator robot visual servoing and guidance for dynamic target grasping," in International Symposium on Visual Computing. Springer, 2020, pp. 223–235.

[18] Y. Qiu, B. Li, W. Shi, and X. Zhang, "Visual servo tracking of wheeled mobile robots with unknown extrinsic parameters," IEEE Transactions on Industrial Electronics, vol. 66, no. 11, pp. 8600–8609, 2019.

[19] R. P. D. Vivacqua, M. Bertozzi, P. Cerri, F. N. Martins, and R. F. Vassallo, "Self-localization based on visual lane marking maps: An accurate low-cost approach for autonomous driving," IEEE Transactions on Intelligent Transportation Systems, vol. 19, no. 2, pp. 582–597, 2017.

[20] R. Lagneau, A. Krupa, and M. Marchal, "Automatic shape control of deformable wires based on model-free visual servoing," IEEE Robotics and Automation Letters, vol. 5, no. 4, pp. 5252–5259, 2020.

[21] A. Ahmadi, L. Nardi, N. Chebrolu, and C. Stachniss, "Visual servoing-based navigation for monitoring row-crop fields," in 2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2020, pp. 4920–4926.

[22] E. Velasco-Sánchez, L. F. Recalde, B. S. Guevara, J. Varela-Aldás, F. A. Candelas, S. T. Puente, and D. C. Gandolfo, "Visual servoing nmpc applied to uavs for photovoltaic array inspection," IEEE Robotics and Automation Letters, vol. 9, no. 3, pp. 2766–2773, 2024.

[23] B. Griffin, V. Florence, and J. Corso, "Video object segmentation-based visual servo control and object depth estimation on a mobile robot," in Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2020, pp. 1647–1657.

[24] B. A. Griffin and J. J. Corso, "Depth from camera motion and object detection," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 1397–1406.

[25] D. Bersan, R. Martins, M. Campos, and E. R. Nascimento, "Semantic map augmentation for robot navigation: A learning approach based on visual and depth data," in 2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE). IEEE, 2018, pp. 45–50.

[26] D. H. Dos Reis, D. Welfer, M. A. De Souza Leite Cuadros, and D. F. T. Gamarra, "Mobile robot navigation using an object recognition software with rgbd images and the yolo algorithm," Applied Artificial Intelligence, vol. 33, no. 14, pp. 1290–1305, 2019.

[27] Y. Hu, G. Liu, Z. Chen, and J. Guo, "Object detection algorithm for wheeled mobile robot based on an improved yolov4," Applied Sciences, vol. 12, no. 9, p. 4769, 2022.

[28] J. W. Wu, W. Cai, S. M. Yu, Z. L. Xu, and X. Y. He, "Optimized visual recognition algorithm in service robots," International Journal of Advanced Robotic Systems, vol. 17, no. 3, p. 1729881420925308, 2020.

[29] H. Liu, D. Li, B. Jiang, J. Zhou, T. Wei, and X. Yao, "Mgbm-yolo: a faster light-weight object detection model for robotic grasping of bolster spring based on image-based visual servoing," Journal of Intelligent & Robotic Systems, vol. 104, no. 4, pp. 1–17, 2022.

[30] I. del Pino, M. A. Munoz-Banon, S. Cova-Rocamora, M. A. Contreras, F. A. Candelas, and F. Torres, "Deeper in blue," Journal of Intelligent & Robotic Systems, vol. 98, no. 1, pp. 207–225, 2020.

[31] P. I. Corke and O. Khatib, Robotics, vision and control: fundamental algorithms in MATLAB. Cham: Springer International Publishing, 2017, vol. 73.

[32] H. Wang, C. Wang, C.-L. Chen, and L. Xie, "F-loam : Fast lidar odometry and mapping," in 2021 IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS (IROS), ser. IEEE International Conference on Intelligent Robots and Systems. IEEE; RSJ, 2021, pp. 4390–4396, iEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), ELECTR NETWORK, SEP 27-OCT 01, 2021.

[33] E. P. Velasco-Sánchez, M. Á. Muñoz-Bañón, F. A. Candelas, S. T. Puente, and F. Torres, "Lilo: Lightweight and low-bias lidar odometry method based on spherical range image filtering," arXiv preprint arXiv:2311.07291, 2023.

[34] B. Espiau, F. Chaumette, and P. Rives, "A new approach to visual servoing in robotics," IEEE Transactions on Robotics and Automation, vol. 8, no. 3, pp. 313–326, 1992.

[35] L. Zhou, Z. Li, and M. Kaess, "Automatic extrinsic calibration of a camera and a 3d lidar using line and plane correspondences," in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 5562–5569.

[36] A. Dhall, K. Chelani, V. Radhakrishnan, and K. M. Krishna, "Lidar-camera calibration using 3d-3d point correspondences," arXiv preprint arXiv:1705.09785, 2017.

[37] E. J. Kirkland, Bilinear Interpolation. Boston, MA: Springer US, 2010, pp. 261–263.

[38] P. Tornero, S. Puente, and P. Gil, "Detection and location of domestic waste for planning its collection using an autonomous robot," in 2022 8th International Conference on Control, Automation and Robotics (ICCAR). IEEE, 2022, pp. 138–144.

[39] M. Á. Muñoz-Bañón, E. Velasco-Sánchez, F. A. Candelas, and F. Torres, "Openstreetmap-based autonomous navigation with lidar naive-valley-path obstacle avoidance," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 12, pp. 24 428–24 438, 2022.

[40] N. D. Munoz-Ceballos and G. Suarez-Rivera, "Performance criteria for evaluating mobile robot navigation algorithms: a review." Revista Iberoamericana de Automática e Informática Industrial, vol. 19, pp. 132–143, 2022.

[41] S. Kumar NT, M. Gawande, N. P. B. M, H. Verma, and P. Rajalakshmi, "Mobile robot terrain mapping for path planning using karto slam and gmapping technique," in 2022 IEEE Global Conference on Computing, Power and Communication Technologies (GlobConPT), 2022, pp. 1–4.

EDISON P. VELASCO SÁNCHEZ received the degree in Electronic Engineering and Instrumentation from the University of the Armed Forces ESPE (Ecuador) in 2015 and a Master's Degree in Automation and Robotics from the University of Alicante (Spain) in 2018. He was a research technician in the ARSI research group at the ESPE University and in the Automation, Robotics and Computer Vision Group (AUROVA) of the University of Alicante. He is currently pursuing a Ph.D. degree in AUROVA at the University of Alicante funding by the Regional Valencian Community Government and the Ministry of Science, Innovation and Universities through the grant PRE2019-088069. His research interests include navigation and autonomous localization in UGVs with cameras and LIDAR sensors.

MIGUEL ÁNGEL MUÑOZ-BAÑÓN received his B.S. degree in telecommunications engineering from the University of Alicante in 2016 and his M.S. degree in artificial intelligence with UNED in 2017. He has worked on different knowledge-transfer projects. He was a Research Technician in various public projects at the Signals, Systems, and telecommunications Group (SST) and Automation, Robotics, and Computer Vision Group (AUROVA) at the University of Alicante. He received a Ph.D. degree in 2022 at the University of Alicante, funded by the Regional Valencian Community Government and the European Regional Development Fund (ERDF) through the grant ACIF/2019/088. His research interests include graph-based simultaneous localization and mapping (Graph-SLAM), geo-referencing using aerial imagery, and machine learning techniques for environment perception.

FRANCISCO A. CANDELAS received the Computer Science Engineer and the Ph.D. degrees in the University of Alicante (Spain), in 1996 and 2001 respectively. He is Associate Professor in the University of Alicante since 2003, where he teaches currently courses about Automation and Robotics Sensors in the Degree in Robotic Engineering. Previously, he was in tenure track from 1999 to 2003. Dr. Candelas also researches in the Automation, Robotics and Computer Vision Group (AUROVA) of the University of Alicante since 1998, and he has involved in several research projects and networks supported by the Spanish Government, as well as development projects in collaboration with regional industry. His main research topics are autonomous robots, robot development, and virtual/remote laboratories for teaching.

SANTIAGO PUENTE received the Computer Science Engineer and the PhD degrees in the University of Alicante (Spain), in 1998 and 2003 respectively. He is a full-time lecture and researcher at the University of Alicante since 2003, where he teaches currently in the Degree in Robotic Engineering. From 2013 to 2019, Dr. Puente has been deputy director of Infrastructures and facilities Polytechnic School of the University of Alicante, Furthermore, from 2019 to 2021 he has been Academic Coordinator of BEng Robotics Engineering. Dr. Puente also researches in the Automation, Robotics and Computer Vision Group (AUROVA) of the University of Alicante since 1999, and he has involved in several research projects and networks supported by the Spanish Government, as well as development projects in collaboration with regional industry. His research interests include automation and robotics (intelligent robotic manipulation, robot perception systems, robot imitation learning, field mobile robots), and e-learning.

FERNANDO TORRES was born in Granada, where he attended primary and high school. He moved to Madrid to undertake a degree in Industrial Engineering at the Polytechnic University of Madrid, where he also carried out his PhD thesis. The last year of his PhD thesis he became a full-time lecturer and researcher at the University of Alicante, and he has worked there ever since. He directs the research group "Automatics, Robotics and Computer Vision" founded in 1996 at the University of Alicante. He is a member of TC 5.1 and TC 9.4 of the IFAC, a Senior Member of the IEEE and a member of CEA. Since july 2018 he is coordinator of the area of Electrical, Electronic and Automatic (IEA) of the Spanish Agency of Statal Research (AEI). His research interests include automation and robotics (intelligent robotic manipulation, visual control of robots, robot perception systems, field mobile robots, advanced automation for industry 4.0, artificial vision engineering), and e-learning. Currently, his research focuses on automation, robotics, and e-learning. In these lines, it currently has more than fifty publications in JCR-ISI journals and more than a hundred papers in international congresses. He was Leader Research in several research projects and he has supervised several PhD in these lines of research.