education sciences

MDPI

Article

Extending Smart Phone Based Techniques to Provide AI Flavored Interaction with DIY Robots, over Wi-Fi and LoRa interfaces

Dimitrios Loukatos * and Konstantinos G. Arvanitis *

Department of Natural Resources Management and Agricultural Engineering, Agricultural University of Athens, 75 Iera Odos Street, Botanikos, 11855 Athens, Greece

* Correspondence: dlouka@aua.gr (D.L.); karvan@aua.gr (K.G.A.); Tel.: +30-210-5294-014 (D.L. & K.G.A.)

Received: 15 May 2019; Accepted: 22 August 2019; Published: 27 August 2019

Abstract: Inspired by the mobile phone market boost, several low cost credit card-sized computers have made the scene, able to support educational applications with artificial intelligence features, intended for students of various levels. This paper describes the learning experience and highlights the technologies used to improve the function of DIY robots. The paper also reports on the students' perceptions of this experience. The students participating in this problem based learning activity, despite having a weak programming background and a confined time schedule, tried to find efficient ways to improve the DIY robotic vehicle construction and better interact with it. Scenario cases under investigation, mainly via smart phones or tablets, involved from touch button to gesture and voice recognition methods exploiting modern AI techniques. The robotic platform used generic hardware, namely arduino and raspberry pi units, and incorporated basic automatic control functionality. Several programming environments, from MIT app inventor to C and python, were used. Apart from cloud based methods to tackle the voice recognition issues, locally running software alternatives were assessed to provide better autonomy. Typically, scenarios were performed through Wi-Fi interfaces, while the whole functionality was extended by using LoRa interfaces, to improve the robot's controlling distance. Through experimentation, students were able to apply cutting edge technologies, to construct, integrate, evaluate and improve interaction with custom robotic vehicle solutions. The whole activity involved technologies similar to the ones making the scene in the modern agriculture era that students need to be familiar with, as future professionals.

Keywords: smart control; artificial intelligence; educational robotics; DIY; agricultural vehicles; android app; arduino; raspberry pi; LoRa

1. Introduction

The rapid growth of the mobile phone and tablet industry, boosted by the vast demand, made possible the production of high-end electronic components in large numbers and at very affordable prices. Devices equipped with features like cameras, GPS, compass and motion sensors, high quality screens and speakers are offered at surprisingly low prices. Inspired by this progress, the computer industry enriched its product series with credit card - sized devices of outstanding features. These devices share a lot of similarities with smart phones, but they have a bigger layout and larger connectors to facilitate the interfacing with the physical world. Such devices, of diverse capabilities, ranging from arduino [1] units to raspberrypi [2] ones, can be used in many embedded systems or IoT projects to facilitate modern life. These systems can be very beneficial, from educational aspect as well, allowing for a balanced intervention among interdisciplinary thinking and making activities, that can be very meaningful in terms of pedagogy and science [3,4]. For this reason, during the last years,

we are witnessing a very successful synergy between innovative credit card - sized computers and education [5]. The teaching curriculum of many educational institutions, from primary schools up to universities, has been updated and enriched, in order to incorporate well designed practices of this type, that are usually classified as STEM. Although most of the effort focuses on students of secondary education age, university students can be benefited as well, by participating in similar activities [6,7].

In this regard, students of agricultural engineering participating in the planned activities comprise a noticeable example, because they have to cope with scientific fields that require hands on experiences, construction skills, as well as good theoretical knowledge and design capabilities. Indeed, they have weak prior programming experience and a confined time schedule to demystify the cutting edge technologies making the scene in the agricultural era. Having a clear understanding of the participants' skills gap the authors organized a series of activities, based on problem solving techniques, that can be accomplished by all students, within a short period of time. A very important goal is for students, as future professionals, to be able to adapt to a demanding and rapidly changing technological environment, as agriculture is one of the most important sectors of primary industry and is characterized as sensitive, unstable, complex, dynamic, and highly competitive. In the twenty-first century, according to FAO [8], agricultural productivity should be increased by 60% in order to adequately satisfy the nutritional needs of the constantly growing world population, while the natural resources being available (e.g., land suitable for cultivation or fresh water supplies) are becoming fewer and less qualitative. To successfully tackle with these issues, the sector of agriculture has to become more productive and "climate-smart", by successfully exploiting a variety of existing and emerging technologies [9]. Among them, robotics and autonomous systems (RAS) technologies could positively contribute to the transformation of the agri-food sector [10–12].

This paper highlights educational activities that challenge university students to use problem solving techniques and team collaboration skills so as to be prepared for jobs that are yet to be invented. A good technique for students to become familiar with the afore discussed challenges is to participate in making custom do-it-yourself (DIY) robotic constructions and providing the necessary programming logic to support human – machine interaction scenarios, inspired by a constructionist learning methodology [13,14], successfully tested in students of younger ages [15]. Smart phones/tablets devices and innovative credit card sized systems have an "exotic" potential that should be better exploited in real world scenarios. The idea of hiring similar devices in teaching would be more successful and attractive, if provisioning for scenarios implementing modern features like remote interaction using gestures or sophisticated controls using simple artificial intelligence [16] tools, offering features like cloud based voice or speech synthesis. Students also installed, parameterized and trained a local, off-line artificial intelligence (AI) voice recognition engine to extend the functionality and autonomy of the robotic vehicle.

In recapitulating, systems like the raspberry pi and the arduino uno can play a significant role with their affordable price and their large supporting community. These devices are exhibiting a generic architecture allowing for a flawless cooperation with other important devices, like tablets and smart phones and a wide variety of sensors and actuators. A good method for starting, in order for the remote interaction and control process to be achieved via trivial Wi-Fi links (i.e., the IEEE 802.11 [17]), is by exploiting the very popular and educationally fruitful MIT App Inventor [18] visual programming environment, in conjunction with a pairing software (written in python [19] (or C) that can be run on a raspberry pi, model 3 credit card - sized computer. As students are getting more experienced, they become capable of orchestrating more composite scenarios involving further brews of hardware and software components, providing extended controlling distance and independence from strictly cloud based artificial intelligence solutions.

The whole approach is of a moderate cost and uses widely available generic tools of an open and highly modular nature, thus contributing in democratizing the educational process by providing freedom and access to opportunities for learning by inspecting, making, altering or even by sometimes damaging the operating mechanism of the participating components. Furthermore, actions of

progressive complexity are provisioned in order to provide intermediate inspection points and results. The quality of the students' work was of loosened standards, especially at the early stages of the pilot implementation. The whole effort, especially with university students with limited experience in technology, was targeted towards solutions that bridge the gap between school-level constructions and high-end (and cost) professional-level specialized robotic vehicles.

Apart from this introductory section, the rest of the paper is organized so as to highlight the pedagogical methodology being adopted (in Section 2) and (in Section 3) to present the hardware and software architecture to support the remote interaction with the robotic vehicle, as well as useful implementation details. The paper also highlights characteristic derived enhancements providing better support to the diverse control scenarios (in Section 4), presents the most important technical and pedagogical evaluation remarks (in Section 5) and, finally, concludes on the overall process and gives directions for future work (in Section 6).

2. Pedagogical Methodology Details

Students participating in the design and the implementation stages of this work attended were enrolled in the courses "Computer Science Applications in Agriculture", "Applications of Artificial Intelligence in Agriculture" and "Automatic Control Processes", which were offered during the fourth year of their studies in agricultural engineering. All these students comprised a group of around 20 persons, in total, aged from 21 to 23 years old, 16 males and four females. The duration of the students' participation in the human-robot interaction project was four months. Approximately, 25% of almost each lecture time was dedicated to tasks related with the robotic vehicle project.

The students, through an open discussion process, expressed their interests, skills and abilities, with emphasis to be put on the crafting, programming, information seeking, coordinating and presenting fields. Indeed, in the beginning of the activity, by participating in indicative group project actions, during an ice-breaking and brainstorming phase of one to two weeks students had the opportunity to reveal their diverse styles and personalities. Some people had common interests and knowledge background while others had complementary skills and little prior experience. The professor supervised the final assembly of the students into teams taking into account students' preferences, skills set, and prior performance, their skills sets and their estimated performance. A priority during the team assembly was to select group members with diverse yet complementary characteristics., for participation into the same group. This approach of forming the teams was in accordance with the methods and the goals described in [20], i.e., to achieve maximum diversity within groups of students and homogeneity among groups, but it was less mechanistic allowing them some autonomy in team member selection. The small number of students (i.e., 20 persons approximately) allowed for such an arrangement.

Four separate teams of four to five members each were formed, so as to cover the most crucial areas of the project. More specifically, one to two members of each team were responsible for the electromechanical layout (and the wiring) of the robotic vehicle. Another one to two persons in each team were responsible for programming of the card-sized computers supporting the robot's operation and communication issues. Another one to two members of each team were responsible for developing the application intercepting voice and gesture commands. Finally, one person in each team was responsible for coordination and documentation tasks and for reporting the most recent findings of the team to the whole class. The four teams performed in parallel and all members of each team worked under a loose collaboration schema and their roles were interchangeable, but not in a mandatory to follow circular basis. During the project, students had the option to experiment with separate parts, similar to the ones that were finally put on robotic the vehicle, (e.g., spare motors and drivers, chain wheels, embedded boards and radio interfaces or software modules). Students had the opportunity to learn through composing and/or gluing together experimental software and hardware parts. The better performing variants were incorporated into the final robotic vehicle platform.

The activity was based on the project-based learning model (PBL) [21] and collaborative learning (CL) theory [22]. The PBL nature is justified by the fact that the robotic vehicle's gesture and voice control challenge can be seen as an attractive mean of teaching the important knowledge and skills that students of modern agricultural engineering need to learn. The collaborative learning characteristics are also apparent, since the students within the teams expressed their ideas and opinions worked together to gather information, understand techniques and experiment as they completed the robotic vehicle control tasks. Additionally, such a learning-through-doing methodology enhances active learning, project and time management skills.

The methodology being adopted aims that the students develop both hard skills (i.e., more technical ones) and soft skills such as creativity, teamwork, communication, self-confidence and problem solving capabilities. The whole project can offer valuable engineering experiences of both visual and textual programming of smart phone devices and embedded systems. These systems are equipped with cost effective but quite tricky to handle modules, like Inertial Measurement Units (IMUs) or compass units, that are widely used in modern agricultural applications (e.g., in precision agriculture) and thus, all students should be familiar with them. In parallel, basic techniques can be introduced for combining electric motors with mechanical parts, sensors, and efficiently driving them. In terms of networking, the role of the basic communication protocols, with emphasis in wireless solutions and client- server architectures can be highlighted. Finally, students can have the opportunity to become familiar with the idiosyncrasies of an AI system and its training process. It is worth mentioning that the proposed project was designed to lead to high skills' acquisition according to the (revised) Blooms's taxonomy [23], as students recall knowledge related with the topic of constructing and controlling a machine, understand, design, apply, analyze, test, evaluate and integrate a robot. This means that students act as co-creators and makers.

Nearly 20% of students (i.e., three to five persons) dedicated one to three hours, weekly, on a voluntary basis, for further elaborating with the project's tasks. Typically, the latter students were acting as multipliers for the rest of the class, during the lecture time dedicated to the specific project, supporting peer learning [24] activities. The role of the professor, who was assisted by one laboratory staff colleague, apart from providing the initial motivation, was to encourage the teams in their work or pose fruitful questions and provide useful advice during their crafting and programming activities. In most cases, the groups were self-administered and the supervisor (i.e., the professor) acted as moderator, facilitator and expert only when needed.

3. Architectural Overview and Implementation Details

During this work, students of the Agricultural Engineering Department designed, implemented and properly modified a DIY electric robotic vehicle, so as to intercept simple commands from humans. This section provides the necessary details to highlight their work.

3.1. Electromechanical Layout

This robotic vehicle, made of wood and metal, has one independently controlled motor per side which is equipped with a chain drive mechanism. This layout combines simplicity and cost effectiveness, eliminates the need for extra mechanisms dedicated to steering tasks and leads to more robust constructions that are able to maneuver / turn in narrow areas just by changing the rotation direction of their side wheels. The robot was designed to be powerful enough to move on inclined or quite anomalous terrains, at speeds similar to the ones of a walking man. The dimensions of the robot were 40 by 40 cm, approximately. The selection of all hardware components intended to minimize size and cost and maximize reusability of materials and electronics.

There was the provision so that as this basic robotic chassis was able to host, apart from its basic controlling unit (namely an arduino uno and/or a raspberry pi boards), further equipment such as motor speed drivers and sensors, motion tracking devices or special radio transceivers and speakers. The vehicle incorporates basic automatic control functionality (in terms of speed and direction

stabilization) based on data fusion of signals provided by photo interrupters, as well as by IMU and compass modules. An arduino uno unit tackles the related low level tasks, while the complicated tasks are left to the more powerful raspberry pi (a 3B model unit – RPi3) single board computer. The basic electromechanical layout of the vehicle to study the human - robot interaction scenarios (i.e., both the initial design and the implementation outcome) is depicted in Figure 1.

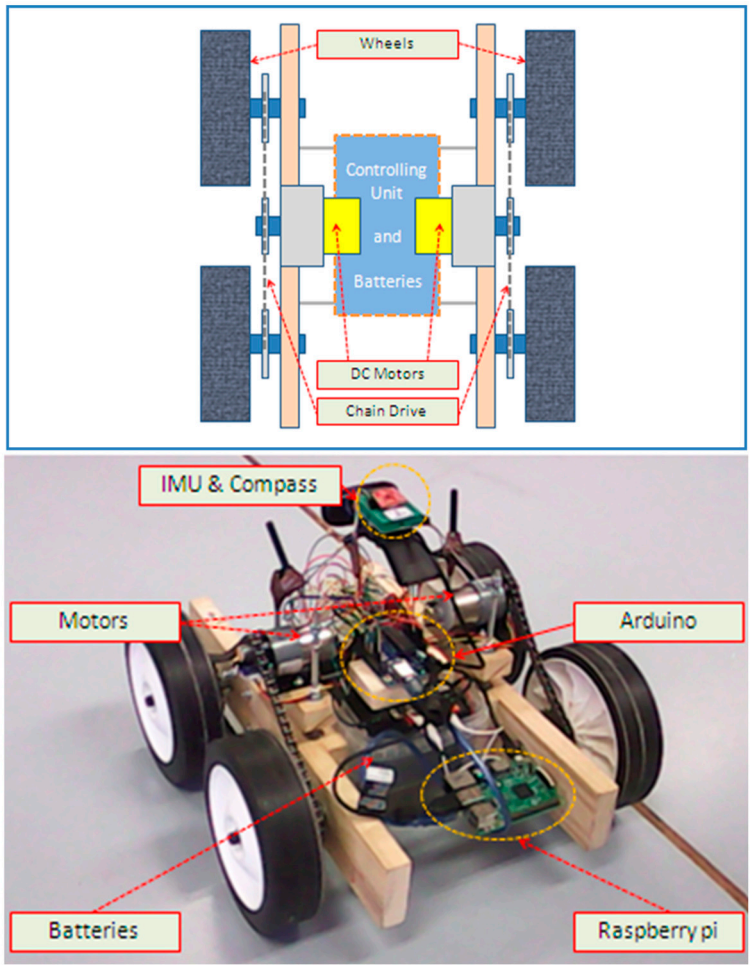

Figure 1. The basic electromechanical layout of the vehicle to study the human - robot interaction scenarios. Top part: Initial design. Bottom part: Implementation.

3.2. Human-Robot Interaction Design

As the vast majority of today's students are very familiar with smart phones/tablets, a good method to investigate, understand, and even design and implement a basic human-machine interaction schema, is programming these smart devices to generate suitable instructions towards the raspberry pi, on the robotic platform, via Wi-Fi. This configuration is not the only one being implemented and examined during this research work but is simple enough for a start and highlights the basic ideas behind human-robot interaction. To establish a fluent cooperation between the smart devices and the robotic vehicle, the MIT App Inventor tool has been used, as this tool provides an attractive and easy way for rapid mobile application creation, with AI characteristics like gesture recognition and cloud based voice recognition or speech synthesis features, even by quite inexperienced users.

The necessary controlling and monitoring commands are generated by a smart phone application, implemented by the students. This application incorporates from conventional touch button controls up to the user's gestures interception or voice recognition mechanisms. These triggers are matched with a preselected set of patterns and used to invoke the corresponding commands. The later commands,

having the form of suitable HTTP [25] request messages, are directed to the robot. The robot, in its turn, typically using its hosted raspberry pi unit, intercepts and handles those HTTP requests, via python or C implemented code, based on a properly modified HTTP server code [26]. The proper adjustments have been made for this code to be executed as a service on the raspberry pi unit (i.e., to start automatically after powering up the raspberry pi unit). This serving code may include driving commands towards the arduino unit, over the USB (serial) interface, or calls to custom Linux (bash) scripts [27], performing more sophisticated actions like sound message invocation [28], speech synthesis [29] or battery status data acquisition. Figure 2 depicts the interoperation among the modules supporting the "smart" robot control actions.

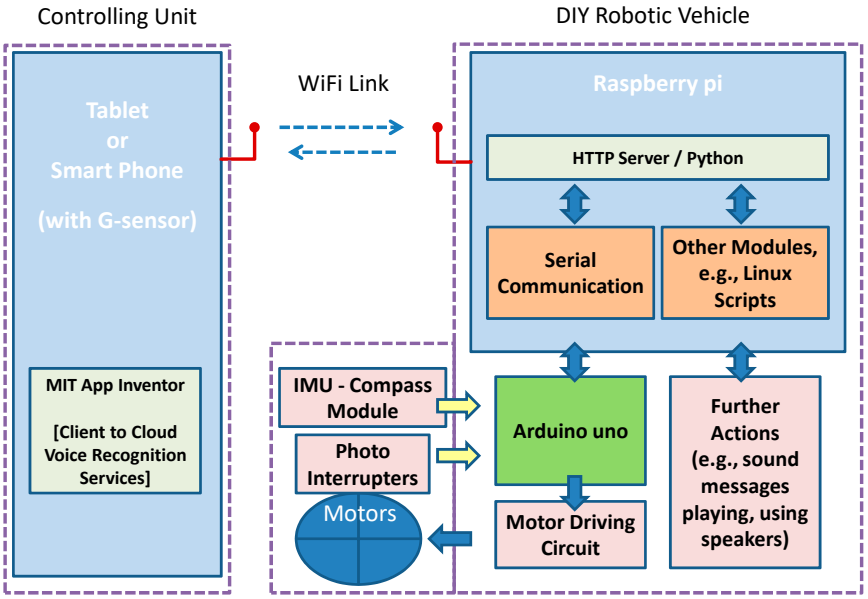

Figure 2. The interoperation among the modules supporting the "smart" robot control actions.

3.3. Human-Robot Interaction Software Details

As discussed in Section 3.2, the smart phone application interface, designed and implemented using the MIT App Inventor visual programming environment, allows for three main controlling options. The simpler one involves typical interaction, through touch buttons, and is mainly serving for debug and practice purposes. The touching of the proper button invokes a simple event that properly sets the "command-to-be-sent" field (referred as global action). Its detailed description is omitted, as it is quite trivial. Another option exploits the accelerometer unit (G-sensor) that the phone has, in order to intercept human gestures. More specifically, as depicted in the top part of Figure 3, the arctangent quantity of the y and x axis accelerometer readings is compared with empirically defined threshold values, in order to decide which command should be formed. Almost horizontal position for the smart device corresponds to a "stop" command. The field "global action" is updated according to the corresponding accelerometer axis values, thus resulting in higher speeds or turns for more extreme gestures. According to the third case, as depicted in the middle and bottom part of Figure 3, whenever the smart phone is shaken (or a corresponding button is pressed), the application listens to voice commands. The spoken content is processed through cloud based voice recognition engines and the string result being returned is compared with a set of predefined strings, defining the command to be sent (i.e., the global action).In all cases, only the field "global action" is updated, while the final HTTP request towards the robotic vehicle is scheduled using a 250ms timer events. This arrangement relieves the smart phone from the CPU intensive dense input data processing (especially data provided by the accelerometer) but requires an extra check, so as only the "fresh" command requests to be served. For simplicity reasons, initialization commands and commands to properly alter the color of the interface buttons are omitted from this description.

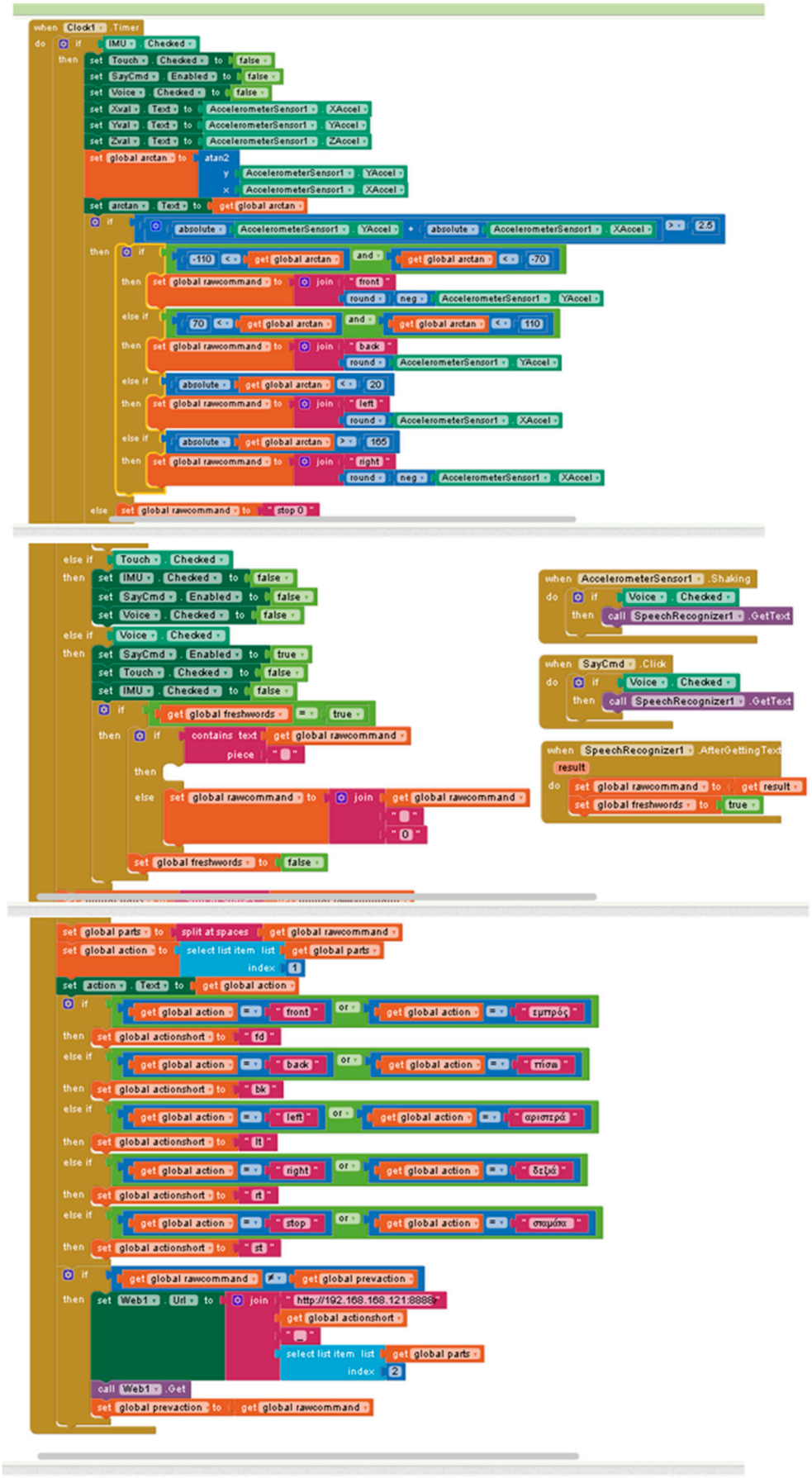

Figure 3. Code blocks in MIT App Inventor environment for interaction with the robotic vehicle.

4. Derived Improvements

The basic platform being presented allows for studying the idiosyncrasies of interacting with a robotic vehicle via gesture or voice commands, using smart phone or tablet devices, over Wi-Fi links. Series of experiments, performed in open space, using this setup, indicated the need for implementing and investigating further derived configurations as well. The most characteristic cases are explained in detail, in Sections 4.1 and 4.2.

4.1. Adding Local Voice Recognition Capabilities

The first derived case focuses on how to operate in voice command mode without relying upon external cloud-based AI services. For this to be done, in the absence of an Internet connection to provide access to a remote server, a locally running software has to tackle the speech recognition issues, to provide interaction with the robotic vehicle. For this reason, in lieu of a smart phone, a raspberry pi unit is used to run the necessary voice recognition service. Embedded boards are now powerful enough to handle such tasks, in a trimmed down manner of course (i.e., for a limited set of words or phrases). In order for the students to become familiar with this alternative, the promising SOPARE [30] software package has been installed and properly parameterized on the raspberry pi unit. Characteristic interaction cases have been assessed, by implementing the necessary plugins, in python, and creating a limited vocabulary through training the system. Typically, 10–15 training samples per each voice command were adequate.

The drastically modular nature of the proposed architecture allows for easy implementation of the necessary changes by making minor code modifications. More specifically, the user can speak towards a USB microphone connected with the raspberry pi unit performing the speech recognition tasks. The latter unit encapsulates the corresponding driving commands into HTTP requests towards the remote raspberry pi unit on the robot. These requests are invoked directly from within the SOPARE plugin mechanism, using the httpie [31] package. The architecture modifications corresponding to this improvement are shown in Figure 4.

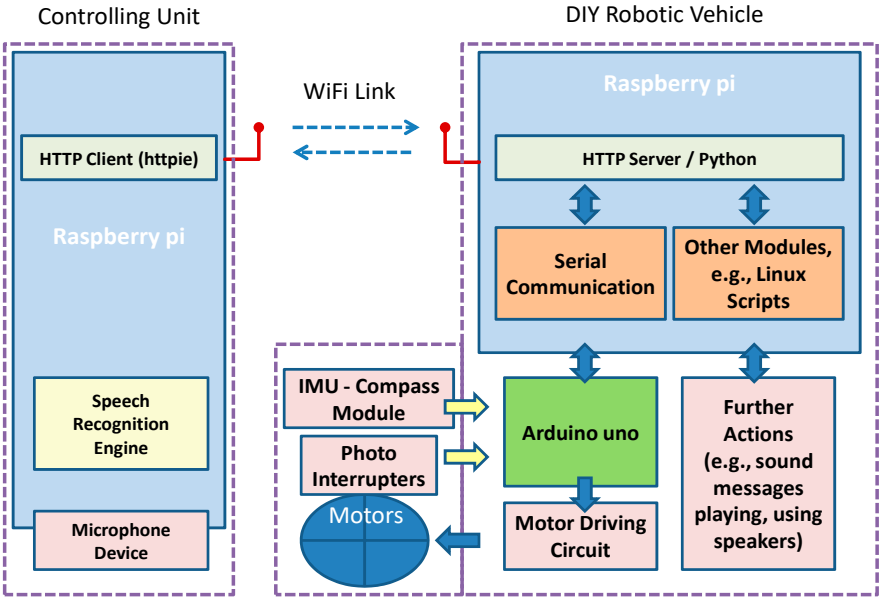

Figure 4. Modifications to support local voice recognition engine over Wi-Fi link.

The message format of the HTTP requests being formed, is identical with the one generated by the MIT App Inventor driving application in the case of using a smart phone and thus, a smart phone may be used in parallel, as a gesture control device or just for viewing purposes, exactly as in the initial scenario. It is important to mention that, grace to the modular architecture, if the user intends to be

close to the robot, the whole voice recognition and command triggering software can be hosted by the raspberry pi unit on the robotic vehicle (and thus, there is no need for the raspberry pi unit at the user's end), without any modification in the preexistent code.

4.2. Extending the Controlling Distance Range

Going a bit further, the second interesting to mention derived case is to provide locally based (offline) voice recognition services and, in parallel, to extend the controlling distance of the robotic vehicle. For this to be done, a LoRa [32] radio interface is added to each communication end. These identical LoRa radio transceiver modules have the form of suitable arduino shields [33], in order to be fit on a corresponding adruino uno unit, each. The accompanying RadioHead library [34], was used to implement a simple client – server schema for communication between these radios, in the arduino environment. The presence of a raspberry pi unit at the user's end, exactly as in the previous case, is necessary. This raspberry pi unit, apart from performing voice recognition tasks, is connected with the arduino unit hosting the LoRa Dragino shield. The voice commands, intercepted by the SOPARE software and interpreted into driving command HTTP requests, are now locally parsed to LoRa messages towards the robot. The architecture modifications that this second improvement implies are shown in Figure 5.

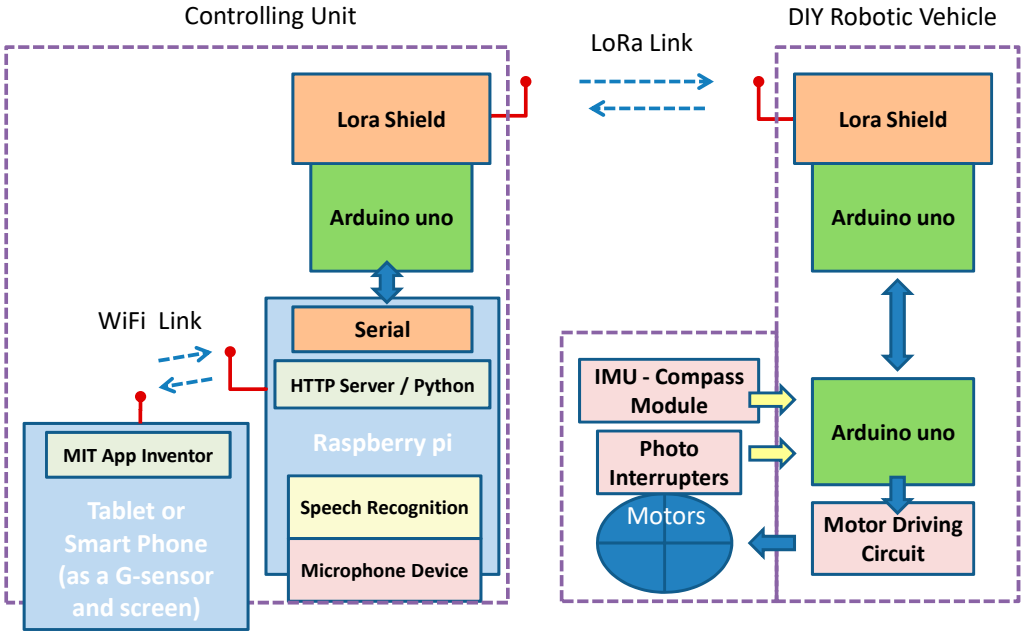

Figure 5. Modifications to support local voice recognition engine over LoRa link.

This configuration generates at user's end redundant local HTTP requests. Indeed, the necessary driving commands could be simply encapsulated into serial commands towards the LoRa equipped arduino unit, by forming the necessary SOPARE plugins. This redundancy is kept for educational and testing purposes and for consistency with smart phone generated commands.

On the robotic vehicle's end, the raspberry pi can be either kept (equipped with an additional arduino and a matching LoRa shield) or omitted (if there is no need for its extra computational power). In the latter case, the arduino hosting the LoRa shield can directly exchange the necessary messages with the arduino unit performing the low level controlling tasks on the robot, through their serial interface. In a more sophisticated manner, only one arduino unit could be used, both for hosting the LoRa module and for controlling the motors of the robot.

Figure 6a depicts the hardware components that are necessary to accompany a raspberry pi unit both for supporting a local voice recognition engine and for extending the robotic vehicle's effective controlling distance through the LoRa radios. Figure 6b depicts the modified robot's layout having the

vehicle's raspberry pi unit totally replaced by the arduino unit hosting the LoRa Dragino shield, while the rest remains untouched.

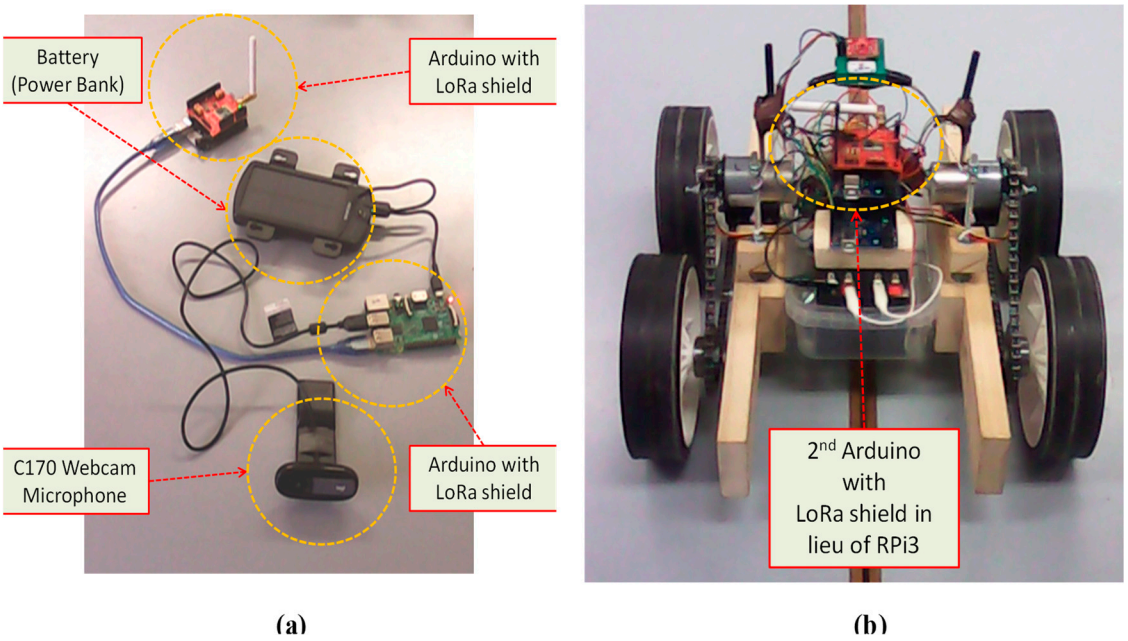

Figure 6. (**a**) Raspberry pi based controlling unit, in case of local voice recognition engine using LoRa link; (**b**) The LoRa equipped robotic vehicle layout.

5. Evaluation of Methods and Discussion

The controlling platform being presented incorporates diverse methods for interaction between humans and a DIY robotic vehicle, offering better flexibility and multilevel assessment skill opportunities, covering a wide range of software and hardware engineering issues as well as pedagogical issues.

5.1. Techical Point of View

During the project implementation, the students performed series of tests in order to evaluate the performance of the proposed variants and made the necessary enhancements.

In terms of delay, the results, using mobile phones, indicated that the cloud based voice recognition mechanism exhibited delays of a few seconds (of about one to three seconds), depending on Internet speed, network traffic congestion and voice service request load conditions. The final command request transfer (over the local Wi-Fi connection) and the execution invocation demanded less than a second to be accomplished. The locally installed (off line) voice recognition mechanism had, in general, a faster total response, varying from less than a second to two seconds.

Evaluating the voice recognition accuracy, for a set of typical commands, the results, when using cloud based services, were better, typically varying from 0.6 to above 0.9, due to the fact that the voice recognition tasks were performed by already well trained AI engines. In case of the local engine, the voice recognition accuracy, for a limited set of commands, varied drastically, typically from bellow 0.5 to 0.9, depending on how meticulously the training process had been done (typically 10-15 times per each command), on the specific selection of the SOPARE's configuration parameters and on the quality of the microphone and audio card being used.

It was verified that cloud based voice recognition techniques provide a quite accurate match between the words being spoken and their textual counterpart and no training stage is required. On the other hand, the possibility of returning textual content completely irrelevant with the words being spoken is not negligible and thus, extra integrity controls should be added to exclude such content.

Added to that, the quality characteristics and even the availability of the underlying Internet connection cannot always be guaranteed. The local (off line) voice recognition engines provide independency from external resources and faster responses, but they require a meticulous training in order to achieve satisfactory matching results and they are computationally exhaustive, sometimes bringing a barely powerful unit, like the popular raspberry pi, to its knees. The response times being experienced are considered as satisfactory ones, taking into account the low speed of the robotic vehicle and the educational nature of the overall approach. Nevertheless, faster alternatives can be investigated, e.g., having as candidate voice recognition software the Snips [35] engine or faster processing boards like the ASUS Tinker Board [36].

The hiring of an HTTP based mechanism to transfer the user's driving commands allows for many educationally meaningful debug points. For instance, the suitable requests to test a specific robot controlling module can easily be generated through a smart phone, a raspberry pi unit or a simple WEB client, thus highlighting the idiosyncrasies of each module and the potential of the network programming techniques. Apparently, this method can lead to considerable delays in communication with the robot (e.g., in case of packet losses) due to the complicated nature (i.e., connection oriented reliable packet delivery) that the TCP [37] protocol, supporting the HTTP requests, has. Alternatives using the UDP [38] protocol are much faster and quite simple and can also be investigated in the future.

In terms of communication range enhancements, the LoRa protocol provided a promising alternative to Wi-Fi links, combining extended distance range and low power consumption. The only tradeoff was the reduced communication rate, compared to the Wi-Fi case. Nevertheless, data rates of 2-3kbps were adequate to serve the client – server communication needs of the discussed human to robot interaction schema. Controlling distances beyond the physical dimensions of the field provided by the University Campus for experimentation (i.e., at the range of 500m) were easily reached. Figure 7 depicts the RSSI [39] values, during communication between the pairing LoRa modules, indicated by blue color, as a function of their distance, at a transmit power configuration of 10dBm. The red points, in the same picture, correspond to the pure Wi-Fi radio case using raspberry pi boards and a conventional access point.

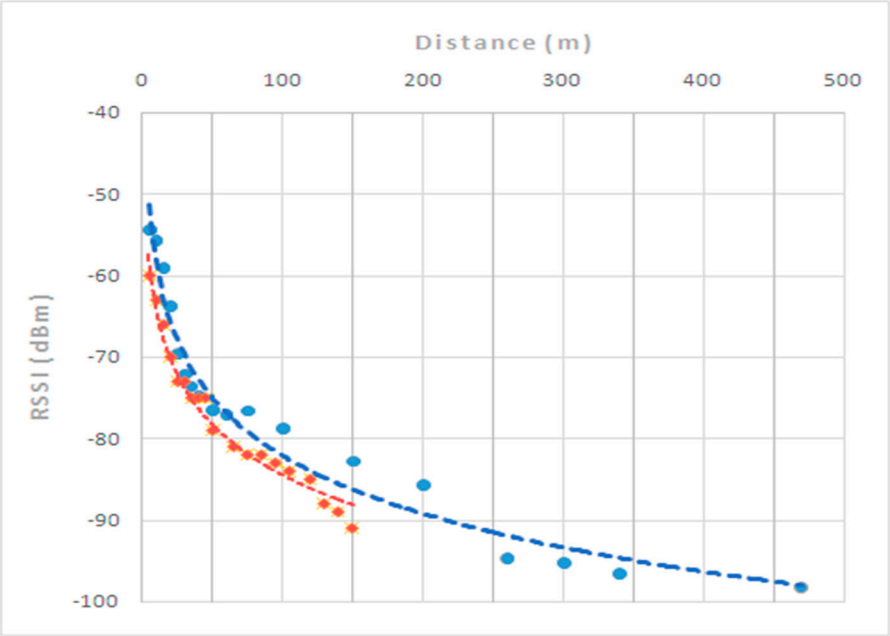

Figure 7. Modifications to support local voice recognition engine over LoRa link.

During experimentation, students also realized that diverse control techniques require careful programming modifications to assure user friendliness. More specifically, gesture based command methods are impressive to use but required a lot of experimentation in order to find the right threshold values allowing for an accurate interaction. In gesture mode, the need for disabling the controlling unit, when not necessary to give a command, was a priority to provide user's comfort during the controlling process. Gesture based interaction is able to generate much more frequent controlling commands, compared with the voice recognition methods. For this reason, voice commands should contain more information than their gesture driven counterparts. For instance, during a voice triggered "turn left" command case, it is crucial to define the extra parameters, as the exact number of degrees that this turn has to perform. This is because the next spoken command could not be processed faster than after one to three seconds, but this is not the case in a gesture triggered "turn left" command, that needs no other arguments, as the consecutive command(e.g., a "move front" command) can be generated and intercepted much faster than three seconds.

The MIT App Inventor programming environment, although initially targeted at pre-collegiate students supporting comparatively simple interaction scenarios, was proved appropriate to support much more composite human-machine interaction cases. Using this tool, students of weak prior programming experience managed to tackle the challenging programming tasks related with the forthcoming technologies, by implementing quite impressive interaction scenarios, in a limited time schedule. The pairing code parts, in C and python, on the raspberry pi or the arduino units, were of reduced complexity but adequate to support a pilot educational implementation.

Alternatives involving less powerful than the raspberry pi systems with Wi-Fi capabilities have been tested as well. These tests involved the WeMosD1 R2 [40] board, which is an ESP8266 based device that combines the user friendly layout of the arduino uno plus a Wi-Fi module for easily implementing HTTP server functionality. However, these systems lack at the moment the large supporting community that both arduino and raspberry pi have.

The ability of the robotic vehicle to dynamically adapt its kinetic behavior to the preferred target velocity or to the turn direction values, simplified the need for detailed and frequently generated commands. It must be noted though that, if the underlying controlling mechanism is not properly designed or configured or if the motion sensors do not provide accurate readings, annoying situations may occur, like a continuous vehicle's swaying left and right or alternating between slow and fast. It is worth mentioning that the inability of the overall control process to be accurate or fast responsive enough, had a welcoming effect: it aimed to reveal the idiosyncrasies of the underlying mechanism, and thus, formed an educationally fruitful environment for the students, who had the opportunity for "hands on" experiences on automatic control paradigm

The colorful bouquet of methods to remotely interact with the robotic vehicle did not eliminate the need of manual overdrive. A man-in-the middle was necessary to inspect the robot's behavior and correct things during failures. Such a controlling mechanism is necessary in order to have a performance landmark and a safe alternative, during the crashes that any experimental educational platform is experiencing during its implementation.

5.2. Pedagogical Point of View

All stages, from the design to the final implementation and testing, formed a meaningful learning environment assisting students and future professionals to demystify several high-end techniques that are used in the modern production, with the smart agriculture era to be a noticeable example.

Despite the limited number of participants (at about 20 persons) a survey was performed to better assess the students' opinions about the platform being implemented. Apart from valuable discussion with the students, during this survey, questionnaires with five points were used, similar to the Likert-type scale [41]. The results were processed using techniques described in [42] and plotted in characteristic bar charts. The main findings of this survey are presented in Figures 8 and 9, while detailed information on the questions used is given in Appendix A.

More specifically, the top part of Figure 8 reflects the students' opinions on the contribution of the proposed techniques to acquire hard skills, i.e., to better understand fundamental software and hardware issues, as well as topics related with the scientific fields of their specialty (i.e., agricultural engineering).They tended to indicate that the specific robotic vehicle interaction AI project assisted in better understanding many technical and scientific issues, like networking, operating system and system interconnection issues. The bottom part of Figure 8 depicts the students' feeling on the capacity of the discussed methodology to assist the acquisition of soft skills, i.e., collaboration, problem solving and presentation capabilities. More specifically, students were enthusiastic about the activity promoting presentation skills while they found satisfactory the contribution of the activity in gaining problem solving and collaboration skills.

Students faced some difficulties in textual programming and in combining the robot's components together and exhibited an increased interest in delivering a "good-looking" construction, despite the abovementioned assembling difficulties.

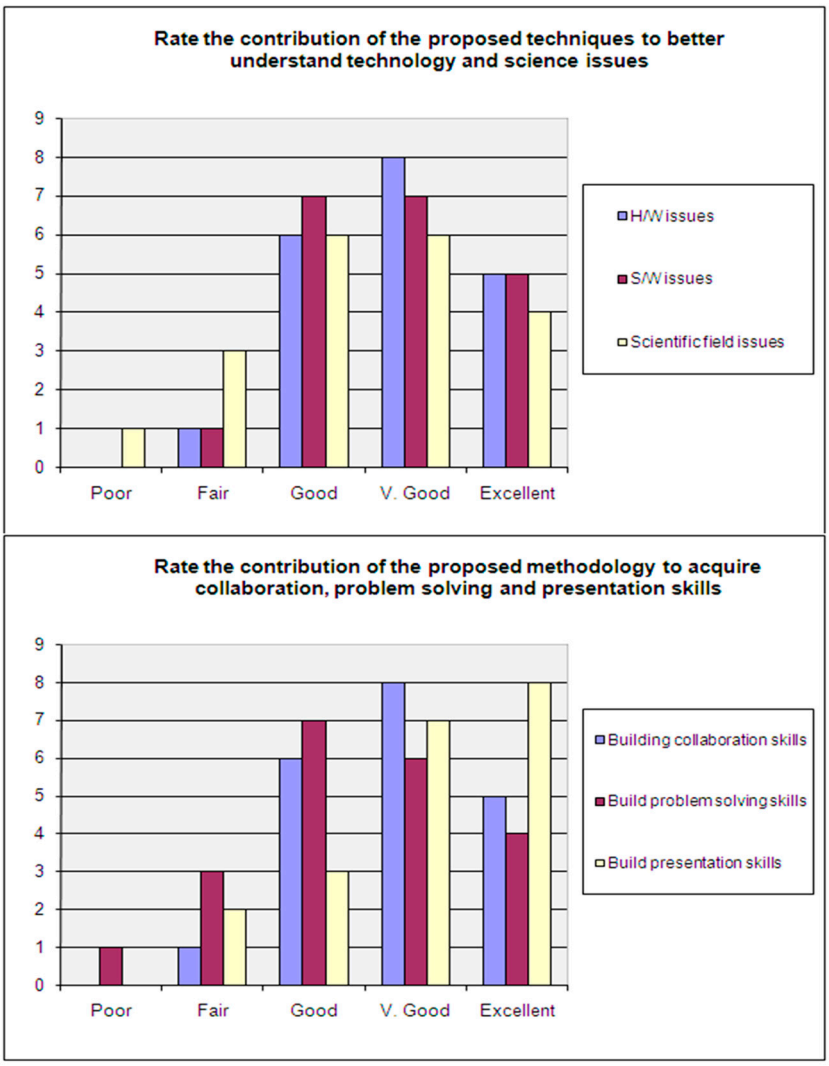

Figure 8. The students' opinions on basic hard (top part) and soft (bottom part) skills they gained.

The top part of Figure 9 assesses the students' opinions on the performance of the group teams, they participated in. The tasks being assigned were completed in a very satisfactory degree, while the cooperation among the members of each team was very decent. It must be noted that the students also found that the peer learning method worked considerably well. The bottom part of Figure 9 presents the students' opinions on the relevance of the proposed activities with the university curriculum.

According to the answers, the vast majority of participants had weak previous experience with similar systems and working methods. The findings also tended to indicate that the students felt that the techniques being used were promoting their potential as future professionals and that they would like to keep similar educational practices active in the university lessons program.

Finally, according to the university statistics based on the official lesson subscription data, the number of students that were willing to participate in this type of activity for the next academic semester was increased by nearly 50% (compared with the number of students participating during the current semester) and this can be considered as a positive sign, as well.

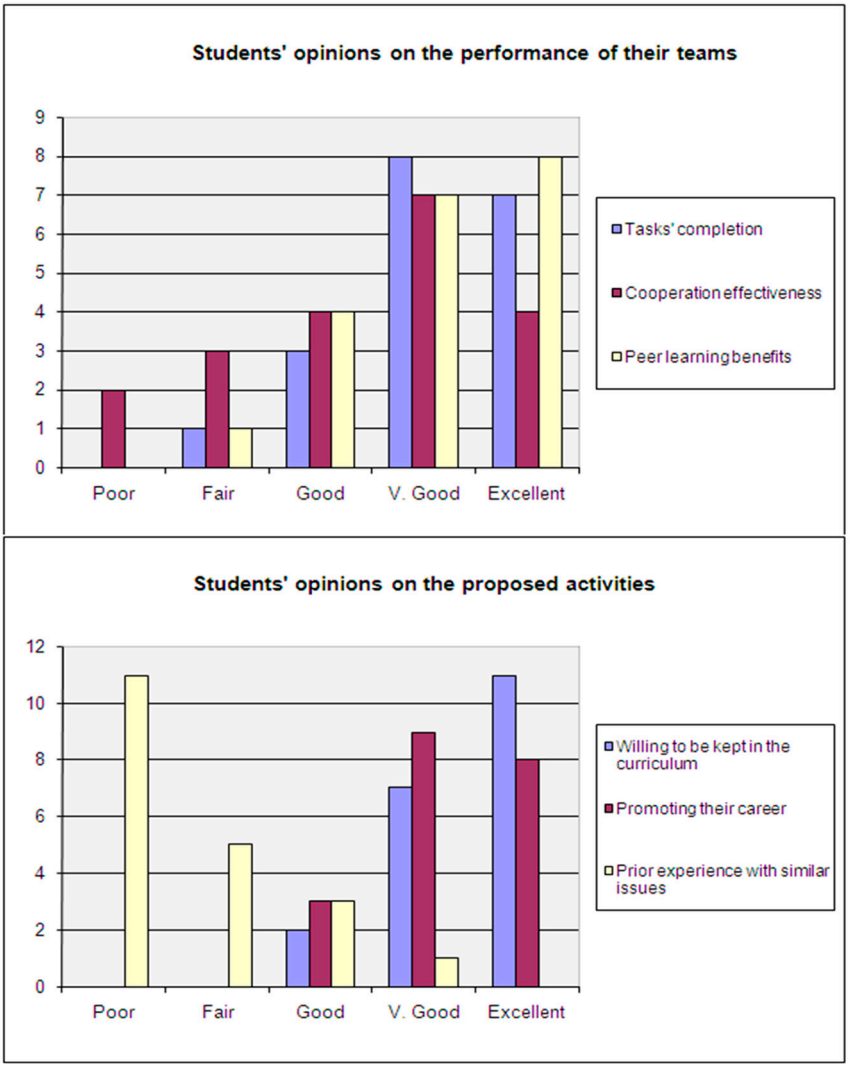

Figure 9. The students' opinions on their team work and performance (top part) and the dynamics of the platform being used (bottom part).

6. Conclusions and Future Work

This paper reported on the learning experiences, the technical solutions and the corresponding perceptions of students of agricultural engineering while trying to find efficient ways to interact with and improve DIY robotic vehicles, intended for simple agricultural use scenarios, in scale. The students participating in this problem based learning activity had a weak programming background and a confined time schedule. Interaction techniques under investigation involved from touch button to gesture and voice recognition, AI flavored methods, mainly through mobile phones or tablets. The robotic platform exploited generic hardware, namely arduino and raspberry pi units, and

incorporated basic automatic control functionality. In conjunction with the MIT App Inventor, important programming tasks have been carried out, via C and python based programming environments. Apart from applications relying upon cloud based AI features, the paper also assessed the case when there is no presence of Internet connection to provide access to a remote speech or voice recognition server. Typically, the remote interaction scenarios were performed through Wi-Fi interfaces, while, later on, the activity involved LoRa interfaces as well, to extend the controlling distance of the robot. In all cases, the setup intended to remain as open and cost effective as possible, to maximize the reusability of electronic components being involved and to exhibit high modularity, thus allowing for several educationally meaningful check points. During experimentation, most students felt that the learning activity had a positive contribution to their acquisition of hard and soft skills. According to their perceptions, by getting involved in designing, modifying and developing stages, participants assessed the whole robotic vehicle interaction platform as beneficial for their professional development and they would like to keep similar activities in the university lessons curriculum.

The communication and processing delays, using raspberry pi and arduino units, were acceptable for a pilot implementation, but the need for further experiments with faster processing boards, better audio equipment, more communication protocol alternatives and further code optimizations should be a priority for the near future. Solar panel assistance would be a valuable asset as well. The proposed low cost human machine interaction methods could be further adapted to provide a bouquet of assistive solutions for the elderly or for persons with disabilities. Furthermore, as LoRa interfaces drastically extend the effective controlling distance of the robot, more sophisticated methods for monitoring the vehicle will be investigated, including fusion with GPS data and additional sensors for a more precise vehicle's trace visualization and planning or machine vision techniques, to facilitate the guidance process and increase the autonomy of the robot. Finally, implementing techniques for assessing the energy fingerprint of robot's activity, under different working scenarios and with larger and more specialized electromechanical layout, tailored for a successful real world operation in the agricultural era, would provide valuable data in the future.

Author Contributions: D.L. conceived the idea for the paper, designed most of the platform, was responsible for the implementation and the evaluation process and wrote the majority of the paper. K.G.A. helped by providing valuable advice and undertook most of the administration tasks.

Funding: This research received no external funding. The journal publication fee is paid for by Konstantinos G. Arvanitis via vouchers.

Acknowledgments: We would like to thank the personnel of the Farm Machinery Laboratory and the students of the Department of Natural Resources Management and Agricultural Engineering of the Agricultural University of Athens for their valuable assistance, during the implementation and the evaluation stages of the proposed platform.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

This appendix contains, in Table A1, a detailed form, the main questions used in the survey described in Section 5. Means for the survey (satisfaction degree) are also provided per question.

Table A1. The main questions used in the survey.

Question Items	Mean
Rate the contribution of the proposed techniques to better understand hardware issues (networking, motor driving, mechanical parts assembling, etc.).	3.85
Rate the contribution of the proposed techniques to better understand software issues (network programming, operating systems, AI, interface design, etc.).	3.80
Rate the contribution of the proposed techniques to better understand scientific field issues provided by your university curriculum.	3.45
Rate the contribution of the proposed techniques to become better in collaboration issues.	3.85
Rate the contribution of the proposed techniques to become better in problem solving issues.	3.60
Rate the contribution of the proposed techniques to become better in presenting a topic in public.	4.05
Assess the degree of completion of the assigned tasks by your team, at the end of the project.	3.90
Assess the degree of the cooperation among the members of your team.	3.40
Rate to what extent you are satisfied of your peers, assisting you to better understand the issues needed to be accomplished.	4.10
Rate to what extent you would like to keep the proposed activities in the university lessons curriculum.	4.45
Rate to what extent the technologies and practices being used would promote your career.	4.25
Assess your previous experiences using similar devices and practices in your student life.	1.70

References

1. Arduino. Arduino Uno Board Description on the Official Arduino Site. 2019. Available online: https://store.arduino.cc/arduino-uno-rev3 (accessed on 18 April 2019).
2. Raspberry. Raspberry Pi 3 Model B Board Description on the Official Raspberry Site. 2019. Available online: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/ (accessed on 12 April 2019).
3. Alimisis, D.; Moro, M.; Menegatti, E. Preface. In *Educational Robotics in the Makers Era, Advances in Intelligent Systems and Computing 560*; Alimisis, D., Ed.; Springer: Berlin, Germany, 2017.
4. Alimisis, D. Educational Robotics: Open questions and new challenges. *Themes Sci. Technol. Educ.* **2013**, *6*, 63–71.
5. Doran, M.V.; Clark, G.W. Enhancing Robotic Experiences throughout the Computing Curriculum. In Proceedings of the SIGCSE'18, Baltimore, MD, USA, 21–24 February 2018; ACM: New York, NY, USA, 2018; pp. 368–371.
6. Alimisis, D.; Loukatos, D. STEM education post-graduate students' training in the eCraft2Learn ecosystem. In Proceedings of the 2nd International Conference on Innovating STEM Education, Athen, Greece, 22–24 June 2018.
7. Eaton, E. Teaching Integrated AI Through Interdisciplinary Project-Driven Courses. *AI Mag.* **2017**, *38*, 13–21. [CrossRef]
8. FAO 'Climate-smart Agriculture Sourcebook'. 2013. Available online: http://www.fao.org/3/i3325e/i3325e.pdf. (accessed on 18 April 2018).
9. Symeonaki, E.G.; Arvanitis, K.G.; Piromalis, D.D. Cloud computing for IoT applications in climate-smart agriculture: A review on the trends and challenges towards sustainability. In *Innovative Approaches and Applications for Sustainable Rural Development*; Odoridis, A., Ragkos, A., Salampasis, M., Eds.; Springer: Cham, Switzerland, 2019; Volume 29, pp. 147–167.
10. Bechar, A.; Vigneault, C. 'Agricultural robots for field operations. Part 2: Operations and systems'. *Biosyst. Eng.* **2017**, *153*, 110–128. [CrossRef]
11. Krishna, K.R. *Push button Agriculture: Robotics, Drones, Satellite-Guided Soil and Crop Management*; Apple Academic Press: Oakville, ON, Canada, 2016; ISBN 978-1-77188-305-4.
12. UK-RAS Network: Robotics & Autonomous Systems Agricultural robotics: The Future of Agricultural Robots, UK-RAS White Papers. Available online: https://arxiv.org/ftp/arxiv/papers/1806/1806.06762.pdf. (accessed on 24 April 2019).

13. Blikstein, P. Digital Fabrication and 'Making' in Education: The Democratization of Invention. In *FabLabs: Of Machines, Makers and Inventors*; Walter-Herrmann, J., Büching, C., Eds.; Transcript Publishers: Bielefeld, Germany, 2013.
14. Schon, S.; Ebner, M.; Kumar, S. The Maker Movement Implications from modern fabrication, new digital gadgets, and hacking for creative learning and teaching. *Laia Canals*. P.A.U. Education, Ed.; eLearning Papers. 2014, pp. 86–100. Available online: https://www.researchgate.net/publication/263655746_The_Maker_Movement_Implications_of_new_digital_gadgets_fabrication_tools_and_spaces_for_creative_learning_and_teaching (accessed on 26 August 2019).
15. Kahn, K.; Winters, N. Child-Friendly Programming Interfaces to AI Cloud Services. In *Data Driven Approaches in Digital Education. EC-TEL 2017. Lecture Notes in Computer*; Lavoué, É., Drachsler, H., Verbert, K., Broisin, J., Pérez-Sanagustín, M., Eds.; Science; Springer: Cham, Germany, 2017; Volume 10474.
16. Available online: https://en.wikipedia.org/wiki/Artificial_intelligence. (accessed on 12 April 2019).
17. Wi-Fi–IEEE 802.11 Standard. 2019. Available online: http://www.ieee802.org/11/ (accessed on 10 May 2019).
18. MIT App Inventor. 2019. Available online: http://appinventor.mit.edu/explore/. (accessed on 10 May 2019).
19. Python Programming Language. 2019. Available online: https://en.wikipedia.org/wiki/Python (accessed on 10 April 2019).
20. Borges, J.; Dias, T.G.; Cunha, J.F. A new group-formation method for student projects. *Eur. J. Eng. Educ.* **2019**, *34*, 573–585. [CrossRef]
21. Markham, T. Project Based Learning. *Teach. Libr.* **2011**, *39*, 38–42.
22. Smith, B.L.; MacGregor, J.T. What is collaborative learning. In *Collaborative Learning: A Sourcebook for Higher Education*; Goodsell, A.S., Maher, M.R., Tinto, V., Eds.; National Center on Postsecondary Teaching, Learning, & Assessment, Syracuse University: Syracuse, NY, USA, 1992.
23. Anderson, L.W.; Krathwohl, D.R. A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives. In *Pearson Education Group*; Allyn & Bacon: Boston, MA, USA, 2001.
24. King, A. Structuring Peer Interaction to Promote High-Level Cognitive Processing. *Theory Pract.* **2002**, *41*, 33–39. [CrossRef]
25. HTTP, Hypertext Transfer Protocol. 2019. Available online: https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol (accessed on 11 May 2019).
26. BaseHTTPServer. Available online: https://wiki.python.org/moin/BaseHttpServer (accessed on 11 May 2019).
27. OS–Miscellaneous Operating System Interfaces (Python). 2019. Available online: https://docs.python.org/3/library/os.html (accessed on 12 April 2019).
28. mpg321-A Simple and Lightweight Command Line MP3 Player. 2019. Available online: http://mpg321.sourceforge.net/ (accessed on 12 April 2019).
29. eSpeak–Speech Synthesizer. 2019. Available online: http://espeak.sourceforge.net/ (accessed on 20 April 2019).
30. Sound Pattern Recognition–SOPARE. 2019. Available online: https://www.bishoph.org/ (accessed on 10 April 2019).
31. HTTPie, A command line HTTP client. 2019. Available online: https://httpie.org/ (accessed on 18 April 2019).
32. LoRa, LoRa Protocol Description on Wikipedia. 2019. Available online: https://en.wikipedia.org/wiki/LoRa (accessed on 14 April 2019).
33. Draguino, The LoRa Draguino shield for arduino. 2019. Available online: http://www.dragino.com/products/module/item/102-lora-shield.html (accessed on 12 April 2019).
34. RadioHead, The RadioHead library to Support LoRa Modules. 2019. Available online: https://www.airspayce.com/mikem/arduino/RadioHead/ (accessed on 11 April 2019).
35. Snips Voice Recognition Engine. 2019. Available online: https://snips.ai/technology/ (accessed on 12 April 2019).
36. ASUS Tinker Boad Credit Card Computer. 2019. Available online: https://www.asus.com/us/Single-Board-Computer/Tinker-Board/ (accessed on 12 April 2019).
37. TCP, Transmission Control Protocol. 2019. Available online: https://el.wikipedia.org/wiki/Transmission_Control_Protocol (accessed on 10 May 2019).
38. UDP, User Datagram Protocol. 2019. Available online: https://en.wikipedia.org/wiki/User_Datagram_Protocol (accessed on 10 May 2019).
39. RSSI, Received Signal Strength Indicator–RSSI. 2019. Available online: https://en.wikipedia.org/wiki/Received_signal_strength_indication. (accessed on 12 April 2019).

40. WeMos D1 R2. 2019. Available online: https://wiki.wemos.cc/products:d1:d1 (accessed on 11 April 2019).
41. Likert, R. A Technique for the Measurement of Attitudes. *Arch. Psychol.* **1932**, *140*, 1–55.
42. Robbins, N.B.; Heiberger, R.M. Plotting Likert and Other Rating Scales. In Proceedings of the 2011 Joint Statistical Meeting, Miami Beach, FL, USA, 30 July–4 August 2011; pp. 1058–1066.