Music Impro App (MIA) - Final report

Rune Rosseland - runebro@ifi.uio.no, Geir Snorre Berge - gsberge@ifi.uio.no

A little project history

The Music Impro App was originally devised as a virtual musical instrument which would support “play, communication and co-creation, between people with and without severe disabilities, on equal terms, in different communicative situations” (Bording et.al 2011). The core idea was to utilize the different sensors available in smartphones to play instruments in response to user movement and gestures. The intention was to make it as easy and accessible as possible for anyone to play an instrument on their smartphone, in accordance with universal design principles.

Since we first got involved in this project in early September a lot has happened on the part of the MIA project group. At first we had some trouble getting in touch with the project group because our contact had just become a father. When contact was established, the outline of the project was as follows: Music Impro App should be a music playing app with a focus on collaboration, playing and improvising music together in a group setting. The technical solution involved an iOS app running on iPod touch devices. The iPods worked as separate instruments that users were played by working the iPods’ hardware sensors like the compass and accelerometer. Gestures used were shaking, waving, hitting and tilting. These devices were syncing wirelessly and through one of the devices that were set as a master-device, the music was created.

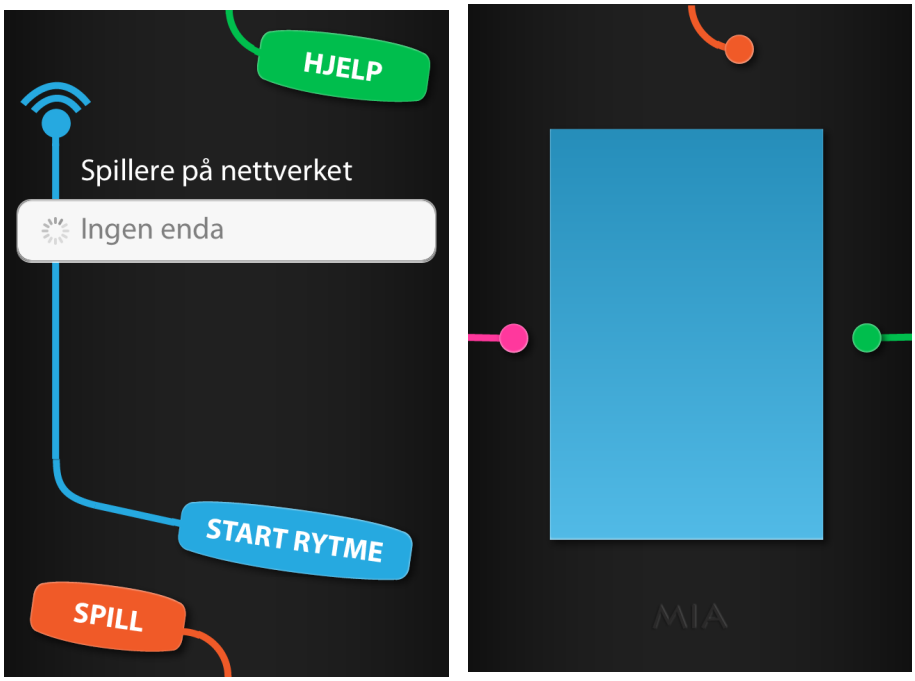


Figure 1 Early design of MIA startscreen and instrument

At that time the project had three directions for target audience and use, two of which were known to us. The first direction was using the app in conjunction with music therapy in fields of health and pedagogy, more specific as a device for stimulation, education, rehabilitation, socializing in an institutional setting. This project was geared towards research into use of the technology to produce health benefits and better quality of life for patients / users.
The other direction was a more commercial one, focusing on playing, programming and recording music through collaboration. The MIA-teams current direction for MIA has now changed from having a focus both on musical therapy and commercial use to being mainly focused in the commercial direction.

Our role at this stage was to come up with concepts for the graphical user interface (GUI) representation and sensorial control of the instruments, representing them in a visual way on mobile terminals through icons and visual feedback. We were also to look at possible future social aspects of the app.

MIA currently has three categories of instruments, with 2-3 separate instruments in each. These categories are synths, strings and percussion, and should be playable “live” on top of a beat that’s either predefined or created through a beat studio. The beat studio is intended to work as something in between a drum machine, a sampler and a sequencer. The graphical user interface is currently represented by a zoomable visual representation of a tree with nodes for functionality and instruments as leaf nodes.

Reasoning for our projects direction forward

We've spent quite some time together with the MIA-team trying to get to grips with how we can contribute to the project. It became apparent we were not able to follow the scheduled progress plan of the MIA-team for their intended December 1st release, as we simply did not have the time nor necessary oversight of their day-to-day progression. The project has been ongoing for over a year, and the participants have invested a lot of time and energy into it, and they know it intimately. For us to involve ourselves directly in the last drive towards release would be near impossible.

We came to an agreement that we should take a more loosely coupled approach to their project, and develop it in our own direction from the current state. In that way, our progress were not locked to their current release schedule and we could come up with concepts that might diverge from the MIA-teams current direction. At the same time, the MIA-team can pick and choose from what we’ve developed, if they find some of our approaches valuable to them. We agreed on the following tasks directly in connection with the current version of MIA:

- Delivering GUI concept sketches for the live musical instruments.
- Developing new interaction methods for the live playing of the musical instruments, specifically looking at how a combination of touch-screen and sensor input best can be

utilized. These are closely connected, and it will be crucial for the success of the app that GUI and interaction design is developed in relation to each other.
- Developing possible high-level interaction solutions for the beat studio part of MIA.
- Delivering GUI concept sketches for the beat studio.
- Possible future expansions. Social aspects, sharing and companion community.

It is in our opinion quite futile to start working on interface and interaction design without having a clear set of guidelines for the design. These points, and more, must be thoroughly analysed and evaluated in order to establish solid guidelines for the project before design and development should start:

- Evaluating competitors and what they are doing in terms of concepts, interaction, interface and target audience.
- Heuristic evaluation of a competing app that bears closest resemblance to MIA.
- Defining a target audience and market segment for MIA.
- Motivation - Why will they want to use the app? What about repeated use?
- Context - Social and geographic?

Competitors

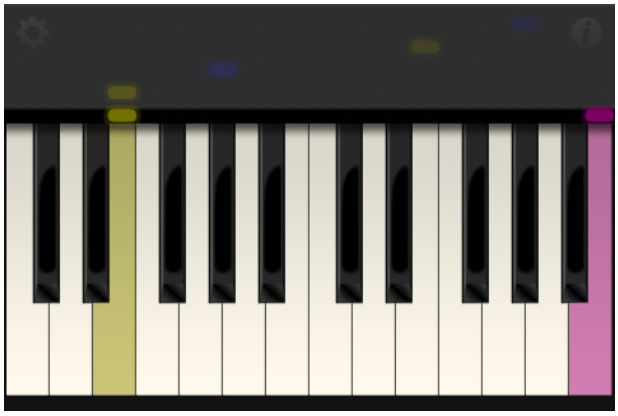
There is a large range of different smartphone apps for making music in some way or another. They range from very simple where the app effectively is an instrument, to full recording studio apps with loads of high quality instruments, audio recording options and export / integration into traditional desktop applications. In addition to straight music-making apps, there are also different apps with a more experiential approach. They focus on making the user control and interact with pre-recorded / famous music pieces, empowering the user to do something they cannot do in real life. The following is a list of smartphone apps illustrating the diversity of available music apps in Apple's App Store. We will try to position MIA within this range.

Bravo Gustavo (0 kr)
Effortlessly get the experience of an orchestra conductor by shaking the phone.

This is the only app we've found that utilizes the accelerometer to control the music. By shaking the phone like a conductor, the user activates a recording of a symphonic orchestra, playing a short 1-3 second excerpt of a famous musical piece, and repeating the motion will play successive excerpts from the same piece. In this way, the user is in control of the progression and rhythm of the performance, but not the actual notes being played. Very easy to understand, fun for a little while, but quickly becomes repetitive and boring. A possibility to buy new songs to play also exists in the app.

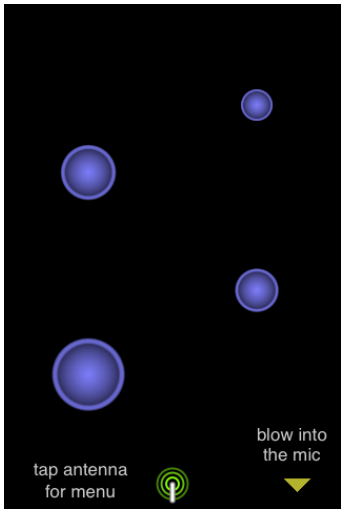
Easy Xylo (0 kr)
Simply play a xylophone on the touchscreen.

This is one of the one-app-one-instrument deals and is very straightforward. Startup presents the user with a colorful xylophone on the display, and you use your fingers to tap the keys you want to play. Very easy to get started, but requires some degree of practice or previous musical experience to play a melody. Could easily be used in a live jamming setting with other smartphone instruments. No recording function or rhythmic assistance.



Tiny Piano (0 kr)
Play a piano in free play, or play famous melodies by tapping the timing for each note.

The user can choose from a list of pre-programmed melodies to play on the piano. Tapping anywhere on screen will play the next note in line. In this way the user can control the rhythmic progression of the piece. There is no indication of how the song is normally played, and the user is therefore dependent on knowing the melody in order to play it right. This makes it possible to give the same piece a number of different interpretations and expressions based on the rhythmic playback, and can give the user a sense of musical mastery. At the end of the song the app rewards the user for the performance and is immediately ready to go again, making it very easy for the user to play 'just one more time'. A fun way to kill time.



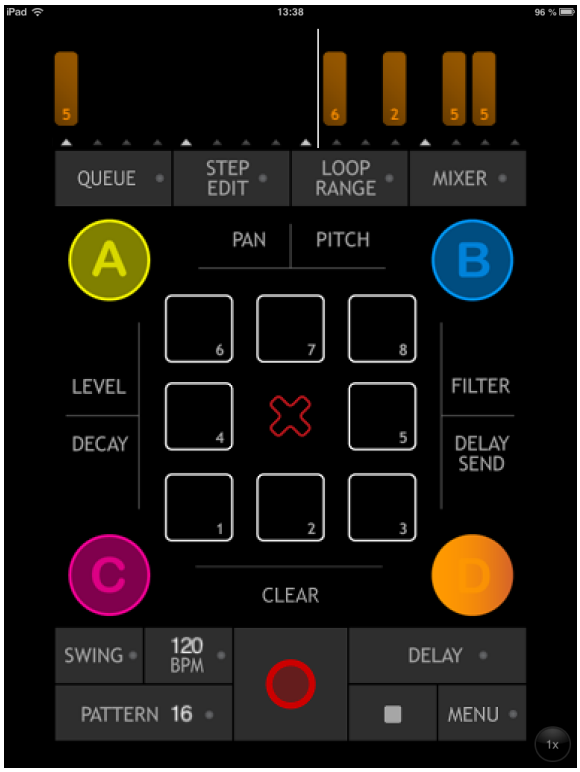
Ocarina (7 kr)
Play a flute by blowing into the microphone and pressing buttons on the touchscreen.

This is another one-app-one-instrument, but with a twist: You can record your performances and share them with other Ocarina users. A button inside the app brings up a graphical representation of the earth, with small bright dots twinkling on locations people have been when recording their songs. By tapping the dots the user can hear the recorded melody. This is a novel and interesting way of sharing.

There is no way of doing anything with the recordings, other than listen to them. In spite of, or maybe because of the lack of other options, the listener can go on a rather special journey through the world, listening to peoples solo performances on their virtual flute.

Figure (7 kr)
Loop-based music studio.

This is a three-track loop studio where the user can make a single loop of music. This might not sound very exciting, but through clever interface design and a thorough understanding of how to build interesting loop-based music they have created a very enjoyable app. The three tracks are pre-defined as 'Drums', 'Bass', and 'Lead', and each track has a list of sound-kits to choose from. The user can play one of the instruments live, record their performance, and edit and / or overdub their recordings. It is also possible to record different filters and effects on each instrument, adding expressiveness to them and personalising the sound. Even though the longest loop you can play is about 15 seconds at 120 bpm (beat per minute), it is still possible to create great variety through 'live' manipulation of the controls. The clever use of interface elements provides a very intuitive and flowing user experience making it very easy for anyone to create good-sounding loops. One of its core strengths is a thorough understanding of the strengths and limitations of smartphones as music production devices. Instead of just replicating the interaction paradigms of traditional music software, which often have just replicated the interaction paradigms of the physical hardware devices they were built to emulate, the makers of Figure have taken full advantage of the touch interface and created powerful new ways of providing expressiveness and functionality on an extremely small screen area.



Triqtraq (14 kr)
Loop-based music studio.

Triqtraq is quite similar to Figure in what it does, but is more advanced and complex. It provides a lot of different instrument kits divided into four categories: Drums, Perc, Bass, and Keys. The user can also modify the kits, replacing single samples to suit their needs, and save their own kits. All the sound kits can be assigned to one of four tracks. Similar to Figure there are different effect and filter options for adding expressiveness and flare. The interface elements are quite clever and are clearly designed specifically for the small screen area of smartphones. Triqtraqs' biggest advantage over Figure is the pattern editor / view. Where Figure allows the user to create one loop at a time, storing them separately from each other, Triqtraq offers 16 slots for storing and playing back different loops in the same session. It is also possible to arrange the different loops into a sequence forming a complete song. This makes for a very powerful app. But the advanced functionality comes at a cost: Triqtraq is providing so much advanced functionality that it makes the interface a little intimidating and confusing for inexperienced users. However, for the money, Triqtraq is a very powerful and well designed app.

Heuristic evaluation

Figure is for us the best and closest to what the MIA project aspires to be, and we intend to do a heuristic evaluation of it to gain an understanding of its strengths and weaknesses. We refer to Jakob Nielsens list of ten recommended heuristics (Nielsen):

Visibility of system status:
When starting the app the user is taken into the main window and the app is 'ready to go' at once. There's no explanation of what to do or expect, just a very pleasing, tidy and inviting interface. In spite of this, within five minutes anyone can make an ok-sounding loop just by playing with the controls to see what happens. The three tracks are color-coded for foolproof understanding of which controls control which instrument. Despite restrictive use of text and its minimalist look, the interface is able to communicate an impressive amount of information in a very unobtrusive way.

Match between system and the real world:
Limited use of text and clear user interface works well. Where used, text is as informative as it can be.

User control and freedom:
Very quick and easy navigation between all parts of the system. ‘Impossible’ to go wrong. Editing / undoing is a little tricky, but solved in a way that significantly reduces the visual representational need of the action.

Consistency and standards:
Very consistent, color-coded interface. Little need for explanation - try it and learn.

Error prevention:
Almost foolproof error prevention, but possibly a little tricky to undo recorded actions.

Recognition rather than recall:
The visual presentation reduces need for recall significantly, but some of the fine-grained controls are not represented visually at all and can only be experienced and then recalled. However, this is not a big drawback, and it contributes towards the minimalism of the interface.

Flexibility and efficiency of use:
Provides an impressive amount of fine-tuned control and flexibility for novices, but lacks ability to develop arrangement into a full song. Does not provide any other input method than the touch screen. No use of sensors.

Aesthetic and minimalist design:
Very bright, pleasing, inviting, informative, minimalist design.

Help users recognize, diagnose, and recover from errors + Help and documentation:
No error messages or documentation. We argue that the need for error messages or documentation for iPhone apps is the result of a design-flaw and should be avoided.

What will MIA be?

“*The goal in designing a mobile phone musical instrument involves: 1) Decide which input modalities to use, 2) manipulate them to be good control for synthesis, 3) pick a matching synthesis algorithm. Ideally the composer should have to spend as little energy of any other cursory requirements.*” (Essl 2008)

We will consider how this relates to our / the MIA project.

1. Input modalities

MIA was from the start conceived as a musical instrument to facilitate co-creation, and the use of sensors was central. The goal was to use the sensor data to both select, trigger and in some cases modulate the instrument being played, enabling gestural control of the instruments. We want to remain true to this goal, but we also acknowledge that the ultimate goal is to make an

intuitive, understandable, meaningful, and enjoyable interaction pattern. Thus, the use of sensors must be applied in a way the makes sense for the desired effect. Dekel et.al. has investigated three different ways of mapping accelerometer data to control musical instruments, and found that

> *subjects preferred the three axis model in which every axis is mapped to a different dimension of music generation (attack, amplitude, and pitch). This mapping was deemed better by subjects over simpler or more complicated mapping models in three of five dimensions (easier to learn, produces ''nicer'' music, and in how easy it is to understand the relationship between gestures performed and the music that is subsequently generated)* (Dekel 2008).

2. Good control for synthesis

We have tried a prototype of MIA, and our experience is that it is very difficult to map accelerometer data to trigger sounds in a predictable way. Timing is absolutely critical for musical co-creation and it is seems close to impossible to use accelerometer threshold points to achieve this. And if one considers the data fed from the accelerometer, it is much more useful for continuous modulation rather than as a binary trigger. The accelerometer is very sensitive, sampled at a very high frame rate, and it produces a continuous stream of data. In our opinion this data is put to much better use as modulation of musical expression. It can give very fine-grained and nuanced variation to the sound output, thereby allowing the user to put their personal touch to the musical output. Tanaka defines

> *expressivity of a musical instrument to mean specific musical affordances of an instrument that allow the musician performing on the instrument to artfully and reliably articulate sound output of varying nature that communicates musical intent, energy, and emotion to the listener. The success of expressivity resides not just in the effectiveness of communication, but in the sense of agency that the system gives back to the performer.* (Tanaka 2010)

Through experimenting with both the MIA prototype and numerous commercially available apps we have concluded that in order to make the interaction as engaging and predictable as possible it is necessary to play one the strengths of the different input modalities on the iPhone. Instead of trying to replicate or approximate the interaction pattern of the instrument being emulated, we want to design the interaction patterns with a clear understanding of the desired musical output. In this context, Tanakas argues for three mapping types working in conjunction to produce the "*articulation of a musical unit. The three mapping types are:*

- **a** *Binary mapping* (on/off, trigger)
- **b** *Basic parametric mapping* (selection of parameters that pre-define the nature of the sound output)
- **c** *Expressive mapping*" (continuous variation of expressive quality of the sound output) (Tanaka 2010)

We envision a mapping structure along these lines. We want to use the touchscreen as a binary trigger control and basic parametric input to pre-set controls that define the sound output. On

top of this we want to use sensors to modulate the sounds triggered on the touchscreen. In this way it is possible to produce fine-grained variation in tonality, intensity, attack, etc. while at the same time giving the user more precise control over the timing of the triggered sounds. Tanaka has also showed that this approach produces a very potent and engaging interaction experience:

> *In the instruments created by Parkinson, the accelerometer is used in conjunction with the touchscreen where buttons onscreen activate individual sounds (Binary mapping), sliders set Basic mapping parameters such as granular synthesis grain size, leaving the X and Y tilt of the accelerometer to modulate Expressive mappings of pitch and time stretching. These instruments can be played by one hand grasping the iPhone, with the thumb manipulating the touchscreen one parameter at a time, and with hand and wrist motion performing the accelerometer. This creates a compelling performance gesture which becomes all the more dynamic with the use of two such instruments, one in each hand.* (Tanaka 2010)

3. Choose a matching synthesis algorithm

At this point we have not had time to look closely at how the synthesis algorithm should be applied, but we will come back to this later.

Target audience

We argue that the main audience for MIA would be is persons who have an interest in creating and playing music, but with little or no expert knowledge in playing musical instruments or musical specialist jargon.

The target audience should also not be required to having previous knowledge of using music creation hardware or software, like Reason, Abelton, Cubase, Logic or Traktor. In this market segment there are already quite a lot of competitors in existence, who try to emulate traditional electronic music creation hardware like mixers, sequencers and drum machines.

By targeting this audience, one could exploit a relatively unused niche in the market, in between the areas of simple “soundboard” apps and semi professional tools already in existence.

How would we cater for this audience?

Playing a musical instrument takes interest, but most of all, a lot of practice. We would argue experienced musicians would also have demands of high degree of control over the instruments in the app. This would especially apply to the live instruments. This means catering to both trained musicians and novices would not necessarily be a good idea in the same user interface.

The app should have a low level of entry for starting to create music, but with enough depth that it would not only be a novelty product like your standard soundboard app. It could also of course

incorporate more advanced functionality for fine tuning the instruments and filters, pitch, etc. in the beat studio, as long as the meddling with the default presets would not be required to start creating music by a novice user.

Context and Social aspects

Smartphones has very quickly become an indispensable tool in people's everyday lives. This makes it a very useful platform for musical co-creation as it is almost always available, wherever people are. But it is also a highly embodied and personal device that is quite rarely used in social, face-to-face contexts as a means for continuous social engagement. You might show a friend something cool / fun / annoying on your smartphone, access some information regarding a discussion or engage in social games through Wi-Fi or Bluetooth, but rarely use apps for continuous engagement.

We are slightly apprehensive of the assumption that people will want to make music together in a wide variety of social settings. In our experience, mobile entertainment applications are used to kill time / boredom. They are used on the bus, at the doctors office waiting for a consultation, waiting for a friend, and so on. Or to support social activities, as mentioned above.

In addition, making music in a mobile (as in public) context puts demands on acceptable levels of ambient noise, and music generation will necessarily make sound which might be disturbing to people in our immediate surroundings. There are also a number of social conventions, which will make it awkward and embarrassing for a lot of people to use their smartphones to make music in public settings.

Requirements for GUI and interface

Demands for a mobile graphical user interface

In contrast to a traditional desktop application, the GUI in an app on a mobile device has a lot of additional possibilities but also quite a lot of limitations. Luca Chittaro mentions some of these in his article Visualizing Information on Mobile Devices:

- *"displays are very limited due to smaller size, lower resolution, fewer colors, and other factors;*
- *the width/height aspect ratio differs greatly from the usual 4:3;*
- *onboard hardware, including the CPU, memory, buses, and graphic hardware is much less powerful;*
- *input peripherals such as tiny keypads, micro-joysticks, and rollers are often inadequate for complex tasks;*
- *input techniques are different, for example, handwriting and pattern recognition on a small surface, one-hand thumb-based input, point-and-tap with stylus;*

- *connectivity is slower, affecting interactivity when a significant quantity of data is stored on remote databases;*
- *form-factor, performance, and input peripherals among different mobile device models vary greatly; and*
- *currently available tools, such as graphics libraries, tend to be low-level or limited."*

(Chittaro, 2006)

Although some of these points are a bit dated, the points about screen size, hardware power, input techniques and connectivity are still highly relevant today. These are points that a developer of mobile apps should be aware of and design for.

Chittaro also mentions the physical environment in which mobile devices are thought to be used in. For example, a mobile device will be used in a variety of light conditions, from direct sunlight to the darkness of a nightclub. This demands a GUI that has good contrast, so it can be used both in good and bad light conditions. This is especially important on OLED screens, as they do not have a backlight in the same way as traditional LCD screens have, and will have a bigger problem with glare in strong light.

Josh Clark, author of the book "Tapworthy: Designing Great iPhone Apps", have some short pointers when designing for the iPhone, but this applies also to most smartphone mobile devices with a touchscreen bought today:

- *"Place important info at the top and sink controls to the bottom.*
- *Design to a 44-pixel rhythm.*
- *Where appropriate, create at-a-glance displays that avoid scrolling.*
- *Whittle onscreen elements to the bare minimum.*
- *Push advanced tools and controls to the background."*

(Clark, 2010)

When designing for mobile devices with touch screens, the designer has to take into consideration that the surface displaying information also is the input interface for the user. This means the users fingers could obscure parts of the interface, so navigation elements should in many cases be placed at the top of the screen.

"The most important or frequently used info should float to the top of the screen above the app's primary buttons and controls. This meets our expectations not only of graphic design—headlines at the top—but also the way we hoist and handle just about any physical device. The screen bottom is the most comfortable thumb zone for a handheld gadget, but that's also where the screen is most likely to be obscured by hovering hands." (Clark, 2010)

There is also the issue of fingers as input device. Finger tips are much larger than previously used styluses and the mouse pointers of the desktop systems, so GUI elements should be "Too big to fail", making a user able to hit the interactive elements comfortably with a fingertip. A rule of thumb, so to speak, for this is designing in a "44px rhythm":

“Just how big is big enough when it comes to tap targets? Well, what's the size of a fingertip? All the platforms offer guidance here, but as usual Apple is the most opinionated, insisting on what I consider the best guideline for all mobile platforms: ***make tap targets a minimum of 44 points, or about 1/4" or 7mm. For the web, a 44px minimum also works well****.”* (Clark, 2012)

This rule of 44px is based on the first iPhone displays which had a resolution of 640x380px, but with newer displays with higher pixel density this still applies in some form, since the displays physical size has not grown dramatically and neither has your average finger. Since the devices are mobile, one should also expect that apps are used when moving. This could be while sitting on a bus, train or tram, but also while walking and running. Having to struggle with small user interface elements would then quickly cause frustration

A very valid point to be considered is social influences and limited attention:

“Occasionally a mobile user will focus his full attention on the device, such as when playing a game, but he won’t do it as often as someone sitting at a keyboard will. Therefore, design for distracted users: make the task sequences easy, quick, and reentrant. And make everything self-explanatory.” (Tidwell, 2010)

In addition to these concepts that are crucial to the success of a mobile GUI, there are also the evergreens of general GUI design to consider. Design a consistent interface in both looks and functionality. Give the user information of where they are in the application through easily understandable navigation elements. Consider readability at all times.

What we’ve done

The prototype

We’ve been able to make a prototype that implements the mapping structure described by Tanaka (2010). A combination of touchscreen and accelerometer input is used in conjunction to provide and compelling and expressive interaction.
The prototype consists of the following components and technologies:

- TouchOSC - an app designed to work as a remote control for music software on computers. It provides different buttons, sliders, and other interface elements that are useful for controlling music software. It can also access the accelerometer and pass on the data to other devices. It communicates via UDP and OSC (Open Sound Control) which is “a content format for messaging among computers, sound synthesizers, and other multimedia devices” (http://en.wikipedia.org/wiki/Open_Sound_Control, accessed 21.11.12).
- A Mac running Max/msp and Logic Pro music software.
- A max/msp-patch to reformat specific messages from TouchOSC into MIDI messages used to control software instruments in Logic Pro.

Figure 2 Touch OSC interface used for prototype

We are able to control the triggering of different notes in the software instrument from an iPhone equipped with TouchOSC. By pressing one of nine buttons onscreen, notes are triggered. Also, by moving a slider, it is possible to pre-select the pitch-range of the nine buttons. Finally, by tilting or shaking the phone along the x-axis (left-to-right in portrait-mode, top-to-bottom in landscape- mode) a wah-wah effect is applied to the given instrument. The result is that the user has very tight control of both the timing and selection of notes, as well as expressive control over the articulation of the sound. Pressing one button will produce sounds immediately, and pressing it continuously will predictably reproduce the same sound. Different shaking and tilting-motions produce varying degree of effect depending on the direction and 'position' of the phone. After exploring the prototype for a bit it is possible to produce fine-tuned expressive control over the instruments, and to predict what effects different gestures and motions will have on the sound output.

Drawbacks of prototype

The TouchOSC app provides a number of preset generic user interfaces containing different buttons and sliders. None of these interfaces allowed us to implement a swiping gesture on the touchscreen in order to simulate strumming a guitar. We therefore had to settle on an interface that allowed us to trigger sounds by pressing buttons. This works very well for a wide range of instruments, including guitar, but it does not allow us to test our chosen trigger-method for the guitar. However, Google Chrome has released a web-app, Jam With Chrome (Google / DinahMoe, 2012), that implements swiping the mouse cursor over virtual guitar-strings in the same manner that we have envisioned for our app. We have used this app in conjunction with our own GUI-designs to illustrate the intended functionality for users during user tests.

The max/msp patch we have made is relatively simple and does have some limitations with regards to triggering several sounds simultaneously. Doing so will result in notes hanging and not being turned off as intended when releasing the last finger from the screen. This can be a bit confusing, but it has also opened for interesting new ways of using the app that was not intended when constructing the code.

The Graphical User Interface

For our GUI concepts we chose to develop two live virtual instruments. We chose the guitar from the string category and piano from the synth category. We decided not to include an instrument from the percussion category.

We tested quite a few bongo, drum and drum kit apps, and their common problems was the inability to play with a fast enough pace and the very limited ability to get timing right, even on an iPad. So because of this limited playability of percussion apps in the context set, we imagine the most useful use of the percussion category would be transfer the functionality of traditional percussion instruments like drum kit to beat studio. Then one could create loops and a steady backbeat. A possibility would be to record short drum breaks for use with the beat studio, but we think playing a “drum kit” with the app in real-time collaborative setting would be near impossible. If a percussion category of instruments should exist, is our opinion in that case that those instruments should be of a simpler kind, like maracas, cowbell, tambourine etc. That way you have only one base sound, and the user would more likely be able to keep the required pace and timing.

Our goal with the GUI was to have it be as self explanatory as possible, and interesting enough to encourage exploration of the interface. It should be easily readable, both concerning text and graphical representations of instrument parts. The GUI and the over all interface should also replicate or at least approximate the interaction pattern of the emulated instrument as possible.

In our instrument concept sketches we tried to bury as much of the advanced functionality as possible, so only the bare essentials to play the instrument in a satisfying manner would remain, in accordance with Clarks’ rules of thumb. We took care to create navigation and interactive elements that were large enough for average fingers on a 4.65 inch screen with a 16:9 aspect ratio and a resolution of 1200x800 pixels. The design would be easily transferrable to other 16:9 ratio screens of similar size, and would with some minor adjustments work well on 2:3 aspect ratio screens of older iPod touches and iPhones.

We chose to design the two live instruments as a horizontal interface. The idea behind this was simply enabling operation of the instruments with two hands, and working as similar as possible as the real thing whilst still being simple. This is particularly essential for the guitar, as its inherent complexity would benefit from being operated with two hands. An example of such successful approximation of interaction patterns is the “Skate” games from Electronic Arts for among others, the PS3 and Xbox 360. In these games, the analog sticks on the game controllers are used by the players thumbs, and the movements of the sticks when performing

tricks are eerily similar to the foot movements a real skater would use to perform the trick on a skateboard.

Having an horizontal GUI would also enable using the thumbs to cover and operate on large parts of the touch screen, at the same time as the other fingers could be used to hold the device, while being on the move, while seated on the subway etc.

Visual Style

The visual style we chose for the GUI is partly based on and an expansion of the already existing GUI sketches of the MIA-team, but in their early sketches there were no style established for the instruments themselves. What struck us when we saw their mock-ups was the similarity to 1950s graphic and interior design. After doing some research we ended up on a combination of the style of early Hanna Barbera cartoons, like the “Flintstones” and “Yogi Bear”, and the early work of graphic designer Saul Bass. Saul Bass did quite a lot of movie posters like “Vertigo”, “The Man with the Golden Arm”, Alfred Hitchcocks’ “Psycho” and “West Side Story”.

The reasoning behind this choice of visual style is as mentioned to build upon what already existed, but keep readability and expressiveness high whilst at the same time keeping it simple. The common denominators of the styles we are inspired by incorporate these traits. They use clean, crisp colors, the shapes has sharp, clear edges, few and subtle gradients, textures and details and high contrasts between elements. A side effect of the strange and hard angles of the almost cardboard cutout like shapes is that it also gives a certain personality and quirkiness to the look.

Figure 3 - The guitar GUI

Playing the guitar

The idea is that the user will strum the strings with her right thumb, following the joint movement of the finger. This would hopefully be a natural movement, free of strain, and would enable both up and down strums as on a real guitar.

The left thumb would be used to select chords on the neck of the guitar, but it would be the strumming that makes the guitar sound. Swiping the thumb along the neck of the guitar would result in a “slide guitar” experience.

By tilting and shaking the device along the Y-axis, the user could modulate the notes playing to their hearts content.

There would of course be an option to reverse the interface for left handed players.

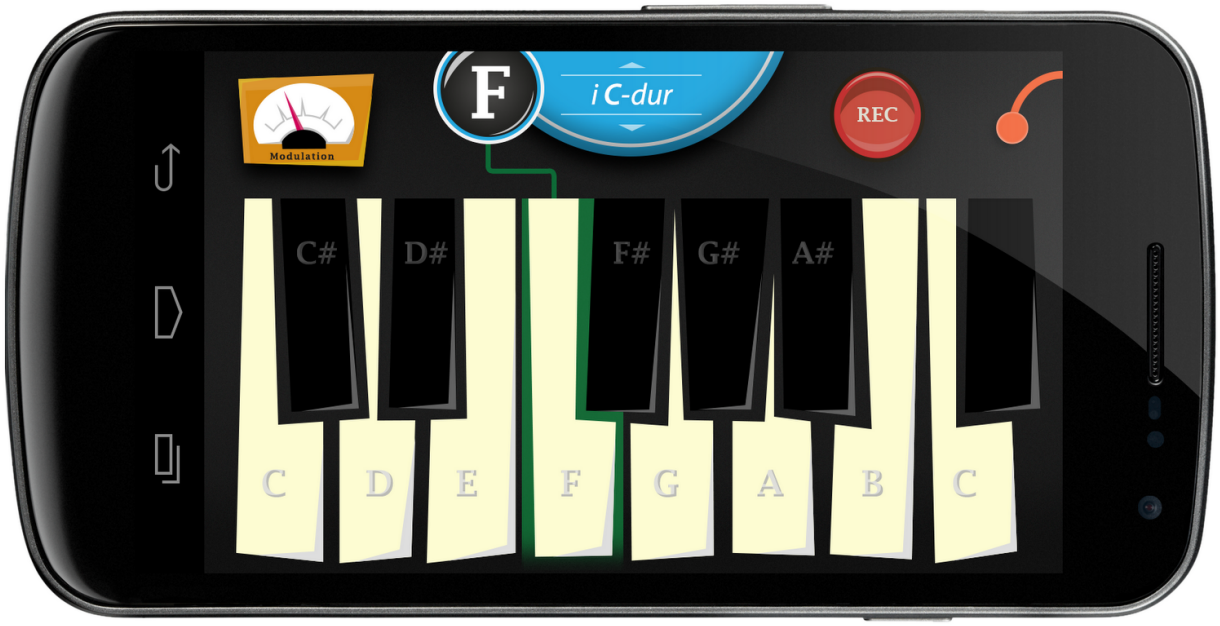


Figure 4 - The piano GUI

Playing the piano

This would pretty much work as a real piano, but with just one scale available at one time. We envision using the thumbs while holding the device as well, but also using 1-3 fingers on each hand while the mobile device is laying on a flat surface.

Interactive elements of the GUI

The “non-playable” interactive elements of the GUI have been placed in such a manner as they would not be accidentally activated while playing the instruments, and also not obscured by the fingers.

When the user plays a note, this is indicated with a green highlight on the corresponding band on the neck of the guitar and on the key of the piano. This is connected with the main interface through a line and shows note and key played.

The main interface is represented by a light blue button in the center of the screen. In addition to showing the note playing, it enables the user to swipe to change the key. If the user presses and holds the main interface, the idea is that a overlay list of avaliable keys and playing modes would appear. Selecting an new mode and key would collaps the list and be effective immediately, but a default mode and key would always be selected. This way, the instrument is playable at once, but more advanced functionality of the instument is available if desired.

In addition to the main interface, a few additional functions are present:

- A visualization of the modulation controlled by the accelerometer is represented by a generic analogue modulation meter.
- A red “REC” button, which could be utilized as a way to quickly activate recording, or as a shortcut to the envisioned “recording studio”. This could be solved a separate screen or as an semi transparent overlay over the instrument itself.
- An orange “handle”. This is the “back to main menu” / start page / overview navigational element. This is not apparent from our sketches, but is used as a pathway for navigation in the original MIA app, so its function would be more self explanatory when seen in the context of the complete app, as the main navigation of MIA is a zoomable tree view.

User test

We’ve done a ‘quick and dirty’ user test with four users to get some quick feedback on our prototype- and GUI- suggestions. Three of the participants were students at IFI while the fourth is a musician. The three students all had some previous experience with instruments, but either very limited or a long time ago. They were all interested in music and open to the idea of using the phone as an instrument The fourth is a hobby musician and has long experience playing the guitar, as well as other instruments.

We printed our GUI-sketches on paper and used these to explain to the participants how we intended the app to be used. In addition ‘Jam With Chrome’ (Google / DinahMoe, 2012) was used to illustrate how the guitar-interface is intended to work. They were then given the prototype and an explanation of how it works, and asked to play with it. After a few minutes of testing and explaining, when the participants felt that they had a good understanding of how the app was supposed to work, they were given a System Usability Scale (SUS) questionnaire. This

is a standardized questionnaire for ‘subjective assessment of usability’ (Wikipedia(2)). We ended with a short semi-open interview.

User feedback

Here are some of the main points we learned from our user tests:

Positive:

- The users liked the responsiveness of the prototype and saw the potential for fine-tuned control of the sound output.
- They also liked the GUI concept. They felt that it was easily understandable (for the most part).
- Easy to understand how to use, giving a low threshold for beginners.
- Possible for novices to learn about music, notes, chords, etc.
- The hobby-musician was able to play melodies and used the accelerometer effect actively. He expressed that the app could almost replace a real guitar.

Needs improvement:

- Some of the GUI elements were unclear. The ‘button’ for returning to the main screen is not labeled and does not indicate what it does.
- The ‘Modulation’-meter was a bit confusing to the users. This would be more obvious in the finished app as it would continually read the accelerometer and move according to the movements of the phone, but we think ‘Effect’ would be a more appropriate name.
- Need to visualize played guitar chords so users can learn how the chords are put together. This will help novices gain an understanding of harmonies and the relation between the different notes.
- Some of the users tried to swipe the finger across the screen to trigger different sounds like a pianist would run his finger up and down the keyboard without lifting it. This is not possible in the TouchOSC interface and is therefore difficult for us to implement in the prototype, but is something we think would be very useful to provide in the finished app.

The SUS-questionnaires are scored according to a specific formula, giving the questionnaire a maximum score of 100. An average SUS-score is 68, and any score above 80 is considered ‘top grade’ (Measuring Usability, 2012). Our participants’ scores were 77.5, 90, 82.5, and 95, giving an average score of 86.25, which is very good. We don’t want to put too much emphasis on these scores as we didn’t include enough participants, and the participants were all older than our intended target group (25 - 32 years old). However, the scores indicate that our concept has some merit.

Suggestions for future expansions

Social expansions

As mentioned earlier, we think it is imperative for the success of the app that it is useful and fun to use by yourself, not being dependent on others to use it. In this way people can get to know the functionality and feel proficient enough to participate in jam-sessions with others. But as people make music by themselves, they will eventually make something they're proud of and want to share with friends. We think it is important to provide a venue where people can share their creations and get feedback from friends / peers. It is also feasible to provide remix-functionality through such a service where users can remix and rebuild other users' creations. In this way it is possible to collaborate asynchronously on the development of a piece of music, or just learn and get inspired by each other. Done the right way there's a good chance of establishing a vibrant community of smartphone musicians.

Expansion of functionality through in-app purchases

With the current sketched out functionality in mind, an obvious way to be able to monetize this app if it gains traction with the general public would be to incorporate in-app purchases of advanced components or expansion packs for the basic instruments. For example, take the basic guitar: One could create purchasable effect pedal packs, additional guitars, amplifier packs and enable the users to combine effect pedals, guitars and amplifiers to create their own personal "sound". The same could be achieved for the other instrument categories, given that the basic instruments work and can be played in a intuitive and satisfying manner.

Feedback session with MIA-team

We've presented our work to the MIA-team, and discussed our ideas and prototype with them. It is clear that our two projects have taken different directions and are overlapping less than before. However, they seemed to appreciate our ideas and will take it into consideration in the further development.

References

1 Bording, J., Madsen, T., Sethre, G., Skotterud, J. O., Vestvik, K. The Music Impro App Final Report, 2011. http://www.uio.no/studier/emner/matnat/ifi/INF5261/v11/studentprojects/music-impro-app/Final%20Report%20-%20Music%20Impro%20App.pdf. Accessed 16.10.12

2 Chittaro, L. Visualizing Information on Mobile Devices. *Computer* 39, 3 (March 2006), 40-45. DOI=10.1109/MC.2006.109 http://dx.doi.org/10.1109/MC.2006.109

3 Clark, J. *Tapworthy: Designing Great iPhone Apps*. O'Reilly Media (June 2010).

4 Clark, J. February 01, 2012. http://www.netmagazine.com/features/designing-touch, accessed 16.10.12.

5 Dekel, A., Dekel, G. Mogmi: Mobile gesture music instrument, *Proceedings of the 5th International Mobile Music Workshop 2008*, p. 6-10

6 Essl, G., Wang, G., Rohs, M. Developments and challenges turning mobile phones into generic music performance platforms, in *Proceedings of the 5th International Mobile Music Workshop 2008,* p.11-14.

7 Google / DinahMoe. Jam With Chrome, http://www.jamwithchrome.com/, accessed 13.11.2012

8 Nielsen, J. 10 Heuristics for User Interface Design, http://www.useit.com/papers/heuristic/heuristic_list.html, accessed 16.10.2012.

9 Measuring Usability, http://www.measuringusability.com/sus.php, accessed 20.11.2012.

10 Tanaka, A. Mapping Out Instruments, Affordances, and Mobiles, in *Proceedings of the 2010 Conference on New Interfaces for Musical Expression (NIME 2010), Sydney, Australia.*

11 Tidwell, J. *Designing Interfaces, 2nd Edition, Patterns for Effective Interaction Design*, pages 442-443. O'Reilly Media (December 2010).

12 Wikipedia(1). Open Sound Control, http://en.wikipedia.org/wiki/Open_Sound_Control, accessed 21.11.2012.

13 Wikipedia(2). System Usability Scale, http://en.wikipedia.org/wiki/System_usability_scale, accessed 20.11.2012