Nearly-integral Manipulation of Rotary Widgets

Rodrigo Almeida and Pierre Cubaud

Laboratoire CEDRIC, Conservatoire National des Arts et Métiers
292, Rue Saint-Martin, 75003 Paris, France
{rodrigo.almeida,cubaud}@cnam.fr

Abstract. We present a work in progress that investigates the manipulation of virtual rotary knobs using a device with three degrees-of-freedom. We draw a brief parallel between handling real knobs in professional sound appliances and interacting with desktop rotary widgets. Then, we present an interaction technique aimed to support a natural mapping, to reduce the activation time, and to enhance the fluidity along the gestures that compose this activity.

Key words: Interaction Techniques, Rotary Widgets, 3DOF Devices

1 Introduction

We started this study observing the framework of activity of a sound engineer, trying to understand the specificities of rapidly controlling many rotary knobs and linear sliders. This activity is marked by concurrency and strong time constraints: when one mixes sound live, controlling two knobs at the same time, or works in audio post-production, fine-tuning multiple tracks, milliseconds matter. A mixing table has a large number of sound parameters, but for each one of them there is an associated physical control. Our goal is to make from this particular and critical scenario a more general case of study: the interaction with 'rotary widgets'. We see today a whole class of software aimed to support varied audio activities. Many of them present graphic interfaces whose elements mimic the controls of their physical analogue (Fig.1). However, such transposition raises a critical point: the impoverishing of the gestural dimension. The numerous dials and sliders of the professional sound appliances become graphic controls, but their manipulation depends on a single physical device - the mouse. The interest in this loss of expressiveness is not new[6]; between the mixing table and the mouse, there is a continuum with control surfaces and other sophisticated devices. Such solutions may however be expensive, space-consuming, application specific, and, sometimes, unable to space-multiplex the totality of parameters of the target application. The interaction technique presented in this paper tries to improve the control of multiple rotary widgets in a desktop perspective. Therefore, our point of departure is a mouse-like device that senses its orientation. We attack the problem from three different angles: reducing the activation time to manipulate each widget, providing a more coherent mapping between the device's movement and the visual feedback, and adding more fluidity to the transition of gestures that compose the overall activity.

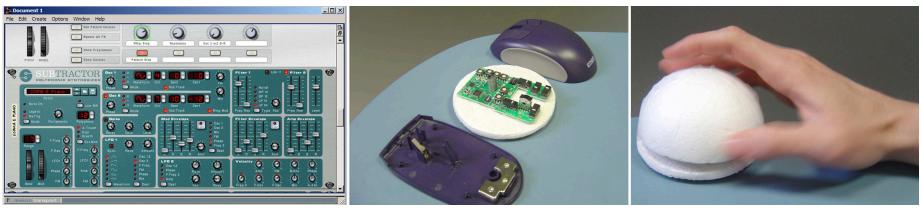

Fig. 1. Left: Screenshot of part of the interface of *Propellerhead's Reason*. Center: The *Wacom 4D Mouse* opened. Right: The 3DOF device with a hemispheric shape.

2 Rotary Widget Interaction Analysis

The handling of rotary widgets in sound software is often addressed through clicking on one of them and moving the mouse. We will analyze this interaction as a fourfold problem: phase one, one moves the cursor to pick a knob; phase two, one activates this knob, e.g., pushing mouse button down; phase three, one assigns a new value to this knob; and phase four, the knob is deactivated, e.g., mouse button up. The object of phase one is extensively studied in target acquisition research and it will not be treated here. The phase two is pragmatically decisive, for it binds the movement made to get to the widget to the one made to rotate it. In phase three, there are two common mappings in order to rotate the widget: moving the mouse vertically or making it circularly. The first relies upon the metaphor of a vertical slider, but the required movement is incompatible with the widget's visual affordance; the second requires an awkward and unusual movement. Finally, in phase four, one deactivates the knob by releasing the mouse button; at this point, the cursor may be quite far from the adjusted knob and the user must drive it to the following one. The scenario described relies upon temporal activation; our system explores instead spatial activation[2]. In such configuration, since the widget is activated and deactivated when the cursor enters and leaves its area, phases two and four need no longer explicit actions. Nonetheless, in order to make it work, the cursor must rest over the widget during its manipulation. This would be impossible using the above techniques because the same transducers that are used to drive the cursor are also used to rotate the widget. A third transducer, enabling the mouse to sense its yaw angle, frees the position transducers and also supports a natural mapping between the user's manipulation and the widget rotation. In an integral task, the manipulation is improved through the parallel control of the multiple DOFs[4]; despite selection and rotation being sequential operations, we believe that the transition from the first to the second may become more fluid if the same gesture driving the cursor to the widget starts rotating it - in a nearly-integral movement.

3 Prototyping the Spatial Activation

The orientation along the vertical axis of a mouse-like device can be sensed through different techniques: two position trackers, magnetic tracker, accelerom-

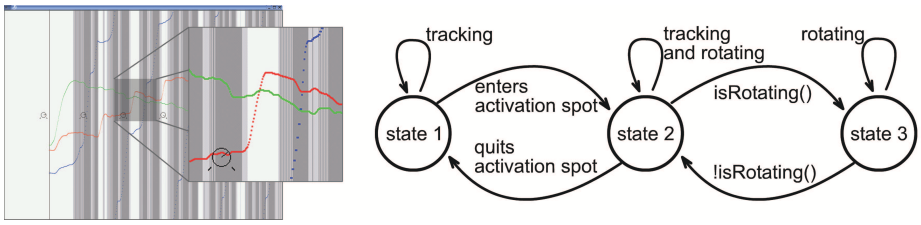

Fig. 2. Left: Screenshot of wire frame knobs interface together with input data visualization. Right: A three-state diagram of our system.

eter, and computer vision[1]. We used the *Wacom 4D mouse*, a cordless device, part of the *Wacom Intuos 4* tablet set, that senses absolute orientation[1]. Its main drawbacks are its dimensions (12.5cm by 7cm) and its asymmetric shape, which constrains the ways in which one can grasp it. Its output data have however three advantages over other prototypes': steadiness, responsiveness, and precision (0.2 degree). We took the mouse's board out of its house and dressed it under a ten centimeters polystyrene hemisphere (Fig.1), making a symmetric smaller shape. We developed a simple six rotary knob graphic interface using *OpenGL*, *XInput*, and the *Linux Wacom* driver. Testing the technique indicated us that it would be difficult to conserve the cursor over the selected knob while rotating the mouse; when one rotates a movable device, it may swing and abandon its initial position. Such oscillations move the cursor, which accidentally leaves the widget's activation spot. Our approach is to detect the intentional rotation and, during it, lock the cursor on the center of the spot. Such detection must, however, take into account the following possibilities: **(1)** the cursor go through a widget, which is not the final target, and, despite its apparently linear trajectory, orientation data vary; **(2)** the user is still rotating a widget when he or she starts driving the cursor to another widget; **(3)** the most recent mouse data packet may not correctly convey the trend of the manipulation.

A three-state model (Fig.2) illustrates how we managed the above cases[3]. In *state 1*, rotating the device has no effect on the system; *state 2* is ambiguous and moving the device rotates the widget but also tracks the cursor; in *state 3*, rotation is clearly intentional, it rotates the widget while the cursor is kept locked on the center of the activated widget. There are three strong conditions to get into and to persist in *state 3*: the variations of the two position data must be below a certain threshold and the variation of the orientation data must be both continuously growing and above another threshold. *State 2* solves **(1)** and **(2)**: the cursor will not be blocked while going through a widget, neither will the concurrent movements of rotating and of leaving the widget be hindered. As to conveying a trend of the movement, we applied the filter used by Oshita in our data[5]. We set the filter's constant to a high value, e.g., 0.8, in order to strongly bias the output values by the most recent data, so that switching from *state 3* to *state 2* be enough responsive.

[1] We thank M. Beaudouin-Lafon for informing us about this device.

Implementing this three-state model and setting the system's thresholds were supported by a data visualization interface, which helped us to know during which moments position and orientation data varied simultaneously and also to understand such variations. In Fig.2, each column of pixels in the window represents a step in the event loop and it is pushed one pixel to the left at every step. The X, Y and Yaw values are represented by red, green and blue pixels whose heights represent the values, mapped to the screen's height, of the device's data at that step. When the system is in *state 2*, the background color of the column of pixels is switched from white to light gray; dark gray when it is in *state 3*.

4 Conclusion and Future Work

We identified some aspects that could be improved in rotary widget manipulation and proposed a technique based on 3DOF devices to tackle them. The interaction afforded by our exploratory interface indicates the feasibility of such approach. The activity investigated is a special case of sequential manipulation, dubbed nearly-integral, where the first operation requires DOFs that are not those required by the second. The performance of this technique should still be evaluated in comparison with the traditional model, using a smaller better-balanced device[2], and we would like it to be tested by advanced sound edition software users as well. Finally, we would like to explore other potential nearly-integral interaction scenarios, such as picking an item in a long drop-down list, in order to test the validity and the extensibility of this approach.

References

1. R. Almeida and P. Cubaud. Supporting 3d window manipulation with a yawing mouse. In *Proc. of NordiCHI'06*, pages 477–480, New York, NY, USA, 2006. ACM Press.
2. M. Beaudouin-Lafon. Instrumental interaction: an interaction model for designing post-wimp user interfaces. In *Proc. of CHI '00*, pages 446–453, New York, NY, USA, 2000. ACM Press.
3. W. Buxton. A three-state model of graphical input. In *Proc. of INTERACT '90*, pages 449–456. North-Holland, 1990.
4. R. J. K. Jacob and L. E. Sibert. The perceptual structure of multidimensional input device selection. In *Proc. of CHI '92*, pages 211–218, New York, NY, USA, 1992. ACM Press.
5. M. Oshita. Pen-to-mime: Pen-based interactive control of a human figure. *Computers & Graphics*, 29(6):931–945, 2005.
6. J. Patten, B. Recht, and H. Ishii. Audiopad: a tag-based interface for musical performance. In *Proc. of NIME '02*, pages 1–6, Singapore, Singapore, 2002. National University of Singapore.

[2] For example, the *3Style Mouse* (http://www.cylo.com.au)