AI4U: Modular Framework for AI Application Design

Kamil Wołoszyn[1], K. Turchan[1], M. Rąpała[2], and K. Piotrowski[1(✉)]

[1] IHP - Leibniz-Institut für innovative Mikroelektronik, Frankfurt (Oder), Germany
{woloszyn,turchan,piotrowski}@ihp-microelectronics.com
[2] CBK PAN - Centrum Badan Kosmicznych PAN, Zielona Gora, Poland
mrapala@cbk.waw.pl
https://www.ihp-microelectronics.com , https://cbkpan.pl

Abstract. This paper presents the concepts of a universal, modular framework that shall enable rapid development of AI applications. The goal of the conceptual tool is the versatility in terms of changing the environment, as well as integration of different types of sensors to perform a specific task. The framework follows the Sens4U approach that will facilitate the entire process of building the AI applications and simplify the testing process.

Keywords: AI · Artificial Intelligence · AI Application · Framework

1 Motivation

Artificial intelligence (AI) has grown in popularity in recent years. AI has been applied in many scenarios, from intelligent assistants to advanced real-time vision systems. AI enables devices to make decisions based on data collected from cameras or various sensors. This data is processed and the device responds correspondingly. Artificial intelligence (AI) is a part of computer science domain which builds systems or machines that in specific tasks can imitate human intelligence. They are also able to constantly improve their performance according to collected data. Artificial intelligence is often associated with humanoid robots that are believed to replace humans, but their real purpose is to increase human capabilities instead [1]. Artificial intelligence is divided by the mechanism of complexity.

Artificial Weak/ Narrow Intelligence is the type we are using nowadays. Most of current AI systems can be classified as narrow artificial intelligence. All software using technologies such as natural language processing, Machine learning, pattern recognition, data mining is considered as narrow artificial intelligence. Narrow artificial intelligence focuses mainly on tasks in which human beings can be outperformed.

Artificial Strong/General Intelligence is the concept of an intelligent system with a great deal of knowledge and cognitive ability, which will be able to

C. Biele et al. (Eds.): MIDI 2022, LNNS 710, pp. 56–66, 2023.
https://doi.org/10.1007/978-3-031-37649-8_6

independently perform tasks even those not yet known. It should think similarly to a person and with comparable or even better efficiency. At present, there is no machine that can understand the world as a human being can.

Artificial Super Intelligence is the most popular term circulating in the computer science world in recent times. An artificial super intelligence will hypothetically be able to interpret and to understand human behaviour and/or intelligence. It will be able to surpass human intelligence in mathematical tasks, science, medicine or applying solutions to different types of problems. What is more, artificial super intelligence is assumed to be able to evoke understanding, emotions or to be capable of having its own desires.

Artificial intelligence is an idea that integrates human intelligence with machines, as shown in Fig. 1. An issue related to artificial intelligence is ML (Machine Learning), which is one of the subcategories, that belongs to the area of artificial intelligence. Its main goal is to allow computers to learn using specific algorithms.

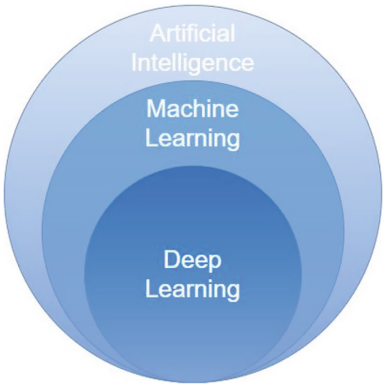

Fig. 1. Artificial intelligence structure.

This paper presents the concept of a universal, modular tool that will enable rapid construction of AI applications. The goal of this conceptual tool is its versatility in changing environment and different sensor types integration needed to perform a specific task. The concept of the proposed tool using the Sens4U approach [2], will improve the construction process, testability and reliability of the application. The idea of Sens4U is based on accelerating the application construction for non-experts in the field by using the modularity of available libraries.

Currently, there are many tools available for training deep neural networks, and many examples of software codes that support the sensors needed to create an intelligent robot or implementation of trained models. However, putting these together and making it fully function requires a large time investment.

When working on the SpaceRegion project, the initial focus was to build a stationary robot with no mobility capability. The robot is equipped with video cameras and servos to control the cameras, and its task was to observe and track detected objects. In the course of the work, it was decided that a mobile robot would be additionally built for the project. Initially the mobile robot was supposed to have the same functionality as the stationary robot, but during the code implementation to the mobile robot it turned out that most of the software code was the same, and only the code responsible for controlling the robot chassis has changed significantly.

This gave rise to the idea of building a universal tool that would speed up the process of developing AI applications. This approach can be also used to move one application from a particular scenario to another, for example a consumer solution to a space solution, without re-implementing the application from scratch. A universal tool must define application areas and requirements that simplify the migration process between scenarios. According to the Sens4U approach, the application area of the proposed tool will be expandable, by adding support for new sensors or updating those already implemented in the conceptual tool.

This paper is structured as follows. Section 2 discusses the related work, while Sect. 3 describes the proposed approach, followed by the example scenarios given in Sect. 4. Section 5 concludes the paper.

2 Related Work

At present, one can find systems that provide users with libraries and tools to help building robot applications, such as Robot Operating System (ROS) [3] or CubeSystem [4]. ROS is a meta operating system focused on building robots, which main goal is to support users in recycling of previously written code for robotics research or development purposes. ROS has a Gazebo [5] robot simulator that allows applications to be developed virtually, prior to the real robot, saving time and cost of modifications that might arise during construction. Another advantage of the ROS system is language versatility, thanks to which software can be written in such programming languages as Python [6], C++ [7], Lisp [8]. The big disadvantage of this system is the lack of multi-platform The stable version work only on Unix [9] systems, but there are also experimental versions for Windows system.

ROS is open source software which allows the community to create new solutions. CubeSytem on the other hand is a collection of hardware and programmable components that enable rapid prototyping of robots. CubeSystem in combination with the RobLib library [10] allows customization and linking to a distinct set of libraries that are tailored to a specific application. For example, the RobLib library contains general functions for controlling robots based on a two-wheeled differential drive. It is generally difficult to find systems or sets of libraries that have standardized functionality for sensors of the same type. The operation complexity level of the application is raised by a number of additional

software, libraries and an adequate operation system that have to be installed. Fulling of such requirements can be problematic for a user unfamiliar with the field. Given the existing software and libraries, it seems reasonable to present the concept of a tool that is functional, modular, and cross-platform to support the construction of a complete application.

3 Proposed Approach

The concept of the tool for fast and modular construction of the application is shown in Fig. 2. The fact is almost every available sensor on the market has ready-made hardware drivers as well as dedicated software. The aim of the proposed tool is to create software, which will be a set of dedicated libraries, but any introduced modification in the conceptual tool will be applicable on different sensors. As a result, it will always return the same output, assuming that all the requirements specified during the construction of the conceptual tool are met.

The idea for the proposed tool emerged during working on the development of a computer vision application. The programming language used to build the application is Python version 3. Python is a high-level programming language, and thanks to its simplicity and versatility, it has become an ideal tool for building AI systems.

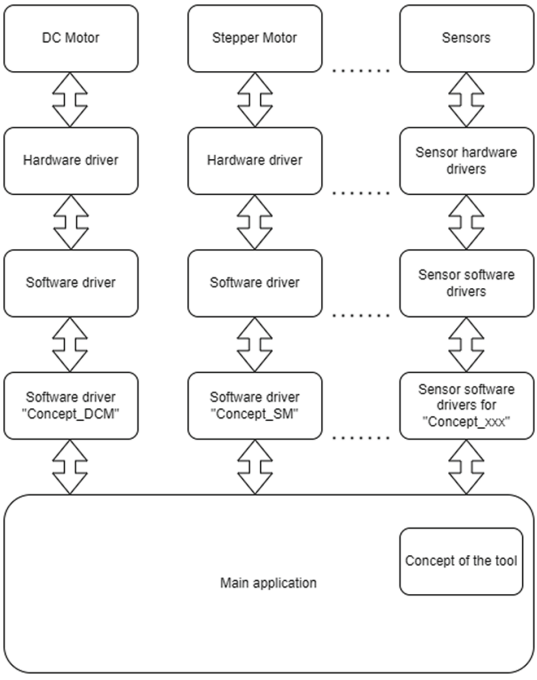

Fig. 2. Concept of the proposed tool.

The base AI application that was created for the project had to observe the environment, recognise people in the same room and check from time to time whether the person is still in the same place. Based on the people present in the room, the robot estimated in which office it was located. The basic application was initially built on a stationary robot, which is the QBO ONE [11] produced by Thecorpora as shown in Fig. 3.

Fig. 3. Robot QBO.

Further, the plan was also to investigate the possibility of a smooth transfer of the application samples to more demanding environments (like space). In order to support that, the application should fulfil its tasks when the original sensors and actors are replaced by devices that fit the new environmental conditions. The application should thus use generic application programming interface (API) to apply the functionalities of the devices and for instance adapt to their physical features (video resolution of a camera module or the resolution of a motor turn).

Replacing a device with another one does not change the application implementation although it can change its internal behaviour and features. The API equips the application with the control over the device and provides information about its features. The application developer who uses the conceptual tool will be able to obtain a list of available functions for a given sensor. When building

the tool, it would be necessary for the user to define the communication interface and to enter the parameters so it could work correctly.

For example, a user is building a robot with built-in DC motors. Using the API functionality of the conceptual tool, the user would be able to call, for example, the getFeaturesDCmotor() method. This method will return information to the user about parameters that should be entered and about available functions for DC motors e.g., turnForward(), turnBackward(), stop().

Another functionality of the tool concept is its modularity, which aims to support the seamless modification of the project. As an example lets take the modification of the mentioned robot. The user replaces the standard DC motors with stepper motors, which brings a number of changes to the created main application. However, when using the proposed conceptual tool, the only modification that the user should provide, would be possibly a name change of the used motors. The basic functionality would be consistent for both motors and the program operation after the motors replacement would not be disturbed.

What's more, after modifications, calling the getFeaturesSmotor () method for stepper motors will display additional functions, besides turnForward (), turnBackward (), stop() stepper motors can also have functions like turnForwardAboutAngle (x_angle), turnBackwardAboutAngle (x_angle) and turnLeftAboutAngle (x_angle), turnRight-AboutAngle (x_angle), setStartPosition() while maintaining the requirements related to the motor mounting direction specified during the tool construction.

The proposed tool will allow expansion with new libraries depending on the users' needs. The main advantage of the conceptual tool would be reliability to environmental changes, and unification of functionality for sensors of the same type, what would result in rapid modifications in the design. During the construction of the tool, it will be possible to extend its operation to support different types of sensors. Moreover, sample functionalities such as object detection on the provided video stream or real-time object tracking from trained deep learning models can be implemented in the tool for artificial intelligence purposes.

For devices with low processing power, the proposed conceptual tool will be able to implement functionality to support AI accelerators. The AI accelerators will enable the satisfactory performance of artificial neural networks combined with computer vision. AI accelerator is, for example, the Intel Neural Compute Stick 2 [12]. One advantage of the proposed tool will be its cross-platform capability, which would allow the tool to run on both Microsoft Windows and Linux platforms.

4 The Application Area

This section presents examples of scenarios from which requirements for the correct operation of the proposed tool may arise. Some of the scenarios have been implemented, some are in the process of being implemented and some may be created in the future for development, experimental or research purposes during the development of the proposed conceptual tool. While creating the application,

scenarios based on terrestrial solutions were used for testing reasons and finding available solutions. However, the conceptual tool also envisions applications in extraterrestrial scenarios.

4.1 Watchman Scenario

The first developed scenario is the Watchman scenario. The main task of this scenario is to monitor a specific room and control people authorized to be in it. The system based on optical sensor and machine learning must be able to detect, recognize and perform verification of a person present in the room. The next step that the created system must take is to react during the detection of a possible intruder. During the detection of an intruder, the task of the system is to take a picture of a person who is not authorized to be in the room and record the event in the database for further verification by the administrator. The system must be adapted to expand the database of people authorized to stay in the room.

4.2 Parkmonitor Scenario

The second scenario is the Parkmonitor scenario. The main task of this scenario is to monitor a specific area and count free and occupied parking spaces. System using optical sensor and machine learning must be able to detect and return information about number of free and occupied parking spaces.

4.3 Tracker Scenario

The third scenario is the Tracker scenario. The main task in this scenario is to track a specific object by an unmanned craft (drone). For example, such an object can be a ball, a cyclist, a person or a car. In general, any model of object or objects that can be trained using neural networks suit this scenario. The system, based on the optical sensor and machine learning, after activating the function, must be able to follow a specific object until it disappears from the frame or the function is disabled. The drone operator must be able to activate and deactivate the tracking function.

4.4 Mapping the Environment Scenario

The fourth scenario is the Mapping the Environment scenario. The main task in this scenario is to build a map of the area on which the mobile robot moves. Created system has to enable real preview of the built map and must have the ability to export the created map of space/area. Moreover, the system using an optical sensor and machine learning has to be able to mark certain objects on the built map. Additionally, the system must have the possibility to choose the option of controlling the vehicle. The first option is to drive autonomously to build a map of the room. The second option after building the map is to drive

to a specific point on the map and the last required option is manual control of the vehicle. A prototype of the mobile vehicle under development is shown in Fig. 4.

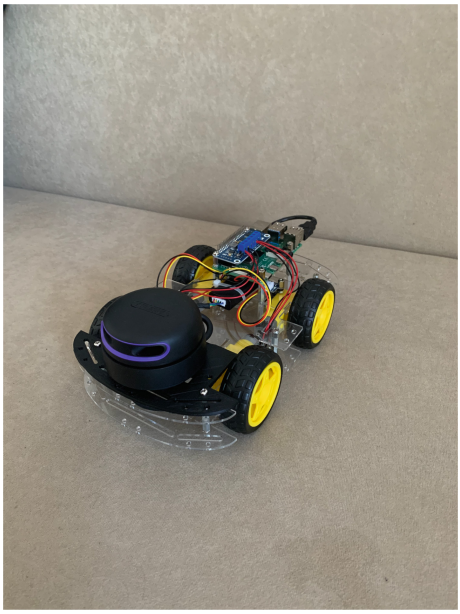

Fig. 4. Mobile vehicle prototype.

4.5 Vehicle Counting

The fifth is the scenario of Vehicle counting. The main task of this scenario is to build a system capable of counting vehicles. Vehicles that will be counted are wheeled vehicles such as: cars, trucks and motorcycles. Counting of vehicles has to be done separately in two directions for vehicles approaching and moving away from the device. Additionally, the device on which the created system will operate must be resistant to external weather conditions and it should be as small as possible. Prototype of vehicle counting device under development is shown in Fig. 5. After installing and testing the device, the system will transmit the vehicle counting results via a network to the middleware. Vehicle counting device prototype consists of Raspberry Pi 4, Raspberry Pi High Quality Camera, 6mm 3MP Wide-Angle Lens and Intel Neural Compute Stick 2.

4.6 Space Surveyer

The sixth scenario is the space scenario of the Space Surveyer. The main task in this scenario is for the mobile robot to explore the terrain and determine the landing site of the lander. The created system must allow communication

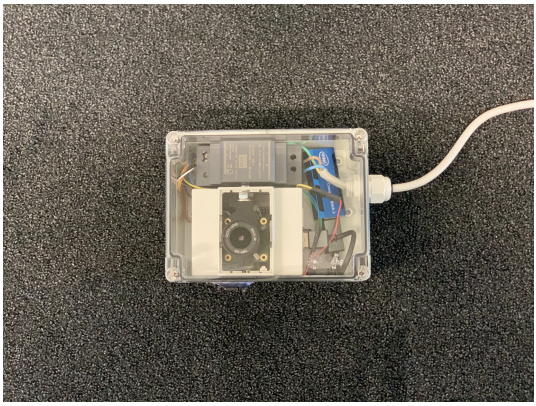

Fig. 5. Vehicle counting device prototype.

between the lander and the mobile robot to exchange information while exploring the terrain. The mobile robot after exploring the terrain must be able to determine the best landing place and send information about it to the lander. The site selection will be determined by the detected obstacles, hills or hardness of the ground. In case the communication between the platforms fails, the mobile robot will be equipped with a light system which will allow the lander to start the landing procedure after detection. The light signal from the mobile robot will determine the center of the selected landing area.

4.7 Mission to Mars

The seventh scenario is the space scenario of Mission to Mars. The main task in this scenario is the autopilot function for the spacecraft. The developed system using optical sensors and machine learning has to be able to track the planet Mars and enable and disable the autopilot function. When the autopilot function is enabled, the system will be able to control and correct the flight to the designated planet.

4.8 First Conclusions

The above scenarios, while each different, have something in common. Namely, each of them contains optical sensors and machine learning. So, building one system with the previously described modular approach will reduce the time needed to build the next system. A user or even a developer involved in building AI systems, after introducing a conceptual tool will not have to write a consecutive system from scratch. The gained time can be spent on the process of building and training the model, which is time consuming.

In the first scenario, the previously mentioned QBO platform was used to build a watchman application capable of monitoring a room. This platform, thanks to the installed servos, allowed searching for objects around the room.

The built-in VGA cameras were used for object detection. For image processing, a programming function library was used, which is OpenCV [13]. It is a library mainly oriented to work with computer vision in real time. This library has the advantage of supporting hardware acceleration and the ability to load machine learning models trained on various available development platforms such as Tensorflow [14], PyTorch [15], Cafe [16] and many others. However, for proper real-time operation, the QBO platform was modified. The Raspbery Pi 3 computer platform was replaced with a Raspberry Pi 4 and due to limited computing capabilities, an AI gas pedal such as Intel Neural Compute Stick 2 was added to the computer platform.

After the modifications, the AI application created in conjunction with the QBO platform enabled seamless real-time monitoring of the room and verification of people from the created photo database. Having built a system from the first scenario, it is easy to build another system, for example for the fifth scenario. Thanks to the modular approach to building applications, the module responsible for the detection of objects from the first scenario can be fully applied in the fifth scenario.

It is enough to change the trained model from people to vehicles and replace part of the code responsible for controlling the camera with a new code created to count the occurring vehicles. The same is true for second scenario. A minor modification of the code and a change of the trained model will allow building a parking lot monitoring application without much effort. In the first scenario, after replacing the module responsible for controlling servos with a module for controlling DC motors, the functionality can be extended to run on a mobile platform without much effort.

5 Conclusions and Further Work

The paper presents the concept of the tool, which is still under development and may change during the implementation process when new problems and challenges are discovered within the SpaceRegion project. Different scenarios based on terrestrial and space solutions are currently developed and implemented. Based on the presented scenarios, the capabilities and requirements of the proposed tool will be adjusted. We will also analyse and compare implemented code for different scenarios in order to find common parts, which will be eventually transferred to the proposed tool. A library will be created consisting of several sensors of the same type in order to test the behaviour of the application during its modification.

Furthermore, sample applications related to artificial intelligence, robotics and computer vision using the conceptual tool will be created. In addition, an analysis of the application implementing modifications under scenario change, will be performed. The main goal of the proposed conceptual tool is to simplify the construction of AI applications.

Acknowledgments. This work was supported by the European Regional Development Fund within the BB-PL INTERREG V A 2014–2020 Programme, "reducing barriers - using the common strengths", project SpaceRegion, grant number 85038043. The

funding institutions had no role in the design of the study, the collection, analyses, or interpretation of data, in the writing of the manuscript, or in the decision to publish the results.

References

1. Jajal, T.D.: Distinguishing between Narrow AI, General AI and Super AI. https://medium.com/mapping-out-2050/distinguishing-between-narrow-ai-general-ai-and-super-ai-a4bc44172e22. Accessed 21 Apr 2022
2. Piotrowski, K., Peter, St.: Sens4U: wireless sensor network applications for environment monitoring made easy. In: Proceedings of the 4th International Workshop on Software (2013)
3. ROS. ROS-Robot Operating System. http://wiki.ros.org/ROS/Introduction. Accessed 22 Mar 2022
4. Birk, A.: Fast robot prototyping with the cubesystem. In: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2004 (2004)
5. GAZEBO. http://gazebosim.org/. Accessed 22 Apr 2022
6. Python. PythonTM. https://www.python.org/. Accessed 22 Mar 2022
7. C++. https://www.cplusplus.com/. Accessed 22 Apr 2022
8. Lisp. https://lisp-lang.org/wiki/. Accessed 22 Apr 2022
9. Unix. https://www.hpc.iastate.edu/guides/unix-introducti-on. Accessed 22 Apr 2022
10. RobLib. http://robotics.jacobsuniversity.de/-CubeSyst-em. Accessed 22 Apr 2022
11. Thecorpora Robotic Company, QBO ONE. https://thecorp-ora.com. Accessed 07 Apr 2022
12. Intel Corporation. Intel®Neural Compute Stick 2 (Intel®NCS2). https://www.intel.com/content/www/us/en/dev-eloper/tools/neural-compute-stick/overview.html. Accessed 22 Mar 2022
13. OpenCV. https://opencv.org/. Accessed 04 Apr 2022
14. Tensorflow. https://www.tensorflow.org/. Accessed 21 Apr 2022
15. PyTorch. https://pytorch.org/. Accessed 21 Apr 2022
16. Café. https://caffe.berkeleyvision.org/. Accessed 21 Apr 2021