# PPIG 2011: User Configurable Machine Vision for Mobiles

Alistair G. Stead

*Computer Laboratory*
*Cambridge University*
*ags46@cam.ac.uk*

Keywords: POP-II.A. End Users, POP-III.C. Visual Languages, POP-IV.B. User Interfaces

## Abstract

We describe efforts to evaluate a new smartphone-based visual programming language, which gives users the ability to map their own musical blob notation to synthesized audio. We use the Cognitive Dimensions of Notations questionnaire (Blackwell & Green, 2000) to gain qualitative feedback, and use traditional observation methods to support our results. Using this methodology, we were able to find that musical and non-musical users both react positively to the system, with no marked difference between the two groups.

## 1. Introduction

As part of an on-mobile programming project, we developed a new domain-specific visual programming language (DSVL). The purpose of this language is to allow end users to create rules, which when triggered by some input, produce synthesised music.

To enable control of the synthesis, the user is able to train a classifier with a smartphone camera on a set of coloured blobs. After training, the user can specify mappings (using the DSVL) between blobs and synth tracks. During the performance stage, the system recognises blobs and triggers the appropriate rules, which then produce synthesized audio.

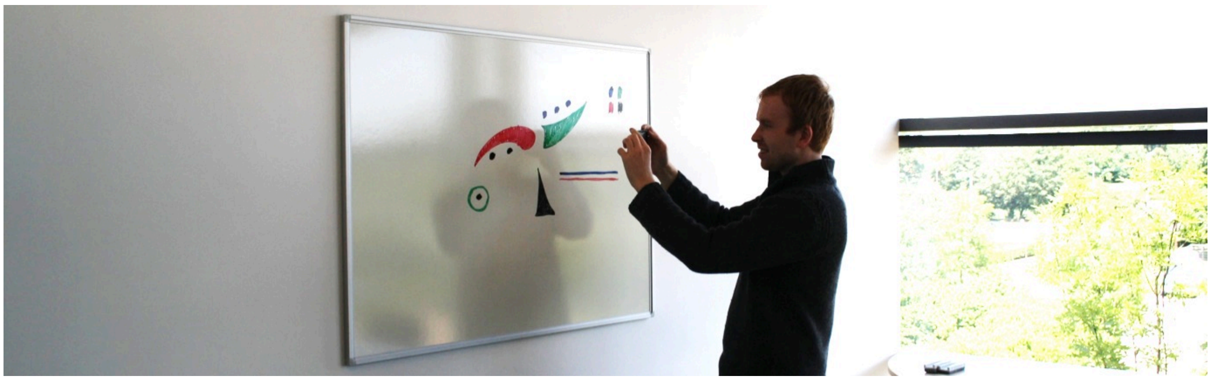

*Figure 1: System Usage Scenario*

DSVLs are restrictive languages, consisting of domain concepts and well-formed rules that evaluate the models. The language is intended to be used primarily by domain experts in digital music performance but we hope to obtain findings that have general relevance to all end users.

### 1.1. Background

Recent developments in technology have resulted in smartphones becoming significantly more powerful and useable. This is in part due to the multitude of sensors now available, including touch screens, proximity sensors, accelerometers, gyroscopes, microphones and cameras. The surge in popularity in this market has encouraged a huge amount of software development and experimentation.

End user programming (EUP) aims to give end users the ability to express powerful programming-like functionality, without being an expert in programming. The type of functionality needed is usually repetitive or unique to the needs of the user, and is therefore not included in mainstream software applications. While EUP has been implemented successfully on personal computers (e.g. Apple Automator (Apple)), there is little evidence of it being implemented on modern smartphones.

Visual languages have long existed in the form of diagrams, which are largely used in a professional context (e.g. design or software engineering). Visual programming (VP) is the specification of programs by manipulating graphical objects. The notion that VP could be more accessible to humans when programming than textual interaction (Blackwell, 1996) drives research into how to best represent concepts to users doing EUP. If we are to consider EUP on smartphones then it is important to choose a language appropriate to the form factor and potential use case scenarios. Touchscreen interaction supports direct manipulation of objects on the screen, which would be useful if using a visual language. Text is inherently difficult to interact with due to the size of the screen, on-screen keyboards occluding information and the small size of the text.

Visual programming allows the user to easily identify relationships between objects and reduce the semantic gap between their mental model of the system and the computational model. These advantages could make programming more appealing to novices. VP is therefore primarily aimed at users with little or no experience.

### 1.2 System Architecture

The system consists of three main stages: training a classifier, programming the condition-action rules, and the performance stage (see Figure 2). The user first trains a classifier on different coloured blobs in the camera's field of view so they can then be identified during the performance. The programming stage allows the user to describe the relationship between the appearance of a particular blob colour and one or more output synth tracks (concurrent synthesised output). After programming, the user can draw multiple blobs that give the system a particular behaviour during the performance stage, depending on the users movement of the camera over the blobs.

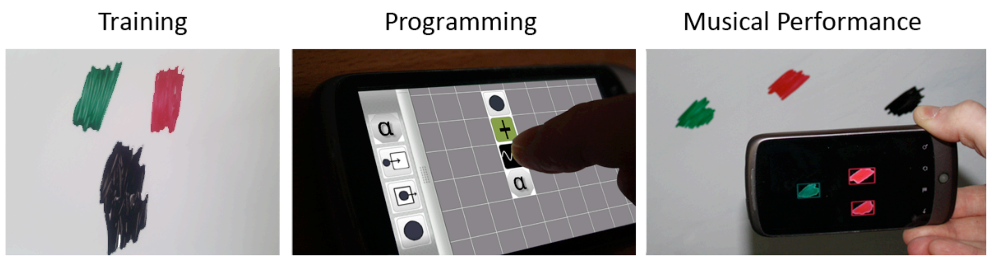

*Figure 2: System Progression*

Figure 2 shows the system being trained on three coloured blobs. A rule is constructed during programming that changes the pitch of synth α to a specified value when a black blob is recognised. During performance, when the black blob appears in the cameras view, the rule triggers and the sound is changed. As no rules refer to green and red blobs, nothing happens when they appear in view.

Our programming language draws from the concept of rewrite rules, as in Agentsheets (Repenning, 1993), BitPict (Furnas, 1991) and ChemTrains (Bell, 1991), where rules are executed if the conditions are satisfied. The rules in this system are specified by the end user and contain conditions that describe the existence of blobs in the cameras field of view. The remainder of the rule affects the properties of the synthesised music output.

## 2. Design of Evaluation Study

Nardi (1993) observed that end-users only take an interest in programming if it helps to improve domain productivity. It is therefore important that we thoroughly test the language in terms of creativity, flexibility and quality of results. These metrics are difficult to directly compare with other digital performance systems due to the differences in methodology. We are primarily interested in the usability of the language rather than comparisons with other digital music systems.

To evaluate effectiveness of the proposed DSVL, we devised a user experiment involving 8 participants who were asked to complete a series of moderately challenging programming tasks, which exposed them to all aspects of the system design. The participants were given a short introduction to explain that it is natural to find something new difficult to use initially and that negative as well as positive feedback was welcome. This was given with the intention of avoiding an experimental demand effect where the user might avoid criticising the system to win praise.

## 2.1 Cognitive Dimensions of Notations Framework

The Cognitive Dimensions of Notations framework (CDs) was developed to provide an evaluation technique for programming environments (Green & Petre, 1996). The CDs began to be used to design system-specific questionnaires, allowing designers to map user feedback directly to each CD. Blackwell and Green (2000) proposed a generalised questionnaire which gives users clear definitions of relevant CD's and allows them to feedback which features they felt were relevant to each CD.

This generic questionnaire is useful when evaluating our system as it carefully explains the terms and definitions to the participant and allows us to directly link responses to CDs of the system. Splitting the feedback into CDs will also help to identify themes. The questionnaire will contain four sections:

1. Background information
2. Definitions
3. Parts of the system
4. Questions about the main notation

"Background information" probes the users' experience with similar tools and their level of expertise when it comes to using such systems. "Definitions" will describe the terms we use to refer to parts of the system, such as notation. "Parts of the system" asks the participant to assign the proportion of their time devoted to performing specific parts of the task. "Questions about the main notation" gives a simple definition of each CD and prompts the participant to recognise features of the system that relate to each CD in terms of their goals. Due to space constraints, the questions will not be described here. For a full description, see (Blackwell & Green, 2000).

The generic CD questionnaire describes the programming language to the participant as a "notational system". This has potential to cause confusion if participants associate the term "notation" with the blobs they are drawing (which act as musical notation) rather than the programming language on the smartphone. To avoid confusion, the distinction was clarified at the beginning of the questionnaire.

## 2.2 Participants

To complete the usability study, we recruited two groups of participants: users with background in digital music performance and users without any background in digital music performance. Making a comparison between these groups allowed us to draw conclusions with external validity.

**User Group 1**

The target user requirements are that participants in group 1 should be very familiar with music and live digital performance. Recruiting domain experts is advantageous due to instant familiarity with domain terms (e.g. pitch, midi notes) and an ability to arrange complex, meaningful pieces of music, thoroughly testing the system.

**User Group 2**

Participants in this group should have no digital music performance background and therefore should be reasonably unfamiliar with domain terms.

**Programming Experience**

The programming experience of the user will not be a factor used to target or group users, as we do not expect programming experience to affect overall usability. This expectation is due to the fact the language is very different from languages in mainstream regular use, and that due to its visual, gamelike, simple design, the tool does not appear to be a programming language. In fact, during usability testing, the participants were not told explicitly that they were using a programming language. This decision was made to avoid potential user-alienation due to possible negative stereotypes attributed to the term (e.g. that it is inherently difficult, requires experience or involves mathematics).

As a safety measure, the user was asked in the final question of the questionnaire if they have had previous programming experience. This was not expected to be an influential experimental factor but could explain any unforeseen variation in usability.

A characteristic that may affect the participants' feedback is previous experience using smartphones. Although seemingly user friendly to those with experience, participants with no experience of touch screens or cameras on the phone could be confused. Although it is interesting to compare feedback of such participants, there was no requirement for smartphone experience during recruitment.

## 2.3 Core Design

We provided eight single-participant sessions over a period of 48 minutes, after which, the participant was asked to complete the usability questionnaire. The time was partitioned into 5 minutes for the initial tutorial, 7 minutes for each of the four tasks, and up to 15 minutes experimentation time.

The initial tutorial was a typed introduction to the system that gave an overview of the different categories of tiles, the "palette" object (allowing users to drag new tiles onto the grid) and the classification and performance process. Screenshots of the tiles and the grid were given to enable the users to understand which parts of the system were being described.

At the start of the experiment, the participant was instructed to express anything they found difficult or negative. A microphone was positioned next to the participant to avoid conflict when the system produced sound. Using methods described by Clarke (2001) the views were transcribed and cross-referenced with feedback given in the questionnaire to highlight areas where (1) verbal and questionnaire feedback overlap, (2) verbal feedback was not made but participants recorded an issue in the questionnaire and (3) verbal feedback was given but not recorded as an issue in the questionnaire. Participants were asked to vocalise only criticism during the study in an effort to minimise the cognitive load used during the well-known "Think aloud" technique. This allowed participants to focus on creativity and the task, rather than trying to justify their actions.

The exploration phase of the experiment was similarly monitored. Key metrics were recorded by observation to provide some insight into how the system is being used. These metrics included: the largest rule size for each performance; the number of colours of blobs used during each performance the number of rules used in each performance; the time taken to prepare each performance and the time taken to carry out the performance.

The experiment was conducted in controlled conditions using a Nexus One with Android 2.3.4; a MacBook Pro for sound synthesis; a whiteboard with coloured markers and a microphone for recording participant comments.

## 2.4 Tasks

The tasks given to the users were intended to provide exposure to all aspects of the system. The main aspects of the language are the different types of tile available:

- **Synth** – referring to the synth track to be manipulated
- **Triggers** – conditional statements which have to be true for the rule to execute
- **Value** – a value which can be applied to a property
- **Properties** – a property of the synth track a value can be applied to

These tile types are referred to more formally in technical descriptions of the system. The formal terms have not been revealed as they could potentially confuse or intimidate the users. This intuition was confirmed when running the pilot experiment and discussing the terms with the participant.

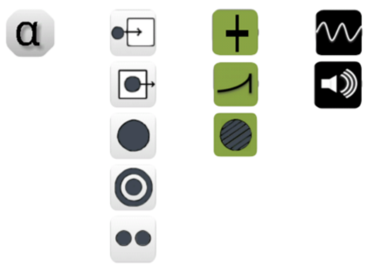

*Figure 3: Synth, Triggers, Value and Property Tile Categories*

To ensure full exposure to the system, each category of tile was tested in isolation using individual tasks. We used a consistent template where participants were required to work only with the tile of interest. The participants were given a step-by-step instruction sheet and were able to ask questions if they had difficulties resulting in non-completion of the task. This verbal prompting was noted and will be discussed during evaluation of the results. As well as thoroughly testing features, this methodology allows the users to learn, ensuring that as complexity of tasks increases, they are not overwhelmed.

## 3. Data Analysis

Due to the nature of the experiment there is limited statistical analysis we can do given qualitative feedback from the questionnaire. The transcript of any negative user experiences will be used in conjunction with this to strengthen validity of the claims and any conclusions that are drawn.

We use the quantitative data from the exploratory phase of each session to make some observations of how much expression and experimentation the language encourages. For example, the mean number of rules or mean rule length might tell us how confident the participant was to build a complex program of their own. The mean time taken for performances may tell us if participants quickly get bored or find that their rules produce a synth that they were not expecting. This data may also be used to strengthen any conclusions we can make from the questionnaire and verbal feedback.

In theory, the digital music performance users will be more at ease with the notation due to their knowledge of the domain terms and familiarity with thinking about music in a technical and abstract manner. We expect the following outcomes from experimentation:

- Digital performance users should find the system easier to use.
- Previous smartphone experience is not expected to influence results.
- Programming experience is not expected to influence results.

## 4. Results

In this section, we describe quantitative results gathered during experimentation and highlight interesting qualitative feedback gathered in the questionnaire and during the experiment. The CDs questionnaire amassed feedback for thirteen cognitive dimensions. We discuss common themes from questionnaire feedback, interesting or unique results, and results from CD's that are particularly relevant to this system. Detailed results can be found on the PPIG website, ppig.org/data-repository.

| Participants | Programming Experience | Smartphone Experience | Similar System Use |
|---|---|---|---|
| P1 | Intermediate | Yes | SC |
| P2 | Intermediate | No | Max, PD, SC, Chuck |
| P3 | Intermediate | Yes | SC |
| P4 | Basic | No | SC, Max, PD |

*Table 1 – Participants Group 1 Background (SC – SuperCollider, PD – Pure Data)*

All participants from User Group 1 have experience with Supercollider (SC) and similar musical synthesis systems. These systems are not particularly similar to our system in the way they work, but use similar terms and can be used for live coding.

| Participants | Programming Experience | Smartphone Experience | Similar System Use |
|---|---|---|---|
| P5 | None | Yes | None |
| P6 | Basic | No | None |
| P7 | None | No | None |
| P8 | Basic | Yes | None |

*Table 2 – Participants Group 2 Background*

Participants in Group 2 have little to no programming experience. There is an unplanned 50/50 balance of participants' smartphone experience across both groups.

### 4.1 Group 1 (Digital Music Performance – P1, P2, P3, P4)

**Visibility -** P3 and P4 described the pitch and time slider bars as difficult to change precisely in the Time Value settings dialog (see Figure 11), due to the limited space for the sliders. Task 3 requires the participant to change the pitch to a specific midi note; P3 and P4 noted difficulty in pinpointing that exact midi note. P2 described the screen as "too small" during the tasks and in the questionnaire.

P4 noted that finding the settings dialog and changing the blob colour might be difficult for a first time user. P4 also noted that the palette was useful in finding tiles.

**Viscosity -** P4 noted "Blobs often get moved around unwillingly. Double tapping sometimes doesn't work and note value is hard to see / define". P2 described a difficult change as moving tiles as "they don't fall accurately". The difficulty in getting precise pitch/time was again mentioned by P3 and P4.

**Diffuseness -** P4 took issue with the need to use two tiles to make a sound instead of one tile that would set the pitch *and* the volume. The other candidates appear to be aware of the things that take more space to describe, such as adding more effects or lots of rules.

**Progressive Evaluation -** Two candidates found checking work easy by going back to the main menu and clicking "Perform". P2 found this "not so easy; you need to change 'modes' which is a little cumbersome." P4 thought checking progress would require taking notes. All participants agree that they can try out partially-completed versions of the product.

**Provisionality -** The participants appear to assume "sketch" is the same as writing notation. They agree they can "sketch" rules and test them before finishing. P3 and P4 agree that when precision is not required, they can pay less attention to the note and still have a working product. P1 states that not being precise allows them to "try several versions of the desired synth on the grid."

**Premature Commitment -** P1 and P3 preferred to work in the same order as the tasks. With regards to decisions that needed to be made in advance, P1 noted that they needed to devise the effects they wanted to produce with each colour/blob. P4 said interestingly that decisions can be changed later but some unexpected results may occur.

**Error Proneness -** P1 found that they tried to delete tiles by dragging them to the palette. This does not delete the tiles, just places them on the grid space under the palette. P2 accidentally pressed buttons around the phone while programming, such as the volume button and the home button. When asked about small irritating slips, P1 stated that they accidentally put blocks of different rules side by side. P2 admitted not feeling competent in handling a smartphone in general.

**Others -** P1 would like to have more actions available. P2 would want a "more reliable general interface". P4 stated that the value tiles could be improved, referring to the issue of occlusion.

## 4.2. Group 2 (Non Digital Musical Performance – P5, P6, P7, P8)

**Visibility -** P5 stated "double clicking is fairly standard but holding down would be nicer." Interestingly, P6 stated that at first understanding the icons/concepts/changes were difficult but improved later, presumably as they progressed through the tasks. On comparing or combining different parts, P5 stated that it is difficult to see exact frequencies at the same time.

**Viscosity -** P7 and P8 correctly identified what notation takes up more space (many rules or a rule with many tiles). P5 wrongly asserted that all rules have a fixed length. P6 referred to the learning curve, saying "at first more difficult to see what changing rules actually means". P7 was unsure how the frequency and time changes affected output. P5 and P8 found no changes particularly difficult.

**Diffuseness -** P7 stated that the notation was long winded and "needs a tile for everything". P4 stated that it not always easy as the screen only has a finite area. P7 and P8 understood that several triggers and things that required many subparts took up more notation space.

**Hard Mental Operations -** P5 found it mentally challenging to "figure out what the tiles do initially". P6 found it challenging to "comprehend what a change meant". P7 found understanding the meaning of the tiles challenging. P8 found "associating a piece of music in one's head with the order of the triggers" challenging. P5 found the tile/frequency block complex the first time they used it.

**Closeness of Mapping -** P6 found the two concepts were "quite different – the sound vs. the rules" and therefore notation was not closely mapped results. P1 stated that "creating a new rule just to adjust something you could adjust by moving one block" was strange.

**Role Expressiveness -** P6 found notation for a synth track, hard to interpret. P7 found the different types of tile difficult to interpret. P5 and P7 found the use of both pitch and volume tiles confusing.

**Hidden Dependencies -** Two participants stated that the dependencies were visible and two participants stated they were not visible, with P8 giving the example of pitch and volume. When creating a large description, P7 noted that if using too many tiles, they would not all fit on the screen at once. P5 noted that forgetting the exact frequency values is a problem.

**Progressive Evaluation -** Participants stated that it was easy to stop to check work so far. Participants gave conflicting responses when asked if they could try out partially completed systems. Two participants interpreted partially completed as not a whole rule and answered no (due to the constraint of having to have a valid grid). The other two candidates interpreted "partially complete" as having a subset of the final set of rules ready and answered yes. In both cases participants were correct.

**Provisionality -** P6 noted "you can make things relative rather than focus on precise numbers".

**Others -** P5 suggested a "burst of sound as well as constant". P8 would have liked more triggers.

**Quantitative Results**

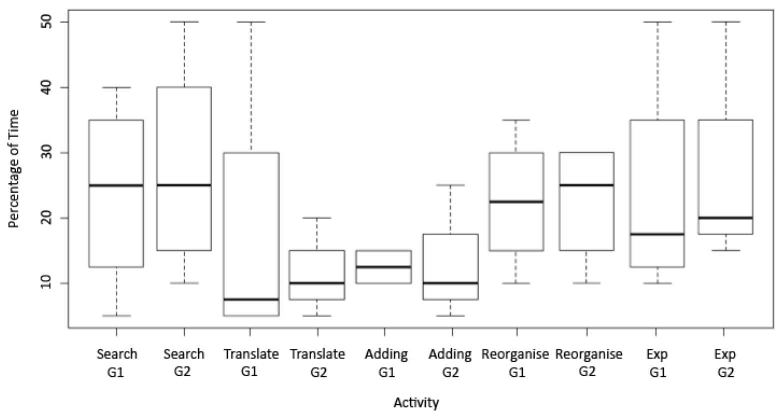

*Figure 4: Group 1 (G1) and Group 2 (G2) Reported time Allocation*

*(Exp = Exploratory Design)*

The participants were asked to give the percentage of their time they allocated to the following:

- Searching for information within the notation.
- Translating substantial amounts of information from some other source into the system.
- Adding small bits of information to a description that you have previously created.
- Reorganising and restructuring descriptions that you have previously created.
- Playing around with new ideas in the notation, without being sure what will result

We can see from results shown in Figure 4 that both groups reported time allocation in very similar proportions. The majority of time was spent searching for notation and reorganising. There is a noticeable drop in the translation time between the two groups, these results will be analysed further in the discussion.

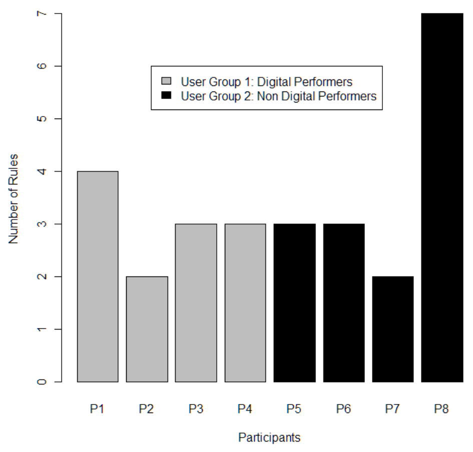

*Figure 5: Most Number of Rules*

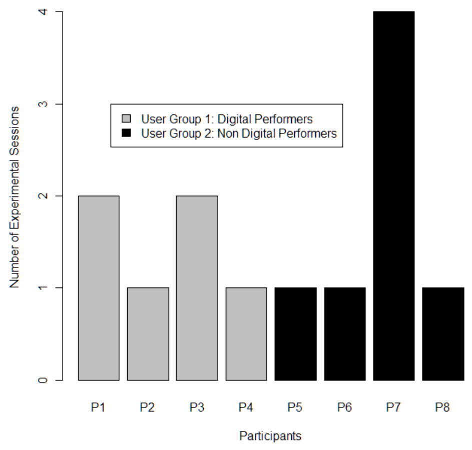

*Figure 6: Sessions Conducted*

Figure 5 shows each participant's highest number of rules during the experimental stage. P8 constructed 7 rules - considerably more than other candidates. No participant used just one rule or one blob, as in the tasks. Figure 6 shows how many sessions (training, programming, composition and performance) each candidate undertook during the experimental stage. P7 conducted 4 sessions in very quick succession, despite not having had previous smartphone experience. Most participants focused on one session, often alternating between programming and experimenting.

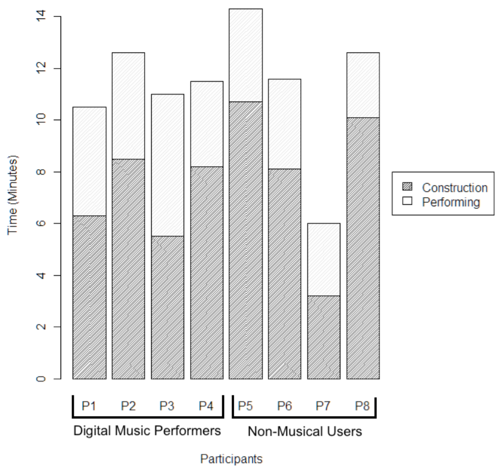

*Figure 7: Longest Time Taken for Experimentation*

The time taken to construct a program was always more than the time taken to run it. Typically the programs produced were of a fairly simple nature and therefore unsurprisingly took less time to run.

## 5. Discussion

The first point to discuss is the usefulness of the questionnaire and the participants' ability to adequately answer the questions posed to them.

Given that the questionnaire is purposely generic, there could not have been bias (from the experiment coordinator) in the way the questions were asked. The questions are intended to be simple for anyone to understand, given definitions in section 1. The majority of participants asked the experiment coordinator at least one question about a question's meaning. The questions forced participants to think of the notation in ways they had not previously; for example, the consistency questions asking which parts of notation are different but should be the same, or are the same but should be different.

All but two CDs were relevant to this system (no possible subsystems). The high number of questions caused the questionnaire to be 25 minutes long, with answers becoming shorter in the latter stages. Some participants' misunderstood questions and occasionally even avoided answering due to complexity (confirmed verbally).

We obtained interesting qualitative results from all participants, highlighting areas for improvement. The questionnaire results were mirrored during the experiment and can be considered valid. The participants' responses were consistent with verbal feedback, supporting validity of this questionnaire as an evaluation tool. Initially, we planned to mine the questionnaire for numerical data by classifying answers and comparing the groups using numerical analysis. It is apparent from variation in participants' responses that some questions may have been interpreted differently. Due to variation and some missing responses, numerical analysis would provide invalid and misleading results.

### 5.1 What does the questionnaire tell us?

P2 described the screen as being "too small". The phone has a screen size of 5x8.1cm and a resolution of 480 x 800. Although the grid size is flexible, each grid space is set to 80x80 pixels, giving 10mm per square of the grid. According to Holz and Baundisch (2011), targets of over 8.6mm have a touch accuracy of 95%. Given no other participant raised this in the questionnaire or vocally, we could attribute this frustration to P2's lack of smartphone experience. The participant may have been alluding to the finite number of possible grid spaces. Although finite, the grid can be moved by dragging with the finger. It is very large and should easily accommodate any set of rules the user would like to use. There is a possibility the candidate may not have noticed this functionality.

P5 stated "double clicking is fairly standard but holding down would be nicer". The participant is referring the settings dialog of a tile. During implementation, both double tap and long tap techniques were implemented and tested. The "long tap" technique was found to be inferior to the double tap technique due to the need for a delay of more than half a second (to avoid the user mistakenly accessing settings when they want to drag the tile). Due to the need to change settings frequently, this delay was frustrating. Using a long press for a dialog is also counter-intuitive as an object appears under the finger. A double tap is intuitively provoking a response from the object being tapped.

P2 also noted in the viscosity section, that double tapping sometimes does not work. This is likely due to mistakenly tapping in the wrong area and could be related to the size of the grid. He also mentions that the tiles "don't fall accurately". The tile drag and drop system ensures that while moving the tile, the grid space it will "land" on, if dropped, is highlighted in yellow. This space is directly under the user's finger, while the tile icon is offset to allow the user to see which tile they are dragging.

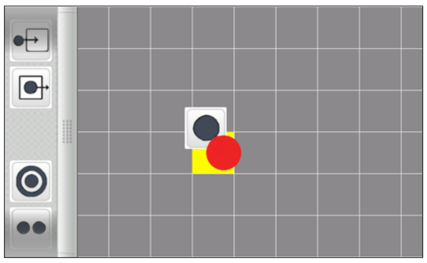

*Figure 8: Dragging a tile from the palette (red blob is the user's finger)*

P4 took issue with the need to use two property tiles instead of one tile that would represent pitch *and* the volume. P1 noted that it seemed strange to use a single value tile to change both pitch and volume. The system was designed to be extensible and allow multiple musical properties to be changed. The "value" and "time value" tiles are generic and applicable to any property. P7 thought the notation was long winded and "needs a tile for everything". The design decision to change properties independently appears to have caused confusion initially for the participants.

P4 stated that it is "not always easy [to compare or combine different notation] as the screen only has a finite area". This is referring to having a large rule that might extend off the grid. During implementation of the system, this possibility was recognised. The solution was to allow tiles to be connected in any order or any cluster, as long as they are directly connected. This may not have been obvious to users as the tasks used the same order of tiles every time for consistency.

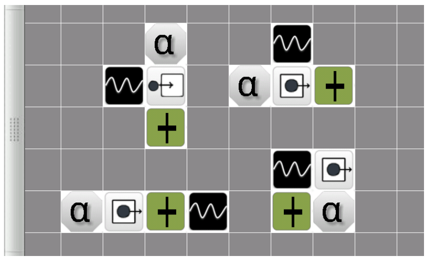

*Figure 9: Subset of possible rule combinations*

Figure 9 shows four possible combinations of tiles that make the same rule. The palette tool can be closed and opened to maximise the grid area available when it is not needed.

Questions devoted to dependency are particularly interesting as four participants appeared to think there were no dependencies and three thought there were (P2 did not answer). The system does not constrain the uniqueness of the rules. Duplicate rules depend on each other (as both would run in parallel) and any two rules referring to the same synth, using the same triggers depend on each other. This is a difficult concept to grasp and realistically cannot be learnt within a short amount of time.

Most participants found trying out test versions of their programs easy (by tapping back and clicking "Perform" on the main menu). P2 found this "not so easy; you need to change 'modes' which is a little cumbersome". During the experiment, this participant pressed several buttons by mistake, including the physical volume and sleep controls. Given lack of smartphone experience, we can conclude that he found the phone difficult to use and not just our system.

When asked about using partially built programs, some participants stated that they could not be used due to grid validation. A valid grid consists of zero or more rules which have: one or more synths, one or more triggers, one value and one or more property tiles.

P4 stated that "decisions can be changed later but some unexpected results may occur" when asked what needs to be considered before writing the notation. A trial and error approach to building a program could lead to interesting unanticipated results. We would like to facilitate experimentation, as well as making the system easy to use for people with a specific idea in mind.

*Figure 10: Current Value tile representation*

When thinking about secondary notation, most participants stated they would record what rules or tiles meant. There is currently no provision for notes or annotation within the application but detailed descriptions of the tiles could be added to the settings dialog. Some participants mentioned it was difficult to see exact values without going into the settings dialog. Currently, the tile shows a dynamic representation of the current value. This could be improved by showing specific values.

The “Time Value” tile changes the current value of a given property of a synth to a new value over the given time period. P4 noted that when he was setting the “Time Value” parameters in the settings dialog, he occluded the value with his finger.

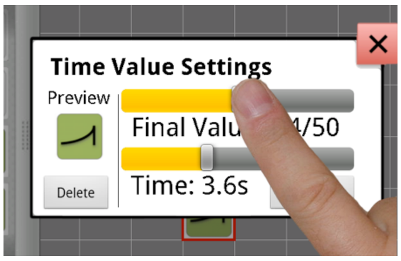

*Figure 11: Demonstration of value occlusion*

Clearly the sliders should be underneath the labels to avoid the occlusion. The user interface uses standard Android slider controls which give the user the ability to set the value to any in the range of 0-99. This value can represent volume percentage as well as midi notes. In the future, we could introduce precision + and – buttons to allow the user to increase and decrease the value.

### 5.2 Group Comparison

From Figure 4 and **Error! Reference source not found.** we can see that both groups reported allocated time in a similar way, with most being spent searching for information within notation or reorganising previously created descriptions. It is possible that this is due to the ease of using trial and error when experimenting with the system. Both sets of users spent least time adding bits of information to the description and translating information from other sources. The large variation of translation time in Group 1 is due to P2, who is marked in Figure 4 as an anomaly due to having unusual issues when using the smartphone. Time allocations were estimated by participants and could therefore be inaccurate.

## 6. Conclusions

The participants’ reaction to the system was encouraging, reacting positively to the tile metaphor and visibility of notation. Almost all participants identified areas for improvement, which is useful when considering future work. Apart from P2 (who had a particularly bad experience with the smartphone), results suggest that the reaction to the system is similar for participants with smartphone experience and those without, which was expected.

Observations of the experiments and analysis of questionnaire results show that some participants did not understand the complex ideas of the system. For example, the notion that tiles in each rule could be positioned anywhere as long as they are directly touching. Also, it is not clear whether all users understood why there was a synth tile and how it affected the audio output. The separation between the value and property tiles seems to have been initially counter intuitive to most participants, but after learning throughout the tasks and experimentation, was understood. Only one participant (P8) used multiple synth tiles to turn the volume of four synths to 100%.

Several participants mentioned a learning curve; they appeared to be comfortable after completing the tasks but may not have fully understood the system’s intricacies. This could be due to the ineffectiveness of communicating complex ideas to users in short written tutorials – several participants misunderstood concepts descripted in the tutorial. The system is very different to others, forcing the creation of a new mental model, rather than relating the system to one used in the past.

The CD questionnaire was effective as it probed the participants’ experience with the system, as well as their mental models and object relationships. The only issue with the questionnaire is the variation in interpretation of questions. Questions are compelling and clear, asking related questions to get the user thinking. The problem may not be with wording of questions, but with the length of the questionnaire and the complexity of the questions in relation to the system.

There was little difference in reaction to the system between groups, which is unexpected. We can conclude from this that both users with and without music background might find this system equally usable. There is a learning curve that users would inevitably have to undergo. All participants welcomed the ability to progressively evaluate notation and did so during experimentation. Figure 4 shows that P8 from Group 2 was the most experimental user, using 7 rules in total.

Users from Group 1 and 2 suggested that to improve the system, a wider range of actions could be introduced and the value tile could be improved. Group 2 users suggested a tile that produces a burst of sound (i.e. changes to a new frequency, then changes back over a given time period).

## 7. Future Work

The feedback given in the questionnaire offers several adjustments to the system, including the change of the "Time Value" tile settings dialog and the "Value" tile. Participants noted it was difficult to see the exact value of these tiles while manipulating other tiles. More work needs to be done to find a representation which makes the exact value obvious. Some participants tried to drag tiles back to the palette to delete them. This could be a better method of tile deletion than the current delete button.

## 8. Acknowledgements

Thank you to Alan Blackwell for his amazing support, perspective and insight while supervising this project. Thank you also to Sam Aaron for providing the audio synthesis system and for his support.

## 9. References

Apple. (n.d.). *Automator: Your Personal Automation Assistant*. From www.apple.com/macos/features/automator

Bell, B. (1991). ChemTrains: A Visual Programming Language for Building Simulations.

Blackwell, A. F. (1996). Metacognitive Theories of Visual Programming: What do we think we are doing? *IEEE Symposium on Visual Languages* , 240 - 246.

Blackwell, A. F., & Green, T. R. (2000). A Cognitive Dimensions Questionnaire Optimised for Users. *Psychology of Programming Interest Group*, (pp. 137-154).

Clarke, S. (2001). PPIG. (pp. 275-289). Bournemouth: Workshop of the Psychology of Programming.

Furnas, B. V. (1991). New Graphical Reasoning Models for Understand Graphical Interfaces. *SIGCHI on Human factors in computing systems* (pp. 71-78). New Orleans, Louisiana: ACM.

Green, T. R., & Petre, M. (1996). Usability Analysis of Visual Programming Environments: A 'Cognitive Dimensions' Framework. *Journal of Visual Languages & Computing , 7* (2), 131 - 174.

Holz, C., & Baundisch, P. (2011). Understanding Touch. *Proveedings of CHI* (pp. 2501-2510). Vancouver: ACM.

Nardi, B. A. (1993). *A Small Matter of Programming: Perspectives on End-User Computing.* Cambridge. MA: The M.I.T Press.

Repenning, A. (1993). Agentsheets: A Tool for Building Domain-Oriented Dynamic, Visual Environments. *Ph.D. dissertation.* Colorado, USA: University of Colorado.