Omnibot: A Scalable Vision-Based Robot Swarm Platform

Zhao Ma, Jiachen Liang, Hongyi Wang, Shiliang Guo, Peidong Huo, Yin Zhang, Shiyu Zhao

Abstract— **Introducing vision sensing into swarms presents three challenges for developing robot platforms. First, the vision system requires a wide field of view to perceive surrounding robots. Second, vision algorithms demand high computational power, which poses a challenge for real-time vision-in-the-loop simulation. Third, as the swarm scale increases, managing the system becomes increasingly demanding. The main contribution of this paper is the development of a novel mobile robot swarm platform to overcome these challenges. 1) Each robot features a 360-degree omnidirectional vision system comprising four cameras, allowing each robot to detect and interact with the surrounding robots. 2) It has a novel ROS-based distributed swarm simulation system, which can effectively utilize the onboard computational resources of multiple robots to achieve parallel vision-in-the-loop simulation. 3) It features a novel swarm management system that allows real-time monitoring and debugging of multiple robots. These innovative designs provide a novel swarm platform that can facilitate the study of versatile vision-based swarm tasks.**

Fig. 1: Omnibots.

I. Introduction

Robotic swarm systems, encompassing rich scientific research problems and extensive potential applications, have been studied extensively over the past two decades. Existing studies primarily fall into two categories: theoretical research and practical platform development. These two categories complement each other, with practical platforms serving as crucial tools to validate theoretical algorithms. Current robot swarm platforms can be classified into aerial [1]–[3], ground [4], [5], surface [6], [7], and underwater [8], [9] systems.

Considering ground robot swarms exhibit higher advantages in terms of experimental cost and convenience, we are interested in ground robot swarm platforms. The existing ground swarm platforms can be classified into two categories according to whether they emphasize on vision systems. For example, platforms including Kilobot [4], Droplet [10], Alice [11], Zooids [12], e-puck [13], and HeRo2.0 [14] do not focus on vision-based swarm tasks, although they may provide expandable camera boards. By contrast, the platforms like Turtlebot3 [15], Khepera IV [16], and ROSbot2.0 [17] integrate vision systems onboard. However, many challenges such as omnidirectional vision, vision-in-the-loop simulation, and swarm management are not specifically addressed by these platforms.

Our research group has long been dedicated to both theoretical research [18] and platform development [19]–[21] of multi-robot systems. In particular, we recently developed a swarm system consisting of 50 ground robots [5]. By the proposed control strategy, this system can efficiently accomplish various swarm tasks with high adaptability. However, each robot in this swarm lacks essential onboard modules such as vision sensing, inter-robot communication, and computing resources. In this paper, we will extend this system by proposing a second-generation robot swarm platform.

The swarm platform proposed in this paper is illustrated in Fig. 1 and Fig. 2. Each robot comprises essential onboard modules for vision, navigation, computation, and communication, making it suitable for diverse and complex swarm tasks. Here, each robot is called *Omnibot* due to the following reasons. First, it contains an omnidirectional vision system. Second, it can execute omnidirectional movement. Third, its shape is a disc with omnidirectional symmetry.

In comparison to existing robot swarm systems, the innovation of omnibots lies in three aspects.

1) Inter-robot information acquisition is essential for swarm systems: each robot needs to acquire the states of its neighboring robots, such as their positions and velocities. Currently, wireless communication is the primary means for inter-robot information acquisition. However, wireless communication lacks scalability to large numbers of robots due to the need for time or frequency division communication. By contrast, vision sensing provides a scalable solution since vision is a passive sensing approach, which avoids mutual interference among robots. In nature, vision is also the crucial sensory mechanism in many large-scale animal swarms [22]. Another advantage of vision is that it can provide rich information about the environment, serving as a crucial foundation for enhancing the intelligence of the system. In the past, it was challenging to develop vision algorithms that can effectively acquire the needed information. However, recent advancements in deep learning [23]–[25] and visual

This work is supported by the Dean's special projects in the School of Engineering at Westlake University (Grant No. WU2023B013).

All the authors are with the Intelligent Unmanned Systems Laboratory in the School of Engineering at Westlake University, Hangzhou, China. E-mail: {mazhao, liangjiachen, wanghongyi, guoshiliang, huopeidong, zhangyin, zhaoshiyu}@westlake.edu.cn

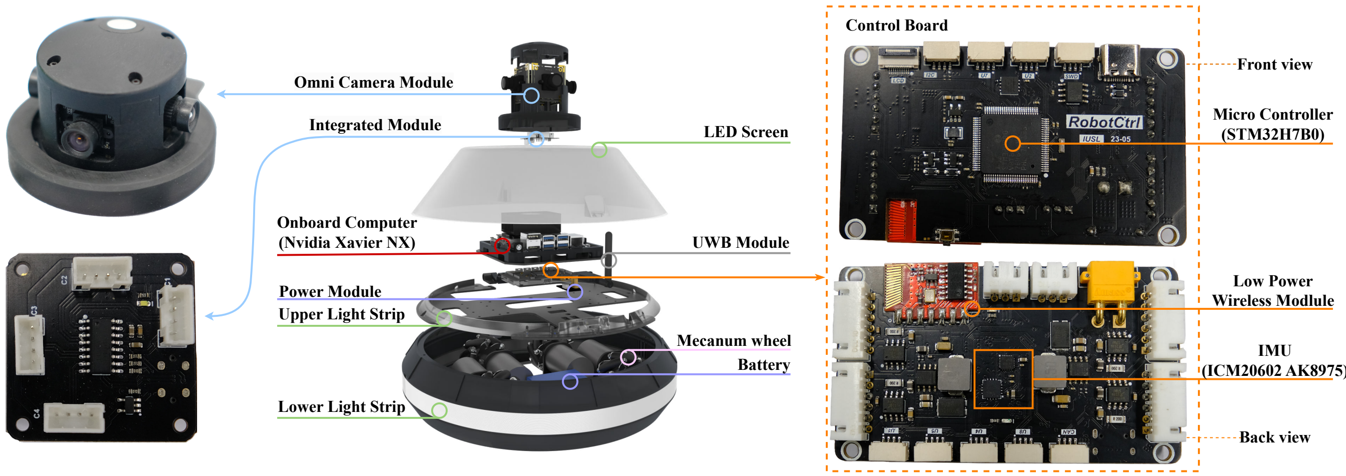


Fig. 2: The hardware architecture of omnibots.

language models [26], [27] present new opportunities for vision perception in robot swarms.

The first novelty is that each omnibot provides an omnidirectional vision system composed of four low-cost cameras. An omnidirectional vision system is crucial since every robot needs to continuously perceive other robots in the swarm for collaboration or collision avoidance. However, most of the existing swarm platforms still lack omnidirectional vision systems, limiting the perception ability of each robot. With omnibots, researchers can develop various vision algorithms to achieve scalable and intelligent swarm systems. On top of omnibots, we have also developed preliminary target recognition and depth estimation algorithms, enabling mutual detection and localization among the robots.

2) Before deploying a swarm system, it is essential to validate the system through simulation. Currently, simulation for swarm systems falls into two categories. The first category does not consider the vision system, assuming that the vision system can already provide the required information. This type of simulation, which focuses primarily on dynamics and control is usually very efficient, enabling the handling of large-scale swarm systems [28]. However, the disadvantage is that it cannot validate vision systems. The second category incorporates vision sensors and vision algorithms into simulation. For instance, in the AirSim/Unreal simulation environment, each robot can execute vision algorithms to process captured images of the virtual reality environment [21]. The advantage is its ability to simulate the vision system with high fidelity, but the disadvantage is a high computational load since vision algorithms usually consume much more time than simulating robot control or dynamics. More importantly, the overall computational load increases rapidly as the number of robots increases, usually making the simulation too slow to be practical.

The second novelty of this work lies in the design of a hardware-in-the-loop swarm simulation system, which can efficiently handle large-scale vision-based swarm simulation. In particular, we establish digital twins of real omnibots in the Gazebo simulation environment. Cameras in the simulation can capture images of the virtual-reality scene and these images are then transferred to and processed on real omnibots. This approach not only allows for parallel processing, significantly improving the simulation efficiency to achieve near-real-time simulation but also fully utilizes the onboard computational resources of omnibots without the need for additional parallel computers.

3) In practical experiments, the time required for maintenance and operation increases rapidly as the swarm scale increases. This is due to the need for numerous operations on each robot, including algorithm deployment, program initialization, startup/shutdown, and battery charging. These operations significantly elevate the cost of conducting large-scale swarm experiments. While Kilobot [4] has addressed some of these issues, such as designing an efficient charging system, it lacks onboard vision sensing and processing modules. When robots involve more modules and operations, a sophisticated swarm management system becomes essential.

The third novelty of this work lies in the development of a swarm management system facilitating both hardware and software operations. In particular, the system consists of online debugging, one-click startup/sleep, one-click program loading, wireless charging, and wireless centralized control. These functions greatly contribute to reducing the management costs to achieve large-scale swarm experiments.

Finally, we present a demonstration experiment based on omnibots. In this experiment, three omnibots are used to achieve a vision-based formation control task. This experiment showcases the capabilities and potential of the platform.

II. Single-Robot System

This section introduces the system of a single omnibot.

A. Overview

An omnibot, as illustrated in Fig. 2, features several modules arranged from top to bottom, including an omnidirectional camera module, onboard computer, robot control board, various types of sensors, power system, motion system, and a light display module. The overall design adopts a disc-shaped symmetrical structure to facilitate the

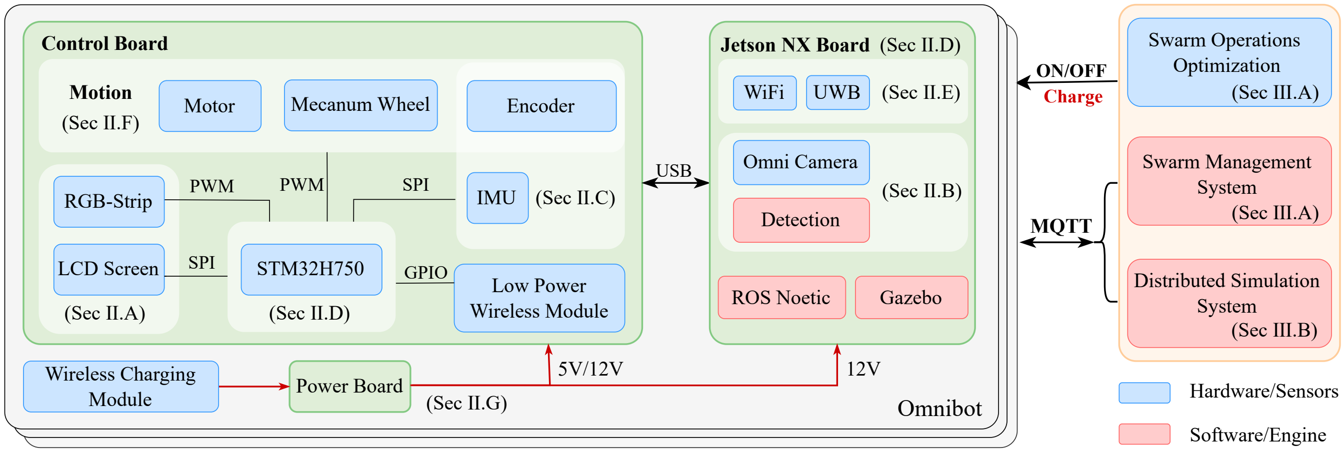


Fig. 3: The functional architecture of omnibots.

unobstructed installation of the omnidirectional camera on the top. The light display module consists of a strip of 131 LED lights. The light display module can be programmed to indicate the different status of the robot.

B. *Vision system*

To provide comprehensive perception capabilities for each robot, we designed an omnidirectional vision module.

Regarding the hardware system, we selected four low-cost cameras, each of which has a horizontal field of view exceeding 90 degrees, to form an omnidirectional vision system with 360-degree perception capability. To achieve a stable connection between the four camera modules and the USB interface of the onboard computer, we designed a connection board to facilitate compactness and maintainability while ensuring efficient data transmission (Fig. 2). We employed a USB port binding mechanism to allocate independent USB interfaces for each camera, enhancing the system's flexibility and configurability. By adopting a generic USB camera driver framework, the majority of cameras could be used in the system without additional drivers. To ensure system stability, we employed ROS time stamps for synchronizing images with sensor data. This design makes camera selection more flexible and provides better support for future system upgrades.

Regarding the algorithms, each omnibot should be able to perform object detection and depth estimation to accomplish various swarm tasks. In terms of object detection, a YOLO-based target detector [23] has been deployed and accelerated in real-time through TensorRT. Object detection is executed on the images captured by each of the four cameras. After that, we use the extrinsic parameters of the cameras to fuse the detection results. The detector is able to detect omnibots and other common objects such as pedestrians. Based on the detection results, the robot can further estimate the distance and velocity of other robots or objects. Of course, the vision and control algorithms are fully customizable: developers should develop their own algorithms according to different tasks.

C. *Other sensors*

To meet the requirements of performing versatile swarm tasks, each omnibot incorporates six types of sensors: a nine-axis IMU, a wheel odometry, a GPS receiver, and a UWB module. Developers can customize sensor data acquisition and data process algorithms based on their specific experimental needs. For instance, the fusion of IMU and odometry data can be jointly used for robot navigation. GPS and UWB modules can be utilized for self-positioning and neighbor communication. The microcontroller board is designed to reserve multiple interfaces such as UART, CAN, and I2C for the convenient expansion of additional sensors.

D. *Computational resources*

To simultaneously address complex vision computations and motion control, we divided the entire computational architecture into two main components: a microcontroller and an onboard computer. These two components complement each other's strengths, ensuring the real-time execution of complex algorithms. The communication between them utilizes USB-FS, providing a bandwidth of 12 M/s.

Due to the page limitation, the details about the onboard computer and microcontroller as well as inter-robot communication, motion system, and power system are omitted in this version. Details can be found in the arxiv version.

III. Swarm Simulation and Management

A. *Swarm management*

To facilitate the operation and maintenance of large-scale swarms, we implement dedicated optimizations on the management of the entire swarm system.

1) Online debugging: To deploy any control or vision algorithms, we usually need to debug online to observe real-time execution results and then make necessary modifications. Therefore, we developed a system to facilitate online debugging over multiple robots. The framework of the system is illustrated in Fig. 4.

This system utilizes ROS as the local message middleware and abstracts overall robot control and sensor data acquisition into ROS topics to reduce code coupling. To accommodate

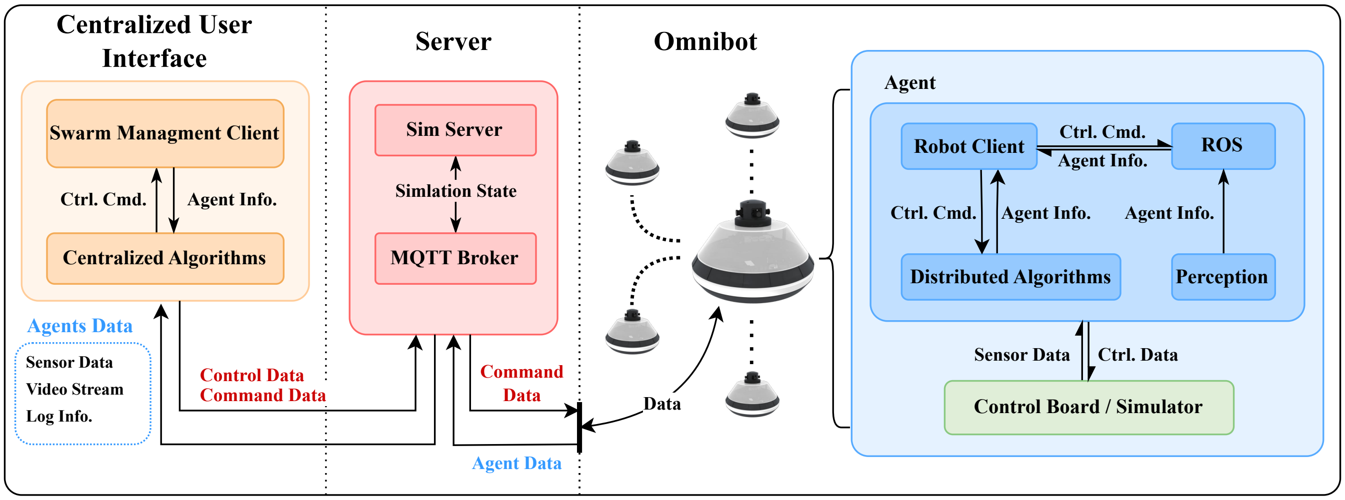


Fig. 4: The architecture of the swarm management and simulation system of omnibots.

the introduction of a visual system and the requirements of multi-robot communication and control, the platform employs MQTT as the wireless message middleware. MQTT's lightweight, reliability, real-time capability, and ease of integration make it an ideal choice for constructing large-scale distributed swarm systems.

Built on this framework, we developed a centralized user interface that allows subscription to camera data from any robot. As shown in Fig. 4, the interface can simultaneously view information from up to 8 cameras of any robot. Additionally, it can publish control commands to manage the motion of multiple robots. Within this framework, the validation of visual algorithms that need deployment becomes straightforward. Developers only need to subscribe to real-time video streams for validation on the centralized interface.

2) One-click activation/deactivation: To enable one-click activation or deactivation of all omnibots, we design a low-power mode for omnibots. In the absence of an activation command, only the low-power wireless module responsible for receiving activation signals and the voltage regulator are operational, resulting in a standby power consumption of only 105uA. Upon receiving the activation command, the entire system starts functioning, powering up the control board first and then initiating the power supply for the onboard computer and motion system.

Based on this design, we have developed two remote signal-emitting devices. One device has a broad beam range, enabling one-click activation or deactivation of all robots. The other, with a narrower beam range and weaker signal, allows targeted activation and deactivation of specific robots.

3) Wireless charging: Effective management of robot charging is an inevitable challenge in large- scale swarm experiments. To address this, we have devised a specialized wireless charging solution, incorporating wireless charging modules and robot charging slots. This system enables unified charging for the swarm robots, streamlining the experimental setup and management processes.

B. Swarm simulation

As the number of robots increases, the computational load of vision-in-the-loop simulation increases rapidly. We address this challenge by deploying a Gazebo simulation platform on the onboard computer of every omnibot, leveraging ROS and MQTT as the underlying message middleware. The structure of this simulation system is illustrated by Fig. 4. Although Gazebo does not support distributed simulation, we have developed a simulation server, depicted in the central module of Fig. 4, to centrally collect the state from each robot and send it back to each robot.

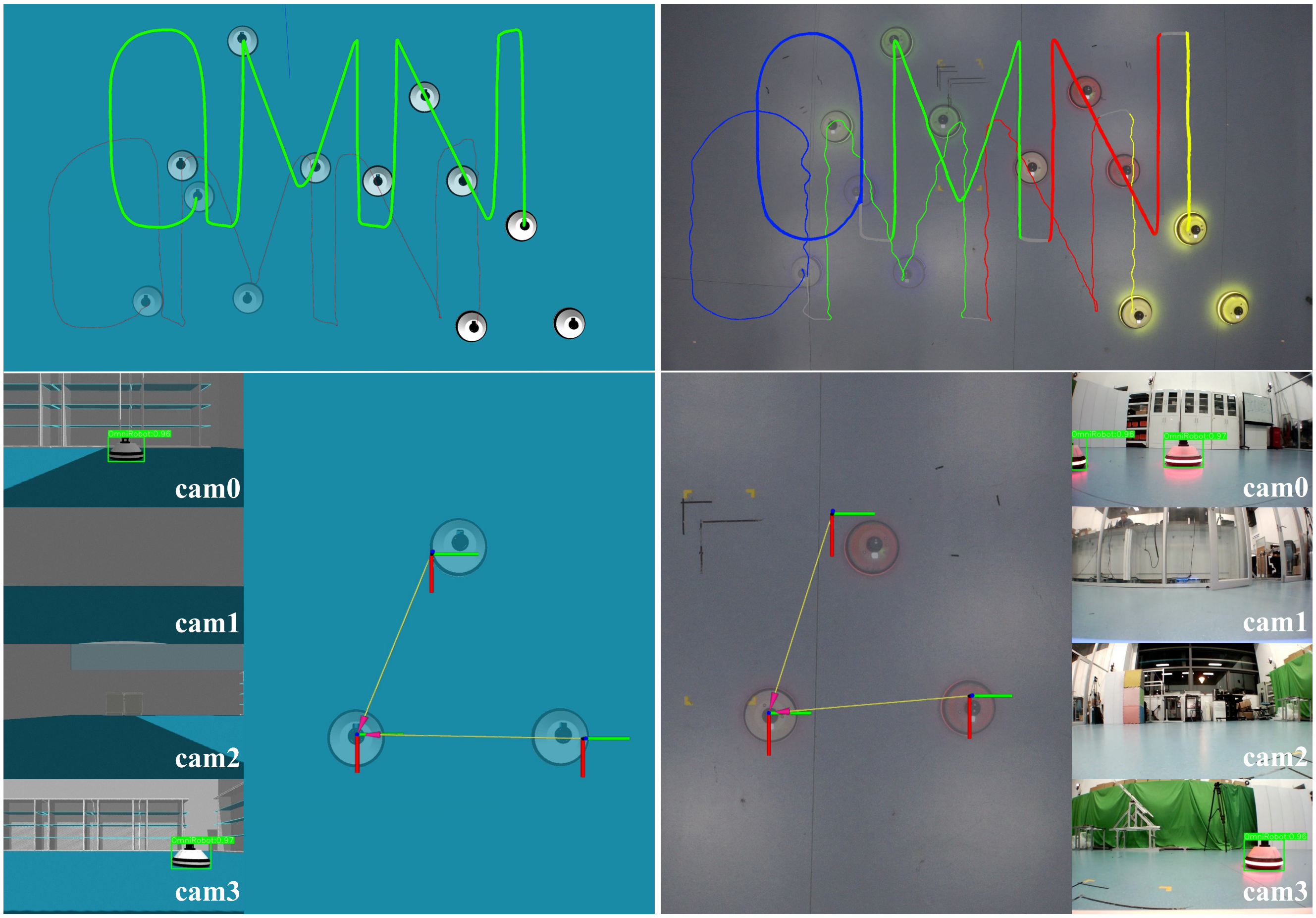
In real-time, each robot's digital twin in the simulation environment uploads its status information, such as position and orientation, to the simulation server. Upon receiving this information, the simulation server organizes and dispatches it. Omnibots that receive the dispatched messages render other omnibots in their simulation environment. This approach ensures scene synchronization for every robot in the same scenario, achieving distributed simulation. Distributed simulation, classified as hardware-in-the-loop, not only validates algorithms but also assesses the feasibility of algorithm deployment. The simulation code and deployment code are fully reusable, facilitating the creation of twin simulations for robotic entities.

IV. Experimental Results

This section shows both simulation and real-world experimental results to validate the proposed system.

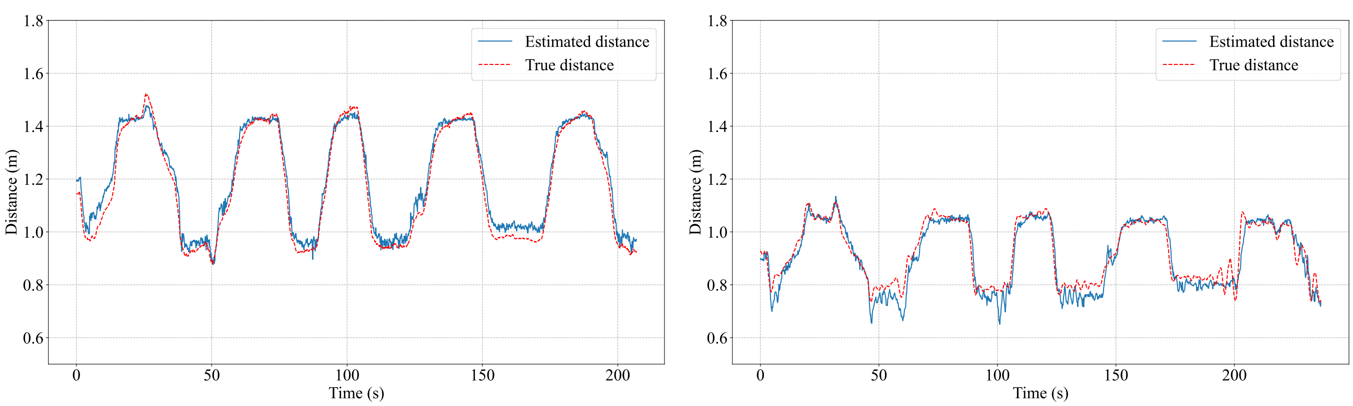
Three omnibots are used in the experiments. One omnibot is designated as the leader, which moves according to the preset motion trajectory in the shape of "OMNI". The other two omnibots act as followers, adjusting their positions based on their visual perception to maintain desired distances from both the other follower and the leader. The three omnibots should maintain an equilateral triangular formation.

Each omnibot uses a Yolov5 network [23] to detect other omnibots. Once an omnibot has been detected in the image, its relative distance can be estimated based on the prior



(a) Simulation (b) Real-world

Fig. 5: Simulation and real-world experimental results. The red-green frameworks in the bottom sub-figures indicate the estimates obtained from vision by the left omnibots.



(a) Simulation (b) Real-world

Fig. 6: Estimation error by vision in the simulation and real-world experiments.

size information of omnibots. In particular, suppose that the physical height of each omnibot is H. The height of the bounding box surrounding the detected omnibot in the image is h. Let f_y be the focal length in the camera's y-axis direction. Then, the distance can be estimated as Hf_y/h. Moreover, the relative bearing can also be calculated from the detected bounding box based on the camera's intrinsic parameters. The details are omitted here since it is straightforward to do that. We employed a simple PID controller to control the inter-robot distance and bearing.

A. *Simulation experiment results*

The simulation results are shown at the top of Fig. 5(a), which presents the trajectories of the three omnibots during the experiment and the formation at different moments. the bottom of Fig. 5(a) shows the images taken by the left omnibot at $t = 170$. Figure 6(a) shows the true and estimated distance between the left omnibot and the leader. As can be seen, the estimation error is around 0.015 m, which is satisfactory for this simple task.

B. Real-world experiment results

We deployed this experiment in a real-world setting. All the algorithms do not need any modifications or re-deployment, demonstrating seamless sim-to-real deployment. The results in the real-world environment are shown at the top of Fig. 5(b). The bottom of Fig. 5(b) shows the images captured by the four cameras of the left omnibot at time $t = 170$.

In the real-world experiment, we posted commands through the swarm management system to set the LED strips on the robots to various colors as the robots form different letter shapes. The distance estimation error is shown in Fig. 6(b), with the ground truth being supplied by the vicon system.

V. Conclusions

This paper introduced a novel robot swarm platform. Such a platform is novel in the following aspects. First, each omnibot has an omnidirectional vision system to support various vision-based swarming tasks. Second, the proposed hardware-in-the-loop swarm simulation system can efficiently handle large-scale vision-in-the-loop swarming simulation. Third, the proposed swarm management system can facilitate both hardware and software operations to reduce the management costs of large-scale swarm experiments. This platform provides a comprehensive infrastructure for study of versatile vision-based swarm tasks.

References

[1] G. Vásárhelyi, C. Virágh, G. Somorjai, T. Nepusz, A. E. Eiben, and T. Vicsek, "Optimized flocking of autonomous drones in confined environments," *Science Robotics*, vol. 3, no. 20, p. eaat3536, 2018.

[2] X. Zhou, X. Wen, Z. Wang, Y. Gao, H. Li, Q. Wang, T. Yang, H. Lu, Y. Cao, C. Xu, *et al.*, "Swarm of micro flying robots in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm5954, 2022.

[3] H. Xu, Y. Zhang, B. Zhou, L. Wang, X. Yao, G. Meng, and S. Shen, "Omni-swarm: A decentralized omnidirectional visual–inertial–uwb state estimation system for aerial swarms," *Ieee Transactions on Robotics*, vol. 38, no. 6, pp. 3374–3394, 2022.

[4] M. Rubenstein, A. Cornejo, and R. Nagpal, "Programmable self-assembly in a thousand-robot swarm," *Science*, vol. 345, no. 6198, pp. 795–799, 2014.

[5] G. Sun, R. Zhou, Z. Ma, Y. Li, R. Groß, Z. Chen, and S. Zhao, "Mean-shift exploration in shape assembly of robot swarms," *Nature Communications*, vol. 14, no. 1, p. 3476, 2023.

[6] B. Hu and H. Zhang, "Bearing-only motional target-surrounding control for multiple unmanned surface vessels," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 4, pp. 3988–3997, 2021.

[7] M. Duarte, V. Costa, J. Gomes, T. Rodrigues, F. Silva, S. M. Oliveira, and A. L. Christensen, "Evolution of collective behaviors for a real swarm of aquatic surface robots," *PloS One*, vol. 11, no. 3, p. e0151834, 2016.

[8] F. Berlinger, M. Gauci, and R. Nagpal, "Implicit coordination for 3D underwater collective behaviors in a fish-inspired robot swarm," *Science Robotics*, vol. 6, no. 50, p. eabd8668, 2021.

[9] J. S. Jaffe, P. J. Franks, P. L. Roberts, D. Mirza, C. Schurgers, R. Kastner, and A. Boch, "A swarm of autonomous miniature underwater robot drifters for exploring submesoscale ocean dynamics," *Nature Communications*, vol. 8, no. 1, p. 14189, 2017.

[10] J. Klingner, A. Kanakia, N. Farrow, D. Reishus, and N. Correll, "A stick-slip omnidirectional powertrain for low-cost swarm robotics: Mechanism, calibration, and control," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 846–851, IEEE, 2014.

[11] G. Caprari and R. Siegwart, "Design and control of the mobile micro robot alice," in *Proceedings of the 2nd International Symposium on Autonomous Minirobots for Research and Edutainment, AMiRE 2003: 18-20 February 2003, Brisbane, Australia*, pp. 23–32, CITI, 2003.

[12] M. Le Goc, L. H. Kim, A. Parsaei, J.-D. Fekete, P. Dragicevic, and S. Follmer, "Zooids: Building blocks for swarm user interfaces," in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pp. 97–109, 2016.

[13] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. Zufferey, D. Floreano, and A. Martinoli, "The e-puck, a robot designed for education in engineering," in *Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions*, vol. 1, pp. 59–65, IPCB: Instituto Politécnico de Castelo Branco, 2009.

[14] P. Rezeck, H. Azpúrua, M. F. Corrêa, and L. Chaimowicz, "Hero 2.0: A low-cost robot for swarm robotics research," *Autonomous Robots*, pp. 1–25, 2023.

[15] R. Amsters and P. Slaets, "Turtlebot 3 as a robotics education platform," in *Robotics in Education: Current Research and Innovations 10*, pp. 170–181, Springer, 2020.

[16] J. M. Soares, I. Navarro, and A. Martinoli, "The Khepera IV mobile robot: performance evaluation, sensory data and software toolbox," in *Robot 2015: Second Iberian Robotics Conference: Advances in Robotics, Volume 1*, pp. 767–781, Springer, 2016.

[17] RadekJarema, "Husarion rosbot 2.0 pro." Website, 2021. `https://wiki.ros.org/Robots/ROSbot-2.0-PRO`.

[18] S. Zhao and D. Zelazo, "Bearing rigidity theory and its applications for control and estimation of network systems: Life beyond distance rigidity," *IEEE Control Systems Magazine*, vol. 39, no. 2, pp. 66–83, 2019.

[19] J. Li, Z. Ning, S. He, C.-H. Lee, and S. Zhao, "Three-dimensional bearing-only target following via observability-enhanced helical guidance," *IEEE Transactions on Robotics*, vol. 39, no. 2, pp. 1509–1526, 2022.

[20] C. Zheng, Y. Mi, H. Guo, H. Chen, F. Chen, J. Jia, Z. Lin, and S. Zhao, "Optimal spatial-temporal triangulation for bearing-only cooperative motion estimation," *arXiv preprint arXiv:2310.15846*, 2023.

[21] Z. Ning, Y. Zhang, J. Li, Z. Chen, and S. Zhao, "A bearing-angle approach for unknown target motion analysis based on visual measurements," *The International Journal of Robotics Research*, pp. 1–20, Feb. 2024.

[22] K. Li, L. Li, R. Groß, and S. Zhao, "A collective perception model for neighbor selection in groups based on visual attention mechanisms," *New Journal of Physics*, 2024.

[23] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, Y. Kwon, K. Michael, J. Fang, Z. Yifu, C. Wong, D. Montes, *et al.*, "ultralytics/yolov5: v7. 0-yolov5 sota realtime instance segmentation," *Zenodo*, 2022.

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[25] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, *et al.*, "Segment anything," *arXiv preprint arXiv:2304.02643*, 2023.

[26] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, pp. 8748–8763, PMLR, 2021.

[27] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Conditional prompt learning for vision-language models," 2022.

[28] J. Li, L. Li, and S. Zhao, "Predator-prey survival pressure is sufficient to evolve swarming behaviors," *New Journal of Physics*, vol. 25, no. 9, p. 092001, 2023.