1. Abstract
The Arduino prototyping platform is a creative outlet for any user to produce technology through easily usable hardware and software. However, the steep learning curve of the Arduino programming language holds back many of its users. Through graphical and tangible programming languages users can effectively learn how to program for the Arduino platform. Incorporating tangible human computer interaction in Arduino technology yields positive results in learning ability and enthusiasm of programming language learnt. A new language for Arduino can be developed through interactive and participatory design practices which intends to be valuable for novice programmers.

2. Introduction & Motivation
Arduino is an open source electronics prototyping platform (Arduino, 2005) that is used by expert to novice programmers. These novice programmers can be anyone from a child to artists. The aim of Arduino is to provide flexible, easy-to-use hardware and software but the difficulty lies in the Arduino programming language which controls the hardware and is controlled by a set of C/C++ functions. This language is difficult for novice users.

This paper aims to look at the current work done with Arduino programming language and the difficulties faced by novice programmers. It looks at the challenges and use of participatory design to help users with the learning of the Arduino programming language.

Firstly, the paper looks into the Arduino programming language then the paper explores current studies which have used Arduino to teach programming. Followed by the role of participatory and interactive design in development. Other topics explored are tangible human and computer interaction and its particular use in Arduino projects. Lastly, comparing tangible and graphical programming for the most effective in learning a programming language.
3. Presentation of Findings

3.1. Programming Language

The higher level short comings of Arduino is the Arduino programming language itself. Programs are compiled into a C/C++ but Java, Python, Processing are used on a personal computer to run communications with the Arduino. The process of code to implementation can be a complex set of steps for a novice programmer as the code implementation is either C/C++, python or Java. This problem forms part of a larger set of problems related to programming in general not exclusively to Arduino. In programming as Musical Instrument which aims to deal with the psychology of programming (Blackwell, 2005). Blackwell, describes the problem as a conflict between what users want and what users need.

The broader problem of programming language design as seen by a non-programmer is one that is very complex (Miller, 1974). The scope of this paper is not to explore these fundamental issues which face programming in relation to the broad topic of computer science. The aim is to explore the options available that may lead to positive programming attitude towards Arduino by all its users.

TurTan uses Logo which is a graphical programming language to encourage its users to be creative and explore the potentials of programming (Gallardo et al., 2008). This example of a tangible programming language proved to be a positive with users, much of this positivity being credited to its tangibility attribute. Users found it to be enjoyable and easy to understand. Gallardo concluded that a tangible programming language yields good learning of basic programming language skills with its users.

3.2. Using Arduino to Teach Programming

The AdMoVeo study that has been running since 2008 with 400 students was conducted for industrial design students with the main aim to increase the basic programming skills of students (Hu & Alers, 2010). The study was conducting with the knowledge that the ability to link theory and practice is difficult. This ability to link the two frustrates users/students. The study created a robotic platform based Arduino. Hu et al. designed their own software so there was no C++, only a Processing program environment. The project was successful as it “motivated and encouraged students to explore their creativity with their passions in graphical and behavioural design” (Hu & Alers, 2010). Along with “inviting the students to apply and grow knowledge in their design projects” (Hu & Alers, 2010).
Arduino was also used to teach child friendly programming languages (Eisenberg et. al, 2009). Confliction on ideas about what exactly children should be able to learn. These being recursion, procedure and variables, utilising graphical programming or robots that are programmable. It is shown that these topics are important to children learning of a programming language as it provides meaningful information to programming. One of the implementations was a paper based Arduino prototype. Ambient programming which is informal , moment-to-moment ways of programming was shown to be effective with children (Eisenberg et. al, 2009).. Arduino in particular with children is useful but it is “hardly a revolutionary design” (Eisenberg et. al, 2009).

There exists a gap in programming education for junior high school students, when a student is in elementary school a purely graphical environment is learnt and complex text based programming is learnt in high school (Cheung et al., 2009). The study aimed to address the gap in-between i.e. junior high school. Bricklayer is a text enhanced graphical programming language that was proposed (Cheung et al., 2009). This mix between the two methods of programming allowed the programming experience to be more tangible and immediately rewarding for the students. Giving students to gradually build up their confidence in programming.

3.3. Participatory design

Arduino holds potential in terms of its ability to effectively achieve participatory design (Hribernik et al., 2011). In this study, two products were developed using Arduino as the base in collaborations with its users. Evidence from both products developed showed the success of participatory design. These were shown to be effective especially when role playing and storytelling tools and methods were used as part of the participatory design process.

The Interactive Building project is another project which supports the success of using Arduino in interactive and participatory design (Bier, 2012).

3.4. Tangible HCI

Varesano’s master’s thesis explores the current situation in the Arduino programming language (Varesano, 2011). Currently, Arduino users program using the Arduino IDE which is the same as coding in C/C++. The current programming process is purely text based.
Each program written is called a sketch. Before being deployed onto the Arduino board the sketch is compiled and converted into a C program. Throughout Varesano’s paper he implements a variety of technologies built on top of Arduino to achieve tangible human computer interaction technology. These examples are the Palla prototype, translated from Italian means ball, which was implemented as a spherical game controller. The aim of Varesano’s future work is to incorporate user experience into the tangible component of prototypes he builds.

3.5. Tangible vs. graphical programming languages

Work done by Horn shows that tangible languages have an advantage over single mouse graphical interfaces (Horn et al., 2009). The tangibility aspect of the programming language is more inviting and more conductive to collaborative interaction. Both languages are equivalent in their ability to achieve understanding with a user and engaging its user.

4. Conclusion

Arduino in conjunction with participatory design and interactive design yields successful results. Once users are knowledgeable of the Arduino programming language they are able to implement a variety of solutions to numerous problems. The challenge is obtaining that knowledge for all users of Arduino, in particular the novice programmers. Tangible and graphical programming languages have been proven to be successful in stimulating and educating novice programmers. Both have been found to make an improvement in learning when used with participatory and interactive design. In terms of learning how to program, tangible programming has shown better results.

Thus, there is an opportunity to develop a tangible or graphical programming language for Arduino. Possibly a combination of the two along with adopting participatory and interactive design practices could yield an improved Arduino programming language that all of its users can use with ease and creativity. Focussing on how the users want to program and not how they should theoretically be programming.

5. References


