Heterogeneous thread-based parallel software for scalable industrial machinery

Miguel O. Martínez-Rach[1][0000−0001−7071−8921], Otoniel López-Granado[1][0000−0002−6968−061X], Carlos Pérez-Vidal[2][0000−0003−0943−8693], Ricardo Morales[3], and Héctor Migallón[1][0000−0002−4937−0905]

[1] Computer Engineering Department, Miguel Hernández University, Elche, Spain
mmrach,otoniel,hmigallon@umh.es
[2] Systems Engineering and Automation Department, Miguel Hernández University, Elche, Spain carlos.perez@umh.es
[3] Research and Development Department, Jovisa S.L., Muro de Alcoy, Spain
ricardo@jovisa.es

Abstract. Computing in industry is highly integrated, and the development of versatile and scalable industrial machinery requires the development of equally versatile and scalable software that efficiently utilizes the computing platform embedded in such industrial machinery. Any conventional computer today, even if designed for industrial use, is a shared memory or multi-core parallel architecture. This architecture must be efficiently exploited in the design and development of industrial machines that, in addition to controlling common systems such as robots, conveyors, etc., must collect and process information from the environment, with increasingly complex sensors that provide greater amounts of information and with more complex processing that increases the overall computational cost. To meet the temporal requirements, this work presents the software design of an industrial machine equipped with at least one hyperspectral sensor. This industrial equipment is designed as an add-on that must be adapted to the needs of the industrial plant in which it is integrated.

Keywords: Industry 4.0 · heterogeneous computing · HSI.

1 Introduction

The characteristics of the Fourth Industrial Revolution [1] include interoperability, resource optimization, and adaptability to future changes. In other words, new production systems must be able to integrate into systems with other devices with which they may or may not communicate. If machines can adapt to perform similar tasks in the same industry, resources will be optimized. It must also be able to adapt to integrate new technologies or relevant advances in the technologies used.

In the proof of concept, developed in collaboration with the company JOVISA S.L., a scalable software adapted to a selective parts collection system has

been designed. These parts, which come from different sources, can be selective collection waste, such as plastics, in which case it is necessary to differentiate between materials with similar characteristics, or industrial waste, such as packaging waste, in which case it is necessary to differentiate between cardboard, wood and plastic, for example. The use of conventional vision technology could be useful for the latter application, but not for the former, so a design requirement is the use of HSI (Hyperspectral Imaging) technology to be able to distinguish any type of material by its chemical composition [2]. It is logical that the computational cost of the two examples is not comparable.

The computational cost also depends on many other factors, such as: the spatial and spectral resolution of the HSI sensor and/or the number of HSI sensors, which affects the amount of data to be processed; the minimum size of the objects to be collected; the amount of different materials to be discriminated; the number of collection robots; the speed and size of the conveyor belt. In short, we are dealing with a non-stationary production system.

The goal is to configure a software architecture running on a conventional multicore computer system that adapts to the maximum possible combinations of the picking system. It should be noted that this picking system is a system to be integrated with other processing lines and must be designed ad hoc for each installation.

2 Parallel software skeleton

The requirements of each ad-hoc system will determine the computational resources to be used. In the first level of work distribution, the different heterogeneous tasks are distributed by assigning a processing thread to each of the basic tasks, ensuring that all the computational threads are mapped onto different cores. It is worth mentioning that one of the most widely used software tools for the development of robotic systems, such as the one that integrates the presented work, is ROS (Robot Operating System) [3], which is characterized by being a system with a distributed architecture with communication through messages, i.e. an architecture with great analogy to MPI; in this way, the developers of these environments can exploit the parallelism of the computational architecture by developing different modules that communicate through ROS.

In the system presented, the developments made with ROS are just another module that is part of the system. Not all of the infrastructure can be developed on the basis of ROS because of the large amount of information to be collected and processed, coming from one or more HSI cameras and probably from cameras in the visible RGB spectrum, which cannot be transmitted by messages, on the one hand because of the time cost of these communications and on the other hand because of the increased memory requirements due to the fact that the memory is not shared.

The initial development based on ROS results in a practical impossibility to meet the time requirements necessary to work in real time. Therefore, the software structure shown in Fig. 1 is designed. As can be seen, the main thread

is responsible for creating the first layer of heterogeneous work distribution, for this layer we're going to use operating system (OS) threads. It is worth noting that the development environment was QT, a multi-OS cross-platform development framework with an inter-thread communication system based on signals and slots, retaining the benefits of a shared memory system.

The main tasks to be distributed, see Fig. 1, are: management of the GUI assigned to the main thread; acquisition of data received from the HSI camera (sensors thread); segmentation of objects to determine edges, sizes and centroids (object detection thread); management of queues of objects to be collected, responsible for sending orders to the robot arms (dispatch objects thread); management of ROS processes, mainly communication with PLCs (ROS ecosystem thread); a thread for storing data in cloud servers is provided if external communication is available (communications thread).

As mentioned above, the goal is to have a hardware scalable system. If the system requires more than one camera, either HSI or RGB, the camera thread will spawn an OpenMP thread for each camera installed, which will be responsible for communicating with it. Each of these threads must classify the type of material for each pixel based on the HSI information. This classification task is critical and must be completed before new data becomes available. If this time requirement is not met, this processing is accelerated by generating new OpenMP threads in a nested parallel region, being the third level of parallelization or work sharing.

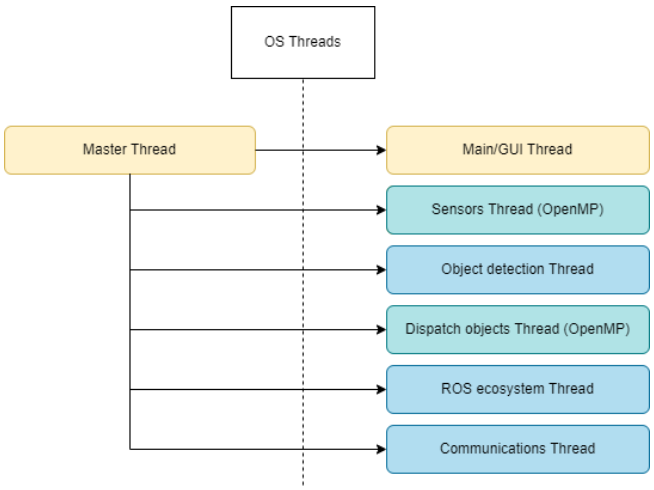

Fig. 1. Block diagram and thread assignment.

If we consider a system with two HSI cameras and one RGB camera, and the HSI classification requires 3 threads in the nested region, while the RGB does not require acceleration, the number of cores used by the sensors thread will be

7. Furthermore, if we consider that two robotic arms are installed, the dispatch object thread spawns another thread and, therefore, with respect to the number of cores the system requirement is 13.

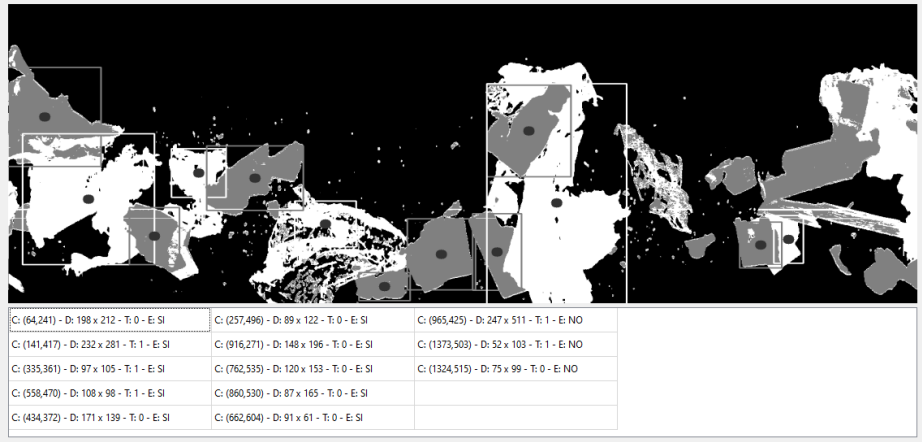

Fig. 2. GUI displayed classification and object detection

Fig. 2 shows the image that appears in the developing GUI when the data is classified and segmented, i.e. the information about the detected objects and the classification according to their chemical composition is already available.

Acknowledgments. This research was partially supported by the Spanish Ministry of Science and Innovation and the Research State Agency under Grant PID2021-123627OB-C55 co-financed by FEDER funds (MCIN/AEI/FEDER/UE), and by the project CPP2021-008593, funded by MCIN/AEI/10.13039/501100011033 and by the European Union-NextGenerationEU/PRTR.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Schwab, K.: The fourth industrial revolution. Currency (2017)
2. G.J. Edelman, E. Gaston, T.G. van Leeuwen, P.J. Cullen, M.C.G. Aalders; Hyperspectral imaging for non-contact analysis of forensic traces, Forensic Science International, Volume 223, Issues 1–3, (2012). https://doi.org/10.1016/j.forsciint.2012.09.012
3. P. Tsarouchi, S. Makris, G. Michalos, A.-S. Matthaiakis, X. Chatzigeorgiou, A. Athanasatos, M. Stefos, P. Aivaliotis, G. Chryssolouris, ROS Based Coordination of Human Robot Cooperative Assembly Tasks-An Industrial Case Study, Procedia CIRP, Volume 37, (2015). https://doi.org/10.1016/j.procir.2015.08.045