Proceedings Paper:

Wang, F. orcid.org/0000-0003-2102-8670, Hutabarat, W. orcid.org/0000-0001-7393-7695 and Tiwari, A. orcid.org/0000-0002-6197-1519 (2024) A lightweight target following architecture for indoor mobile robot. In: Proceedings of 2024 IEEE International Conference on Industrial Technology (ICIT). 25th IEEE International Conference on Industrial Technology (ICIT 2024), 25-27 Mar 2024, Bristol, United Kingdom. Institute of Electrical and Electronics Engineers (IEEE) , pp. 1-5. ISBN 9798350340266

https://doi.org/10.1109/icit58233.2024.10540750

A Lightweight Target Following Architecture for Indoor Mobile Robot

Fuli Wang
Department of Automatic Control and Systems Engineering
The University of Sheffield
Sheffield, United Kingdom
fuli.wang@sheffield.ac.uk

Windo Hutabarat
Department of Automatic Control and Systems Engineering
The University of Sheffield
Sheffield, United Kingdom
w.hutabarat@sheffield.ac.uk

Ashutosh Tiwari
Department of Automatic Control and Systems Engineering
The University of Sheffield
Sheffield, United Kingdom
a.tiwari@sheffield.ac.uk

***Abstract*—Manufacturing efficiency and transport operations are being significantly improved by mobile robots. As the implementation of a configurable, lightweight, and state-of-the-art robotic system is required for current manufacturing sectors to maximize the use of mobile robots, this paper presents a software architecture comprised of perception and action systems for human following in real-time. The perception system uses the YOLOv8 computer vision model to identify and estimate the human pose. Additionally, an algorithm is developed using the 3D information from a stereo camera to determine which target and whether to follow. Once the perception system perceives the desired target information, the action system can control and coordinate the robot using the ROS. Experimental results from the physical robot demonstrate the feasibility of the system architecture.**

Keywords—mobile robot, perception, software architecture, target following

I. INTRODUCTION

While the story of mobile indoor robot navigation is increasingly intellectualized, it relies on the performance of sensors, computing resources, and artificial intelligence algorithms. To meet the needs of different scenarios, such as warehousing, logistics, and household, mobile robots, especially wheeled robots, are often equipped with sensors and software systems to realize their specific functions. The target following is one crucial function for automated guided vehicles and service robots, which comprises perception and action systems. Currently, Simultaneous Localization and Mapping (SLAM) is a broad perception technology for autonomous mobile robots, which can utilize the camera, sonar, or lidar for localization and mapping. For instance, radar SLAM is less affected by environmental changes and can be used outdoors when facing severe weather conditions [1]. For indoor applications, as the lighting condition is good, vision or lidar SLAM is mainly used to perceive the environment. A few literature reviews have been to overview the SLAM in detail recently. Literature [2] presents the significant aspects and trends of SLAM. Survey [3] focuses on the visual SLAM applications that cover vision sensors, machine vision algorithms, and deep learning methods. However, SLAM is considerably demanding regarding sensors and computational resources; for instance, demonstrating a modern visual SLAM usually requires a GPU with even more than 10G of memory.

Lightweight vision-based navigation is the key to cost reduction and widespread application [4], especially for flexible manufacturing or assembly workshops with a well-structured environment, where mobile robots are often used to load goods and tools. Therefore, it is necessary to develop a lightweight and configurable target-following system for such robots. Recently, some vision-based target-following systems have been developed. For example, [5] developed a system comprising visual tracking, target re-detection, and visual servo parts; [6] developed a human following system using the deep learning method and color features. However, these systems fall slightly short in terms of reconfigurability. Therefore, this paper presents a configurable and lightweight architecture for mobile robots.

First, considering the reconfigurability, the Robot Operating System (ROS) is chosen as the main framework for software development. Using the ROS frame system, sensors and robots can be linked smoothly [7]. This provides the flexibility to integrate multiple components and realizes some basic robot movement functions via ROS nodes.

Secondly, the perception system uses a stereo camera based on the YOLOv8 object detection model [8]. This is because the stereo camera can give feedback on the 3D information of the detected objects, and the YOLO, as a lightweight object detection algorithm, plays a crucial role in robot perception systems for variable applications [9], [10]. Additionally, search and filter functions in the perception algorithm have been developed to find and track the expected person, which can guide the robot to follow correctly. This perception algorithm also estimates the human pose, allowing the robot to perform some reaction/action based on the human pose, such as emergency braking.

Finally, the proposed architecture has been integrated and validated on a physical mobile robot: a custom-built Clearpath Dingo indoor mobile robot.

II. SOFTWARE ARCHITECTURE

A. Perception System

The perception process comprises both detection and localization. Any of the state-of-the-art deep learning methods could achieve the task of detecting the following target. Therefore, YOLOv8 has been used in the system, as it is the latest version of YOLO with fast, accurate, and easy-to-use charities. Notably, the algorithm can be run on a modern laptop within high-speed detection times, implying that this lightweight computer vision model can also be implemented on the embedded hardware. The proposed system is designed to detect a person as a guide for robot following. Herein, a human pose estimation model trained based on the COCO

This work was supported by Innovate UK through the project: Landing Advances for a New Decade One "LANDOne" (Project number: 10002411).

Keypoint dataset [11] has been used in the algorithm. Therefore, the robot can detect/localize the human and estimate the human pose.

A stereo camera was used to perceive the environment to fulfill the detection and localization requirements. Specifically, the camera can obtain 2D images and 3D point clouds from the environment. Once the camera inputs this information to the perception system, the YOLO algorithm can process the 2D image for human detection and pose estimation. Then, the camera can access the 3D point cloud based on the 2D information to localize the detected human.

While the perception system is ready to detect and localize the human, the system needs to guide the robot further to find the right person to follow if there is more than one person in the robot's view. As shown in Fig.1, assuming the robot faces two or three persons, the perception system can perceive each person's location and default the one closer to the robot's central view as the target to follow.

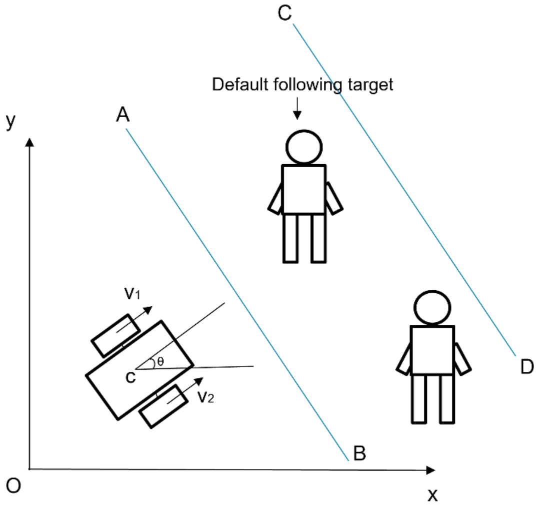

Fig. 1. Diagram of a mobile robot with detected targets.

As shown above, to enhance security, the robot executes the movement command only when the target is located within the area where 'ABCD' is defined. Herein, the 'ABCD' is defined as the following area. This prevents collisions and the robot from trying to follow a target that is too far away. Besides determining the following target, the perception system can also determine whether the target is located on the left or right side of the robot based on its coordinates, as shown in Fig. 2. For example, when the target is within the 'ECF' area, the robot only needs to move forward. Otherwise, the robot will first change direction (perform a left/right turn operation). As a result, 'ECF' is defined as the forward area.

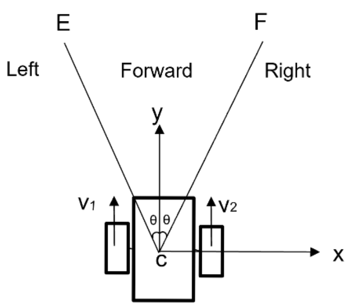

Fig. 2. Robot perceives the direction of the target.

Additionally, the perception system can estimate the human pose. Therefore, once the person shows the robot a "stop" pose, the robot will abort all movement commands. Herein, a pose of arms crossed in front of the chest is used as a 'stop' sign. The algorithm below recorded the filtering and searching operations based on the stereo camera's input.

Algorithm: The perception algorithm with filtering and searching operations

Require: image, point cloud // *from the camera*
Initialize the following and forward areas
model() // *load the YOLOv8 model*
repeat
  results = model(image) // *all detected objects*
  Transfer the results into 3D coordinates using point cloud
  if (there are persons in the following area)
    Select a person as the guide who is closer to the robot
  end if
  if (the guide is located in the turning area)
    Send the required steering command to the action system
  elif(the guide is located in the forward area)
    Send the forward command to the action system
  end if
  Update the image and point cloud from the camera
until cannot detect the desired guide or detect a stop pose

B. Action System Based on the Feedback of Perception

As this paper provides a perception system that can be used as a visual servo for robot control, any control algorithms/methods would be used to drive the robot. This paper uses essential ROS functions/nodes to adjust/publish the robot's linear and angular velocity to achieve movement. Specifically, if a turn left/right command is obtained, the action system will publish a positive/negative angular velocity value via the ROS node. The forward/backward movement can also be realized by publishing positive/negative linear velocity.

Note that this paper focuses not on precise control algorithms; the proposed action system ensures that the robot can perform basic steering and forward/backward motions. Moreover, each steering is performed at a fixed angle θ, and the forward movement is done at a limited speed while avoiding collision with the target. Each motion is performed or changed based on real-time feedback from the perception system, realizing the dynamic following of the target. A flowchart of the action system is shown in Fig.3.

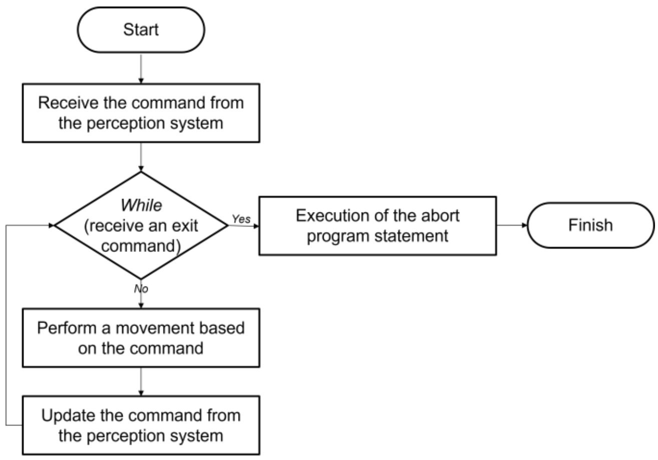

Fig. 3. Flowchart of the action system.

As shown in Fig.3, the action system follows the command from the perception system. If the perception system cannot search for an expected guide, it will not give any order to the action system, and the robot will stay in its current position.

C. *The Overall System Architecture*

As shown in Fig.4, the illustrated perception-action system architecture comprises the user interface, perception, and action parts. To simplify the use of the system, the user interface provides a few functions to initialize the robot, launch the perception system along with the camera, perform the guide-following loop, and exit the system.

The proposed robotic system was applied to a Dingo indoor mobile robot. As shown in Fig.5, the mobile robot comprises a ZED stereo camera for validating the proposed system architecture.

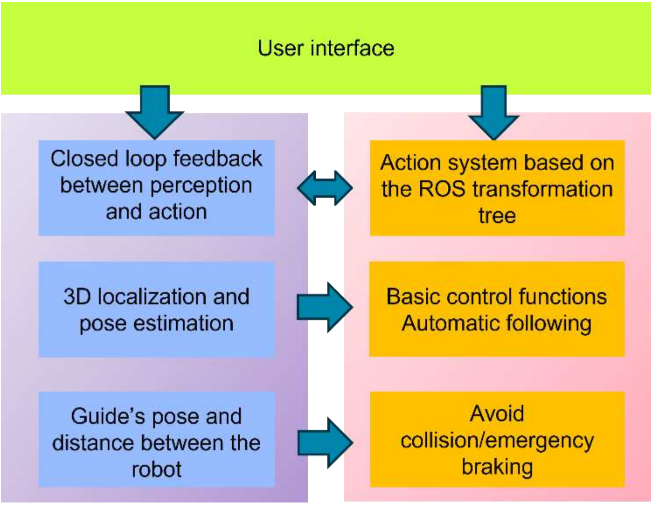

Fig. 4. Overview of the Robotic system architecture.

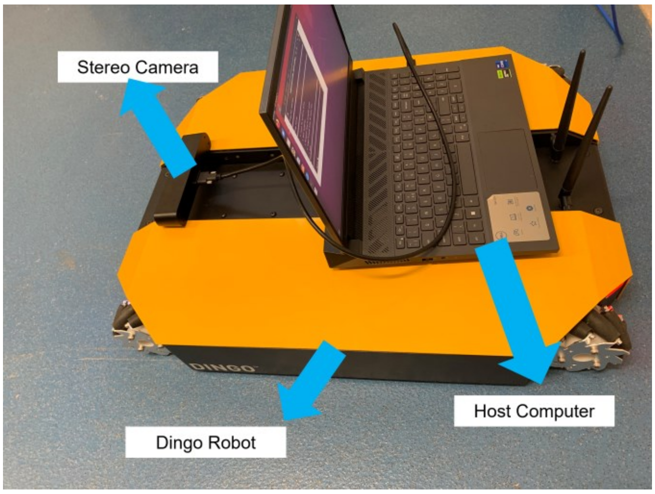

Fig. 5. Dingo indoor mobile robot with a stereo camera.

III. VALIDATION OF PROPOSED ARCHITECTURE

A. *Perception Algorithm Validation*

To illustrate how the perception system works, the system was launched on a laptop to detect a person with different positions and poses. As shown in Fig.6, the person gradually moved far away from the robot/camera and gave a stop pose at the end.

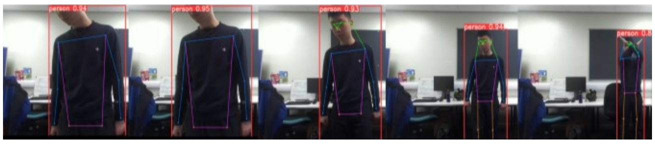

Fig. 6. Person detection and pose estimation process in real-time.

To show the feedback from the perception system more clearly, as shown in Fig.7, a few frames of the above process were extracted with the perception output.

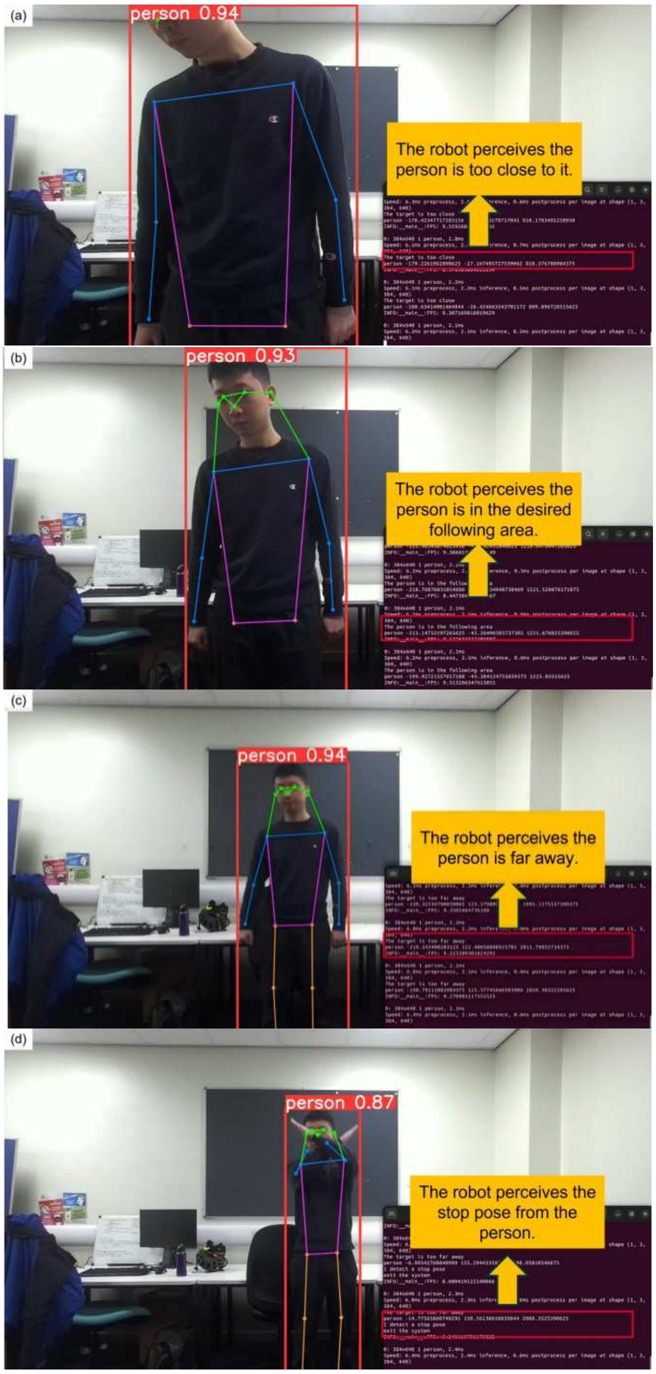

Fig. 7. Series of recognition results from the perception system.

From the first three pictures, we can see that the perception system can perceive the person's orientation and distance (3D coordinates value) to the robot, guiding the robot on whether to follow or not. Similarly, when the robot moves, it can perceive whether the person is in front, left, or right. The last picture of Fig.7 illustrates the pose estimation of the perception system. Once it detects a 'stop' pose, the robot's action system can exit immediately.

Note that the defined 'following area' can be set based on the environment. The above testing was executed in an office room; therefore, in the above testing, the robot was set to follow a person who was within 0.5-2 m of it. Thus, the robot can keep following the guide who walks within this range.

B. *Error analysis*

Once the robotic system runs, the perception system will keep updating the target's position in real-time. The action system will execute each movement based on the target's current status. To analyze the system's error, the target person and robot were allowed to remain motionless, and the target position points obtained for the action system were recorded and plotted in Fig. 8. Since the mobile robots perform translational movements on the floor, all plots herein were shown in 2D coordinates.

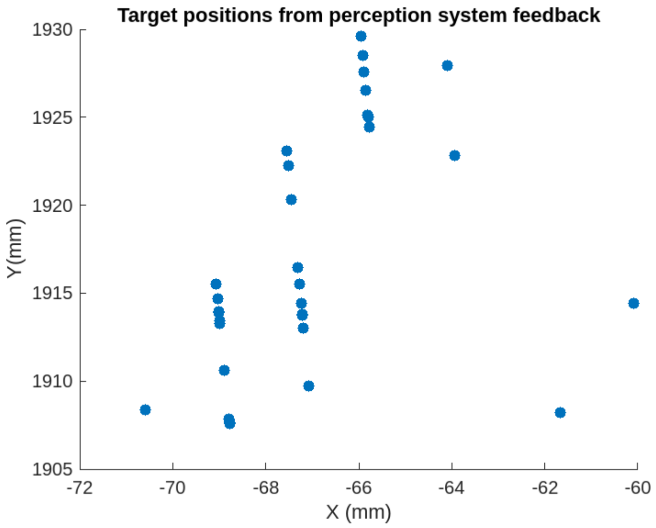

Fig. 8. Target points obtained from the perception system.

As the target person and robot were kept stationary, all recorded points should ideally be the same. However, this is hard to reach due to the presence of errors. As we can see from the above diagram, the distance between the two furthest apart points in the X-direction is 10.15 millimeters, while in the Y-direction, the gap is 22 millimeters.

Note that this error is the result of the feedback from the perception system and may vary if a different camera is selected. In addition, if there are still errors in the robot's hardware, the final error results would be superimposed.

C. *Validation of the Perception-action System on the Robot*

To show how the proposed robotic system works on the physical robot. The perception-action system was integrated to test the target-following experiments. The proposed system was implemented on a laptop with an 8G GPU. As the host computer, this laptop launched the perception-action system and communicated with the Dingo robot via ROS nodes. One experiment was taken as an example in Fig.9. In this experiment, the robot could follow a guide in a narrow aisles. The robot was set to follow a person within 0.8-4.5 m from it.

As shown in Fig.9, the proposed robotic system has been launched, and the robot perspective is shown on the laptop screen. As the robot perceives a guide from its perspective, the guide's 3D information is constantly updated. The action system can access the data and drive the robot to approach the guide.

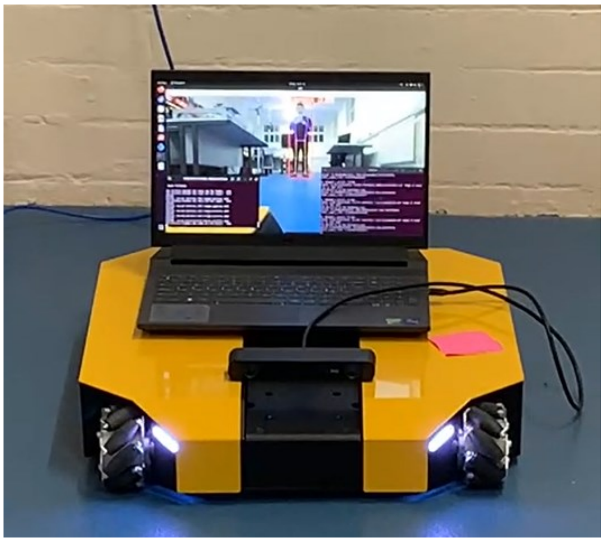

Fig. 9. Picture of the robotic system in operation.

Fig. 10. Series of experiment results from the perception system.

Additionally, as can be seen from the robot's perspective (Fig. 10), the robot can approach the detected person properly using the proposed system. A video of the experiment process might be found at: https://youtu.be/lT3UhhUMGnM.

IV. CONCLUSION

This paper presented a lightweight perception-cation system for human-following mobile robots in a well-structured indoor environment. The perception system comprises a state-of-the-art computer vision model and an algorithm with filtering and searching operations, which can make the robot perceive the human's orientation, distance, and pose. Then, the system can guide the robot to start following the human or stop based on the perception system feedback in real-time. The experimental results demonstrated a certain level of robustness of the system.

Besides being easy to use, the system is configurable to add new components and functions. Therefore, in future work, the robotic system can add more processes to access other sensors, such as lidar and IMU, to improve the accuracy of positioning and the ability to avoid obstacles. Additionally, more poses of humans will be added for the perception system to detect. Thus, it can make the robot more intelligent to do other reactions/movements based on the human poses.

REFERENCES

[1] Z. Hong, Y. Petillot, A. Wallace, and S. Wang, "RadarSLAM: A robust simultaneous localization and mapping system for all weather conditions," *Int J Rob Res*, vol. 41, no. 5, pp. 519–542, 2022, doi: 10.1177/02783649221080483.

[2] H. Taheri and Z. C. Xia, "SLAM; definition and evolution," *Eng Appl Artif Intell*, vol. 97, p. 104032, 2021, doi: https://doi.org/10.1016/j.engappai.2020.104032.

[3] I. Abaspur Kazerouni, L. Fitzgerald, G. Dooly, and D. Toal, "A survey of state-of-the-art on visual SLAM," *Expert Syst Appl*, vol. 205, p. 117734, 2022, doi: https://doi.org/10.1016/j.eswa.2022.117734.

[4] T. Ran, L. Yuan, and J. b. Zhang, "Scene perception based visual navigation of mobile robot in indoor environment," *ISA Trans*, vol. 109, pp. 389–400, 2021, doi: https://doi.org/10.1016/j.isatra.2020.10.023.

[5] M. Zhang, X. Liu, D. Xu, Z. Cao, and J. Yu, "Vision-Based Target-Following Guider for Mobile Robot," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 12, pp. 9360–9371, 2019, doi: 10.1109/TIE.2019.2893829.

[6] R. Algabri and M.-T. Choi, "Deep-Learning-Based Indoor Human Following of Mobile Robot Using Color Feature," *Sensors*, vol. 20, no. 9, 2020, doi: 10.3390/s20092699.

[7] L. Geer, D. Gu, F. Wang, V. Mohan, and R. Dowling, "Novel Software Architecture for an Autonomous Agricultural Robotic Fruit Harvesting System," in *2022 27th International Conference on Automation and Computing (ICAC)*, 2022, pp. 1–6. doi: 10.1109/ICAC55051.2022.9911161.

[8] G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics." Jan. 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[9] G. Liu, Y. Hu, Z. Chen, J. Guo, and P. Ni, "Lightweight object detection algorithm for robots with improved YOLOv5," *Eng Appl Artif Intell*, vol. 123, p. 106217, 2023, doi: https://doi.org/10.1016/j.engappai.2023.106217.

[10] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A Review of Yolo Algorithm Developments," *Procedia Comput Sci*, vol. 199, pp. 1066–1073, 2022, doi: https://doi.org/10.1016/j.procs.2022.01.135.

[11] T.-Y. Lin *et al.*, "Microsoft COCO: Common Objects in Context." 2015.