Ref. Ares(2016)4893285 - 31/08/2016

MOBILE ROBOT CONTROLLER PROTOTYPE [DELIVERABLE 5.1]

Wilfried Lepuschitz

ER4STEM - EDUCATIONAL ROBOTICS FOR STEM

The ER4STEM project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 665972

TABLE OF CONTENTS

The ER4STEM project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 665972

TABLE OF REFERENCES

The ER4STEM project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 665972

DOCUMENT REVISION HISTORY

Version Number	Date	Description	Author
V1	10.08.2016	First Version	Wilfried LEPUSCHITZ
V2	12.08.2016	Review	Ivaylo GUEORGUIEV
V3	30.08.2016	Final Version	Wilfried LEPUSCHITZ

CONTRIBUTORS

Name	Beneficiary	Section affected
Wilfried LEPUSCHITZ	PRIA	all
Clemens KOZA	PRIA	2, 3
Ivaylo GUEORGUIEV	ESI CEE	all

DISCLAIMER

This Deliverable reflects only the author's view. Neither the author(s) nor the REA are responsible for any use that may be made of the information it contains.

The ER4STEM project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 665972

1 EXECUTIVE SUMMARY

1.1 ROLE/PURPOSE/OBJECTIVE OF THE DELIVERABLE

This document describes the prototype of Pocket Bot, a graphical programming environment for the educational robotics controller Hedgehog to be used on smartphones. In the beginning, the Hedgehog controller is introduced, which was mainly developed during the project SCORE! (funded by the Austrian Research Promotion Agency). Then, insights about Pocket Code are given, which acted as basis for Pocket Bot. Finally, details about Pocket Bot are reported.

1.2 RELATIONSHIP TO OTHER ER4STEM DELIVERABLES

Any learnings from educational robotics platforms, and therefore also from Pocket Bot, influence the framework developed in ER4STEM. Therefore, a link to the deliverables of WP1 exists. Future developments of the graphical programming environment in project year 2 will be influenced by the learnings of the framework of year 1.

It is envisaged that Pocket Bot will be used in future activities in ER4STEM. Mainly this will be at workshops conducted in WP2, which is why a relationship to further deliverables of this work package exist.

Pocket Bot shall be added to the repository developed in WP 5, which means a linkage to D5.4 will exist depending on that deliverable's granularity.

1.3 STRUCTURE OF THE DOCUMENT

Section 2 introduces the Hedgehog controller. Section 3 reports on Pocket Bot. The remaining sections give a summary and conclusion and provide a glossary as well as references.

The ER4STEM project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 665972

2 INTRODUCTION TO HEDGEHOG (ANDRIX)[1]

School budgets are often too small for introducing robotics at young ages due to the required investments in equipment. This led to the development of Hedgehog (see Figure 1), a low priced robot controller that involves smartphones and similar mobile devices. Hedgehog was previously denoted as Andrix, when it was only possible to connect it with smartphones based on Android. Hedgehog was initially developed by PRIA in the frame of the project Smartphone Controlled Robots for Education and Industry (SCORE!), which was funded by the Austrian Research Promotion Agency. The aim was to bring robotics to the personal, educational and science domains by providing a simple, flexible and affordable robot controller.

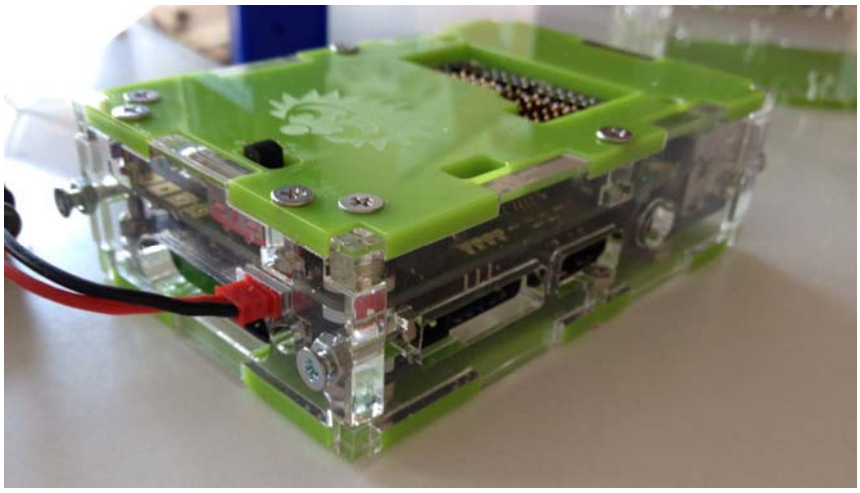

Figure 1: Hedgehog controller in its case.

Hedgehog fits well to the criteria for good practice we have identified in D.1.1 "BEST PRACTICE AND REQUIREMENTS" (pp 35-37), more specifically to criteria such as:

- Tools - Technology used: it is considered as good practice if the educational robotic event is based on technology that follows the latest trends and is similar to what young people are using in their everyday life e.g. mobile and cloud solutions.
- Tools - Type of artefacts produced: it is considered as good practice if the artefacts produced during the educational robotic event are interesting and engaging; participants are interested to use the artefacts and to apply them in different domains of their lives.
- Sustainability - Cost of the activity: The activity requires materials or tools that are reasonably priced compared to other related activities. An example of a relatively cost-effective technology with good cost-to-quality ratio is the LEGO technology and as for a relevantly low initial investment the Arduino technology.

[1] Large parts of the content of this section was presented as an article during the Robotics in Education (RiE) 2015 conference as a part of scientific dissemination activities of the project "SCORE!". Clemens Koza, Christoph Krofitsch, Wilfried Lepuschitz, Gottfried Koppensteiner: "Hedgehog: A Smart Phone Driven Controller for Educational Robotics"

The ER4STEM project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 665972

2.1 ARCHITECTURE OF HEDGEHOG

Figure 2 shows the Hedgehog architecture. The Hedgehog controller is composed of two layers, the high-level control (HLC) and low-level control (LLC), named so for their deliberative and reactive roles, respectively [4]. The HLC consists of the mobile device and its software; the LLC is composed of the hardware controller (HWC) and the software controller (SWC). Inter-module communication is done using a simple stateless protocol.

The HWC is responsible for time-critical tasks and is effectively a black box to the user. These tasks include generating pulse-width modulated signals for motors and servos, or communicating with external hardware such as an iRobot Create via additional interfaces like universal synchronous/asynchronous receiver/transmitter (USART) or inter-integrated circuit (I^2C). The SWC is a single-board computer on which the user can run arbitrary applications and provides connectivity to HLC devices via WiFi or Ethernet. The current implementation uses a Raspberry Pi 3B as SWC and an Android device as HLC, also an iOS variant is currently in development.

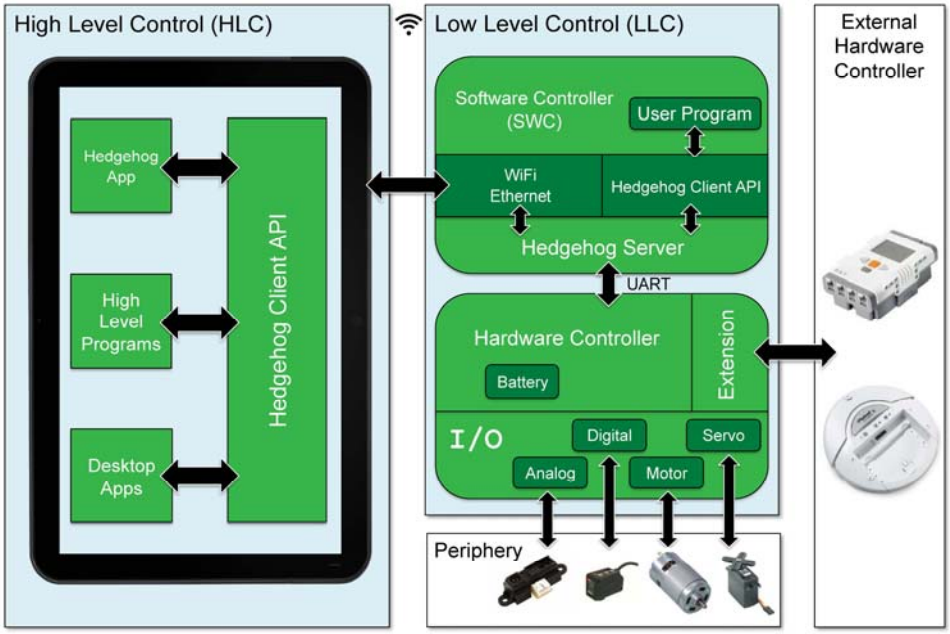

Figure 2: Architecture of Hedgehog.

Like the SWC, the HLC can run arbitrary software – apps in the case of Android or iOS – but the use of the Hedgehog app is assumed for education purposes. It provides

- overview screens to test actuators and sensors. This eliminates the need for a screen found on most conventional educational controllers.
- an integrated development environment (IDE) with debugging facilities to create programs to run on the SWC. This makes separate development machines unnecessary for courses.

By writing applications for the SWC, development license issues, especially on the iOS platform, are avoided.

The ER4STEM project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 665972

2.2 HEDGEHOG IN USE

The Hedgehog app's IDE supports programming in Python. To accommodate for the heavy use of punctuation in most programming languages, additional code template buttons, e.g. for inserting loops, are provided in the IDE.

User programs can either be run on an HLC device, sending commands to the Hedgehog Controller via WiFi or Ethernet, or downloaded to the controller once to run locally. The former has the advantage of quicker write-test cycles, while the latter provides better latency. Users can even combine the approaches – write programs that run on one controller, but control multiple Hedgehogs at the same time.

For debugging, commands such as "Debugging Break Action", which suspends the running SWC program, are available to the HLC. These commands correspond to commands and events of Gnu Debugger (GDB), although the use of GDB on the SWC is an implementation detail. Figure 3 shows two screen shots from a Hedgehog debug session.

The Hedgehog controller represents a powerful platform for educational robotics. Using smart phones facilitates existing resources and provides a well-understood, attractive working environment. At the same time, Hedgehog can leverage future improvements in those devices. Overall, this makes Hedgehog a worthwhile low-cost, long-term investment for schools.

Figure 3: Two screenshots (cropped) from a debug session. Left: Standard output. Right: Setting breakpoints in the source code while the program is suspended.

The ER4STEM project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 665972

3 GRAPHICAL PROGRAMMING FOR HEDGEHOG: POCKET BOT

To further broaden the Hedgehog Light ecosystem towards younger audiences, it was integrated with a graphical programming environment called Pocket Code [1]. The resulting extension for applying Pocket Code to educational robotics was denoted as Pocket Bot.

3.1 POCKET CODE

A Pocket Code program consists of a background and different objects. Both of these can have scripts that are triggered by different events, such as the program starting, an object being tapped, or two objects colliding. Scripts are composed of sequential blocks that each accomplish a tasks. Blocks are separated into categories, such as “Control”, “Motion”, “Data”, etc.

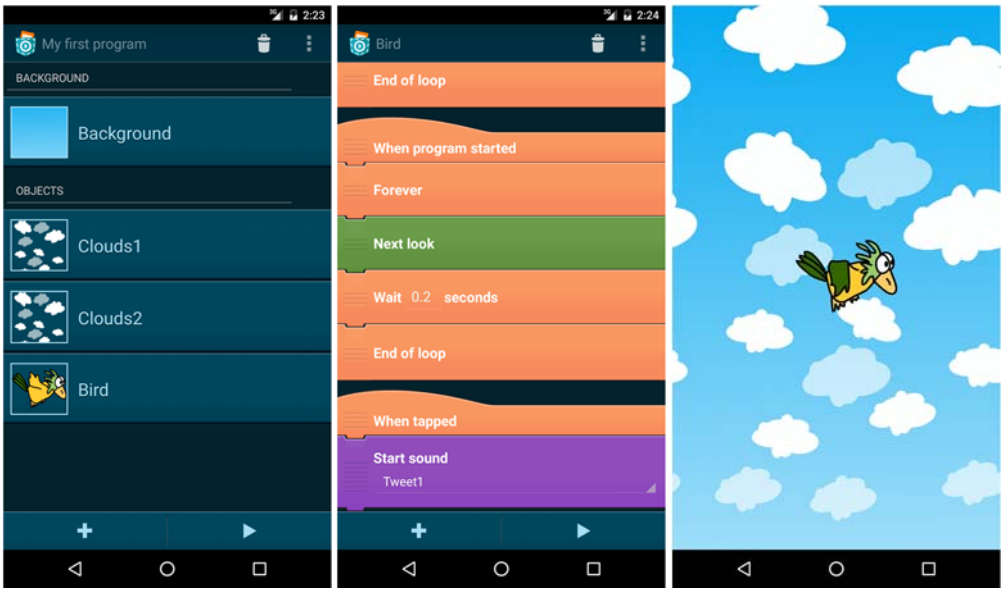

Figure 4: The demo program’s home screen (left); part of the Bird object’s script – loop through looks for “flapping”, and tweet when tapped (middle); and the program’s stage (right).

As an example for Pocket Code, the middle of Figure 4 shows the script of a demo app’s Bird object. When the program is started (using the “Play” button at the bottom of the screen), the “When program started” script is executed: it runs indefinitely and cycles through the Bird object’s different looks, making it appear to flap every 0.2 seconds. The “When tapped” script plays a sound whenever the bird is tapped. The right image shows that “stage”, which is shown when the program starts. The bird, clouds, and background are visible. As can be seen, the bird’s sprite is different from the one in the left image, caused by the “flapping” loop.

Instead of a given number, arbitrary formulas can be used to calculate parameters like a delay, for example relying on an object’s velocity or variable values. In contrast to most imperative languages, there is a clear distinction between blocks (procedures with side effects and without return values, which can appear as instructions) and functions (without side effects but with a return value, which can only appear as part of a formula).

As illustrated by the demo program, Pocket Code allows users to create simple imperative programs, teaching control structures such as loops and conditionals, as well as usage of parameters and

The ER4STEM project has received funding from the European Union’s Horizon 2020 research and innovation program under grant agreement No. 665972

calculations. Moving objects, changing looks and emitting sounds provides users with immediate feedback – just as controlling robots – and thus encourages them to experiment.

3.2 HEDGEHOG LIGHT JAVA API

The original implementation of the Hedgehog Light API is in Python, whereas Pocket Code and Android are based on Java. Integrating Hedgehog Light and Pocket Code is facilitated by Hedgehog's use of a simple protocol based on cross-platform available libraries.

Messages are encoded and decoded using Google Protobuf [2], which provides a simple API, efficient encoding, and facilitates compatibility in the face of protocol changes. Encoded messages are transmitted using ZeroMQ [3], which provides high-level network connections with automatic reconnect capability, improving reliability. Both Protobuf and ZeroMQ are compatible with Java and Android, making the use in Pocket Code easy. For Pocket Bot, only a subset of the Hedgehog Light API was ported to Java at this stage; advanced protocol features such as interacting with subprocesses were intentionally left out. The port was actually implemented in Scala, a Java compatible language that provides more conciseness in many regards.

3.3 MAPPING THE API TO POCKET CODE ELEMENTS

To develop Pocket Bot as a robot extension to Pocket Code, a new category, "Hedgehog", was added to the app (see Figure 5). As described earlier, there are two main elements for bringing functionality to Pocket Code: bricks and functions. These two map cleanly to actuators and sensors. The following elements were already implemented:

- a brick for setting a motor's power,
- a brick for setting a servo's position,
- a brick for turning a servo off,
- a function for reading digital sensor values, and
- a function for reading analog sensor values.

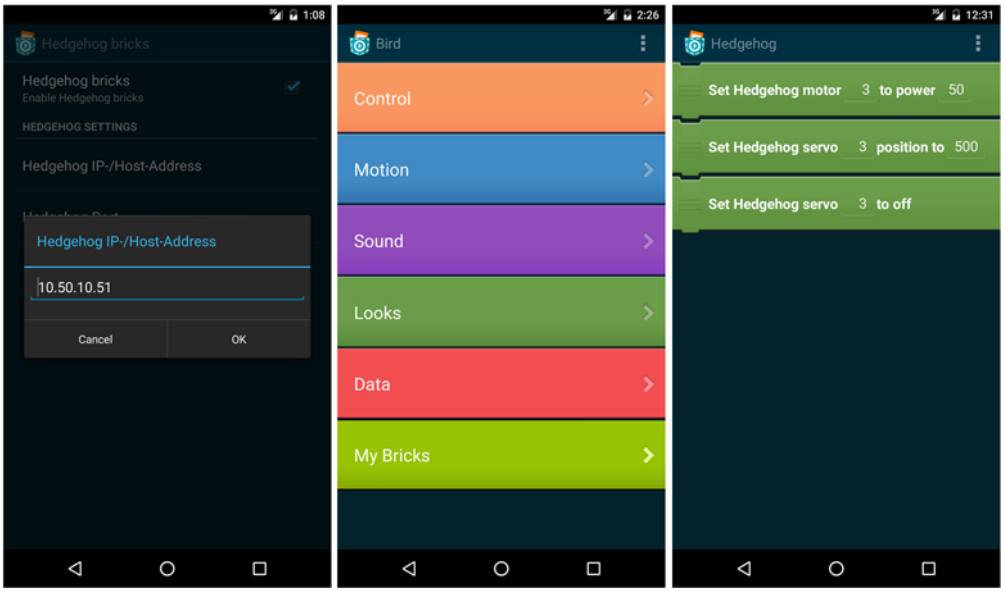

Figure 5: Hedgehog settings (left); added Hedgehog category (middle); and blocks (right).

The ER4STEM project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 665972

3.4 INTEGRATION

Pocket Code does not provide a programming interface that would allow Pocket Bot to be developed as a third party plugin to the app, but it uses – at least in the relevant places – clean separation of concerns, which makes it possible to make the necessary changes without interfering too much with the rest of the app. However, this means that Pocket Bot cannot be used as an optional part of the app: it needs to be distributed either with the official, main Pocket Code app, or as a completely independent fork.

To integrate Pocket Bot, some changes were done in the core of Pocket Code; these included:

- adding brick field constants such as HEDGEHOG_MOTOR_PORT and the HEDGEHOG resource (indicating whether a brick requires a Hedgehog connection) to the Brick class, and adding default values for those fields to the BrickValues class;
- adding the Hedgehog brick category in the CategoryBricksFactory and BrickCategoryFragment classes; and
- adding methods to the ActionFactory, which is responsible for creating the actions corresponding to bricks in a script.

In addition, there are functions for reading out values, i.e. sensors. Modifications necessary in this regard included:

- adding function constants for reading analog and digital sensors in Functions;
- implementing these functions' functionality in FormulaElement;
- registering the functions in various classes, including FormulaEditorListFragment and InternFormulaKeyboardAdapter.

The following measures were also implemented:

- adding Hedgehog specific preferences in preferences.xml and the SettingsActivity class;
- setting up the Hedgehog resource (i.e. connect to Hedgehog) in the PreStageActivity class; and
- adding UI constants, such as colors and texts, in strings.xml and styles.xml. In this stage of development, no translation effort was undertaken.

Apart from these modifications of existing classes, new files were created to achieve the following:

- Adding UIs for the Hedgehog brick category and each Hedgehog brick;
- Adding Brick classes for each Hedgehog brick;
- Adding Action classes for each Hedgehog command;
- Adding the HedgehogService class as an abstraction to the single connected Hedgehog Light controller.

The action classes are the main component in bringing Hedgehog Light functionality to Pocket Code. An action object represents one of the commands in a script that can be executed at the right time. This means that commands are executed on the smart phone and transmitted to the Hedgehog Light controller for execution, usually over WiFi. For functions, the equivalent are the handlers in FormulaElement. Instead of giving each function its own implementation class, all code is in there.

"Downloading" a program to another device and executing it there is not intended by Pocket Code, so there is little support for that. Doing that would require to generate code for the target platform (e.g.

The ER4STEM project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 665972

in Python). This applies not only for Hedgehog blocks, but also for all other blocks, such as Control and Motion blocks, which are highly specific to the Pocket Code programming environment. A downloading approach is therefore not feasible without significant efforts and/or restrictions to the kinds of supported programs.

3.5 USE CASE

To demonstrate Pocket Bot's functionality, a simple program was implemented that uses both Pocket Code and Hedgehog Light features: an object allows the user to interact with the program while it is executing, and the program controls a motor depending on the user's input (see Figure 3). A variable is used to save the state between invocations of the "When tapped" event handler. Although the program is kept simple – which is in line with the complexity that is manageable as a graphical program – it shows how a seamless integration into Pocket Code was achieved. It also shows how a highly motivating use case of Pocket Bot can be achieved with minimal effort: a remote control for robots.

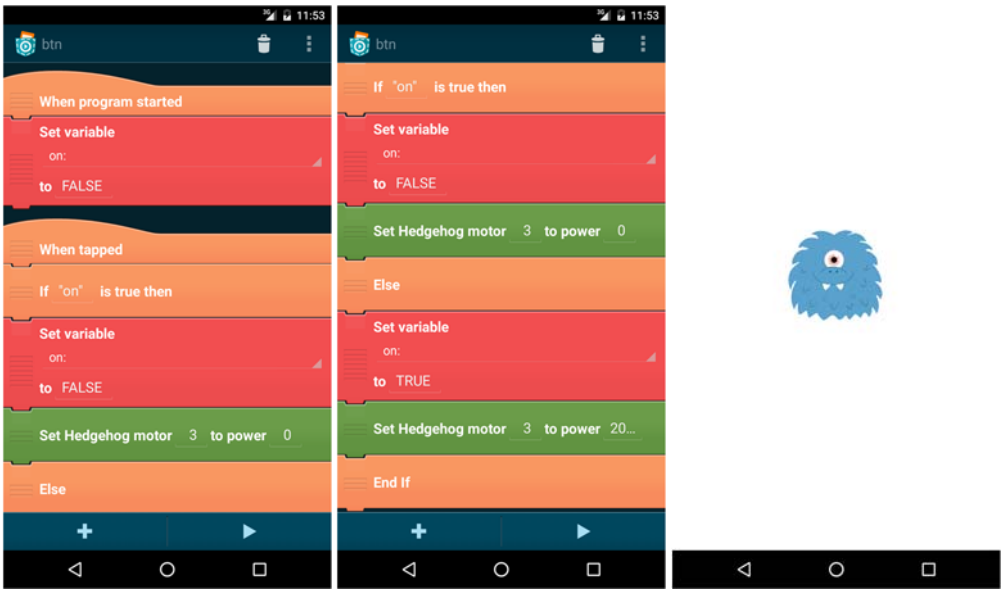

Figure 6: The Pocket Bot demo program (left and middle); tapping the fur ball (right) starts or stops a motor.

3.6 TESTING

During development, Pocket Bot was tested with emulated Android (6.0) devices as well as a dummy Hedgehog Light simulator. In regular intervals, the software was tested successfully with real Android (5.1) and Hedgehog Light devices over WiFi. Tests showed that the app works in principle, and that WiFi latency was not a problem while running the programs.

3.7 OUTLOOK

Some functionality to be implemented in the future includes advanced configuring of I/O ports: the API supports I/O ports to be configured as analogue input, digital input, or digital output. Input ports can be configured to use a pullup, a pulldown resistor, or none. In addition, trigger blocks similar to "When tapped" could be implemented for digital sensors ("When digital <0> becomes <TRUE>") to allow programs to wait for sensor changes.

The ER4STEM project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 665972

4 SUMMARY

This deliverable describes the prototype of Pocket Bot, which provides a graphical programming possibility of the educational robotics controller Hedgehog.

5 CONCLUSION / OUTLOOK

Pocket Bot can be used for graphically programming the controller Hedgehog. It allows the control of actuators and the reading of sensor values using according graphical blocks. Further steps regarding Pocket Bot are the implementation of additional functionality as well as the execution of extensive tests.

In the remainder of the project ER4STEM, the Hedgehog controller will undergo further tests and minor revisions to improve its applicability for educational robotics activities.

Hedgehog contributes to the ER4STEM objectives by representing a versatile educational robotics controller that can be used for workshops and conferences. As it can be programmed using a textual programming language like Python as well as a graphical programming language like Pocket Bot, it is usable for robotics activities covering various age groups.

6 GLOSSARY / ABBREVIATIONS

ER4STEM	Educational Robotics for STEM
HLC	High-Level Control
HWC	Hardware Controller
GDB	Gnu Debugger
I²C	Inter-Integrated Circuit
IDE	Integrated Development Environment
LLC	Low-Level Control
SCORE!	Smartphone Controlled Robots for Education and Industry
SWC	Software Controller
STEM	Science, Technology, Engineering, and Mathematics
USART	Universal Synchronous/Asynchronous Receiver/Transmitter

7 BIBLIOGRAPHY

[1] http://www.catrobat.org/, accessed 2016-08-10

[2] https://developers.google.com/protocol-buffers/, accessed 2016-08-10

[3] http://zeromq.org/, accessed 2016-08-10

The ER4STEM project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 665972