Description of a Self-adaptive Architecture for Upper-limb Rehabilitation

Alexis Heloir
DFKI Saarbrücken &
LAMIH-UMR CNRS 8201
alexis.heloir@dfki.de

Sylvain Haudegond and
Yoann Lebrun
Play Research Lab -
CCI-Grand Hainaut,
Valenciennes, France

Fabrizio Nunnari
DFKI
66123 Saarbrücken University
Campus, Germany

Christophe Kolski
LAMIH-UMR CNRS 8201
Le Mont Houy, Univ.
Valenciennes, France

ABSTRACT

This paper presents a natural and intuitive user interface architecture that uses a consumer-range 3D hand capture device to interactively edit objects in 3D space. While running, the system monitors the user's behaviors and performance in order to maintain an up-to-date model of the user. This model then drives on the fly the re-arrangement and re-parameterization of a rule-based system that controls the interaction. A preliminary user study let us define the initial parameters of this self-adaptive system. We believe that the self-adaptive aspects of the architecture we propose is well suited to the problematics of rehabilitation. This system can, from the beginning, adapt to both the user's impairments and needs, then follow and adapt its interaction logic according to the user's progress. Such a system would, for instance, enable a clinician or a therapist to design tailored rehabilitation activities accounting for the patient's exact physical and physiological condition.

Keywords

Training tools for rehabilitation, Motor rehabilitation, Virtual rehabilitation, Gesture based interaction

1. INTRODUCTION

Recent advances in consumer-range interaction devices like the Kinect[1] or the Leap Motion[2] has opened the door to an unpreceded range of new user interfaces, interaction modalities and metaphors where gesture and bodily interaction are the cornerstones [8]. Taking inspiration from these trends, we propose a self adaptive architecture that has the potential to assist users in 3D authoring task. We believe that the interaction metaphors we propose and evaluate in this paper can be used for building self-adaptive rehabilitation applications that focus on the upper limbs (arms and wrist). In a preliminary experiment we evaluate the performances of fifteen subjects performing 3D authoring tasks using a hand tracker / keyboard combination and a classical mouse / keyboard combination. The experiment is aiming at understanding how better the performance could be when using a hand tracker instead of a mouse. In this paper, we present the self adaptive architecture and the protocol used for assessing its usability, efficiency and ergonomics.

[1]http://www.xbox.com/kinect (25 Feb 2014)
[2]http://www.leapmotion.com (25 Feb 2014)

2. SELF-ADAPTIVE ARCHITECTURE

Existing work proved that specific input devices and interactive user interfaces (UI) can improve the authoring and animation process[3]. On the one hand, some devices are better suited to record animation by capturing the motion or the dynamics of the user [1], on the other hand, other interaction devices are more suited to single pose edit [6]. Only few system are actually suited to both methods [5] and to our knowledge, no proposed architecture accounted for the possibility to switch seamlessly between the two modes during an editing session. Our system is innovative because not only it allows novice users to naturally edit complex animations using natural input devices, it also adapts itself to the user's performance and experience profile.

The system proposed in this paper extends a previous architecture that didn't account for adaptaptivity [2]. This system is built upon the Leap Motion device. It provides a natural animation authoring interface that tracks and record the position and orientation of the user's hands and map them to an object that is manipulated in a 3D editing space. The overall architecture of our authoring system is summarized in Figure 1. It follows a feedback-controlled loop model where the user input (hand motion) is filtered out and analyzed by a component called the Motion Analyzer. This component infers a set of mid-level motion primitives and sends them to the interaction manager: a reconfigurable rule

PervasiveHealth 2014, May 20-23, Oldenburg, Germany

DOI 10.4108/icst.pervasivehealth.2014.255246

based system in charge of triggering the right interaction mode according to its input motion primitives. The interaction manager delivers a flow of edit actions. This flow is continuously analyzed by the Status and performance assessor. All the components work at a high frequency, typically required by interactive systems.

The system should also be capable of recognizing and preventing the deterioration of the user's performance level. This could be done by enforcing good practices or by suggesting pauses. Indeed gestural interaction, when used extensively, might induce fatigue in the arm. To avoid this, the assessor maintains a vector of descriptors that can be viewed as a simplistic user model accounting for proficiency and fatigue level. When the fatigue level increases too much, the system suggests a short break to the user. Also, the Status and Performance Assessor is continuously tuning the rules of the interaction manager to improve the user's comfort and level of performance.

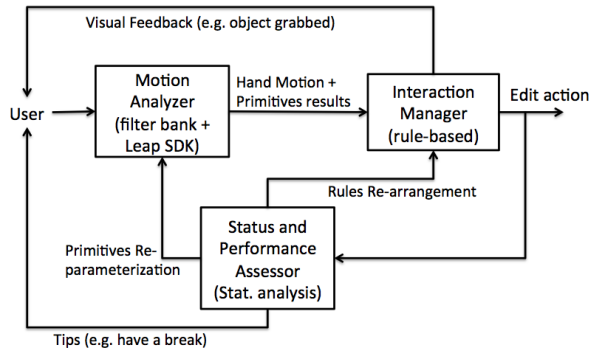

Figure 1: Proposed architecture, inspired by a feedback controller pattern

2.1 Motion Analyzer

The activity of the Motion Analyzer is based on the information stream received from the hand tracking device. The Leap Motion service provides data as a fast paced (30 Hz ca.) sequence of frames through a web socket local connection. Each frame contains, among others, information about the following elements: number of detected hands and a set of data-structure instances storing the position and the orientation of each palm and each detected fingertip. The Motion Analyzer filters out and analyzes the flow of frames streamed from the Leap. It consists of a set of primitive functions performing the analysis on a time-sliding window buffer of the Leap frames received in the last 2 seconds.

2.2 Interaction Manager

The Interaction Manager is an online reconfigurable rule-based / production system in charge of switching to the right interaction state according to the values computed by the mid-level motion primitives introduced in the previous section. The system handles three interaction states, as depicted in the right side of Fig. 1:

- HOVER: this state is active when a hand has been detected by the tracking system, but no editing action is actually carried on,
- GRAB: active when the user has selected an object and is moving it (visual hint),
- IDLE: active when no hand is visible by the tracking device.

When defined, new motion analysis primitives are bound to a selection of dynamic variables. They mostly represent time and distance thresholds. The value of these variables is updated on the fly according to how the user performs.

The first rules govern the basic state transition triggered by the presence/absence of the hand. When a new hand is detected the system switches to state HOVER, regardless of its previous state. Similarly, when a hand tracking is lost the system switches to the IDLE state.

The rest of the interaction is based on the principle that when the user wants to start an editing action he/she needs to stabilize his/her hand into the sensor action space. When the hand is stable for enough time, the selected 3D object will start following the hand. This rule requires the user hand to be somehow far from the position where an object was dropped (GRAB state exited) in order to avoid undesired re-grabbing. Stabilizing the hand again terminates the editing action.

2.3 Self-adaptivity – Status and Performance Assessor

In the previous section, we saw how the state-transition was governed by a set of rules in the Interaction Manager. The interaction Manager's logic can be modified by reparameterizing the primitive functions composing its rules and their arrangement. Such changes have an influence on the interaction dynamics and the goal of the Status and Performance Assessor is to guarantee that the current rule arrangement and parameterization maximize the user's comfort and efficiency.

We monitor aspects of the user interaction, such as his/her ability to keep his/her hand still, to control the velocity profile of the movement, as well as the reaction time to visual cues. This architecture is user-agnostic, which means that it does not build and track a model of the user; it rather tracks the short-term evolution of the user interaction and apply adaptation strategies in order to either increase the user's efficiency or to limit the decrease of performance level. Table 1 lists the variables we use to measure user performances. The values of the variables are calculated through the observation of the last $N_o = 25$ recognized actions. In the following we describe each assessment variable and its influence on the interaction manager's ruleset parameterization.

The value e is the exponential moving average of the last 25 edit actions its evolution over time gives us a hint about the user's ability to perform faster or slower edit actions. If e decreases, we assume that the user needs a more responsive system. Practically, e is used as an adjustment for a feedback gain that multiplies GRAB_START/STOP_STABILITY_TIME by $(1 + e_i)$ at each iteration.

Table 1: List of variables used to assess user performance

Name	meaning
e	action edit time
m	action linear hand movement
c	cancelled actions
l	lost hand tracking
f	fast hand movement detected

m is computed exactly the same way as e on the distance that is traveled by the hand during an action. Since an experienced user is capable of accomplishing an edit with only a few large actions (GRABS) of the hand in space, A decreasing value of m indicates the user tendency to perform longer movements, including large positioning (hand is fast) and fine positioning (hand is slow) of the manipulated object. This suggests that the user is gaining in efficiency and that the interactions rules must be accommodated.

The value of c depends on how many of the last N_o operations have been canceled. An operation is canceled when the user presses the ESC key in order to restore the 3D object in its initial position. For each operation with $1 < i < N_o$, we consider $c_i = 0$ if the operation has not been canceled and $c_i = 1$ id it did. We calculate the tendency by interpolating a line among the sampled results. The tendency c' is calculated as the tangent of the interpolated line. We consider a positive tendency as the fact that that too many operations started when the user didn't really mean to do.

The value of l is calculated, similarly to c, by counting how many times the hand tracking has been lost while performing the last N_o operations. An increasing tendency tells us that the user hand is exiting too often from the editing space. This means that the user hardly feel uncomfortable in extreme hand positions. We use this as hint that we can increase the sensitivity of the overall system, i.e., increase the ratio between the quantity of motion performed by the dragged 3D object with respect to the same quantity of motion performed by the hand in real world.

The value of f is calculated similarly to c and l, by counting how many times the GRAB state has exited because a fast hand movement has been detected. If this occurs too frequently, the system may suggest the user to have a short break so that he/she could recover from the fatigue that occurs when the subjects holds his/her arms in extension for a too long period of time. We are conducting further tests involving long edit session to correctly adjust this variable.

3. PRELIMINARY EXPERIMENT

The goal of this preliminary experiment is to assess the efficiency and usability of hand-tracking input interfaces in basic 3D authoring tasks: docking objects in 3D space. Using a mouse-based interface or a multitouch screen, users can control at most three to four degrees of freedom at the same time ([X, Y, scroll] or [X, Y, pinch, rotate]). In contrast, a 3D input device like the Leap Motion provides a direct mapping between the physical space of the user's hand and the edit space along six degrees of freedom (Rotation and Translation). In theory, users could simultaneously move and rotate objects in the 3D space, thereby perform edit tasks faster. We thus expect direct 3D manipulation to perform better than the mouse and keyboard, at least for 3D object positioning. For single target selection, Sears and Shneiderman [7] have shown that direct-touch outperforms the mouse.

3.1 Task and Experiment design

We compare the performance of a 3D positioning task across two input conditions: 1) Mouse and Keyboard (M&K) and 2) novel input system based on Hand-Tracker and Keyboard (HT&K). This comparison, however, can only be performed on subjects who already have experience with 3D software. The evaluation consisted into docking a 3D brick (translation and orientation) in a 3D environment, as illustrated in Fig. 2.

3.2 Subjects and Apparatus

We conducted the preliminary study on sixteen subjects. Subject were art and animation students in third or fourth year of a renower 3D and animation scholl called Supinfocom from the Rubika group[3]. These students can be considered as experienced 3D modeller and animators are are using software like Maya or 3D Studio Max on a daily basis. Subjects accomplished the tasks with both traditional Mouse and Keyboard (M&K) input system and with the Hand-Tracker and Keyboard (HT&K).

The study has been conducted on a PC Laptop equipped bith an intel Core i7-2630QM processor (2.0 GHz) with 8 GB of DDR3 RAM and a Nvidia Geforce GT 540M video card running windows seven and connected to a 22 inches monitor (resolution 1680x1050) at about 60 cm of distance from the eyes. An evaluator was sitting next to the subject, monitoring the advancement of the experiment, switching between tasks and the (de)activation the logging system. The 3D editor was Blender version 2.66.1. We developed a set of Python add-ons to map the leap Motion input onto 3D objects position. We are publishing on-line[4] the sources that are necessary to build and reproduce the described experiment.

[3]http://supinfocom.rubika-edu.com/
[4]http://slsi.dfki.de/software-and-resources/

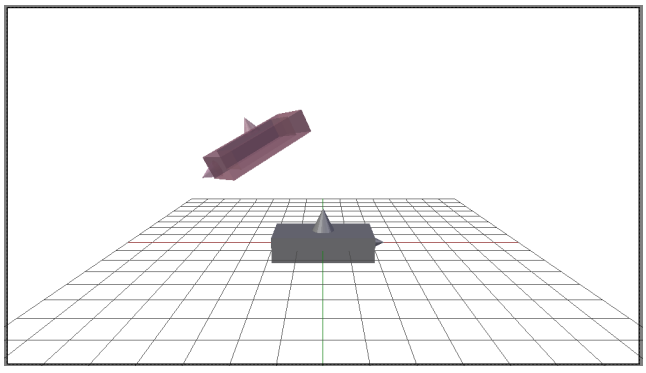

Figure 2: This is what the user views when performing the task: a target brick and a brick that needs to be docked

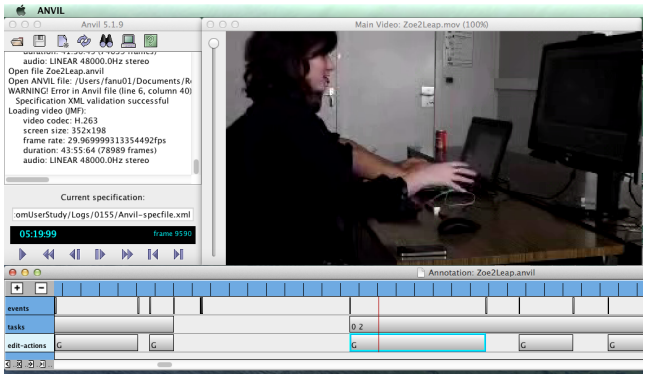

Figure 3: Screenshot of the Anvil annotation tool whith the log and annotation channels synchonized

3.3 Preliminary Results

For each trial, we recorded the time spent by the subject while manipulating the user interface (hitting one of the G,T,R or F key). We started the timer immediately after the subject touched the first edit key to begin a trail (task) and stopped the timer as soon as the task was done, fulfilled, performed. We distinguished between the time spend while moving objects in the scene (i.e. re-locating and/or rotating an object) from the time spent in switching between different editing modes.

We instrumented the Blender software to keep track of and log all the actions performed by the users. Then we processed the log, synchronized them with the video and imported them back into the Anvil annotation tool [4]. Figure 3 shows the Anvil user interface featuring the multiple tracks imported and the tracks used for manual annotations. This data is currently being analysed by fellow experts in psychology and ergonomics and enriched with annual annotations describing confort level and other high level cues. Even if we have not yet performed a systematic and quantitative analysis of the data we could observe that the performance gains obtained by the HT&K compared to M&K increase with the task complexity. We could also observe that, after 10 minutes, the HT&K started to induce fatigue, often compensated by postural changes.

3.4 Insights

We believe that self-adaptation can be useful not only among different users, but also during a long working session of a single user, by accounting for his level of fatigue. Also, the system we propose would be beneficial in the context of rehabilitation: it would indeed enable a clinician or a therapist to design tailored rehabilitation activities accounting for the patient's exact physical and physiological condition.

4. CONCLUSION AND FUTURE WORK

To sum up, we presented a self-adaptive system that has the potential to help injured patients to perform rehabilitation exercises remotely from their home. Their performance and their progress towards recovery would be assessed on the fly by the system and suitable guidance would be provided at the right time. If the system would detect that the patient would not progress, the system would call a trained clinician for assistance. The interaction metaphor we proposed in this paper is suited to the upper limbs (hands and wrists). However, we believe that the self-adaptive architecture we described would also be suited to the rehabilitation of other body parts. The experiment we ran still needs to be analysed but we hypothesize that fatigue has a significant influence on the user's performance and comfort level. To take this fatigue level into account in a relevant manner will be our next challenge.

In this system, user adaptation is enforced by tuning the rules driving the interaction according to the parameters of a basic user model that is inferred at runtime from the user's behavior. Even if the system is supposed to converge towards an optimal set of rules, adapted from the user's behavior and performance, it needs to start with a set of rule that is generic enough to fit all users. The protocol presented in this paper involved users who have experience with regard to 3D editing, as a consequence, the set of generic rules that we would infer from this user study might not fit users that are novice with regard to 3D editing. We are thus planning to conduct a similar study (in may) with users who lack experience in 3D. We finally consider conducting a study with patients going through a rehabilitation of the upper limbs program.

5. ACKNOWLEDGMENT

Authors would especially like to thank the students who participated to the study as well as Azad Lusbaronian, head of studies at Supinfocom / Rubika SAS for his involvement and the essential help he provided.

6. REFERENCES

[1] J. Chai and J. K. jessica Hodgins. Performance animation from low-dimensional control signals. In *Proc. of ACM Transactions on Graphics*, 2005.

[2] A. Heloir and F. Nunnari. Towards an intuitive sign language animation authoring environment for the deaf. In *Third International Symposium on Sign Language Translation and Avatar Technology*, Chicago, Oct. 2013.

[3] K. Kin, M. Agrawala, and T. DeRose. Determining the benefits of direct-touch, bimanual, and multifinger input on a multitouch workstation. In *Graphics Interface 2009*, GI'09, pages 119–124, 2009.

[4] M. Kipp. Anvil - a generic annotation tool for multimodal dialogue. In P. Dalsgaard, B. Lindberg, H. Benner, and Z.-H. Tan, editors, *INTERSPEECH*, pages 1367–1370. ISCA.

[5] M. Kipp and Q. Nguyen. Multitouch puppetry: Creating coordinated 3d motion for an articulated arm. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 147–156, New York, NY, USA, 2010. ACM.

[6] J. Lin, T. Igarashi, J. Mitani, M. Liao, and Y. He. A sketching interface for sitting pose design in the virtual environment. *IEEE Transactions on Visualization and Computer Graphics*, 18(11):1979–1991, Nov. 2012.

[7] A. Sears and B. Shneiderman. High precision touchscreens: design strategies and comparisons with a mouse. *International Journal of Man-Machine Studies*, 34(4):593–613, Apr. 1991.

[8] D. Wigdor. *Brave NUI world: designing natural user interfaces for touch and gesture*. Morgan Kaufmann, Burlington, Mass, 2011.