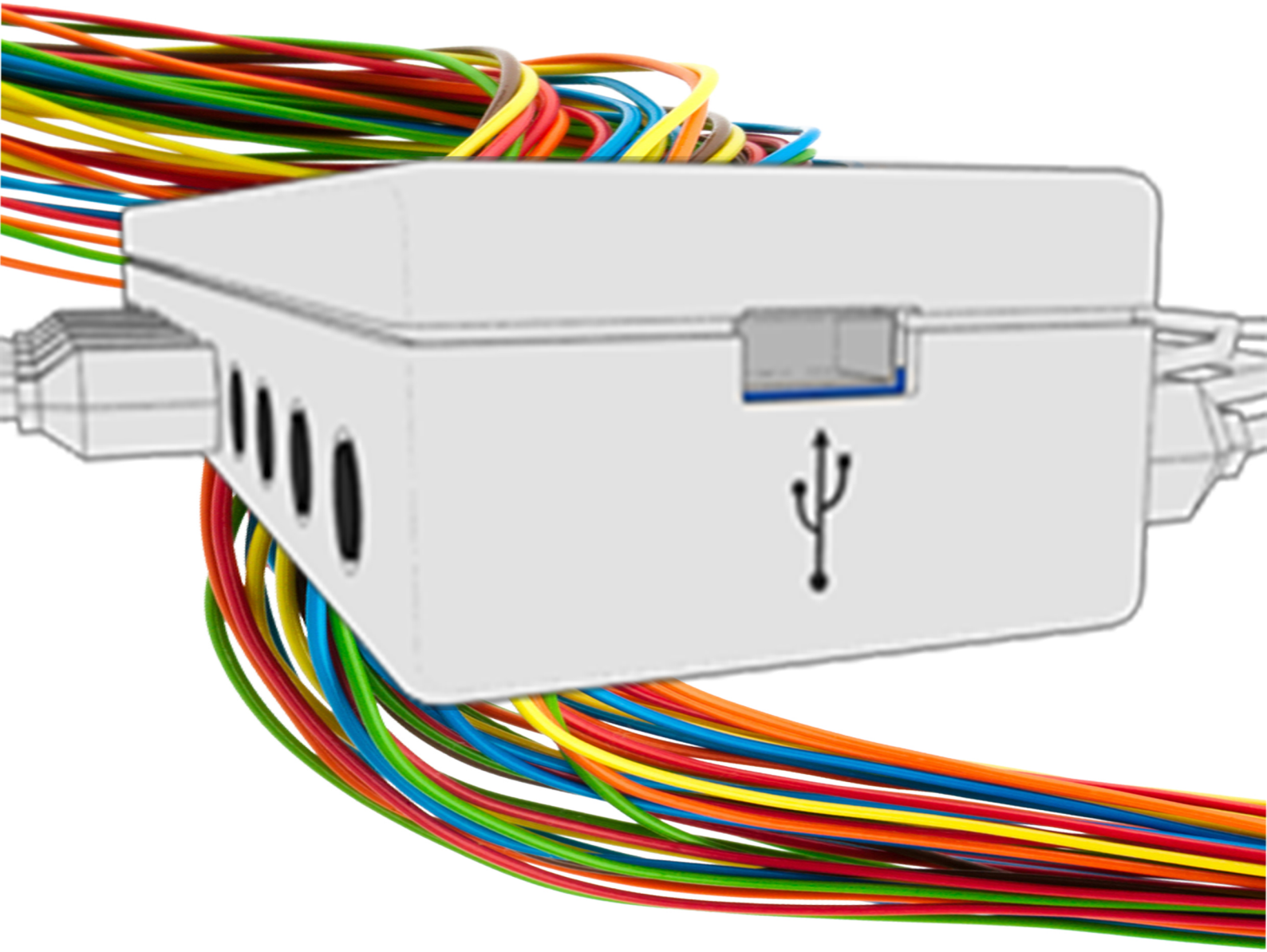
AALBORG UNIVERSITY
DENMARK

Simplifying analogue sensing and actuation in education



Author: Apostolis Kourtogiannis
Supervisor: Brian Bemman

Aalborg, 2018

Medialogy 10, Spring 2018
Group MTA 181039
Members: Apostolis Kourtogiannis
Supervisor: Brian Bemman

Acknowledgements

I would like to thank everyone that participated in this study, all interviewees and participants in the experiment. Also SEA for helping in broadening my perspective over the project. Last but not least, my supervisor Brian Bemman for helping me articulate my thoughts better.

Abstract

The purpose of this study is to examine how we can simplify the procedures when working with analogue sensing and actuation of physical objects. For the purpose of this study an Arduino-like board and a software user interface were made to test them with participants. Results gave indications that the proposed software and hardware was comparatively easy fore participants to wokr with. More work needs to be done to make the procedures when working with analogue sensing and actuation of physical objects even simpler.

Table of contents

Simplifying analogue sensing and actuation in education

Keywords: Prototyping, Arduino, User Experience

1. Introduction

Introducing individuals to the principles of entrepreneurship from an early age has been shown to encourage the development of original ideas later in life[1]. Original ideas form the basis of new products and services and so encouraging entrepreneurship is considered a priority by many countries [Blenker et al., 2011). Apart from the economic benefits, creating a general entrepreneurial mindset in individuals can enrich all aspects of life. Doing so has been called “entrepreneurship as everyday practice” (Blenker et al., 2012).

Towards introducing entrepreneurship in education, Gibb argues that it should firstly be “child centered in primary school, subject centered in secondary education, and discipline centered at university”(Gibb, 2008). This approach seems to have many common points with proposed models for teaching entrepreneurship in all levels of education such as the ones by Rasmussen and Nybye from 2013 (Rasmussen and Nybye, 2013). Shortly after, in 2015, the Danish Foundation of Entrepreneurship came to support this spirit, pinpointing the necessity of supporting, both in knowledge and actual material, all schools in teaching these skills from an early age (School Education Gateaway, 2015).

These initiatives that took place in Denmark seem not to differ significantly from other OECD countries that two decades ago argued for the need of introducing entrepreneurial skills. Since then, a number of benefits have resulted in countries from such an approach, for example, economic growth that comes from new kinds of products and services. Additionally, there are other benefits such as “individual growth, increased school engagement and improved equality” (Lackeus, 2017).

Knowledge Intensive Entrepreneurship (shortened as KIE) is a term often used to refer to the field that focuses on how to develop and implement entrepreneurial ideas into an economy. KIE has become of major importance in the knowledge economy and can play an important role in innovation and economic growth of countries (Malerba, 2016). Societal influences and public policy are considered crucial in KIE ventures for being able to access resources and ideas (Mc Kelvey et al., 2013).

A large role in coming up with new products and ideas is building prototypes in order to test them first. Analogue sensing and actuation can be an important aspect when making

prototypes that involve using technology and programming. With analogue sensing we receive values from measuring physical data such as the pressure or the temperature coming from a sensor. Actuation is controlling the behavior of physical objects. In accordance with this spirit, this study attempts to investigate how building interactive systems with analogue sensing and actuation can be developed further in education, towards strengthening KIE ventures on a regional or national level. For this reason, this study has these three steps: (1) conduct a qualitative study by interviewing people in the field of analogue sensing and actuation to find out what is important for them and the market, (2) build an analogue sensing and actuation prototype consisting of an "Arduino-like" hardware board and software user interface, and test it with users and non users of similar products. The tests entail the time it takes for users to complete some tasks and filling up a questionnaire. Then, (3) present an improved design of the board and the software user interface. The ideas for the new version mainly derive from the feedback from step 1 and 2.

Given these three steps of action for this study, the next section, section 2 refers to the basic aspects of the project. Subsection 2.1 provides an introduction to prototyping for analogue sensing and actuation and 2.2 about pedagogical issues in relation to entrepreneurship and programming in particular. Then it extends to previous work in the field of analogue sensing and actuation, subsection 2.3, and to User Experience analysis in research in subsection 2.4. The next section are referring to the prototypes made for this study and the presentation of the 3 steps mentioned. Lastly the is a discussion and conclusion section.

2. Background

2.1 Prototyping

Prototyping is considered to be a crucial area in entrepreneurship and being able to build fast interactive systems can be an important skill towards this end. The difference between a prototype and a final product appears to vary from case to case: a prototype of a satellite, for example, will usually be quite close to the actual final product. On the other hand, the first versions of a video game may have to go through many different kinds of corrections in order to make the final product that offers the intended playing experience (Wall et al., 1991). Depending on how close to the final product is a prototype, there is the division into two categories, *high fidelity* and *low fidelity* prototyping. Traditionally, in low fidelity prototyping we see sketches or simple texts and manual imitation of the desired functions. These kinds of prototypes are used primarily to present and communicate an idea of a

product. In high fidelity prototyping, actual prototypes are built that are close to the intended final product. High-fidelity prototypes are fully functional on their own, allowing a complete experience of the product (Rudd et al., 1996).

There have been discussions over these two types of prototyping and their implications in relation to how realistically they represent products, along with different types of usability testing methods. A main argument against low fidelity prototyping is primarily that it fails to offer the actual sensation of the supposed product (Walker et al., 2002). High fidelity prototypes are less flexible to substantial changes but are closer to the final product experience. Knowledge in software and hardware is an important factor towards making easier and better high fidelity prototypes. In many cases, high fidelity prototyping might be the only option towards designing and testing a product. One could argue about the strengths and weaknesses of these two kinds of making prototypes, however, what is the most important for both, is to give a clear picture of the product or idea in general. Figure 1. shows how an idea for an app for the stock market is communicated through low fidelity to high fidelity prototypes. It is evident how the low fidelity prototype on the left gives more space to imagination and suggestions than the third one where things become more specific. This method has been used by many entrepreneurs especially into the app market.

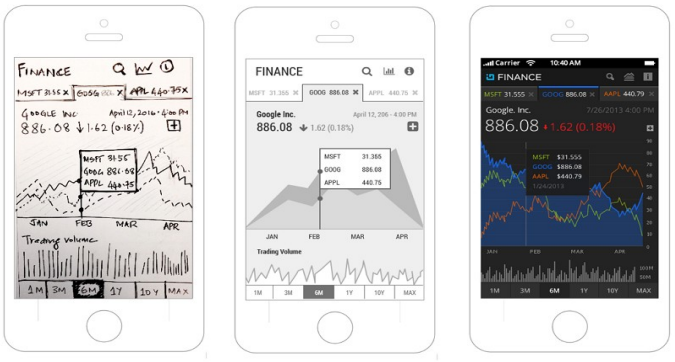


Figure 1. High and low fidelity prototyping towards making a product. On the left a simple sketch on a piece of paper, then in the middle a digitally made version and on the right the final design of the user interface of the app.

2.2 Pedagogical Aspects

It has been a few decades since most countries and economies acknowledged programming as one of the most important factors in many different areas. Indicatively, in the Netherlands in 2002 Informatics was presented as an entirely new generation discipline, related to Business, Physics, Engineering, Linguistics, Economy, Philosophy, Psychology, and Social Sciences in general (Mulder, 2002) . In education, learning experiences are

usually engaging when they “tap into the young participants’ natural curiosity and interest in discovery, and when they serve to motivate, rather than discourage, their eagerness to try new activities”(Community Network for Youth Development, 2001). Towards this end, there has been and still is a difficulty in acquainting many students with programming, as it is thought as something difficult to learn in many cases (Van Diepen, 2005). Three decades ago in 1989, DuBoulay (DuBoulay, 1989) attempted to locate and highlight where the problems were for people to become skilled in programming:

(1) there is difficulty in finding out the functions of the different programs and what can they offer to the needs of each student.
(2) it is not easy to understand the properties in general of a program or machine.
(3) it is hard to achieve a level of knowledge where the student is able to think, design and build real solutions to real problems.

As presented later in this paper, these above three issues have been taken into account in the design of the software user interface and hardware. Of course, towards teaching programming in secondary education the role of the teachers becomes important. The different personalities of teachers and the teaching methods can make a significant difference for the student. This applies not only for the programming teachers in particular, but any teacher that would want to blend programming skills with entrepreneurial thought in general. An interesting study for this report shows how teachers lacked the ability to integrate entrepreneurial thought in their teaching and how there was a distance between the aims and the results of their lessons (Ikävalko et al., 2009). This has been one of the motives for this project, creating tools that would make things easier both for the students and the teachers. As presented in the next section of this paper, there have indeed been many efforts and programs that have offered various solutions to these issues. Some of these efforts are simpler user interfaces, tutorials and communities on line to help any beginner in a specific program. A part in engaging youngsters in their learning is showing them how to use their new skills to connect more with different communities around them (Community Network for Youth Development,, 2001).

2.3 Previous work

As Constructionist Learning theory suggests, children learn better when they construct knowledge voluntarily, for purposes that are important to them(Papert, 1980). This way they become easier engaged in creating something tangible for them, whether that is animations in a program, small robots or other kinds of artifacts (Papert,1980 and Resnick et al., 1996). For this purpose, computer programming environments have been developed in order to support the construction of knowledge through playing with real world objects and

programming. An important factor for engaging children in programming, is using physical objects. This way the entire procedure is closer to one for constructing a toy, something that children are already familiar with. What follows is an extensive discussion about different components of programming software and hardware products towards such goals.

2.3.1 Software

Creating interactive systems that involve programming has been an area of interest for many different fields, professional or not. Traditionally, such tasks would involve programming with languages such as C++ and others that require the user to write code: a process regarded difficult by the average beginner in programming. Having to focus on syntax does not encourage the students to get engaged and create more complex structures (Gal-Ezer,1994). Therefore, there have been efforts to design programs with which such processes could be simplified for the users, whether that would be in education, in various professions or simply for hobbyists (Myers et al.,2000).

Towards helping beginners, there are also the functions a program may have, the on line communities for supporting the users and the user interfaces of the programs. A first step and basic aspect of this evolution in user interfaces would be visualizing the functions of code in order for the user to identify and work with them easier. These kinds of interfaces are widely known as Graphical User Interfaces (GUI) and have been initially used for programs with commercial use and marketed to people with no prior programming knowledge (White Paper, 2009). Clicking open a folder on a screen is a simple example of this, making interaction with the computer real time and easier to grasp. Figure 2. portrays the differences between the two user interfaces, GUI and using code.

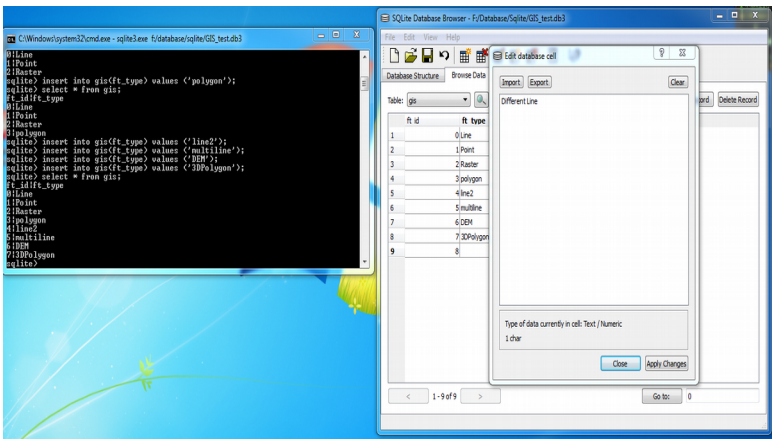


Figure 2. A command line interface using code on the left with the desired functions visualized.

As Marinilli (Marinilli 2002 and Marinilli 2003) states in his work, in order for programs to both get the user's attention and make things clearer for them, there have been various techniques adopted. These are presented in the table under and are discussed further.

Animation	Visual elements that appear on a screen can attract the user's attention. Animations will be used in general to signal various activities in progress, express the GUI internal state etc. Towards making things easier for the user, there is also a concern that animations should not become disturbing to the user
Color	As with animation, colors can offer a clearer and an aesthetically more pleasing environment for the user. Again, however, they should be used in a manner that does not confuse (e.g. too many colors)
Sound	Here also this must be used in moderation. It needs to be pinpointed also though that when dealing with disabled people, sound signals can offer an effective feedback system.
Graphic Adornments	Special graphics, bold fonts etc. to highlight specific issues or elements. This also needs to be implemented in a manner and volume so that it does not become disruptive for the user.

Lastly, one of the most important characteristics is that they give direct feedback to the user. Unlike older user interfaces using code, where the code needs to be compiled each time in order to see the results, here the post processing of the commands is instant. This gives the possibility to the user to see at each moment the consequences of his or her programming, contributing a better and faster understanding of the overall procedure of programming (Depcik, 2004). At the same time it can boost spontaneous creativity as well. However, according to Reed (Reed, 2002) this way of programming might not be ideal for becoming skilled in programming. His argument is that users can reach desired results without having to understand in detail what is happening with programming. A reflection of this fact is often met in art, where artists rather experiment than study how to program, in order to make artifacts aesthetically pleasing.

In relation to making it easier for someone to program, the overall tendency from new users is abandoning the procedures of traditional ways of programming in which the user has to write code (in a C++environment, for example). This becomes evident with a quick look at the various programs that use GUI's not requiring previous programming knowledge involving syntax(Postscapes, 2009), such as Scratch, Max/MSP and VVVV that are presented in the next subsection. What usually one may see in many GUI's dedicated to learning how to program is that the user can construct sets of functions only by connecting different visual items representing these functions with each other. There are various GUI'S/

programming languages of this kind that are known also as “node-based” or “visual” programming (www.vvvv.org, https://cycling74.com/products/max/). Writing code translates with these programs into constructing “a patch”. The logic behind the construction of these so-called patches in these programs appears in most cases to have a tree-branch logic, where different “bricks” also called Nodes, are connected with each other creating this way a system that can have multiple functions. These visual items (or nodes) are in essence encapsulated code. They usually appear to have input pins on the top and output pins at the bottom to connect with each other. Apart from not having to type code and lose time with syntax, visual programming languages reduce the possibility of errors occurring in a patch. This is because every node by default has perfect functionality, thus the main challenge for the beginner programmer is to see how to connect correctly the different nodes.

Scratch

A characteristic and very popular example of one such program that is presented here is Scratch. As one may see in Figure 3, with Scratch the user simply connects different nodes in order to create a patch.

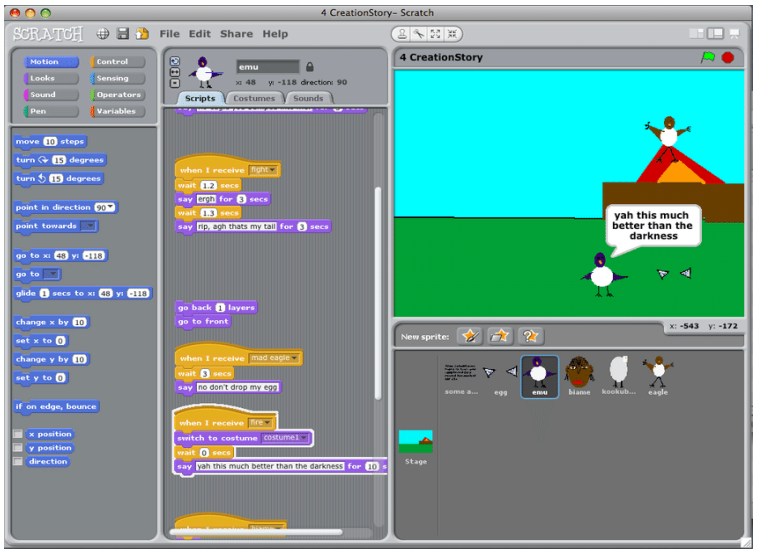


Figure 3. The user interface of Scratch. Apart from looking friendly especially for children, its programming elements have names and colors making it easier for the beginner to construct patches.

The creators of Scratch claim it is a program for everyone in a new age where digital fluency extends more than into browsing on the internet, but creating, designing and remixing. The company's motto is “Programming for All”, having a core audience being between the ages of 8 and 16 and 1500 uploads of interactive patches/ projects daily on

average. Apart from having many young users, the specific company has managed to establish for itself a fame that with Scratch everyone can program and do interesting things. In the company's paper from 2009, they presented their principles for the user interface of the program and how convenient it should be for anyone. Along with that, they highlighted user cases of children in secondary education, how they evolved as programmers and eventually became widely known on line through their work (Resnick et al., 2009). From their findings one can understand how the company has managed to engage a large number of children in programming worldwide.

Max/MSP

Max/MSP follows a similar logic to Scratch for creating patches. As one may notice in Figure 5. Max/MSP also uses nodes in order to construct patches. The program is known for almost a decade now and has a wide range of functions. Though with it the user can do many things, Max/MSP is used extensively by many artists in the music industry. The program is considered easy to learn with on line tutorials and books assisting users for tasks such as creating adaptive interactive instruments for composition and performance activities, measurement tools to conduct research and others (Manzo, 2011).

Figure 4. shows how a patch in Max-MSP looks like. It is a patch that allows one to read and use easily the different notes of a keyboard. Even if Max/MSP is considered a more “serious” program than Scratch, it was also marketed to people with no programming background and has already a large on line community helping the beginners with their projects.

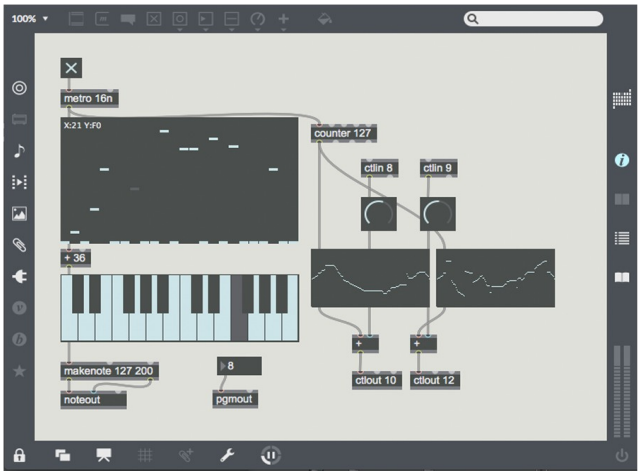


Figure 4. The user interface of Max/MSP. The user can see in the program instantly which notes are being used each time and set the functions the user wants with the tools that appear on the sides.

VVVV

As we see in Figure 5. VVVV looks very similar to Max/MSP. Also here we have a gray background and gray nodes for the user and the logic of constructing patches is more or less the same. What differs are the different possibilities the programs have. VVVV is considered among users better for projects with visuals and Max/MSP for sound. One reason for this distinction, is that the two programs offer specific nodes that make things easier for the user for specific operations. Additionally, different functions exist either in the one or other program. So with VVVV a sound artist will not be able to do as much and as quick as with Max/MSP. This portrays what DuBoulay highlights (in the Pedagogical Aspects section 2.2) as the first of the three major problems towards having new users become good at programming: finding the correct program for their needs.

One question that may arise is how deep can someone go with such programs as a programmer? The answer is quite far. One can understand this by having a look at all the connections of the VVVV patch in Figure 5. While it is a complex patch the user can see instantly the result on the monitor screen that appears on the top left. The software user interface of the proposed product has a similar logic, allowing an instant read of values and an easy connection to other nodes. The board also is encoded this way so that it is read directly by all other programs that users are already accustomed to.

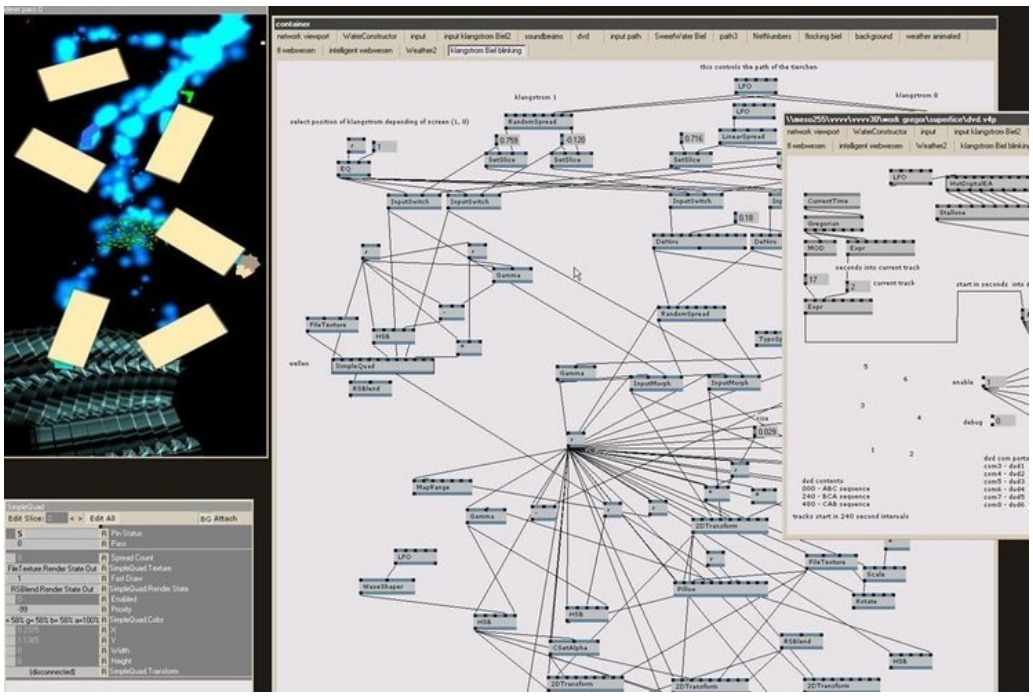


Figure 5. A complex VVVV patch for visuals. It is apparent that there are quite many functions happening at the same time. For a cleaner workspace the user can also create smaller patches within a patch, that is the one on the right.

If speaking only for software, the user interface of the proposed product of this project does not differ significantly from the one we see in Scratch. It also operates in a “building with blocks” logic. As mentioned previously the project is about making easier to program using analogue sensing and actuation of physical objects. In order to control a physical object one has to use a board to connect it to the computer first. The most known boards for this task are the Arduino boards (see on the very next session). Scratch is also able to control physical objects with an Arduino board, but the user has to upload specific extended codes first to it. This can be problematic for various reasons, such as the different models of Arduino boards not being compatible they are with the different versions of Scratch (https://en.scratch-wiki.info/wiki/Hardware_That_Can_Connect_to_Scratch, https://github.com/khanning/scratch-arduino-extension/issues/45). Similar issues with controlling physical objects with an Arduino board are met also with similar other programms like VVVV or Max/MSP that are presented right after. In the proposed product, the software and the hardware (board) are communicating directly in order to avoid such issues. The board also is encoded this way so that it is read straight forward by any program, like any simple usb device. Lastly, the values coming from the sensors or for the motors appear real time on the computer screen for the user (see also later section on the experiment conducted with users). This can help the user to understand faster what is going on inside the patch.

2.3.2. Hardware

As mentioned in the last paragraph of the previous section, in order to have physical objects controlled by a computer, another physical device is needed. The devices for connecting a computer with a physical object (or the physical world in general), would usually be electronic boards that act as an intermediator between the two sides.

Apart from coding itself, controlling a device may include some kind of sensing and/or actuating. As far as sensing is concerned, there are different sensors that measure various elements such as temperature, pressure, position, moisture and others, that can provide a system with data about the physical world. The same applies for different motors or other actuating devices that receive values either from a sensor or a simple piece of code within a system. Not unlike sensors, actuators also vary depending on the nature of a project.

Therefore actuation may extend from using a motor that is moving items to a system of devices for watering plants for example.

Though there are many products that offer stable solutions for sensing or controlling, the Arduino boards are by far the most well known and used. There are different models that come in different sizes and characteristics depending on the desired use. Despite the fact that Arduino boards are widely used in all areas of the product proposed in this paper, education, professionals and hobbyists, they are not generally regarded as something easy to use by a novice. One reason is that in their basic form the boards typically work with non-visual programming languages in the Arduino software program.

This barrier can be lifted with the use of protocols such as Firmata (http://firmata.org/wiki/Main_Page). Firmata is an Arduino library which simplifies communication over the USB serial port, allowing reading, sending and receiving values from an Arduino board directly into another program such as VVVV. In essence, what Firmata does for the user, is to make the Arduino board talk to the host computer via the USB connection like any other serial device to the application hosting it.

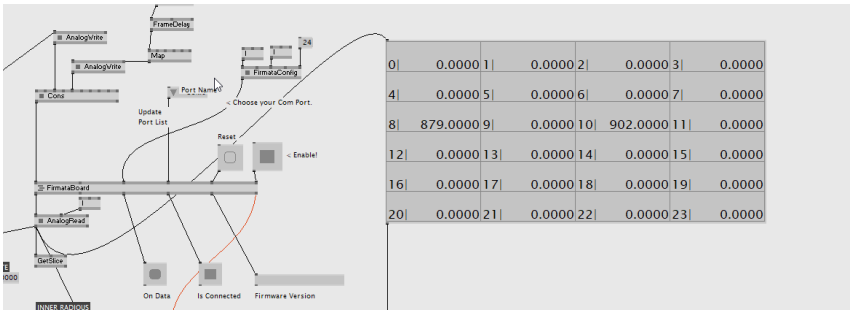


Figure 6. A screenshot in VVVV using Firmata to read directly the data from pressure sensors. The moment the screenshot was taken two pressure sensors were pressed from port 8 and 10 of the board. These values can be seen in the table on the right with all the values from all 24 ports of the board. Nothing happens in the other ports, thus the value they give is zero.

Figure 6. shows how data from the board are read directly in VVVV. The Firmata library was implemented in the boards of the product proposed in this report, allowing someone to see instantly the values from the sensors or to actuators in any program, without having to download any library as they would have to with other boards.

Additionally, many times projects using Arduino boards entail the challenge of making correct electric circuits. This can very easily evolve into very complex systems that apart from being difficult to make, also require knowledge about circuitry. The complexity of such tasks is clearly shown in Figure 7, a simple project using an Arduino board. Another concern also is that this may become dangerous for the users in case something is not grounded properly.

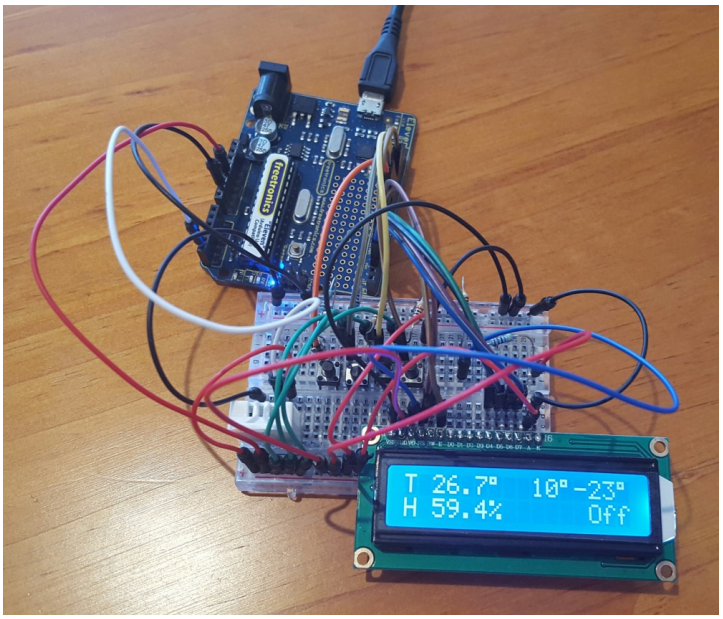


Figure 7. A comparatively simple project with an Arduino board. The board uses different sensors. The values for the sensors appear on the small screen connected to the board.Every input pin of the sensors and screen must be connected to a specific pin on the board through jumper cables.

There have been efforts such as the prototype by Kelemen et al. (Kelemen et al., 2014), where they build boards that could accept jack cables (like the small jack stereo cables for headphones) instead of the conventional so-called “jumper cables” used in Arduino. The good thing with jack cables is that they can carry a significant number of channels of information and that results in reducing the overall number of cables and connections one needs to make. A simple jack stereo cable for headphones has three channels of information, while there are also other kinds of jack cables that have even more channels.

The boards presented by Kelemen also had all vital components (processor, clock source, memory, power regulator) that one sees on an Arduino board, allowing easy sensing and actuation. As they mention, these boards were made in order to help students learn programming, however, there is no mention in their work about combining these boards either with Firmata or Firmata-like libraries or other programs that do not require coding. The board of the product proposed in this report, apart from having more inputs outputs (24) can also offer wireless communication (from the board to the computer) and a battery option in case someone wants the board not connected with a USB to a computer. However the same USB port for connecting the board to a computer can also be used to give electricity to the board from a socket (with a USB adapter in front).

2.4 User Experience Analysis and Qualitative Research

One of the central goals of this research is to see how the users of the product proposed in this report, experience the whole procedure of making systems that involve programming, analogue sensing and actuation. *User experience analysis* aims at obtaining a view on the qualities of the experience a user has while interacting with a product (McCarthy and Wright, 2004). While the term “experience” has always been an object of thorough analysis for all different domains, from Architecture to Medicine, there has not been a concrete definition for it. Mattingly argues how it is difficult to give a concrete definition for experience as this task relates to narratives and the sense making properties in the minds of humans (Mattingly, 1998). Despite the elusive nature of the term there have been several methods suggested to obtain valid information about a particular experience, behavior or aspects of it people might have. If research aims to obtain information about such issues, then qualitative methods should be preferred (Hancock et al., 2007).

Qualitative research focuses on reaching an understanding of why and how things are in our social world (Hancock et al., 2007). Therefore, if a study is interested in exploring how people view or experience something, “exploring a new area where issues are not yet understood or properly identified, assessing whether a new service is implementable, looking at ‘real-life’ context, or a sensitive topic where you need flexibility to avoid causing distress” (Hancock et al., 2007), then qualitative research should be applied to find answers. There are different approaches to conduct qualitative research from case to case. These methods often consist of case studies, ethnography or focus groups, for example (Khambete andAthavankar,2010).

As Bryman argues (Bryman, 2008), a case study is a research method that uses the detailed and intensive analysis of a single case in order to draw general conclusions over a topic. Ethnography, on the other hand, studies social behaviors, perceptions and behaviors that occur within groups, teams, regions, organizations, communities, even circumstances (Patcho, 2015). Unlike these two methods where information is just extracted, in focus groups, the participants influence each other through their answers to the ideas and contributions during the discussion. Within this method the researcher stimulates the discussion with comments or suggested topics. The data received with this method are the transcripts of the group discussions along with the moderator's reflections and annotations (Reeves, 2008).

Additionally, there is also grounded theory where the data collection, analysis and research progress evolve in parallel. Initially the data are collected (mainly through interviewing) and cover different aspects of the topics one wants to investigate (Freitas et al., 1998). The first

set of data are analyzed immediately and the results redefine the next set of data collection activity (which new or other kind of data to collect), from where to collect them and which aspects require special attention. Every pattern that is discovered is treated as a basic object of further analysis (Corbin, 1990). In other words, grounded theory helps to come up with new concepts, pattern recognition and theoretical ideas, emerging out of the repeated renewal and analysis of the qualitative data and the context (Eriksson, 2008).

For more flexibility, the ground theory method may also entail semi-structured interviews that allow the possibility of new data to emerge at any point. Semi-structured interviews are often conducted in qualitative research studies. Based on a pre-defined set of general questions, themes and concepts, the researcher conducts the interview like a physiotherapist would interview a patient in clinical practice; “sticking loosely to a recognizable plan, but allowing for deviations where the interviewee decides that new information is needed” (Nicholls, 2009). This approach was adopted in the first phase of this project where interviews with people from the field were made. The people interviewed were encouraged to add anything they would consider important towards the topics we discussed. Their comments have been a great help in the final phase, where new features for an improved product are presented.

2.4.1 Quantitative Research and Task Analysis

Though qualitative research can give significant information towards a deeper understanding of a topic, quantitative research can also be helpful by measuring specific aspects of it. As presented in later sections of this project, quantitative data were extracted from people using the proposed product. One method of quantitative research in user experience analysis is *Task Analysis*. As Kanji (Kanji, 2015) states, task analysis is about identifying and capturing a user’s workflow. Kanji considers Task Analysis a stronger form of research than ethnographic observation or contextual inquiry, in which the target output is a breakdown of what a user does, step-by-step. This way one can obtain more specific information about the user’s experience. Kieras (Kieras, 1994) on the other hand, goes deeper into Task Analysis by presenting the method known as GOMS that stands for Goals, Operators, Methods and Selection Rules. As its name implies, GOMS attempts to describe the procedural knowledge a user must have in order to execute specific tasks in a system or a device. This method entails a series of tasks for the users that based on the performance of the users can then give specific measurable data. In the User Experience test for the needs of this study, a similar approach was followed, obtaining quantitative information about the time each participant needed in order to complete some tasks. Additionally data from a 7 point Likert scale questionnaire were collected for further analysis.

3. Field of the Research

As mentioned in the background section, there is a concern at all levels of education to integrate the teaching of programming. In general, it is regarded that this will offer more problem solving skills to the students. Some of the main concerns towards this would be, why should we teach programming and what concepts within programming need to be taught. According to DuBoulay (DuBoulay, 1989), it is important to find out what are usually the difficulties that students encounter when trying to learning how to program (Kelleher et al., 2003).

The challenge extends not only into making programming feel friendlier, but also into providing eventually the new users with skills that can offer solutions to real problems. One important aspect would be to find ways to introduce programming more as constructive play for children, involving the use of objects and materials that can be used for programming. Quite recently in 2017, Merkouris et al. (Merkouris et. Al, 2017) pinpointed the importance of using physical objects in programming for young ages. In their work they state how pedagogical research has shown that physical representations and tangible interactive objects can help young students learning how to program. They came to this conclusion by measuring through questionnaires the emotional engagement, attitudes, and computer programming performance. They found that students were more excited and had a larger appetite to learn how to program with small robots rather than the desktop computer.

The same conclusions are presented by other researchers as well in the field (Scaradozzi et al.,2015, Bazylev et al., 2014, Tochacek et al.,2015 and Ospennikova et al., 2015), pinpointing the need of programming physical objects in order to increase the engagement for children. What follows is a presentation of three products that attempt to acquaint children with programming, Lego, Little Bits and Smart Gurlz. These products give an indication of how programming is becoming more and more important for children's education. The proposed product of this paper is also addressed to children, though it is thought as more appropriate for ages a little older when they are in the secondary education. One main aspect of it is that is should cover the gap, both in ages and difficulty to learn, between the above mentioned programs and hardware.

LEGO

For almost two decades now, Lego has been releasing products that involve basic programming applied on various Lego constructions for children. These products are known as LEGO Mindstorm or simple “LM” (http://mindstorms.lego.com). LM has had quite a success with many schools and organizations in different countries adopting them to teach children programming. Indicatively, the work of Atmatzidou et al. (Atmatzidou et al., 2008), pinpoints how LM appears to help children learning how to program. Atmatzidou and her team offered courses to children both in elementary and secondary education. In their paper they show how these courses contributed to the familiarization of students with structured programming principles, a positive influence according to them, on developing problem solving skills later. The same conclusion was expressed also from Sartatzemi et al. (Sartatzemi et al., 2008) that did a similar study with children and LM. The results of their work showed that physical models indeed comprise a valuable solution to introducing programming to children as they appear rather interesting to them.

A common point between these two studies with LM, is that they both stress the importance of having physical objects in order to engage the children more. According to Atmatzidou et al. this way of teaching is known as “Edutainment”, the educational approach that combines games with physical objects and learning. Figure 8. shows a construction using LM products. As one may notice there is a small device on it that applies the code to its moving parts. This is in actuality a very simplified Arduino board with fewer possibilities.

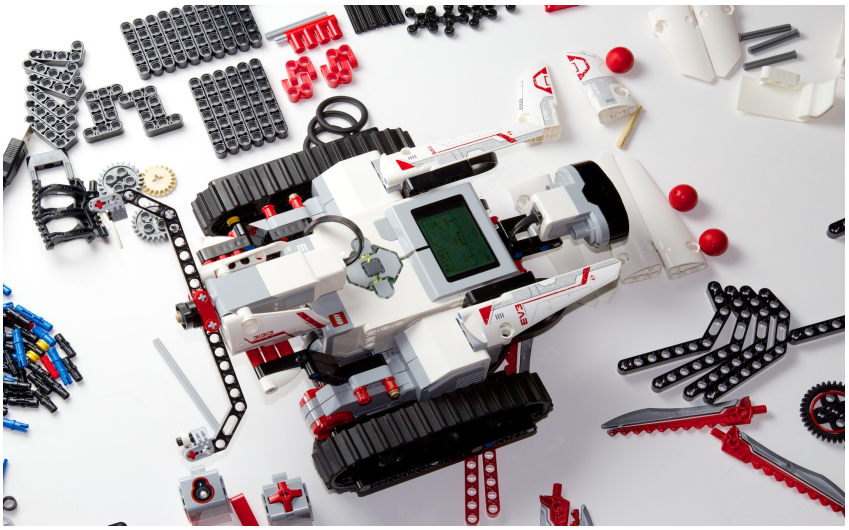


Figure 8. The user attaches the board with the small screen on the construction and then connect it to the motors in order to make it move. Before that, the board had to be programmed.

Little Bits

Little Bits (www.littlebits.com) products are very relevant to the ones of LEGO. The company is sells “kits” that consist of building blocks that fit in circuit boards, buttons and switches, wires, and other components that interlock magnetically. This makes them very easy to use. As Figure 9 shows, by connecting the different parts one can easily create an interactive system. The Little Bits products are addressed almost exclusively to small children as they are very safe to use. However, there is a limit to how complex structures someone can make with them. This was witnessed also by the manager of Coding Pirates in Aalborg (an organization that teaches children robotics all over Denmark, see Step 1 section), that mentioned that after a while children have fully explored them and ask for something more complex.

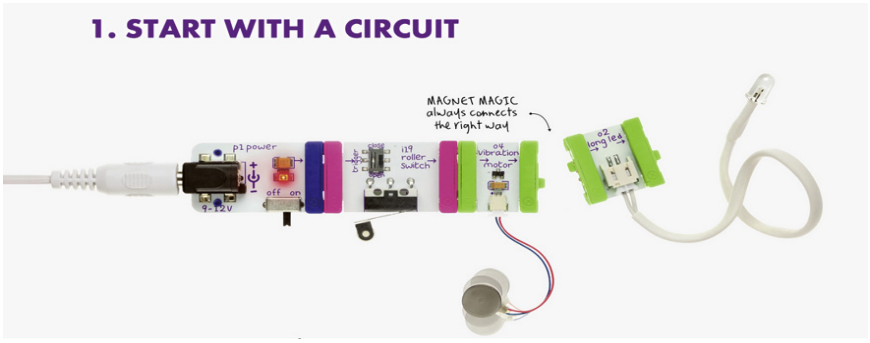


Figure 9. A small operational system with sensors using different components from Little Bits. As stated in the picture components are made so that they always connect the right way.

Smart Gurlz

Smart Gurlz (www.smartgurlz.com) offers products similar to Little Bits but exclusively to young girls before secondary education. Their products look like Barbie dolls for girls, only that they can be programmed to do things. The company comes from Denmark and has managed to attract much attention in the media for their innovative products and concepts. As stated in an article about them in Daily Motion, Smart Gurlz encourages girls to go into coding (https://www.dailymotion.com/video/x6ivmy3). The concept of the company was presented at the television show “Shark Tank” where entrepreneurs go and pitch their ideas. Smart Gurlz managed in that show to find investors and one of them was the owner of Virgin, Richard Branson. Branson stated that he “loved the idea but didn’t think he could add enough value to make an investment himself” (https://www.bizjournals.com/washington/news/2017/11/13/local-startup-s-segway-riding-doll-scores-shark.html)

The proposed product in this paper differs very much in orientation from Smart Gurlz. One main difference is that we think of the product being able to have more advanced functions and being more demanding from the user. However, products like the ones of Smart Gurlz are considered important to be presented here in order to give a broader picture about the circumstances in which the project operates. It is evident that in the future there will be more children than today with some programming knowledge. In such a scenery it is thought that a product like ours will be easily introduced. Figure 10 shows the founder of Smart Gurlz, showing one of their products.

Figure 10. Founder of Smart Gurlz, Sharmi Albrechtsen. One may notice the similarity with the user interface of Scratch. The company itself emphasizes in their web page how similar their software is to Scratch.

4. Presentation of the prototype for the proposed product

The name of the proposed product in this paper is “Oktoware”. In the team behind it is also another graduate from Medialogy of AAU, Bjørn Thorlacius. The product is thought to cover all the steps towards easy analogue sensing and actuation by offering both software and hardware solutions. Before and during this semester we have been developing the hardware and coming up with different ideas for the user interface. What follows is a presentation of these different parts of the product and the reasons they were designed so.

4.1 User Interface

For the purposes of this study, the user interface was made and functioned during the experiment with users within VVVV. Even if there is the plan to have our own software entirely in the future, we thought it would be better to introduce Oktoware through an interface that a large portion of users are already familiar with. This is a direction that we are thinking of following in the future in the design of our own software. As mentioned in the previous section, Little Bits and Smart Gurlz have also adopted a similar logic in the design of their software, resembling Scratch. This way it should be easier for new users to grasp the program.

However, the User Interface used for the experiment of this study has not been just a VVVV patch, but an imitation of the user interface that we would like to have. We wanted the users to come across with more or less how we envision our user interface. For this to be accomplished, several tasks had to be made first.

First of all the board itself was programmed so that is compatible with all programs that can work with Arduino-like boards. We considered it crucial that the hardware should adapt easily to any user's needs that are used to work in specific programs that can support analogue sensing and actuation. Therefore, in order to have a satisfactory imitation of the desired user interface for the experiment, the programming of the board happened initially in the Arduino software. As shown in the later hardware section, there were different components on the board that had to be configured in specific ways. With this programming we achieved the board being read properly in VVVV.

The next stage was to program in VVVV a system that handles the data in order to then have a user interface in which the user does not have to struggle with such issues. One of these issues for example would be to decipher which information corresponds to a specific sensor or device. These configurations and new connections where made in VVVV in a patch that runs hidden in parallel with the actual user interface. Figure 6. from the previous section is a screenshot of the VVVV patch that handled all these issues. Figure 6. shows only a part of the main patch that took care of these issues. This patch had other sub-patches as well in it.

Figure 11. and 12. below show what the users came across with our proposed product. In Figure 11 the users can see instantly which ports of the board are in use and what values they are giving. All they have to do is to activate the sensors in order to see what values they are giving. As shown in Figure 12, in order to give a better understanding instantly to the user, there is also a screen that shows the board and which port of it are active.

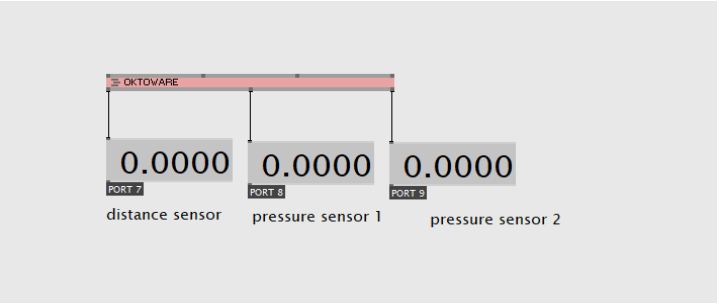


Figure 11. The workspace for the user with three sensors on the board. The ports in use are read automatically by the program while the user can write comments on each of the items (what kind of sensors they are etc.).

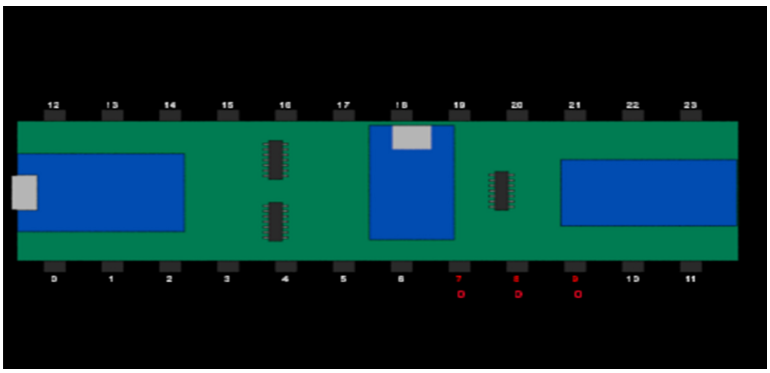


Figure 12. A screenshot of the screen that shows real time to the user which ports of the board are in use with red, so ports 7,8 and 9.

One argument why a node-based programming environment like VVVV was used as a platform is that with such user interfaces one can construct complex patches. This is clearly shown in Figure 5. in this report where we can see a comparatively complex VVVV patch with many connections. What needs to be taken into account for this issue is that we have a board with 24 ports for sensors and another 8 for motors. Therefore, it is expected that users would want to use our product for complex structures such as those in Figure 5. A user interface like the one of Smart Gurlz or even Scratch, for example, was thought not a good solution for such scenarios by our team. Another possibility that VVVV- like programs offer is that users can encapsulate some parts of a patch if they wish. This results in having just one node on the screen that inside it has an entire patch if opened, something very practical when constructing complex patches.

4.2 Hardware

The hardware part of the proposed product is the board and converters from jumper cables to mini jacks. For creating the board we first had to think of the attributes that we would want it to have. These mainly were about having 24 ports for sensors and 8 for actuating devices as we envision of the product being used for complex projects. Then we would have to think about the shape and dimensions of the board that could have this number of ports. That process included finding first which components we would use and place on the boards to have the desired function. Their size and other characteristics have affected the overall design of the board.

We came up with a shape for the board with the dimensions 15 cm length, 3,2 cm width and approx. 1 cm height. This shape was chosen not only because it was found ergonomic but also because it was more appropriate for its inner circuitry in relation with other shapes. The PCB design of the initial board without any components was made in Denmark with the help also of an experienced in PCB (Printed Circuit Boards) design friend of our team. Then the PCB design was sent to China for manufacturing 20 boards in total. Figure 13. shows the PCB design that was sent for manufacturing and Figure 14. shows one of our boards right after we received it from China, without any components on it yet.

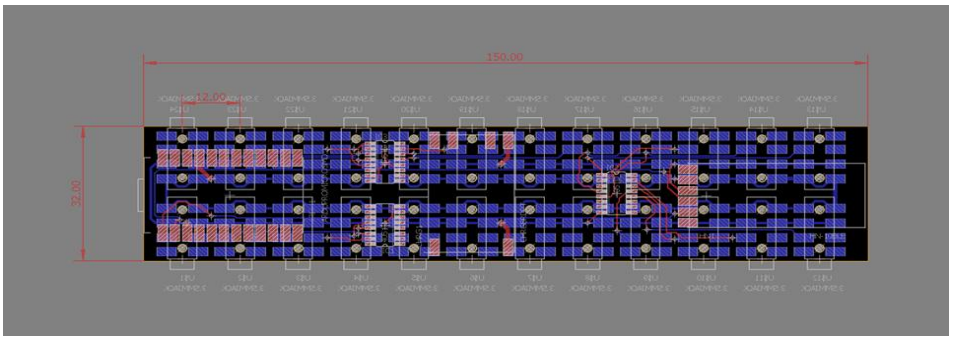


Figure 13. The final PCB design that was sent to China for manufacturing.

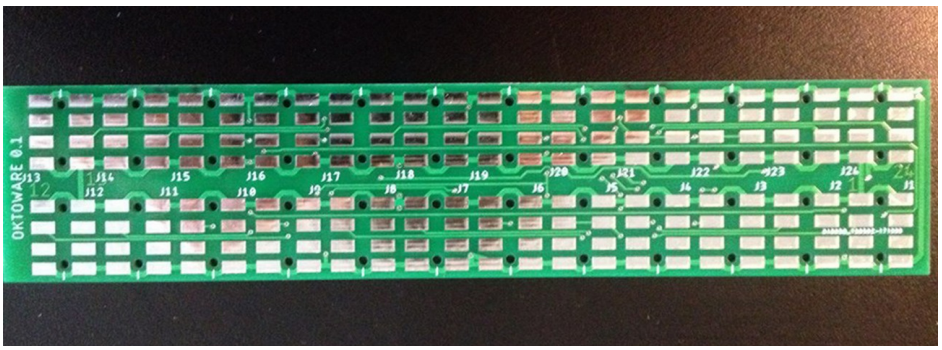


Figure 14. The board in its initial form with no components on it. One may notice on the left the tentative name of the product printed on the board.

The components for the boards had been ordered and received shortly before the boards arrived from China. These components were:

- Mini jack sockets for the sensors

- Mini Arduino boards for configuring our board to a computer

-Multiplexer chips in order to have the desired number of ports

- Pololu boards for controlling motors

-Chips for charging batteries and charging batteries

-Bluetooth chip for wireless communication between the board and a computer.

Figure 15. shows the two sides of completed boards. On the one side, the board on the top, one can see the 24 mini jack inputs for sensors soldered on the board. On the board on the bottom we see the above mentioned components soldered on the board. Something important to mention here, the most important thing for our team in this phase has been to ensure that we have full functionality in our prototypes. This explains the unconventional placing of the Pololu board for example, and other details that we noticed that need to be changed. Pololu boards are boards that are used for actuation. In the final design of the boards (see later sections) all these components will also be designed and manufactured together with the rest of the boards.

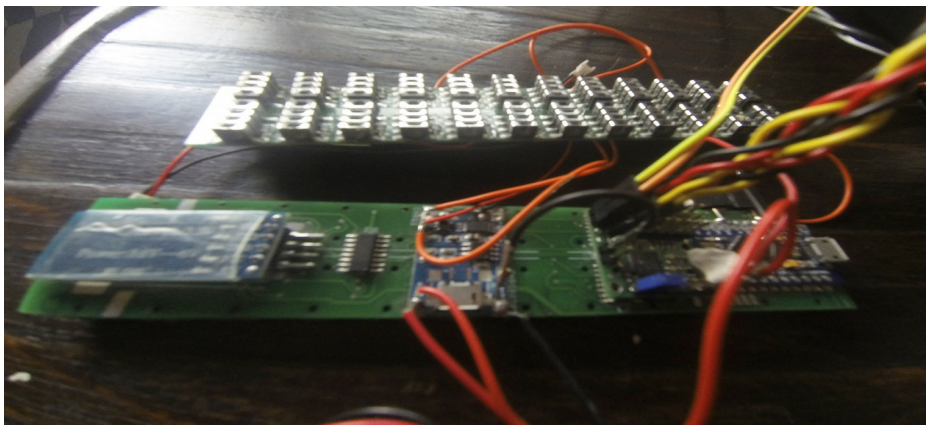


Figure 15. The two sides of the prototype boards in their final stage. On the board on the bottom we see first on the left the Bluetooth chip, then the battery charging chip in the middle. After it the Pololu board for controlling devices, and lastly on the right the mini Arduino chip for connecting the board to a computer. The multiplexers are under the Pololu board.

Lastly, Figure 16. shows the mini jack converters we made for sensors. As mentioned a mini jack cable can carry three channels of information which is enough for the vast majority of sensors. The converters were made by soldering the outputs pins of components that connect to a sensor to mini jack sockets. After our interviews with people in the field

(see in the very next section), we have come across ways to have our boards working also with sensors that demand a higher number of channels in order to work properly.

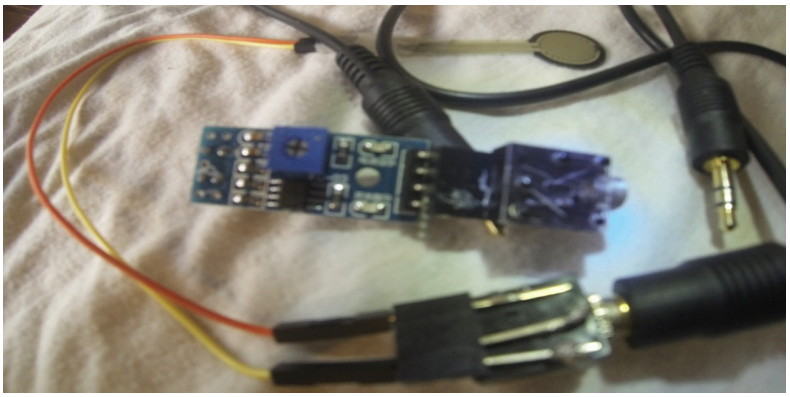


Figure 16. The two sides of the mini jack converters used for pressure and distance sensors. Unlike the pressure sensor, the distance sensor connects directly to the converter without any jumper cables.

5. Plan of Action

The boards, converters and software programming of the proposed program were more or less ready shortly after the beginning of this semester. From that point it was decided that the plan of action for this study should have three steps. The first step would be interviewing people in the field to see what they like. The second to conduct an experiment with users to see how they react to the proposed product. The last step was analyzing the feedback from the first two steps and coming out with improvements for the proposed product. What follows is a presentation of these steps.

5.1. Step 1: Interviewing people in the field

SEA is a department of Aalborg University that helps students with their ideas for start up companies. Our team applied to enter this program of Aalborg University. The procedure to get accepted at SEA entails among others a presentation of the idea in front of a panel of experts. Shortly after our presentation we were informed that we got accepted and placed in the so- called Discovery phase as a start up at SEA. This phase suggests primarily to the students to conduct interviews with experts in the field to find out more details about the market. According to SEA this is a necessary step to go through before moving on with the

design and production of a product. During this phase we were also given a mentor, Ivan *Butler* for better navigation in such situations.

As being told by SEA, an important distinction between people in the field are the *users* and the *payers*. In our case the users would be the students that would use our product and payers the teachers or anyone else in a position to decide whether buying a product like ours or not. We were asked by SEA to focus more on the payers. The central motto given to us by our mentor was to “just find out what they like”. Therefore, it was suggested to us that the interviews should not be limited to a specific structure, allowing this way the interviewees to tell us anything they might think it is useful.

Our team interviewed university teachers, managers of organizations that teach robotics to children and artists that use similar products in their work and performances. We interviewed ten people in total. Both members of the team were present at every interview taking notes and discussing them later. The average duration of these interviews was around an hour. We were advised not to record the interviews so that people feel more relaxed. The questions we asked had to do with specific aspects that each interviewee was more relevant with. Therefore some interviews focus more on technological aspects than others for example. Something pinpointed to us by SEA was not to repeat the same interview ea ch time, but progress from interview to interview, towards reaching a deeper understanding of the field we attempt to enter.

5.1.2 Results of the Market Interviews

Every interview had great value for us because it highlighted different aspects towards making a better product. At the same time we were able to obtain a broader view of the field in which our product is situated. All the interviews gave us very useful feedback, especially the interviews with Casper Møller Bartholomæussen and Markus Löchtefeld that are analyzed in the next paragraphs.

Casper is the manager of Coding Pirates (https://codingpirates.dk/) in Aalborg. Coding Pirates is the largest organization in Denmark that teaches robotics to small children (from the age of 5 and more) in around 50 cities across the country. We have had an interview also with a teacher from this organization that seemed to pinpoint the same issues that Casper did. One of them was the importance of seeing real time the values coming from a sensor or sent to a device. Casper mentioned that both the LM products they were using and some Arduino boards were not giving them this possibility. This occurs because they would have to first upload a sketch to the boards and then see its behavior. This procedure was thought to break the momentum of enthusiasm for children as they have to wait a few seconds each

time to see the results. Another important information was that Coding Pirates also use Little Bits in their projects. According to Casper the Little Bits products are very easy for anyone to grasp their functions but children get bored with them soon as the functions they offer are rather limited. Something that our team considered a good sign was that Casper found our product interesting and appropriate for an organization like Coding Pirates. The reason was not only because they could read real time the values coming from a sensor or sent to device, but also because they would have to use simple mini jack instead of jumper cables. Products like LM, Little Bits and Smart Gurlz usually have just one specific physical object the user can actuate, leaving not much space for creativity for users to make their own constructions. That was something pinpointed by Casper, adding that it is important that our product offers easily the possibility of making original interactive constructions without having to worry about circuitry and other issues like with the Arduino boards and software.

Markus Löchtefeld on the other hand, is an Assistant Professor in Medialogy in Aalborg University. Markus is an expert in electronics and he had no difficulty understanding fast the character of our product. Knowing his background in electronics we brought to the interview one of our boards so that he gives us first of all a technical feedback on how we had made our boards. He told us that we have chosen indeed the correct components to make our boards function they way we wanted and he also liked the overall design. He mentioned that he would like to have such a board for himself, primarily because of the practicality that mini jack cables offer in relation to jumper cables.

Additionally, he said that perhaps it would be a good idea to create a board just with mini jack inputs that gets connected to an Arduino as an add on. This, even if it would be practical from Arduino users is something we think does not fit with perspective we have for our product. One concern Markus expressed was about the software we are planning to create. He found almost no value in making one as there are so many out there that possibly we cannot compete in efficiency, plus the fact that people are already accustomed into using them instead. Though this is true indeed, we still have the opinion that we should have our own software, while the board can also work with other programs as well (see later section Step 3).

Another issue raised by Markus was whether our boards could handle data coming from sensors that use more than three channels of information. One kind of these sensors are the so-called I2C sensors that are widely used. Being able to read values from such sensors as the I2C ones, has been a topic that concerned us, as they can give very precise data used in many different projects. Though we have not yet tried to make it, we believe we have found the solution to it by adding some more code on the board section (see Step 3 section).

5.2 Step 2: Testing the product with users

5.2.1 Procedure

As mentioned in the background section, a user experience test was conducted with ten users in total. I tried to follow the principle of simulating real life conditions as much as possible in the experiment (Ross and Morrison, 2005). Therefore, I came up with a series of actions trying to secure that the overall experience for the participants would be very close to what they would experience if they came across our product. The first step was to give the participants a small introduction about the product and for what problems it can be used. Spending 2-3 minutes explaining to them about the product was thought to be the equivalent for them reading or watching an introductory video about a product.

Once the participants told me that they understood the concept of the product, I went to show them in the user interface software the most basic functions. These functions were about connecting nodes (therefore different devices and sensors extensively) and manipulating values that are either sent to items or received by the sensors. This is done with the use of a keyboard and a mouse. After this short introduction the participants were asked to connect some nodes and manipulate their values in order to see how the program works. At this point no motor or sensor was affected, it was only for understanding how the program works. Once they made this then the actual test took place. This test consisted of three tasks controlling a visual item on a screen and two motors using three analogue sensors. These tasks are presented right after. At the same time I measured with a stopwatch the time it took for every participant to complete each task. Finally, the participants had to answer a questionnaire of six questions in total.

Figure 17. shows the experiment set up. The participants were sitting on a desk and had three sensors to use. Two were pressure sensors and one an infrared distance sensor. Two rotating motors we also on the table. One of the motors had attached a stick to point so that the user could see clearly how much the motor was rotating each time. Additionally I marked with a tape three different positions within the rotation range of this motor. This was done so that I could ask the participants in the last task to make the motor point at one of these positions specifically. I found that this way I could see how easy can participants understand how to connect initially and then control a motor with a sensor with our product. In the video for this project one can see exactly how the experiment was set up.

Figure 17. The experiment set up. The user uses a keyboard and a mouse to set the connections between the sensors, the motors and the visual on the computer screen.

Figure 18. shows what participants came across during the experiment. A patch showing only the incoming values from the sensors and a screen showing which ports of the board were active. Additionally, there were nodes to connect the nodes with the values coming from the sensors. These nodes were connected and affected the two motors that were on the table and the rotation and inner radius of a segment that appeared on a second screen. So in total four possible different outputs for three incoming values. Lastly there was the “Map” node, that would allow the users to set the desired output from a sensor to an item. After the completion of each task the participants should state that they fully understood what was happening in the program. The three tasks were:

-Connect one specific sensor to a specific output

-Connect each of all three sensors to a specific output. All four outputs were used with one sensor connected to two outputs.

-The same as task 2 but with different connections an additionally, one motor should point at a specific point when the pressure sensor connected to it is fully pressed.

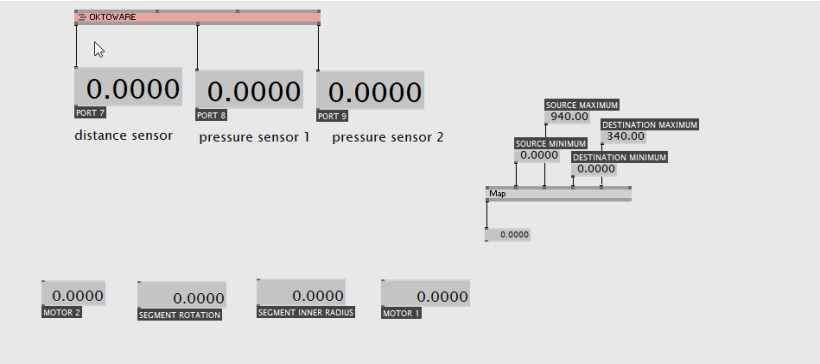


Figure 18. The interface used for the experiment. On the right, the “Map” node¨ with explanatory comments on its input pins.

5.2.2 Results of measuring time

Figures 19,20 and 21 show the results of the participants for each task. In all three graphs each participant has his/her own color. Under each figure a detailed presentation of the results per task.

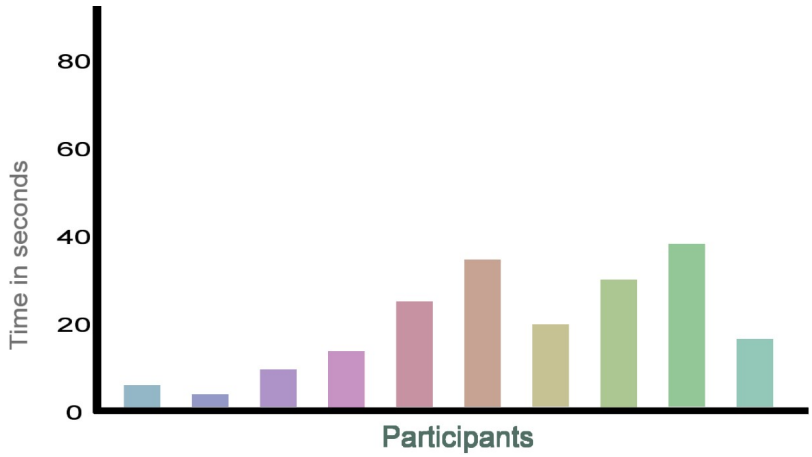


Figure 19. The results for the first task, connecting one input from a sensor to an item

For the first task the scores in seconds: 6.1, 3.5, 10.3, 15.5, 29.0, 40.5, 22.7, 35.1, 44.8, 18.7. The average score is 22.62 seconds and the median value is 34.75 seconds. The fastest participant was at 3.5 seconds and the slowest at 44.8, giving the total range of 41.3 seconds for this task.

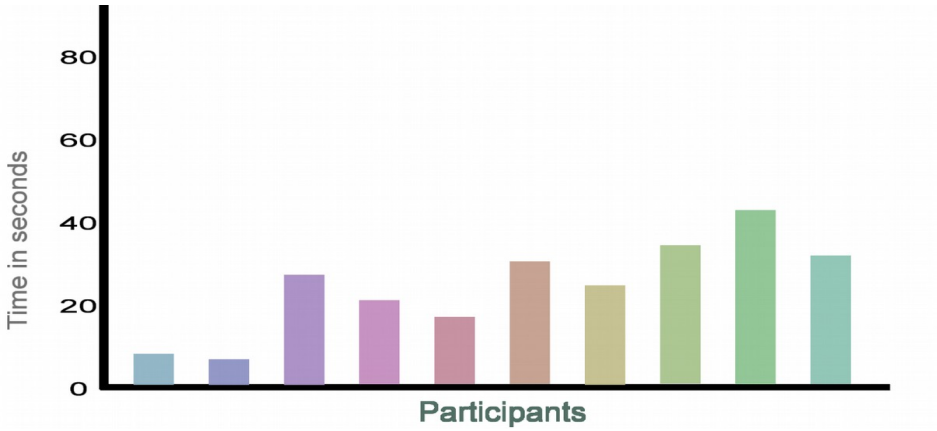


Figure 20. The results for the second task, connecting all three inputs from sensors to four items

The scores in seconds for the second task: 9.0, 7.5, 32.1, 24.5, 19.5, 35.7, 29.0, 40.3, 50.7, 37.4. The average score for this task is 28.57 seconds and the median value is 34.75

seconds. The fastest participant was at 7.5 seconds and the slowest at 50.7, giving a total range of 43.2 seconds for this task.

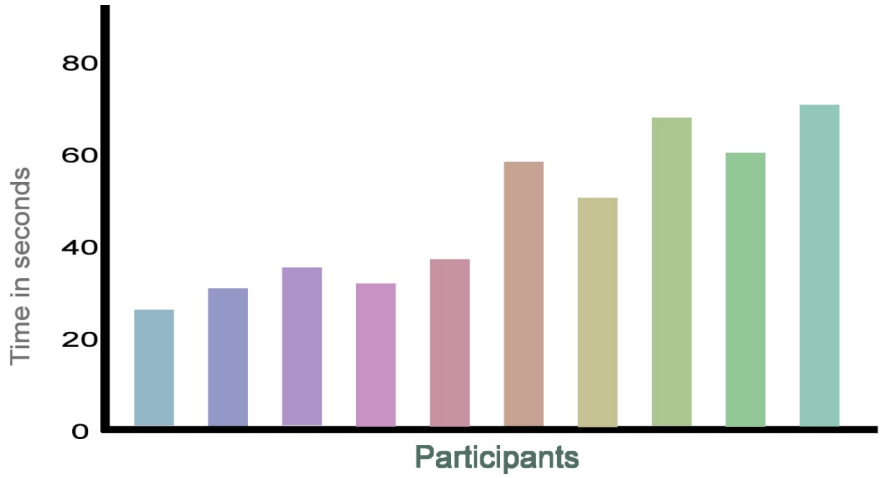


Figure 21. The results for the second task, connecting all three inputs from sensors to four items, one of them with specific values needed in order for the motor to point at a specific position when a pressure sensor was fully pressed.

Finally, the scores in seconds for the third task: 25.3, 30.1, 34.5, 31.3, 36.6, 57.7, 50.1, 67.3, 59.9, 70.3. The average score for this task is 46.31 seconds and the median value is 47.15 seconds. The fastest participant was at 25.3 seconds and the slowest at 70.3 seconds, giving a total range of 45 seconds for this task.

5.2.3 The questionnaire and its results

The questionnaire consisted of the following six questions:

1. From 1 to 7 how easy did you find working with our software and hardware?

2. From 1 to 7 how engaging did you find the procedure?

3. From 1 to 7 how much do you see yourself using something like this for ideas of yours?

4. From 1 to 7 what was the difficulty level making the tasks asked?

5. From 1 to 7 how hard do you think it would be for someone to make things with this product?

6. From 1 to 7 how boring did you find completing the tasks?

The questionnaire was thought to examine three areas with a pair of questions for each area, randomly put in it. Questions 1 and 4 was to see how easy or difficult was for the participants to complete the tasks asked. To find out how interesting was the whole procedure for the participants, the questions 2 and 6 were used. Lastly, questions 3 and 5 were used to see how appealing participants found the proposed product generally. Along with the questions there were notes on what the numbers from 1 to 7 were corresponding. Therefore, for question 1 for example 1 would be corresponding to “very easy”, 2 to 5 to “moderately easy” and 6 to 7 to “hard”. As one may notice the pairs of questions 1/4 and 2/6 are asking information for the same topic from an opposite point of view. That was to check the validity of the answers. If a participant for example gave a 1 to question 1 and a 5 to question 4, that would be an invalid result. In that respect, none of the participants delivered invalid answers. The lowest, highest and average scores for all six questions:

-Question 1. Average value 3.6, Lowest value 1, Highest value 6

-Question 2. Average value 3.2, Lowest value 1, Highest value 5

-Question 3. Average value 3.9, Lowest value 2, Highest value 7

-Question 4. Average value 1.9, Lowest value 1, Highest value 3

-Question 5. Average value 3.3, Lowest value 1, Highest value 5

-Question 6. Average value 2.3, Lowest value 1, Highest value 3

After each experiment was complete I spent a few minutes with the participants, asking for further feedback on the experiment and the idea of the product. Half of the participants, the first five in all results presented, were Medialogy students. Perhaps this can explain the slightly better time scores for completing the tasks asked in the previous section. The same slight difference was reflected in the questionnaires both for how interesting and easy they found the product to work with.

5.2.4 Discussion on the results of the experiment

The results gave indications that the proposed product was comparatively easy for participants to work with. With the exception of one instance all the participants were able to complete the tasks in less than a minute. However, this is not regarded as a definite sign that the proposed program is indeed easy to use. It needs to be taken into account that the

participants were given moments before completing the task, a brief reference to the nodes that they had to use in order to complete these tasks. It is thought by our team that in normal life conditions someone would first have to learn many other things about a program before attempting to do something specific with it. One topic that emerged from these tests was to see how to make it easy for any beginner to learn how to do specific tasks as fast as possible with our product. Therefore, even if the tests are statistically insignificant due to the small number of participants, they gave significant indications on how we should approach introducing our product to new users.

Towards the goal of making things easier for someone to understand how to work with our product we came to realize that an important factor will be the educational material we will need to have on line for users to access. As mentioned in the Previous work section, the companies in the field release many tutorials, helping patches etc. for this reason and this is what i tried to imitate for the participants in the tests. A look at the results of the questionnaire show that participants found completing the tasks comparatively easy. If so, his is the level we should reach for any user in the future.

5.3 Step 3: Improvements for the proposed product

The third step of this study was to suggest improvements and further development of the proposed product. The following subsections analyze the software and hardware aspects of this third step.

5.3.1 Suggested improvements to the software

As mentioned in previous sections, we see ourselves using a user interface very similar to Max/MSP and VVVV. We believe that the node-based style of programming is the most appropriate for the functions we want our software to have. Apart from that, this is a way of programming that many users and professionals are already accustomed to. A quick look at the user interfaces of programs for small children (like the ones presented in the “Field of project” section 3) indicates that in the next years there will be many youngsters that will be familiar with this kind of programming. Therefore, it would be easier for our product to be introduced to them. Additionally, though we have a software and hardware product, we want users to be able to use separately one of them if they want. This way a Scratch user for example should be able to use our hardware in Scratch.

However, we do not see our software as of secondary importance in relation to the hardware at all. On the one hand, Markus was correct when he mentioned that trying to create a program to compete with other similar (and established in the market) would be very hard and require lots of time. On the other hand, we do not see our program as a competitor to such programs even if our software is thought to be working also with other hardware devices like Arduino boards. We think of our software as a program dedicated to handling and manipulating data from many kinds of sensors to motors. And this is how we will be introducing it in the future. Having the software such a character and mission, we do not see it having more than 50 nodes maximum. Programs like Max/MSP or VVVV have hundreds, used for many other purposes like visual and sound manipulation. We know that we will have both nodes that have identical functions with the ones we see in programs like VVVV, but also others that correspond to specific ports of the board.

In combination with on line tutorials, having a small number of nodes can make the program very easy to learn. This does not translate into having software with limited capabilities. As shown in Figure 5. with such user interfaces one can create very complex structures. Additionally, given the fact that the software is dedicated to data handling and manipulation, 50 nodes should be enough to offer many possibilities to the user. In parallel, apart from the nodes we will be creating configurations for other more specific devices. For example: a user should very easily be able to connect a Kinect to our software and with our board control motors.

Figure 22. is a screenshot from a VVVV patch in which I experiment with the design of the user interface of our program. Starting from the right, we have a tool box that is optional for the users to use, but can be helpful for beginers. The different black signs correspond to different kinds of nodes and functions. Once clicked the different nodes of each category appear with a description of what they do. This way users can easily find the tools they want to construct a patch. Next to them we see two columns with 12 colored items each. These 24 items correspond to the 24 mini jack inputs of our hardware. If a sensor is connected to a port, its number appears bigger in the tool box bar in order to indicate this to the user. One may notice that the one column has circles while the other one squares. This is so because it is thought that half of the ports of the board will have some extra functions in relation to others and we want to somehow indicate this in the software. One of these extra functions is for example handling data from an I2C sensor, something that Markus mentioned to us and we think also it is important. In the main workspace area we see the construction of a patch. The little colored squares represent the nodes that we will have in our software. The color they have corresponds to the port they are connected. This way the user can see faster what is happening in the patch. These colored squares in the picture can also be an encapsulated sub patch for space economy. It will have a color, however, again, indicating which port or

ports are related to it. Once a sensor is connected to the board then if used in the patch, it appears automatically with the values coming from the sensor on it.

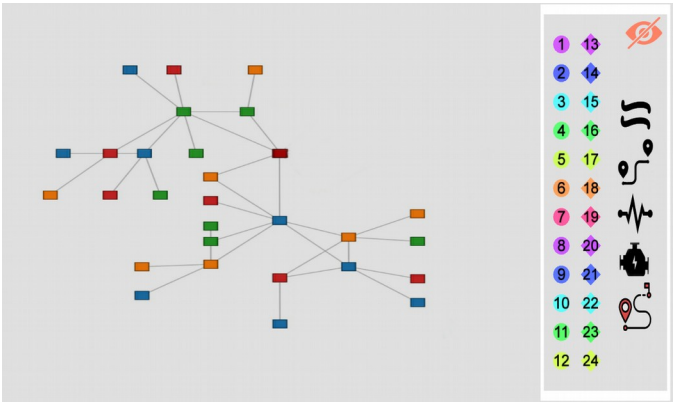


Figure 22. A screenshot in VVVV experimenting with the user interface of the software.

5.3.2 Suggested improvements on hardware

As mentioned in previous section, our first prototype boards were ready before conducting the user experience tests. We received all the components and assembled the boards in March. It has been very constructive for us not only testing them with users but also speaking about them during this semester. During this period we have also been testing ourselves the functionality and stability of these boards. Right now we are at the phase of designing new boards and ordering the components needed for them.

Even if the overall concept and design has not changed drastically, there are many technical details that we are changing, like parts of the coding within the board. One aspect of this issue is to have I2C sensors working with the board, something noted also by Markus. For this to happen we are thinking of using an alternative way of connection of some of the mini jack ports on the board. This way two of them will be able to handle six channels of information coming from one I2C sensor. Another topic is to have the board being better configured by other programs. So far we have had the board working with VVVV and the Arduino software with no problems, but we have to rewrite the code in order to reach the desired state. Lastly, we came up with a way of connecting Pololu-like boards for actuating motors on what we consider it should be the main board, just with a mini jack. That is done by sending the information as a string through the channels of a mini jack cable.

So far we have had all components for analogue sensing and actuation on one board. In the future we intend to have two different boards, one for sensing and one for actuation. Both of

the boards will use mini jack converters but this way someone could buy only the one if he or she wants to. The boards will be connecting with each other only with a mini jack cable. At the same time, one electricity source is charging both of the boards if needed.

We have seen on some occasions installations crushing due only to one weak cable connection for example. This has been one more reason why we wanted to use mini jacks and avoid any soldering for the user. We envision our boards not only being easy to use but also to be reliable for “heavy duty” work, for example by having the boards placed in a plastic case securing them from dust, sun light etc. Figures 23 and 24 show some of our design thoughts for our new generation boards. One may notice the similarity with the ones used for this semester. We came to the decision using more or less the same shape for the boards, also after positive feedback we received for it from various people during this semester. Lastly, each mini jack port on the new boards will have a specific color that corresponds to a specific node in our software program. It is estimated that with the completion of this semester we are going to start building our second generation of prototypes in order to do further testing and for pitching our business idea.

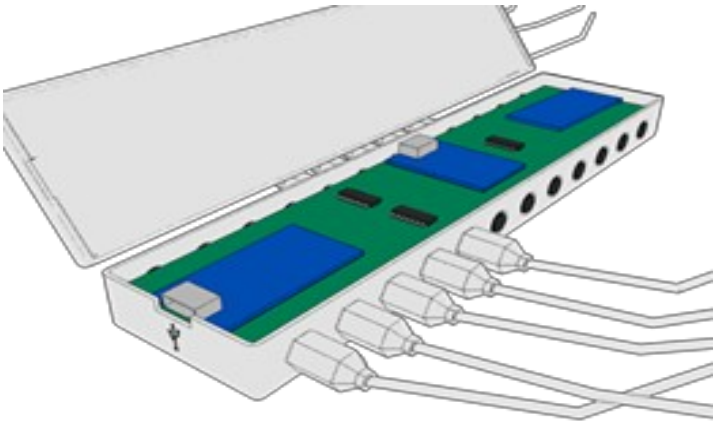


Figure 23. The new boards, redesigned in order to fit precisely in a plastic case.

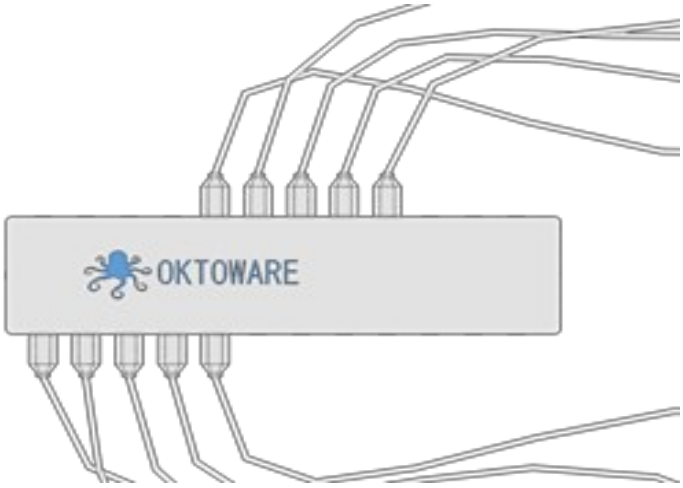


Figure 24. How we envision the board should look like while in use.

6. Conclusion

The work done in this semester is considered just the beginning towards creating a competitive product. In general the feedback we received from experts in the field was positive and we read this as an indication that our idea might be valid after all. The results also showed that the users did not find it hard to work with our product. This has been received as some kind of reward for our team, especially for the boards as we had no experience with all the procedures that manufacturing a boardsentails.

However, and especially after being introduced to market practices by SEA, we realize that in order to make it we need to take care of other aspects as well. One of them would be to learn to communicate the idea of our product better and where and when to do something like this. As both of the members of the team are coming from Medialogy, this is an area we have not been familiar with. In parallel with making our software and hardware product this is something we will have to spend time on it in the future.

7. References

Atmatzidou S., Markelis I., Stavros Demetriadis: “The use of LEGO Mindstorms in elementary and secondary education: game as a way of triggering learning”, Workshop Proceedings of SIMPAR, Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS Venice(Italy) 2008 November,3-4 ISBN 978-88-95872-01-8, pages 22-30, 2008

Bazylev D., Margun A., Zimenko K, Kremlev A., Rukuiza E., “Participation in robotics competition as motivation for learning”, Procedia - Social and Behavioral Sciences 152, pp. 835 – 840, 2014

Blenker, P., Korsgaard, S., Neergaard, H. & Thrane, C., “The questions we care about: paradigms and progression in entrepreneurship education”, pages 417-427, Industry and Higher Education”,25, 2011

Blenker, P., Frederiksen, S. H., Korsgaard, S., Muller, S., Neergaard, H. & Thrane, C., “Entrepreneurship as everyday practice: towards a personalized pedagogy of enterprise education”,pages 417-430, Industry and Higher Education, 26, 2012

Bryman, A., “Social research methods”, New York: Oxford University Press., 2008

Community Network for Youth Development, “Youth Development Guide: Engaging young people in after school programming”, Chapter 7, page 147, SNYD, San Francisco, 2001

Corbin, J., Strauss, A.: Grounded Theory Research: Procedures, Canons and Evaluative Criteria. Qualitative Sociology Volume 13(1), 3-21, 1990

Depcik C., Assanis D. N., “Graphical User Interfaces in an EngineeringEducational Environment”, University of Michigan, 2004

Du Boulay, B., “Some difficulties of learning to program. In: Solo way”, E., Spohrer, J.C. (Eds.),Studying the Novice Programmer, pages 283–299, London, Lawrence Erlbaum Associates, Holland 1989

Eriksson, P., Kovalainen, A.: Qualitative Methods in Business Research 1st edition, SAGE, London (2008)

Freitas H., Oliveira M., Jenkins M.,Popjoy O., “ The Focus Group, a qualitative research method. ISRC, Merrick School of Business, University of Baltimore (MD, EUA), WP ISRC No. 010298, February 1998.

Gal-Ezer J., “Teaching Software Designing Skills”, The open University of Israel, 1994

Gibb, A., “Entrepreneurship and enterprise education in schools and colleges: insights from UK practice”, page 122, International Journal of Entrepreneurship Education, 6, 48, 2008

Hancock B., Ockleford E. and Windridge K., “An Introduction to Qualitative Research,” Trent RDSU, page 4, 2007

http://firmata.org/wiki/Main_Page

http://mindstorms.lego.com

https://www.bizjournals.com/washington/news/2017/11/13/local-startup-s-segway-riding-

doll-scores-shark.html

https://codingpirates.dk/

https://cycling74.com/products/max/

https://www.dailymotion.com/video/x6ivmy3

https://en.scratch-wiki.info/wiki/Hardware_That_Can_Connect_to_Scratch

https://github.com/khanning/scratch-arduino-extension/issues/45

https://www.postscapes.com/iot-visual-programming-tools/

Ikävalko M., Ruskovaara E., Seikkula-Leino J., “Rediscovering teacher’s role in entrepreneurship education”, Lappeenranta University of Technology, Finland, 2009

Kanji F., “UX Research & Testing Techniques”, Akendi, 2015

Kelemen M., Kelemenova T., Virgala I.,Mikova L.,Liptak T., Technical university of Kosice, “Rapid Control Prototyping of Embedded Systems Based on Microcontroller”, Faculty of Mechanical Engineering, Letna 9, 04200 Kosice, Slovakia, 2014

Kelleher C., Pausch R., “Lowering the Barriers to Programming: a survey of programming environments and languages for novice programmers”, Carnegie Mellon University, 2003

Khambete P., Athavankar U.,“Grounded Theory: An effective Method for User Experience Design Research, page 12, retrieved from: http://www.idc.iitb.ac.in/resources/dt-aug-2010/Grounded%20Theory.pdf

Kieras D., “A guide to GOMS Task Analysis”, University of Michigan, 1994

Lackeus M., “Entrepreneurship in Education: What Why When How, Entrepreneurship360 background paper”, Organisation for Economic Cooperation and Development (OECD). Available at:
http://www.oecd.org/cfe/leed/BGP_entrepreneruship-in-education, 2017

Malerba F., “Knowledge intensive entrepreneurship: going beyond the Schumpeterian entrepreneur”, The Invernizzi Center for Research on Innovation, Organization, Strategy and Entrepreneurship, FAPESP, Sao Paolo, 2016

Marinilli, M., “The Theory Behind User Interface Design, Part One”, Download: http://www.developer.com/design/article.php/1545991, 2002

Marinilli, M., “The Theory Behind User Interface Design, Part Two”, Download: http://www.developer.com/design/article.php/156468, 2003

Manzo V.J.,: “Max/MSP/Jitter for Music: A Practical Guide to Developing Interactive Music Systems for Education and More”, ISBN: 9780199777679, 2011

Mattingly C., “Healing Dramas and Clinical Plots: The Narrative Structure of Experience”, Cambridge University Press, 1998

McCarthy J., Wright, “Technology as Experience”, MIT Press, Cambridge, MA., 2004

Mc Kelvey M., Heidenmann Lassen A., “Managing Knowledge Intensive Entrepreneurship”,Chapter 3, page 72, Cheltenham United Kingdom, 2013

Merkouris A., Chorianopoulos K.,Kameas A., “Teaching Programming in Secondary Education Through Embodied Computing Platforms: Robotics and Wearables”, Journal ACM Transactions on Computing Education (TOCE), Volume 17 Issue 2, Article No. 9,June 2017

Mulder, F.,“van BÈTA – naar DELTA-discipline” (in English: Computer Science: from a BÈTA to a DELTA subject). Informatica, Tinfon, 11, 48, 2002

Myers B., Hudson S. E., Pausch R., “Past, Present and Future of User Interface Software Tools,” ACM Transactions on Computer Human Interaction, pp. 3-28, 2000

Nicholls D., “Qualitative Research: Part Three – Methods”, Auckland University of Technology, International Journal of Therapy and Rehabilitation, 2009

Ospennikova E., Ershov M., Illjin I., “Educational Robotics as an Innovative Educational Technology”, Procedia- Social and Behavioral Sciences, Volume 214, pp. 18 – 26, 2015

Papert. S., “Mindstorms: Children, Computers, And Powerful Ideas. Basic Books, New York, 1980

Pacho T. O., “Exploring Participants' Experiences Unsing Case Study”, Faculty of Education, University of Hamburg, International Journal of Humanities and Social Science Vol. 5, No. 4, April 2015

Rasmussen, A. & Nybye, N., “Entrepreneurship Education: Progression Model”,Odense C, Denmark: The Danish Foundation for Entrepreneurship – Young Enterprise, 2013

Reed P. K., “A Comparison of Programming Languages For Graphical User Interface Programming”, University of Tennessee-Knoxville, 2002

Reeves S., Kuper A., Hodges B. D., “Qualitative research methodologies: ethnography”, 2008

Resnick M., Martin F., Sargent R., and Silverman B., “Programmable bricks: Toys to think with”, IBM Syst. J. 35, 3.4 (1996), pages 443–452, 1996

Resnick M., Maloney J., Monroy-Hernandez, Rusk N., Eastmond E., Brennan K., MillnerA.,Rosebaum E.,Silver J.,Silverman B.,Kafai Y., “Scratch: Programming for All”, Communications of the acm, vol. 52, no. 11, 2009

Ross S.M., Morrison G.R.: Experimental Research Methods, University of Memphis, Wayne State University, pp. 1021- 1043, 2005

Rudd, J., Stern, K. and Isensee, S., “Low vs. high fidelity prototyping debate. Interactions” 3(1), pages 76-85, ACM Press, 1996

Sartatzemi M., Dagdilelis V., Kagani K., “Teaching Introductory Programming Concepts with Lego Mindstorms in Greek High Schools: A Two Year Experience”, University of Macedonia, 2008

Scaradozzi D.,Sorbi L.,Pedale A., Valzano M., Vergine C., “Teaching robotics at the primary school: an innovative approach”, Italy 2015

School Education Gateaway, article retrieved from: https://www.schooleducationgateway.eu/downloads/entrepreneurship/Denmark_151022.pdf

Tochacek D., Lapes J., Fuglik V., “Developing Technological Knowledge and Programming Skills of Secondary Schools Students through the Educational Robotics Projects”, Procedia -Social and Behavioral Sciences, Volume 217, pp. 377- 381, 2015

Van Diepen, N.,“Elf redenen waarom programmeren zo moeilijk is” (in English: Eleven reasons why programming is so difficult), pages 105–107, Tinfon, 14, 2005

Walker M., Takayama L., Landay J. A., “High fidelity or low fidelity, paper or computer? Coosing attributes when testing web prototypes”, Computer Science Division, Univerisity of California, Berkeley, 2002

Wall M. B., Ulrich K. T., Flowers W. C., “Evaluating Prototyping Technologies for Product Design”, Massachusets Institute of Technology, USA, 1991

White Paper, “Best Practices for Developing a Graphical User Interface”, QSI Corporation, 2009

www.littlebits.com

www.smartgurlz.com

www.vvvv.org