An Open-source Software-hardware Integration Scheme for Embodied Human Perception in Service Robotics

Iaroslav Okunevich, Vincent Hilaire, Stéphane Galland, Olivier Lamotte, Yassine Ruichek, Zhi Yan

An Open-source Software-hardware Integration Scheme for Embodied Human Perception in Service Robotics

Iaroslav Okunevich, Vincent Hilaire, Stéphane Galland, Olivier Lamotte, Yassine Ruichek, Zhi Yan*

***Abstract*— Perception of human beings is one of the basic capabilities of service robots and is the prerequisite for interaction between robots and humans. Although enabling hardware and software technologies have made great strides, there are not many open-source solutions that organically integrate the two. To address this shortfall, this paper introduces an open-source scheme of hardware and software integration for robotic embodied human perception. The embodied entity includes a robot chassis, a computing unit based on ARM architecture, a 3D lidar, a 2D lidar and four RGB-D cameras for robot exterior perception, a display panel for human-robot interaction, a set of LED lights to show the robot's status and a sonar strip for low-level obstacle avoidance. The perception software is fully based on the Robot Operating System (ROS) which allows high modularity, fully deployed to the embodied entity and running at a rate of 30 Hz. The entire integration solution is very portable and publicly available at `https://github.com/Nedzhaken/human_aware_navigation`.**

I. Introduction

Embodied perception of humans is one of the capabilities that service robots need to possess [1], [2]. Its enabling technology includes hardware, software, and how to properly integrate the two. Hardware mainly includes sensors such as cameras and lidars for exteroception, as well as computing units that process sensory data. Cameras can provide rich semantics, but are sensitive to lighting conditions and cannot provide accurate distance information. While the capabilities of lidar are just the opposite [3]. Therefore, multimodal perception has advantages over unimodal perception, but the challenge lies in how to integrate them effectively [4]. Typical considerations include the field-of-view (FoV, i.e., the covered sensing area) of different sensors, the information redundancy of different modalities, the complementarity and interactivity between sensor capabilities, and even how to resolve conflicts between them [5]. To have an intuitive understanding, Fig. 1 shows our instrumented robot. The robot is a prototype for navigation in public spaces with large-scale long-term human detection requirements.

On the other hand, the computing power of embedded devices and the rapid development of GPUs make it possible to truly realize robot embodied intelligence. Therefore, the development of robot software systems is increasingly focused on deployability on CPU-GPU hybrid platforms. Furthermore, in the face of increasingly complex embodied entities, high modularization and loose integration are the trends.

This work was supported by the Bourgogne-Franche-Comté regional research project LOST-CoRoNa.

All authors are with UTBM, CIAD UR 7533, F-90010 Belfort, France. `firstname.lastname@utbm.fr`

*Corresponding Author.

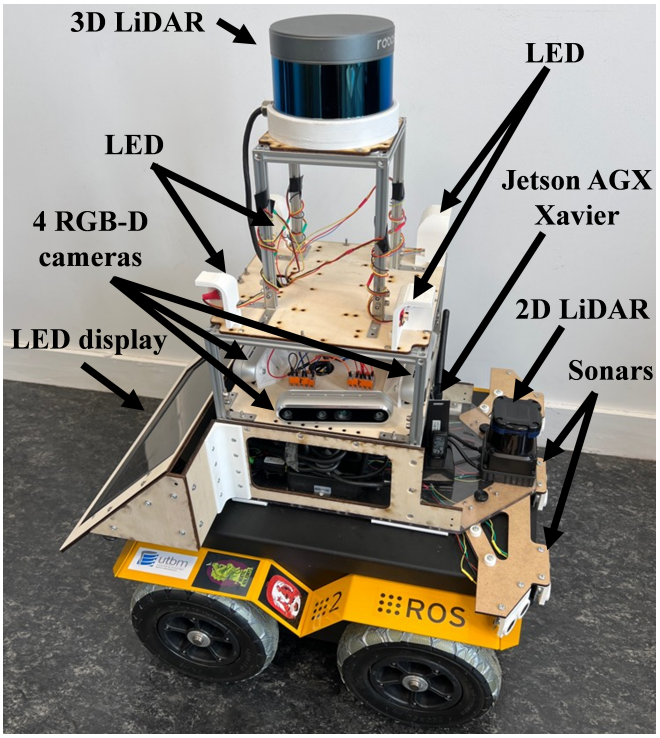


Fig. 1. Our instrumented mobile robot for human-aware navigation in public spaces.

However, what principles should be adopted to modularize the system, how to effectively define the interfaces between different modules, avoid excessive software encapsulation, maximize the reusability of modules, and deploy the entire system to the edge, etc. are still open problems.

Judging from the existing technology, on the one hand, there are various hardware and software to support robot embodied human perception, but open-source software-hardware integration solutions are rarely introduced. On the other hand, solutions running on the x64 platform are common, but few run on the ARM platform and can be holistically deployed to the edge. In order to fill this research gap, this paper proposes a software-hardware integration scheme that

- is open-source and highly modular, in which the perception and the control components are physically decoupled, which means that the entire perception system can run independently of the robotic chassis and has very good portability;
- enables the entire robot software system to be deployed to the edge and operate at 30 Hz, where the perception

stack is fully deployed to ARM-based embedded devices, which creates conditions for the true realization of embodied intelligence.

II. Related Work

Software and hardware integration is one of the fundamental issues in robotics. However, despite its self-evident importance, there is relatively little literature devoted to introducing and exploring this topic. Early reported approaches focused on overall system availability [6], subject to the hardware and software at the time. Later, with the development of technology, functionality was considered more in the integration. For example, system integration of a robotic arm with a mobile base was introduced [7], [8], enabling the mobile robot to have object manipulation capabilities. In recent years, some researchers have begun to pay attention to the performance indicators that integrated systems need to meet to eventually form industrial products. For example, an integrated solution for cleaning robots in public places was proposed in [9], in which the reliability of human and dirt detection was emphatically evaluated; the flexibility of coupled integration of position sensors, with the aim of improving the application of self-driving cars in challenging environments, is presented in [10]; and the real-time performance of an integration scheme for forestry robot perception pipelines is evaluated in [11]. In contrast, the integration scheme proposed in this paper considers reliability, flexibility, and real-time in combination, rather than any one aspect among them.

III. Software-hardware Integration

An autonomous mobile robot typically contains four major components: sensors, actuators, computing units, and power supplies. Based on the principle of "not reinventing the wheel", our solution first includes a mature commercial robot chassis, which solves the three problems of actuator, computing unit for robot control, and power supply. The robot chassis is a Clearpath Jackal UGV with four driving wheels. A CPU-based single board computer (SBC) is integrated into the chassis for tasks with real-time requirements, such as robot motion control and communication with essential – typically high-frequency – sensors including four wheel encoders, an inertial measurement unit (IMU) and a 2D lidar. The chassis is powered by a 270-watt-hour lithium-ion battery pack, which also powers additional computing units and sensors, supporting approximately two hours of autonomy for the entire robotic system. Regarding the software and hardware integration of the embodied human perception system, we will next introduce it from five aspects: sensors, computing units, communications, peripherals, and software. As we mentioned before, the perception system is designed with system portability in mind and does not target a specific robot chassis.

A. Sensors

The selection of exteroceptive sensors mainly considers three aspects: 1) long-term robot autonomy, especially the perception system needs to be robust to various lighting conditions; 2) the largest possible range of perception, including breadth and depth, and try to avoid dead spots; 3) acquisition of environmental semantic information for distance information redundancy, especially including color and texture information. In addition, secondary aspects are also considered: 1) low-level and reliable obstacle avoidance; 2) response to boundary conditions such as glass walls. For the above, a multimodal sensory system is required. In our scheme, a 3D lidar, a 2D lidar, four RGB-D cameras and a sonar belt are integrated. Their specifications are shown in Table I, and their sensing coverage areas are shown in Fig. 2.

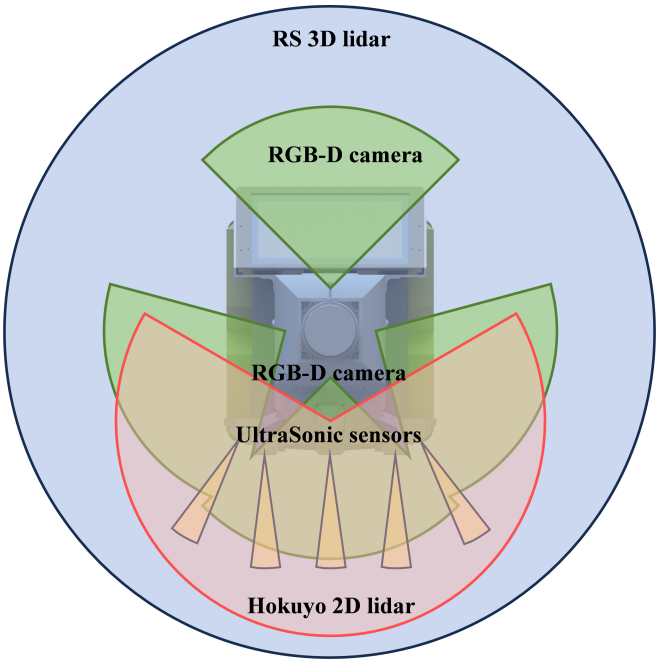


Fig. 2. Schematic representation (not to scale) of sensing areas of the integrated sensors. The RGB-D cameras and the 3D lidar are used for human detection and tracking around the robot. The 2D lidar is applied in simultaneous localization and mapping (SLAM). The sonar belt is used for obstacle avoidance.

TABLE I
Various Sensor Specifications Integrated in Our Scheme

Sensor	Data release frequency	Measure. distance	Horizontal FoV	Vertical FoV
3D lidar	10 Hz	150 m	360°	30°
2D lidar	40 Hz	30 m	270°	-
RGB-D cam.	30 Hz	6 m	87°	58°
Sonar	40 Hz	4 m	15°	-

The 3D lidar (a Robosense RS-LiDAR-16) is mounted on top of the entire perception system, 80 cm above the ground. Like the 2D lidar, the sensor in this modality is not sensitive to lighting conditions and is suitable for day and night use. Since the 3D lidar typically has panoramic scanning capability and detection capability of up to tens of meters, the robot can detect humans or other dynamic objects as early as possible [12], [13], and therefore have more time to react accordingly. The shortcoming of lidar

is that it can only provide a set of sparse coordinate points, which is not enough to understand the environmental context. While some lidars provide intensity information, the value of the intensity is entirely determined by the lidar's internal signal processing unit, which is usually a black box (for commercial reasons) [14]. Therefore, it is necessary to add visual sensors.

The 2D lidar is mounted 30 cm above the ground, facing forward. Integrating this sensor instead of using a one-stop solution based on the 3D lidar, as the lidar installed at a low position is more in line with the needs of SLAM, and can also cover the perception blind spot of the 3D lidar installed at a high position. In addition, the use of 2D lidar, which has higher measurement frequency, accuracy, and resolution than 3D lidar, is also beneficial for localization and collision-free navigation of the robot in the deployment environment. In our case, a Hokuyo UTM-30LX-EW 2D lidar is integrated.

Four Intel RealSense D455 RGB-D cameras are installed 32.7 cm from the ground and placed towards all sides of the robot to provide a quasi-panoramic view. Due to our safety and social interaction requirements for the robot, the left and right cameras are slightly shifted to the front in order to close the blind spot of perception. The negative impact of this setting is to increase the visual blind spots on both sides of the rear. A possible complement would be to add a fifth camera to achieve panoramic coverage, which would also increase the cost of software-hardware integration. An economical and effective solution is to use our multi-modal perception solution so that the robot can rely on the panoramic perception of the 3D lidar together with the back-end information fusion algorithm to achieve the effect of blind filling. Although the cameras are sensitive to lighting conditions, their ability to provide rich environmental semantics and reliable close-range object detection has made them one of the standard components of mobile robots today.

Finally, five HC-SR04 UltraSonic sensors are mounted 23 cm from the ground to enhance collision detection in the forward direction and to cope with some edge cases such as glass walls.

B. Computing Units

In addition to the CPU-based computing unit inside the robot chassis, we integrated a CPU-GPU-based computing unit, Nvidia Jetson AGX Xavier, outside the robot. With peak computing power of up to 32 TOPS and high-speed I/O performance of 750 Gbps, it supports the data transfer and computing resource needs of our human detection and tracking system based on the 3D lidar and RGB-D cameras.

C. Communications

The connectivity between the various components of the robot is shown in Fig. 3. Communications between all hardware components are based on wired connections. As a robotic system, the data transfer rate and communication bandwidth must be fully considered to avoid undue challenges to the algorithm's fault tolerance and to avoid

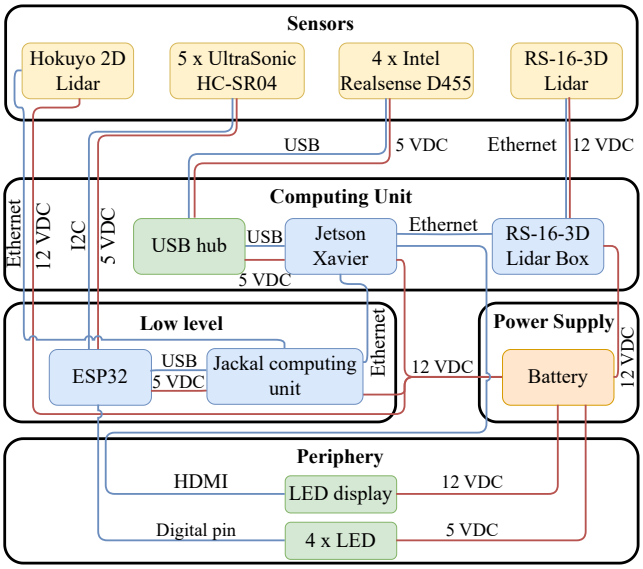


Fig. 3. Schematic diagram of the connections between the various components of the robot. The red and blue lines represent the power and data connections respectively.

catastrophic data delay and loss. Based on this principle, the 2D and 3D lidars are respectively connected to the Jackal SBC and the Nvidia AGX through Ethernet cables to ensure that the sensory data is processed by the corresponding computing units in time. The Jackal SBC and the Nvidia AGX are directly connected via an Ethernet cable and maintain soft time synchronization. Additionally, the four RGB-D cameras are connected to the Nvidia AGX via an USB hub.

D. Peripherals

As shown at the bottom of Fig. 3, the installed peripherals include a set of LED lights to display the status of the robot and a LED display for human-robot interaction.

E. Software

The software system has been fully implemented into ROS [15] Melodic distribution with high modularity, as shown in Fig. 4. It is divided into two parts: human perception and robot control, which are deployed on the two aforementioned computing units with loose coupling.

The human perception stack is built on a tracking-by-detection pipeline. Specifically, human detection is performed on the data generated by the camera and the 3D lidar. Camera-based detection uses off-the-shelf YOLOv2 [16] detector and first obtains the 2D bounding box of people from the RGB image. Then it utilizes the registered depth information to project the 2D bounding box to 3D and calculates the 3D coordinates of the detected human (i.e., the centroid of the 3D bounding box, same hereinafter). The lidar-based detection utilizes the Adaptive Clustering (GPU version) SVM detector [12] to obtain the human 3D coordinates directly from the point cloud. To reduce false alarms, we developed a ROS package to remove background point clouds, which correspond to occupied and unknown

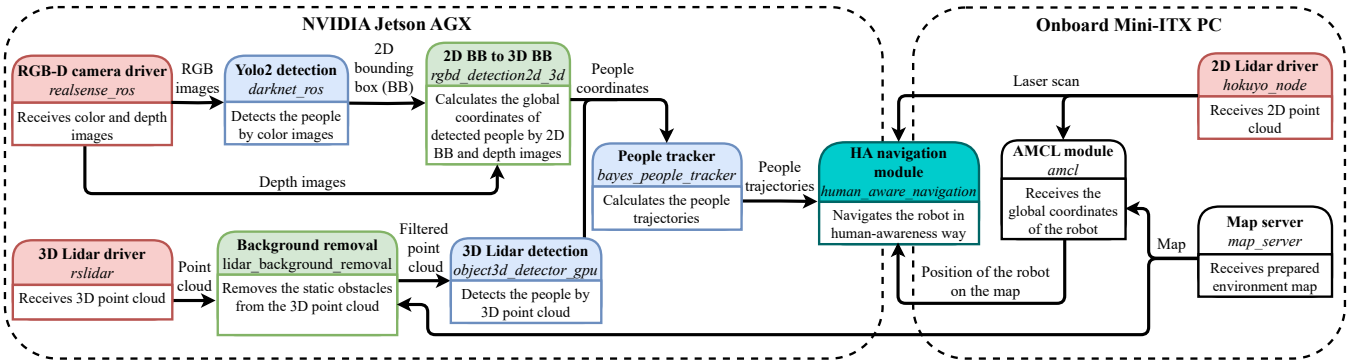


Fig. 4. Our current human-aware robot navigation software design. The red blocks refer to drivers, which are responsible for the sensory data representation. The blue blocks represent state-of-the-art methods. The light green blocks refer to the modules we developed. The white blocks represent the modules provided by the ROS navigation stack. The turquoise block represents the human-aware navigation module.

cells in the grid map. Then, a multi-target tracker [17] is used and high-level information fusion is performed on detections from different detectors. The final output is a human position with covariance, speed, trajectory, and other information [18]. It can be seen that in our current system, the velocity of pedestrians is estimated by software, i.e. the tracker. Hardware such as millimeter-wave radar can also directly provide object velocity information based on the Doppler effect [19].

Given that the integration scheme we developed will eventually be used for human-aware navigation, additional clarification is necessary here. The robot navigation can be done with or without a map. For the former, we integrate the map server and the AMCL modules provided by ROS on the robot's onboard PC, so that the robot can know its global position in the map reference frame. In the absence of a map, the robot performs ego-estimation based on proprioception and/or exteroception. But no matter what kind of navigation, under our software architecture, it can easily incorporate human information around the robot, thereby generating path planning with human awareness. This claim corresponds to the turquoise module in Fig. 4. According to its dependencies, the human-aware navigation module can be flexibly deployed on the SBC side or the AGX side. Benchmarking of different human-aware navigation modules with the developed integration scheme is presented in our work [20].

IV. Evaluation

We evaluate the integrated system at micro and macro levels respectively. At the micro level, we specify the inputs and outputs of each module and show its average operating frequency. At the macro level, we evaluate the effectiveness of the perception stack by designing several test cases to evaluate the results from system input to output using an end-to-end approach. In our paper, we consider system performance in a software engineering sense (CPU, GPU, and RAM usage of the computing unit), rather in an algorithmic sense (accuracy of human detection and tracking). We conduct laboratory experiments to ensure robust detection of experimental participants and to minimize the impact of the algorithm on the integrated system performance.

A. Micro level

From Table II we can see that both the 3D lidar driver and the background removal module operate at a frequency of 10 Hz, which corresponds to the preset spinning frequency of the 3D lidar. The subsequent 3D object detector runs at a higher frequency of 20 Hz to ensure timely processing of the point cloud data. This result is higher than the best performance (11 Hz) reported in [21]. Furthermore, although the RGB-D camera driver and YOLOv2 detector run at 30 Hz and 44 Hz respectively, the current module responsible for projecting 2D object bounding boxes to 3D can only run at a lower frequency of 25 Hz. This result is slightly inferior to the best performance (28 Hz) reported in [21], at the expense of better modularity. Ultimately, the multi-target tracker operates at 30 Hz, which is on par with the data release frequency of the RGB-D cameras and much greater than that of the 3D lidar. To measure all frequencies reported in this paper we use "rostopic hz" command from *rostopic* ROS package.

Additionally, it is worth knowing that the operating frequencies of the 2D lidar driver and the localization module are both 40 Hz, which is equal to the data release frequency of the 2D lidar. This means that the actual pose of the robot is updated with the 2D lidar data without any delay, thus providing conditions for safe robot control.

B. Macro level

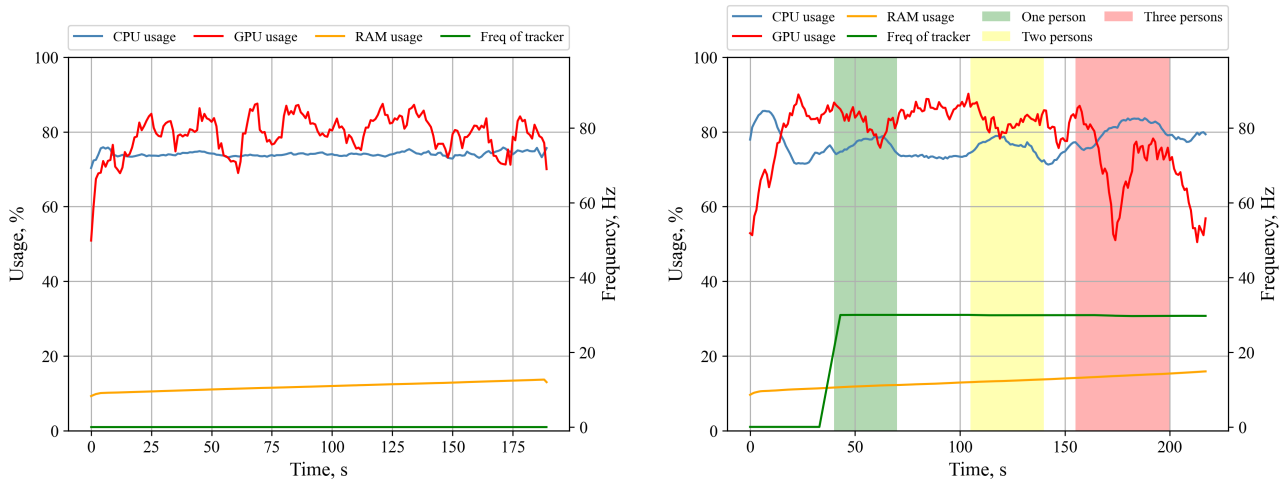
At the macro level, we track the Nvidia AGX's CPU, GPU, and RAM usage in the absence and presence of humans. In addition, we measure the working frequency of the multi-target tracker with different numbers of humans. During the experiment, the number of people varies from one to three. The maximum number of people is due to limited space in the experimental room. As the output of the tracker is also the output of the entire perception stack, measuring it provides a glimpse into the effectiveness of the latter.

The experiments include two scenarios, each repeated ten times. Due to very small deviations in results between

TABLE II

INPUTS, OUTPUTS AND OPERATING FREQUENCIES OF THE SOFTWARE MODULES

Module (ROS package name)	**Input**	**Output**	**Frequency**
3D lidar driver (*rslidar*)	Raw data	3D point cloud	10 Hz [SD = 500]
Background removal (*lidar_background_removal*)	3D point cloud, map	3D point cloud	10 Hz [SD = 77]
3D object detector (*object3d_detector_gpu*)	3D point cloud	3D bounding box	20 Hz [SD = 500]
RGB-D camera driver (*realsense_ros*)	Raw data	Color and depth images	30 Hz [SD = 143]
YOLOv2 2D object detector (*darknet_ros*)	Color image	2D bounding box	44 Hz [SD = 400]
2D bounding box to 3D (*rgbd_detection2d_3d*)	2D bounding box, depth image	3D bounding box	25 Hz [SD = 11]
Multi-target tracker (*bayes_people_tracker*)	3D bounding box	Human trajectory and more	30 Hz [SD = 333]
2D lidar driver (*hokuyo_node*)	Raw data	2D point	40 Hz [SD = 5000]
Localization (*amcl*)	2D point, map	Robot pose	40 Hz [SD = 48]



(a) The performance without people around the robot

(b) The performance with one or more people around the robot

Fig. 5. The performance of the developed perception system.

individual trials, we report results from only one of the trials. The first scenario aims to measure the system parameters without any humans around the robot, the results of which are shown in Fig. 5(a). It can be seen that the CPU usage is about 75%. The GPU usage is less stable than the CPU, with an average of 80%. The RAM usage rate starts to increase from 10%, and the reason for this will be analyzed below. As expected, the multi-target tracker runs at zero frequency because there are no humans around the robot.

The second scenario evaluates the system with an increasing number of people around the robot. Specifically, the robot was first deployed in an empty room. Then, a person entered the room and started to leave after 30 seconds. Next, two people entered the room and also started to leave after 30 seconds. Finally, three people came into the room and started to leave after 30 seconds. During the experiment, people were asked to move around the robot at normal walking speed. Through the entire process, the perception system correctly detected and tracked all participants. The performance of the corresponding three modules, as shown in the blue blocks in Fig. 4, has been widely evaluated in the original papers [16], [12], [17]. The experimental results are shown in Fig. 5(b).

The green, yellow, and red areas in the respective subplots illustrate the time periods when one, two, and three people enter and leave the room, respectively. The size of the area increases with the number of people since more people means more time needed to enter and leave the room. Experimental results show that CPU usage increases when humans are detected. However, the relationship between this increase and the number of people remains unclear. Instead, the GPU usage drops first after detecting people and then returns to the value before the drop. The RAM occupancy does not show a noticeable response to human detection. The working frequency of the multi-target tracker is stable at around 30 Hz after the appearance of humans, and it is not sensitive to the number of people. To find out why RAM usage grows with runtime, we performed additional digging. First, the presence of humans is unnecessary, as the growth can occur with or without humans. Similar to an ablation experiment, we search for the source of the problem by disabling individual modules in the system. The experimental results are shown in Fig. 6. Specifically, we tested 1) only 2D object detector, 2) only 3D lidar driver, 3) only 3D object detector, and 4) the entire perception system. Fig. 6 shows that the 3D object detector is the source of the continuous increase in memory footprint. The reason is that this detector calls the CUDA library optimized for AGX[1],

[1]https://github.com/NVIDIA-AI-IOT/cuPCL

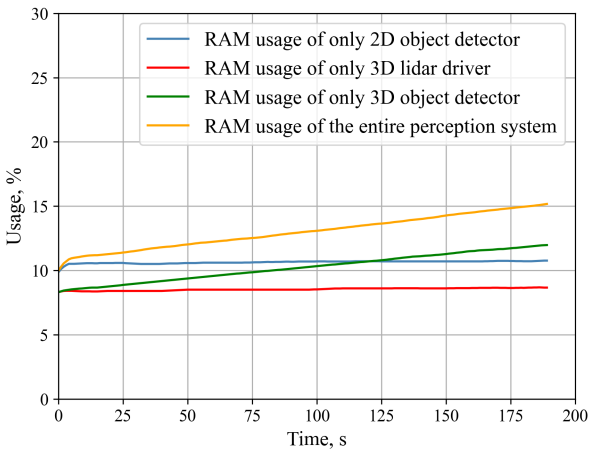


Fig. 6. The RAM footprint results for various perception system configurations without human presence.

which is currently a closed source. Therefore, the current problem of growth can only be located and cannot be solved temporarily.

Based on the above experiments, our insights can be summarized in two aspects.

- Compared to the system running at 20 Hz reported in [18], our integrated system exhibited higher input and output frequencies of the people tracker. Additionally taking the results reported in [21] as a reference, the above results show the broad industrial application prospects of our proposed scheme.
- The increasing RAM usage hinders the long-term deployment of the system, so the 3D object detector should be improved.

V. Conclusions

In this paper, we presented a software-hardware integration scheme for embodied human perception in service robotics. The instrumentation details and ROS package collection are publicly available to the community. The developed system is designed for embedded hardware and can be deployed holistically to the edge. Experimental results illustrate the high performance of the integrated perception system with the frequency of people tracker equal to 30 Hz. However, the 3D object detector should be improved for long-term application. The limit of work duration with the Adaptive Clustering SVM detector depends on the RAM size of a computing unit.

Our future work includes addressing the incremental RAM footprint of the 3D object detector. Additionally, experiments will be conducted in large public spaces with more participants to gain insights into the relationship between system resource requirements and environmental dynamics.

Acknowledgment

We thank RoboSense for sponsoring the 3D lidar, UTBM CRUNCH Lab for support in robotic instrumentation, and all those involved in the experiments.

References

[1] S. Perminov, N. Mikhailovskiy, A. Sedunin, I. Okunevich, I. Kalinov, M. Kurenkov, and D. Tsetserukou, "Ultrabot: Autonomous mobile robot for indoor uv-c disinfection," in *CASE*, 2021, pp. 2147–2152.

[2] T. Vintr, Z. Yan, T. Duckett, and T. Krajnik, "Spatio-temporal representation for long-term anticipation of human presence in service robotics," in *ICRA*, 2019, pp. 2620–2626.

[3] T. Yang, Y. Li, C. Zhao, D. Yao, G. Chen, L. Sun, T. Krajník, and Z. Yan, "3d tof lidar in mobile robotics: A review," *CoRR*, vol. abs/2202.11025, 2022.

[4] Z. Yan, L. Sun, T. Krajnik, and Y. Ruichek, "EU long-term dataset with multiple sensors for autonomous driving," in *Proceedings of IROS*, 2020, pp. 10 697–10 704.

[5] R. Yang, Z. Yan, T. Yang, Y. Wang, and Y. Ruichek, "Efficient online transfer learning for road participants detection in autonomous driving," *IEEE Sensors Journal*, vol. 23, no. 19, pp. 23 522–23 535, 2023.

[6] H. Shin, K. Kon, H. Igarashi, Y. Anbe, K. Kim, S. Hanamoto, R. Yamasaki, S. Toyoshima, N. Sato, T. Kamegawa *et al.*, "Hardware-software integration of a practical mobile robot platform," in *Proceedings of SII*, 2011, pp. 1263–1268.

[7] A. Bubeck, F. Weisshardt, T. Sing, U. Reiser, M. Hägele, and A. Verl, "Implementing best practices for systems integration and distributed software development in service robotics-the care-o-bot® robot family," in *Proceedings of SII*, 2012, pp. 609–614.

[8] T. Yamamoto, K. Terada, A. Ochiai, F. Saito, Y. Asahara, and K. Murase, "Development of human support robot as the research platform of a domestic mobile manipulator," *ROBOMECH Journal*, vol. 6, no. 4, 2019.

[9] Z. Yan, S. Schreiberhuber, G. Halmetschlager, T. Duckett, M. Vincze, and N. Bellotto, "Robot perception of static and dynamic objects with an autonomous floor scrubber," *Intelligent Service Robotics*, vol. 13, no. 3, pp. 403–417, 2020.

[10] X. Li, X. Wang, J. Liao, X. Li, S. Li, and H. Lyu, "Semi-tightly coupled integration of multi-gnss ppp and s-vins for precise positioning in gnss-challenged environments," *Satellite Navigation*, vol. 2, pp. 1–14, 2021.

[11] M. E. Andrada, J. F. Ferreira, D. Portugal, and M. S. Couceiro, "Integration of an artificial perception system for identification of live flammable material in forestry robotics," in *Proceedings of SII*, 2022, pp. 103–108.

[12] Z. Yan, T. Duckett, and N. Bellotto, "Online learning for human classification in 3D LiDAR-based tracking," in *Proceedings of IROS*, 2017, pp. 864–871.

[13] L. Sun, Z. Yan, S. M. Mellado, M. Hanheide, and T. Duckett, "3DOF pedestrian trajectory prediction learned from long-term autonomous mobile robot deployment data," in *Proceedings of ICRA*, Brisbane, Australia, May 2018, pp. 5942–5948.

[14] T. Yang, Y. Li, Y. Ruichek, and Z. Yan, "Performance modeling a near-infrared tof lidar under fog: A data-driven approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 11 227–11 236, 2021.

[15] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[16] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of CVPR*, 2017, pp. 7263–7271.

[17] N. Bellotto and H. Hu, "Computationally efficient solutions for tracking people with a mobile robot: an experimental evaluation of bayesian filters," *Autonomous Robots*, vol. 28, pp. 425–438, 2010.

[18] Z. Yan, T. Duckett, and N. Bellotto, "Online learning for 3d lidar-based human detection: experimental analysis of point cloud clustering and classification methods," *Autonomous Robots*, vol. 44, no. 2, pp. 147–164, 2020.

[19] F. Majer, Z. Yan, G. Broughton, Y. Ruichek, and T. Krajnik, "Learning to see through haze: Radar-based human detection for adverse weather conditions," in *Proceedings of ECMR*, Prague, Czech Republic, September 2019, pp. 1–7.

[20] I. Okunevich, V. Hilaire, S. Galland, O. Lamotte, L. Shilova, Y. Ruichek, and Z. Yan, "Human-centered benchmarking for socially-compliant robot navigation," in *Proceedings of ECMR*, Coimbra, Portugal, September 2023, pp. 1–7.

[21] T. Linder, N. Vaskevicius, R. Schirmer, and K. O. Arras, "Cross-modal analysis of human detection for robotics: An industrial case study," in *Proceedings of IROS*, 2021, pp. 1–8.