Do-It-Yourself Robot Kit for Educators

Proposal by Spyridon J. Rallis
for Google Summer of Code 2021

Personal Details

Name	:	Spyridon J. Rallis
University	:	Hellenic Mediterranean University; Dpt of Electrical & Computer Engineering
Email	:	`spyros.rallis@gmail.com`
Telephone	:	+30.697.8490700
Residence	:	Stamata, Attica, Greece
Languages	:	Greek (native), English (professional working proficiency)

Currently a graduand, I like to work on projects involving wireless sensor networks, the Internet of Things, prototyping platforms, sensors, and 3D printing and design. I have completed all compulsory courses (the last requirement towards my degree is my BSc thesis), having adequate time for my GSoC project. If I am selected, I shall be able to devote 40 hours / week, although this is not a hard limit, as I like to consider myself a "doer".

Why This Project?

THE VALUE OF THE PROJECT

An educational robot kit is one of the simplest yet most effective ways to educate someone on electronics and programming. It requires the learner to program and operate the robot, strengthening their logical thinking, while providing prompt visual feedback. And while the inclusion of many different sensors and actuators can be quite challenging, it is also fun!

THE VALUE OF EDUCATION

Education is of vital importance for our society, and I try to contribute to it as much as possible. During my years as a student I have invested countless hours being a volunteering laboratory assistant educating others, as well as myself. I have also taken part in the creation of the [OpenIn] Erasmus+ training courses, among a few other programs.

INSPIRATION AND MOTIVATION

Working on a project that sparks your inspiration is essential in order to stay motivated, and I have long been passionate about electronics, programming and DIY.

Technical Knowledge & Equipment

ELECTRONICS

I enjoy working with numerous microcontroller boards and computing platforms, including Arduino, ESP32 and Raspberry Pi, as well as bare-metal microcontrollers (but mostly 8-bit AVRs). I feel confident around electronic components, being able to prototype and test various circuits as well as design some on my own.

Apart from controllers, I have worked with many sensors (digital and analog interfaces), actuators and communication devices, including packet radios and LoRa.

Furthermore, I have successfully designed, prototyped and tested PCBs, both myself (photo-transfer method) and through 3rd-parties (oshpark.com, jlcpcb.com). I have done both through-hole and SMD boards, although I have a solid preference for SMDs.

PROGRAMMING

I am mostly confident with Wiring and C++, although I can also code in Python without too much trouble. Furthermore, I have also used flow based programming tools like Node-RED, which is my tool of choice for rapid back-end development.

3D PRINTING & TOOLS

Both additive and subtractive manufacturing awes me. And whilst getting a CNC in one's home is not really a feasibly option, I am fortunate enough to own and use a 3D printer. I know my way around printing different materials and models, troubleshoot prints, as well as design and test my own models.

For the last few years I have been compiling a small but nicely equipped lab, having multimeters, an oscilloscope, a dual-channel power supply, a soldering station and MANY electronic components, ranging from passives to store-bought modules.

What I'd Like to Do

IMPROVE 3D DESIGNS

While the current design is indeed printable, there are quite a few parts that require some experience printing them (e.g. supports needed). These should be revised in order for the kit to be easily printable without being challenging. Plus, I'd like to try making the kit even more DIY friendly, replacing as many hard-to-find store-bought components for 3D printed ones.

A COMMON SENSOR INTERFACE

Most sensors define their own way of communicating with the outside world (i.e., the controller). Analog sensors require the user to utilise an ADC to read their output voltage (and then convert that to a human-readable value with some equation), while for digital ones, the manipulation of specific registers over I2C is a common requirement. But that is not user-friendly at all. A beginner-friendly robot kit should be more “plug-and-play”.

A common sensor interface will help with that. Tiny-little PCBs can be designed to host the bare sensor alongside with a small microcontroller. The “companion-MCU” will perform the operations needed to get the raw data from the sensor, and then send human-readable numerical data to the main controller via I2C. Wiring is standardised, and the user can program the robot to “getTemperature()” instead of “getVoltage() * ($V_{REF}/ADC_{RESOLUTION}$) - OFFSET”.

The added bonus of this approach? If the I2C addresses are defined by convention (e.g., 0x61 for temperature, 0x62 for humidity, et.c.) the main controller can automatically detect the type of sensor that is connected to it!

CUSTOM-BUILT PCB

Wiring on a breadboard apart from being error-prone, is also a major time waster for classes. Breadboards are handy for “quick-and-dirty” prototyping, but in a time-constrained environment like a course there has to be a better way. With a custom built PCB that will house most of the essential components, students will have more time to experiment and program their robot, rather than trying to debug a loose connector. A custom PCB, together with the common sensor interface, could greatly increase the effectiveness of the kit!

Proposal Timeline

WK 1	The first week will be spent mostly on 3D printing and assembling the robot kit, since a functional version will be required prior to improvising the design. Furthermore, I will pick the sensors for the platform to integrate, since they must fulfil certain criteria.
WK 2	The sensors and microcontrollers will be purchased in order to be evaluated and I will design and order the circuit boards that will host the sensors.
WK 3	Hopefully, the components and the PCBs will be delivered in less than a week's time, in order for me to dive into coding. I suspect some 'stall time' there, waiting for the PCBs to arrive.
WK 4 **WK 5**	Starting with week 4, a number of different sensor-hosting PCBs will be functional, and their controller-side code ready, although a few bugs may still need ironing-out. This is the time for the bigger common sensor interface library to be written. While the sensor boards now provide human-understandable numerical values, there is still the need for an easy and automated way to connect to them from within the exercises.
WK 6 **WK 7**	With the common sensor interface functional, it's time to re-write the existing exercises in order for them to utilise the new interface, and why not add a few more scenarios?
WK 8 **WK 9**	Coming up in weeks 8 & 9 I will design and order the bigger PCB that will host the controller and all required peripherals.
WK 10	The final week will be mostly devoted to tests, debugging and issue hunting. I will make sure that there are as few issues as possible on the software in order for the platform to be used immediately

More Information / Notes

NO TIME FOR 3D DESIGNS?

I suspect stall times at various stages of the projects that can be utilised for the 3D designs. Apart from that, 3D printing takes considerably more time that designing easy and ‘squarish’ parts, which means that many tasks can run in parallel. I opted not to strictly devote a week or two for 3D designs, but rather take advantage of stall periods here and there.

IS THE TIMELINE FLEXIBLE?

In coordination with the project mentors many parts can be given greater (or lesser) importance.

WHAT ABOUT AFTER GSoC?

I tend to support the projects that I undertake for the long-term even if I am not the ‘main’ maintainer any more. That means I will probably make revisions on the PCBs and I will fix bugs even after GSoC.

Thank You!