ControllAR: Appropriation of Visual Feedback on Control Surfaces

Florent Berthaut, Alex Jones

ControllAR: Appropriation of Visual Feedback on Control Surfaces

Florent Berthaut
Univ. Lille, Inria, UMR 9189 - CRIStAL
florent.berthaut@univ-lille1.fr

Alex Jones
University of West England
alexander9.jones@live.uwe.ac.uk

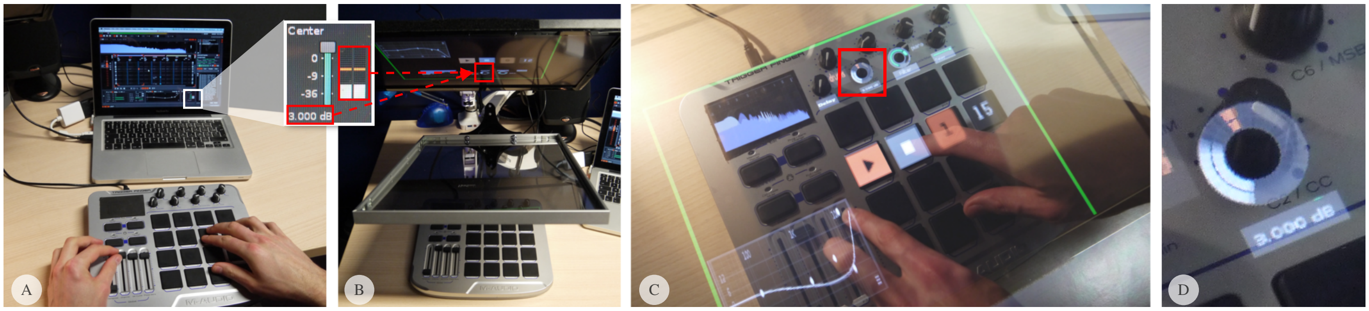

Figure 1. ControllAR enables the appropriation of visual feedback on control surfaces. Elements from the GUIs of controlled applications (A) can be cut, transformed and rearranged (B) by users. They can then be placed on/around sensors on the controller using an optical combiner (C). Close-ups show the value of the volume parameter of a track and the resulting VU meter being placed around the knob that controls it (D), illustrating both process and parameter feedback.

ABSTRACT

Despite the development of touchscreens, many expert systems for working with digital multimedia content, such as in music composition and performance, video editing or visual performance, still rely on control surfaces. This can be due to the accuracy and appropriateness of their sensors, the haptic feedback that they offer, and most importantly the way they can be adapted to the specific subset of gestures and tasks that users need to perform. On the other hand, visual feedback on controllers remains limited and/or fixed, preventing similar personalizing. In this paper, we propose ControllAR, a novel system that facilitates the appropriation of rich visual feedback on control surfaces through remixing of graphical user interfaces and augmented reality display. We then use our system to study current and potential appropriation of visual feedback in the case of digital musical instruments and derive guidelines for designers and developers.

Author Keywords

ControllAR; Augmented Reality; Visual Feedback; Appropriation; Hardware controllers; Control surfaces;

ACM Classification Keywords

H.5.2. Information Interfaces and Presentation (e.g. HCI): User Interfaces

INTRODUCTION

Following on from analog devices for the control of video and audio signals, many software applications for the creation and processing of digital media rely on control surfaces / hardware controllers composed of sensors such as groups of buttons, rotary and linear encoders. They are often preferred to touchscreens because of the variety of gestures that the sensors allow and because of the haptic feedback that they provide and to tabletops with tangibles because of their compactness. Combined with software applications with graphical user interfaces (GUI), these controllers are now used in many contexts : live electronic music performances, music recording/mixing/composition, visual performances, video editing, lights control, and so on.

Control surfaces and appropriation

Over time, users of these systems develop expert skills specific to the hardware and software components. Most importantly, they also customize the systems to match the specific subset of gestures and tasks that they need to perform. As pointed out by Findlater and McGrenere [4], "*the focus of personalized interfaces is generally to improve core task performance, that is, performance of completing known or routine tasks*". This appropriation happens at multiple levels. First the sensors themselves can be modified, either by spatially assembling/arranging existing control surfaces, as depicted in Figure 2.C, or by designing and building them from scratch using dedicated kits or generic components, as described by Greenberg et al. [6]. Gestures applied to these sensors may then vary from user to user. In the case of digital musical instruments (DMI), appropriation may lead to very different playing techniques on the same instrument as demonstrated by Gurevitch et al. [7]. In many cases, the controller and software are interconnected using standard communication protocols such as MIDI and OpenSoundControl. While presets are sometimes available for widespread controllers, users may also freely decide on the mappings between sensors and parameters. The choice of these mappings is essential for the efficiency and expressiveness of the system, as explained in the context of DMIs by Hunt et al. [10]. Mappings may change during the interaction with the system. For instance, multiple sets (or pages) of mappings can be created in a preparatory phase and selected in real time, therefore allowing the user to access a different set of functions with the same sensors. Furthermore, changing mappings may be part of the actual interaction either when the set of functions available in the application changes, e.g. when tracks are added in a video mixing software, or in the case of ”hackable” instruments [13]. These levels of appropriation relate to the control, i.e. the flow of data from the user to the system.

In any interactive system, the feedback, that is the flow of data from the system, is essential to an expert user. In particular, visual feedback plays an important role in the interaction with the system as for example demonstrated by Liu et al. [11]. However, compared to other components described above, the appropriation of visual feedback remains limited. GUIs usually include detailed representations of application content and parameters such as waveform and spectrum displays, tracks and video clips preview, dynamic labels and widgets. They however constrain visual feedback : a) their arrangement can not be modified to fit the controller or users’ actions; b) they may present more details than actually needed during actions, leading to cognitive overload; c) they are distant from the physical sensors and from the performed gestures. Therefore visual feedback on control surfaces is a growing request from users, as suggested by the apparition of embedded LEDs on most commercial controllers and of screens on the most expensive ones. However appropriation of this feedback is constrained either in the choice of the displayed feedback, the resolution of this feedback or its placement on the controller. For example, in the case of MIDI controllers, some include a limited number of programmable color LEDs (Keith McMillen Quneo, Berhinger BCR 2000) around/behind sensors, others a higher-resolution screen with fixed (Ableton Push, M-audio Trigger Finger) or even programmable content (Akai MPC Touch), but distant from the sensors.

In this paper, we propose a solution that enables the appropriation of rich and co-located visual feedback on control surfaces through GUI remixing and augmented reality displays.

GUI remixing

Research on GUIs has demonstrated the usefulness of appropriation of graphical interfaces in many contexts. By transforming visual elements, data can be visualized in new ways as demonstrated by Brosz et al. [2], Mendez et al. [14] and Tan et al. [16]. Novel GUIs can be designed by composing elements from existing interfaces as demonstrated by Stuerzlinger et al. [15] and Fujima et al. [5]. Extraction of GUI widgets can be informed or decided by the user or even done automatically through pixel based analysis [3]. In contrast, our project focuses not on the graphical interaction with remixed interfaces but on using these interfaces as dynamic visual feedback.

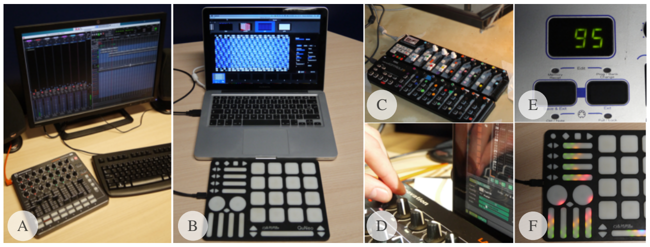

Figure 2. Example hardware controllers and associated applications: A) Digital Audio Workstation, B) Visual performance. C) Appropriation of controllers arrangement and D) of the mappings between sensors and parameters. Visual feedback available on controllers is limited in content or position to fixed screens (E) and LEDs (F).

Visual augmentations for expert interfaces

Adding visual feedback on expert interfaces has been shown to have various benefits. Liu et al. demonstrated an improvement in performance on a value selection task with hardware controllers when adding feedback through an augmented reality display [11]. Augmentation of hardware controllers also opens novel interaction techniques as demonstrated for computer keyboards by Block et al. [1].

In the specific context of DMIs, visual augmentations have been designed for various instruments such as turntables [9], removing the need for musicians to look at their laptop screen. With ControllAR we investigate how custom visual augmentations can be designed by expert users themselves.

Contribution

Our contribution is two-fold. First, we propose ControllAR, a novel approach that facilitates the appropriation of rich visual feedback on control surfaces by remixing graphical user interfaces. We describe its implementation and provide insights on key elements of the software pipeline. Second, we use ControllAR to investigate users practices during a study on digital musical instruments. Based on the results, we provide insights and guidelines to inform the design of applications and hardware that would help appropriation of visual feedback.

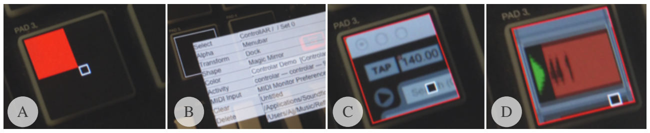

Figure 3. Appropriation process with ControllAR : A) Users draw a *zone* (overlapping a sensor or not) on the controller, and B) select one of the opened desktop windows. C) A *cut* of the selected window is then mapped inside the zone and D) can be dragged and zoomed to match any part or widget of the GUI

CONTROLLAR

Contrary to existing controllers for which the choice of content and spatial organization of visual feedback is limited, ControllAR allows users to select and place freely graphical elements on any controller. It relies on the fact that the GUIs of multimedia software applications, designed for mouse and keyboard interaction, provide rich and high resolution feedback on the manipulated content. With ControllAR, *cuts* can be made in these GUIs and placed in *zones* directly on the controller with either a projector or an optical combiner. This design process is detailed in Figure 3. However, our approach requires a series of steps to ensure that the GUI content can be transformed and displayed as visual feedback on the controllers.

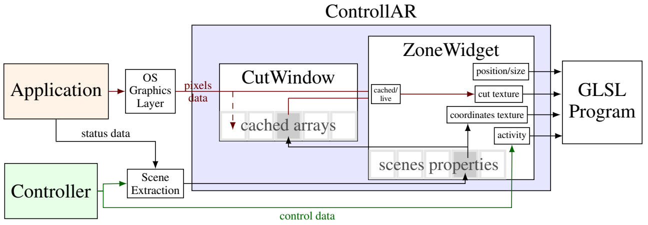

Figure 4. Visual feedback appropriation pipeline: Applications GUIs are captured in CutWindows, transformed in ZoneWidgets and rendered through a GLSL program. Control messages from the controller and data from the application are also captured to handle changing scenes and adapt to user activity.

Appropriation pipeline

In the next sections, we describe the visual appropriation pipeline, depicted in Figure 4, and the key steps that we contribute.

Capturing application GUIs

ControllAR runs on the Mac OS X, GNU/Linux and Microsoft Windows operating systems (OS). Lists of opened application windows and arrays of pixels for each of them can be accessed through the respective system graphics layers and software libraries: CoreGraphics, Xlib, GDI. These three OS feature compositing window managers. In order to allow for visual effects on the windows (such as transparency, blur, live preview ...), the window managers render all windows to offscreen buffers. It is therefore possible to access the pixels of windows even when they are not visible and combine *cuts* from multiple GUIs. This proves to be useful for software applications that rely on multiple windows (e.g. plugins, video preview) which are not always all visible at once. As shown in Figure 4, in ControllAR a dedicated CutWindow manages the pixels array of each currently accessed OS window.

Adaptation to changes in GUIs

Within each application window the arrangement and selection of visual content may change during the interaction. For example, in music production / performance software such as Renoise, a section of the GUI is often dedicated to effects and parameters of the selected audio track. Whenever the selected track changes, all the graphical elements are replaced by the ones associated with the new track. Therefore, the *cut* displayed by a *zone* might be hidden after the change. A key element of our appropriation pipeline is the detection and management of these changes in GUIs. To that extent, ControllAR allows for the creation of multiple *scenes* for each *zone*. Each *scene* can be associated to a state of the GUI and to a set of properties of the *zone*, such as the cut position and size. *Zones* can then be updated only when a specific *scene* is selected, i.e. when the correct content is visible on the GUI. When the *scene* is about to change, the current pixels array for the CutWindow is then cached as shown in Figure 4. Changes in the GUI can be detected either from the controller or from the application. When a sensor on the controller signals a change, the control data sent can be assigned in ControllAR to the selection of *scenes*. When the changes originate from the application, it is not always possible to detect them. However, some applications provide an API which can be used by plugins (for example Max For Live patches in the case of Ableton Live) to retrieve status data, as shown in Figure 4. OpenSoundControl messages can then be sent to ControllAR in order to select the corresponding *scene*. Detection could also be done through pixel-based analysis [3], although with a more variable accuracy and more setup required from users.

Adaptation to changes in mappings

Scenes can also be used to follow changes in mappings rather than changes in the GUIs. Many controllers and applications allow for multiple banks of mappings, i.e. with the same sensors associated to different parameters of the application depending on the bank. By defining *scenes* for each bank of mappings, users may display on the *zones* the correct parts of the GUI for the current bank, for example the labels of parameters for a group of knobs. Changes of banks that are internal to the controller, i.e. sensors simply send different messages/values depending on the bank, require an additional extraction step. For example, we use a Pure-Data patch to detect the apparition of specific messages from the control surface that indicate a change in bank. This patch then sends an OpenSoundControl message to ControllAR in order to set the corresponding *scene* for one or all of the *zones*.

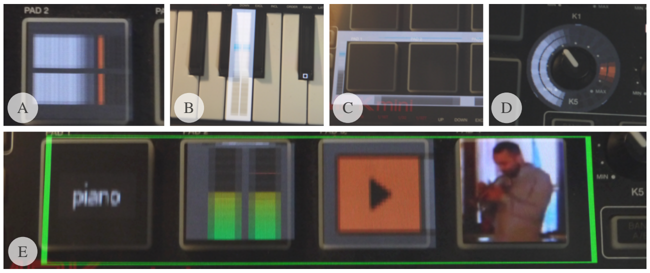

Figure 5. (A-B-C-D) Adaptation of a VU meter *cut* to four different sensors: Single pad, piano key with inverted color, groups of pads, knob. E) Different types of *cut* content on four pads: static output, dynamic output, input/output and external

Adaptation to sensors shape and color

GUIs provide a great variety of graphical elements, which can all be used as visual feedback on the control surfaces. As depicted in Figure 5.E, *cuts* can be : static (e.g. text labels) or dynamic (e.g. VU meter); input only (e.g. buttons), output only, or input and output (e.g. video clips in visual performance software); internal to the application or external (e.g. from a live camera filming a stage). On the contrary to most research on GUI remixing, in ControllAR this content needs to be transformed to match the sensors on which users want to add visual feedback. We draw inspiration from the Transmogrification project [2] and propose a set of transformations that correspond to the shapes of sensors commonly found on hardware controllers. As depicted in Figure 5, a filled square cut of a VU meter that fits a rubber pad (5.A), can be stretched and rotated to fit a piano key (5.B) . It may also be transformed to a frame that surrounds a group of sensors (5.C) or to a circle that wraps around a knob (5.D). Users choose from the sets of shapes and orientations either with a contextual menu or shortcut. They may also resize and move *zones* by dragging them so that they match the sensors once transformed, for example to make a circle shaped zone large enough to surround a knob. In order to render transformed *cuts*, a coordinates texture is first generated for the properties defined for the zone, e.g. transformation and shape, possibly specific to each scene, as depicted in Figure 4. It contains the coordinate inside the window pixels array computed for each pixel of the resulting *zone*. This coordinates image is passed as a texture to the GLSL program together with the texture containing the pixels array of the CutWindow, thus ensuring minimal computation at each frame. Colors of the *cuts* also often need to be modified in order to improve their perception when displayed over sensors with different materials and colors. To that extent, their opacity can be modified and their color inverted, for example to go from light content displayed over a black key, to dark content over a white key. These parameters are passed as uniforms to the GLSL program.

Adaptation to activity

Finally, our pipeline allows for *zones* to be updated dynamically to adapt to the activity of users. ControllAR can be connected to the controller and *zones* associated to messages from corresponding sensors. *Zones* that identify the mappings on sensors can then be hidden when these sensors are used so as to guide gestures but limit visual overload during interaction. Symmetrically, *zones* can be displayed only when the corresponding sensors are used, for example to obtain feedback regarding the exact values of controlled parameters. To that extent, *zones* can be associated with a specific message from the controller, through a learn function, to trigger the corresponding activity.

Augmenting controllers

Appropriation of visual feedback is also limited by the displays of existing control surfaces. We experimented with two hardware solutions that aim at minimizing the constraints on placement of the feedback while ensuring correct perceptual alignment of sensors and augmentations. In addition, we discuss these hardware issues in more details in the guidelines section. The first solution to place a projector above the controller and project zones directly onto its surface. The second solution is the use of an optical combiner. Drawing inspiration from [8], a half-silvered mirror is placed between a screen displaying the cuts and the controller, as shown in Figure 6. Moreover we designed it with adjustable orientations and positions for the mirror and screen so that the angles between them and between the mirror and control surface are equal. When looking through the mirror, the screen and controller surface align perfectly. As explained by Martinez et al. [12], because the optical combiner has a flat surface, this alignment remains consistent at all positions of the observer. This solution preserves the visual quality (i.e. resolution and shape) of the *cuts* and prevents occlusions. When interacting with the controller, users see both their hands and augmentations, therefore ensuring that the visual feedback remains visible at all time. To ensure compactness, in the case of performance applications, the user's laptop can itself be used as the screen above the mirror, removing the need for an additional screen.

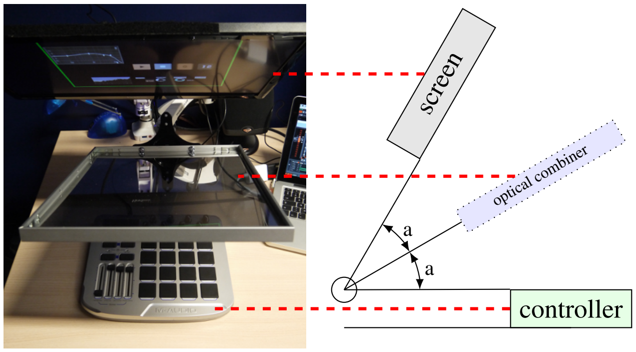

Figure 6. Front view (left) and side schematic view (right) of our hardware solution. We augment the control surfaces with the remixed GUI elements using an optical combiner and a screen. The angles (a) between screen surface and combiner and between combiner and controller surface need to be equal for the augmentation to remain consistent at all viewing positions.

ANALYSIS AND GUIDELINES

In this section, we use ControllAR to investigate the current and potential practices of appropriation of visual feedback. We first conduct a study on users of digital musical instruments, provide an analysis of visual feedback created by the participants, and derive guidelines to facilitate appropriation.

Study on digital musical instruments

This study focuses on current practices of musicians and how these could be expanded using ControllAR. While participants demonstrated and experimented with their instruments and usual performance settings, this study does not provide insights on longer term impact of facilitated appropriation of visual feedback on the way participants design or play their instruments. We conducted the study on ten musicians, aged from 21 to 45 (mean=31.8, sd=9.3). They had been playing music for between 4 and 31 years (mean=16.1, sd=8.6), a digital musical instrument for between 2 and 20 years (mean=9.7, sd=6.5) and the instrument used in the study for between 1 and 14 years (mean=5.2, sd=4.1, only two of them for one year). Seven participants used their instrument for performance only, one for composition only, and two for both composition and performance. Six participants used only one controller, three participants used an arrangement of two controllers and one of three controllers. The participants' control

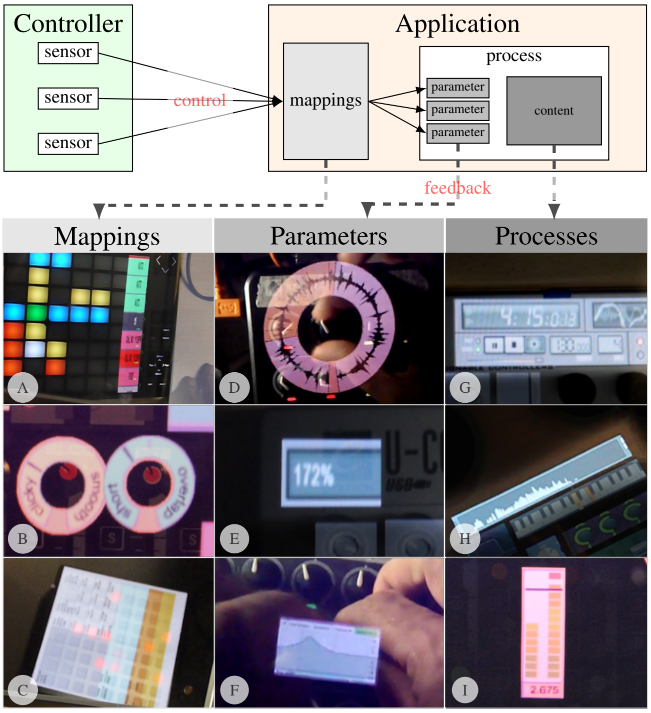

Figure 7. Examples of visual feedback created by the participants during the study grouped by category.

surfaces were composed of between 0 and 24 rotary encoders / knobs (mean=12.4, sd=7), between 0 and 17 linear potentiometers / faders (mean=4.7, sd=6.5), between 8 and 110 buttons (mean=50.6, sd=37.2) and between 0 and 64 pressure and velocity sensitive rubber pads (mean=16.8, sd=26). Other sensors included keys, one ribbon (position) sensor, one leap motion controller and one joystick. Regarding applications, six participants were using Ableton Live (including one with Max For Live plugins), two were using Max/MSP and two were using FL Studio.

The study was composed of four phases and lasted between one and two hours. In the first phase, we used a questionnaire to gather information on the participants' level of musical expertise and their use of DMIs. Then we asked the participants to demonstrate their instrument and explain their use of visual feedback with a guided interview. This phase was filmed for further analysis. We demonstrated the use of ControllAR with a simple instrument created in Pure Data and composed of eight drum loops, a four bands EQ, a delay effect and a soundfile player. Finally, we installed ControllAR on the participants' laptop, placed their controller below the mirror and filmed/interviewed them while they explored its use with their DMI. The study was run only with the optical combiner solution for augmenting the controllers. This choice was made to ensure that visual feedback would be displayed correctly regardless of the material and shape of the sensors on the participants' controllers.

Results and analysis

Participants had positive reactions to the system and saw many possibilities of appropriation of the visual feedback that could help them while playing. P4 stated that "*visual feedback is really important*" for performance/composition , and comparing with touchscreens: "*I find this more interesting than many other options*". Relating to the fact that normal practice is to get used to not having any visual feedback on parameter changes with a controller, so that introducing them aided manipulation, P3 says "*It's quite nice to be able to see what you're doing*". Participants also expressed their interest in using ControllAR to prepare their future performances. P2 says "*you would need to start projects with it, for me at least, because I don't have a fixed instrument*". P1 also commented that an interesting aim for him was to play without needing the screen and that ControllAR was a possible way to do it. P2 pointed out the possibilities of exchanging information with other musicians through ControllAR : "*the guitarist i am playing with, on his midi foot controller, he could have feedback on the various effects*". Participants also highlighted a few limitations of the current hardware solution such as: the size of the area that can be augmented, which may be too small for some large or composite controllers; the size and shape of the mirror which constrains hands and head position of the user; the fact that augmentations would not visible for an audience.

Current practices were extracted from the guided interviews and questionnaires from the first two phases, completed with the analysis of the filmed interviews and demonstrations. The potential practices were extracted from the filmed experimentation with the system in the fourth phase, by extracting and classifying both the *zones* they designed and their comments on the system. During the exploration, each participant created between one and twelve *zones* (mean=5.4, sd=3.7). The results suggest that the visual feedback both currently used by participants and explored during the study with ControllAR originates from three specific sections of the software application: the mappings layer, application parameters and application processes/content. Figure 7 illustrates these three sections with examples taken from the study and indicates their origin in the application. Our results suggest that for each of these categories, feedback can be improved by the use of ControllAR. Although this study was conducted for DMIs, we believe these categories may also apply to other applications.

Mappings feedback reveals the *mappings* section of the application, that is the set of connections between sensors and application parameters. Here feedback is mostly composed of static content such as text, icons or colors. It is used for identifying *static mappings*, either permanently or only temporarily to learn the use of a controller or to remember the mappings when switching between controllers/applications. In their usual practices, two participants used tape with written labels under the sensors to remember the mappings, with several layers when they had to change mappings. One participants had color coded stickers to identify parameters and tracks. Another participant had manuals in PDF format to remember the functions for each button of their control surface.

Feedback of the *mappings* category can also help identify *dynamic or changing mappings*. On the controller used by P9, buttons' LEDs change color to identify which track they control when scrolling through a large number of them.

With ControllAR static labels could easily be stored / removed / retrieved by participants, and placed directly over the sensors. Figure 7.B shows labels used by a participant to permanently identify the mappings around two knobs (P1) and Figure 7.C a user manual in PDF format displayed directly above a group of buttons indicating the function assigned to each (P10). For the *dynamic mappings* on P9's control surface, the LEDs did not exactly match the colors of tracks on screen and provided less information than text labels. As depicted in Figure 7.A, P9 used ControllAR to add labels with correct colors on the right side of the rows of buttons, that will be updated when scrolling through the tracks.

Parameters feedback helps during the interaction by providing visual feedback on the applications *parameters*. They can be subdivided into three types of feedback. *Contextual feedback* provides information on the context of the parameter. This information is usually only available on the application GUI, so participants had to look at their laptop screen. *Value feedback* provides the exact value to which application parameters are set, with the corresponding unit, which allows for reaching specific values. On participants' control surfaces, this feedback is given either for one sensor at a time and at one position, or individually but only for the sensors below the screen (P9). *Interaction feedback* reveals the interaction between application parameters. In their current practice, participants only had access to this feedback on the application GUI on the computer screen.

With ControllAR, *parameters feedback* can be directly integrated with the sensors, allowing users to focus on the control surface instead of having to look at their laptop screen. Examples of *contextual feedback* designed during the study include displaying the changes in linearity or continuity between sensor and parameter values by overlapping the graphical sliders of the application GUI with the physical faders, or as depicted in Figure 7.D showing the control of a range within a waveform (P1). With ControllAR participants placed *value feedback* for each parameter directly over the corresponding sensor, as depicted in Figure 7.E (P4). Finally participants used ControllAR to display *interaction feedback* over the group of corresponding sensors, for example controlling two points on an interpolated curve with two knobs as shown in Figure 7.F.

Processes feedback helps users perceive the global or local state of the application and its processes, in order to guide interaction. It complements other feedback, such as auditory perception in the case of our study. This feedback is usually only displayed on the application GUI, forcing participants to look at their screen, although one control surface allowed for displaying vu-meters when other feedback was hidden. *Local feedback* provides information on the status of more or less autonomous processes within the application, e.g. the progression in a video sequence or waveform or the status of an audio or visual clip. *Global feedback* helps perceive the general state of the application, in order to guide or inform interaction with its components. In their current practices some participants would for example look at their screen for upcoming cues on a timeline or the global pulse.

With ControllAR, *local feedback* was overlapped or placed close to the sensors that were used to control various parameters of a process. For instance, P1 chose to display the VU meter for a track as shown in Figure 7.I. For *global feedback*, participants tried or proposed displaying temporal information such as the bar and beat count used in Figure 7.G (P4), but also information specific to the application such as the global spectrum shown in Figure 7.H, or levels of audio inputs from musicians around the faders controlling their volume during live mixing.

Guidelines for facilitating appropriation

Results from our study highlight the need of appropriation of visual feedback. Although we focused on digital musical instruments, we believe that this conclusion also applies to other fields. However, while our approach enables this appropriation, it remains constrained by design choices of multimedia production applications and could be improved by following three main guidelines.

Facilitate GUI remixing: Appropriation of feedback from GUIs can be simplified if applications enable detaching parts of the interface as separate windows (as done in FL Studio) so that they are continuously rendered for compositing. Otherwise scene mechanisms such as those implemented in ControllAR must be used but cannot guarantee that all GUI elements can be captured at all times. **Expose sections of the application:** To further improve appropriation, by enabling more customized visualizations than remixed GUIs, software applications should provide easy access, via a protocol or API to all information about the three sections used for appropriation : list of mappings, parameters values and contexts, activity of processes and application. **Ensure correct augmentations:** The hardware setup used for augmenting control surfaces must ensure that gestures can be done correctly, for example with enough room for the hands and no visual occlusions, and that augmentations can cover sufficiently large areas, both for large or composite control surfaces, and when the controllers need to be moved during use, for example in music recording/mastering studio contexts.

CONCLUSION

We presented ControllAR, a novel system for the appropriation of visual feedback on control surfaces for multimedia software applications. We also provided an analysis of the current and potential appropriation possibilities leading to guidelines for design. A first perspective for this work is the study of the long-term impact of ControllAR on how musicians design and play their instrument. Another perspective is the extension to handheld / gestural controllers. In fact, many users now combine traditional sensors with gestural sensing (i.e. distance, 3D position, poses, gestures) which provide even less visual feedback on their impact on application parameters. Tracking these controllers or body parts and projecting correctly transformed *cuts* on them could lead to better integration in multimedia production systems.

REFERENCES

1. Block, F., Gellersen, H., and Villar, N. Touch-display keyboards: transforming keyboards into interactive surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM (2010), 1145–1154.

2. Brosz, J., Nacenta, M. A., Pusch, R., Carpendale, S., and Hurter, C. Transmogrification: Causal manipulation of visualizations. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, ACM (New York, NY, USA, 2013), 97–106.

3. Dixon, M., and Fogarty, J. Prefab: implementing advanced behaviors using pixel-based reverse engineering of interface structure. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM (2010), 1525–1534.

4. Findlater, L., and McGrenere, J. Beyond performance: Feature awareness in personalized interfaces. *Int. J. Hum.-Comput. Stud. 68*, 3 (Mar. 2010), 121–137.

5. Fujima, J., Lunzer, A., Hornbæk, K., and Tanaka, Y. Clip, connect, clone: combining application elements to build custom interfaces for information access. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*, ACM (2004), 175–184.

6. Greenberg, S., and Boyle, M. Customizable physical interfaces for interacting with conventional applications. In *Proceedings of the 15th annual ACM symposium on User interface software and technology*, ACM (2002), 31–40.

7. Gurevich, M., Stapleton, P., and Marquez-Borbon, A. Style and constraint in electronic musical instruments. In *International Conference on New Interfaces for Musical Expression* (2010), 106–111.

8. Hachet, M., Bossavit, B., Cohé, A., and de la Rivière, J.-B. Toucheo: Multitouch and stereo combined in a seamless workspace. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, ACM (New York, NY, USA, 2011), 587–592.

9. Heller, F., and Borchers, J. Diskplay: In-track navigation on turntables. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, ACM (New York, NY, USA, 2012), 1829–1832.

10. Hunt, A., and Kirk, R. Mapping strategies for musical performance. *Trends in Gestural Control of Music* (2000), 231–258.

11. Liu, C., Huot, S., Diehl, J., Mackay, W., and Beaudouin-Lafon, M. Evaluating the benefits of real-time feedback in mobile augmented reality with hand-held devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, ACM (New York, NY, USA, 2012), 2973–2976.

12. Martinez Plasencia, D., Berthaut, F., Karnik, A., and Subramanian, S. Through the combining glass. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, ACM (New York, NY, USA, 2014), 341–350.

13. McPherson, A. P., and Zappi, V. Exposing the scaffolding of digital instruments with hardware-software feedback loops. In *International Conference on New Interfaces for Musical Expression* (Baton Rouge, United States, 2015).

14. Méndez, G. G., Nacenta, M. A., and Vandenheste, S. ivolver: Interactive visual language for visualization extraction and reconstruction. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, ACM (New York, NY, USA, 2016), 4073–4085.

15. Stuerzlinger, W., Chapuis, O., Phillips, D., and Roussel, N. User interface façades: towards fully adaptable user interfaces. In *Proceedings of the 19th annual ACM symposium on User interface software and technology*, ACM (2006), 309–318.

16. Tan, D. S., Meyers, B., and Czerwinski, M. Wincuts: Manipulating arbitrary window regions for more effective use of screen space. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '04, ACM (New York, NY, USA, 2004), 1525–1528.