LeapArm - Facilitating Sensory Spaces for Mid-air Gestural Interaction

Ilhan Aslan, Julian Kraus, and Elisabeth André
Human-Centered Multimedia
University of Augsburg
Germany
lastname@hcm-lab.de

NordiCHI '16, October 23-27, 2016, Gothenburg, Sweden
 ISBN 978-1-4503-4763-1/16/10…$15.00
DOI: http://dx.doi.org/10.1145/2971485.2996741

Abstract
We present the LeapArm system, which combines a Leap Motion controller for mid-air gestural human-computer interaction with a robotic arm. LeapArm actuates in real-time the orientation of the controller towards a user's hand, dynamically expanding the sensory space for mid-air interaction. We describe several desktop use scenarios and rapid transitions between them, which are enabled by the LeapArm system and discuss how actuated 3D controllers change the design space for gestural interaction.

Author Keywords
Mid-air gestures; dynamic sensory space.

ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

Introduction
In the last decade, human-computer interfaces based on optical tracking systems have gained popularity, since they can transform "any" mid-air space into sensory space for human-computer interaction. The idea behind optical tracking systems is that cameras capture images of the space in front of them and are capable to recognize a user's body (or a specific body part such as a hand) located in these images. Then,

information about the body, such as posture and the shape the body creates during movement within the "digitalized" space is computed. Ultimately, the technology allows body-based mid-air interaction with computers, such as swipe and zoom hand gestures performed by users in mid-air. While there are potential benefits of mid-air interfaces for different use contexts [8] where use of traditional human-computer interfaces are inappropriate, the design of mid-air interaction is an ongoing challenge with each use context having its own peculiarities and posing different requirements for design.

In our previous work, we have extensively explored mid-air input as a modality for human-computer interaction and designed its use for contexts, such as cars [2], clean rooms in industrial settings [4], and for retail situations [3].

There are some common and systematic constraints of off-the shelf controllers, such as the Xbox Kinect or the Leap Motion. As optical systems they have specific requirements on orientation and proximity towards the in-build sensors. Thus, they require users to perform gestures in a limited and invisible sensory space, causing discomfort and unintended interactions.

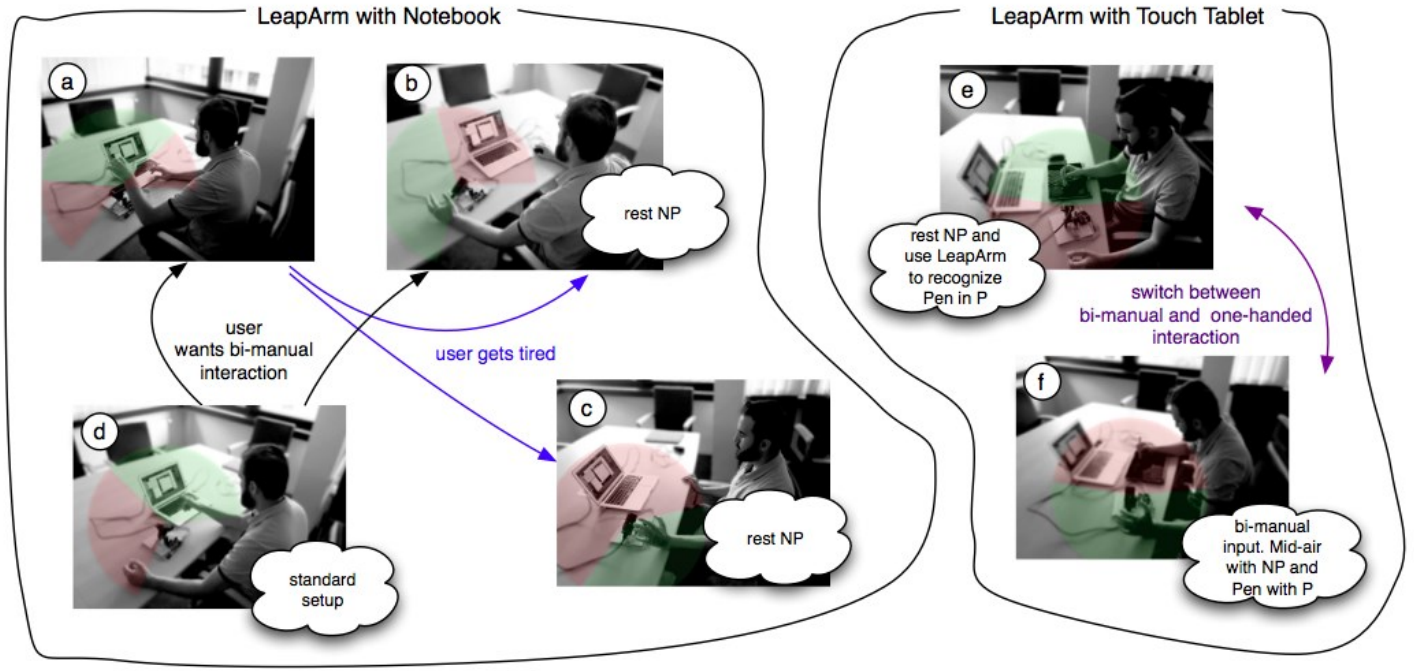

Figure 1: Divers examples of desktop use contexts supported by the LeapArm system. The approximate sensory space created by the orientation and range of the optical system is marked green. With NP we refer to the non-preferred hand and with P to the preferred hand. The user in the pictures is right handed, thus, P refers to the right hand and NP to the left hand.

The colloquial term “the gorilla arm syndrome” refers to associated issues with fatigue and discomfort when postures have to be maintained over long periods and performed in a repetitive manner (e.g., [6, 8]).

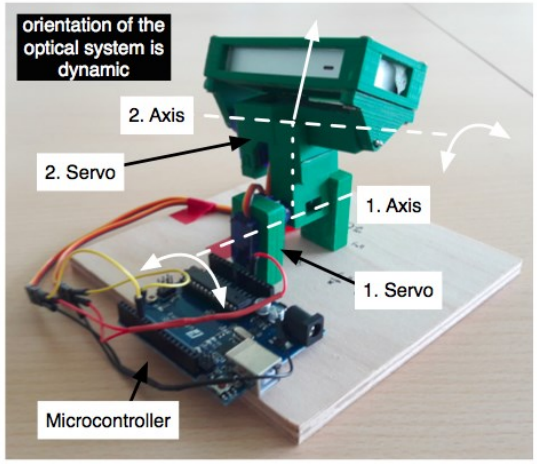

Figure 2: Details of the LeapArm prototype. The microcontroller controls the orientation of the optical system (i.e., Leap Motion) by actuating two servos, which are embedded in the "robotic arm" mount.

Related research (e.g., [5, 7]) has argued that allowing users to rest their arms (e.g., on a chair armchair) while performing hand gestures would limit the inherent effects of gestural mid-air interfaces. However, existing controllers, such as the Leap Motion are not designed for use in rest positions, which might cause bad performance in practice. Consequently Darren et al. [6] have studied the performance of mid-air gestural input with the Leap Motion controller in three different rest positions, and argue that by modeling the interaction space to fit the rest positions' characteristics improves gesture recognition performance.

Motivated by these results in related research and our own experiences with the Leap Motion controller we propose a method to dynamically extend the sensory space of the controller. We argue that doing this not only enables users to take rest positions, but also creates new use contexts (see Figure 1 for examples of desktop use contexts) and allows “seamless” transitions between them. In the following sections, we present the details of the LeapArm prototype and discuss conceptual consequences for gestural interaction design.

LeapArm Prototype

The Leap Motion controller is an exemplary (and widely used) optical tracking system. It is customized to recognize hand and finger movement of users who are typically in a seated position in front of a screen. The controller is small sized and needs to be put on the desktop beside or in front of a computer screen. The range of the Leap Motion is limited to from approximately 25 to 600 millimeters above the device and a field of view of about 150 degrees with recognition rates being worse in the borders of the sensory space than the center of the space [1]. In Figure 1 the approximate sensory space for gestural interaction is marked in green and the space for non-gestural interaction is marked in red.

Hardware and Software

Figure 2 shows hardware details of the LeapArm system. LeapArm consists of a Leap Motion controller, an Arduino microcontroller, two servo motors embedded in the robotic arm, which we modeled and

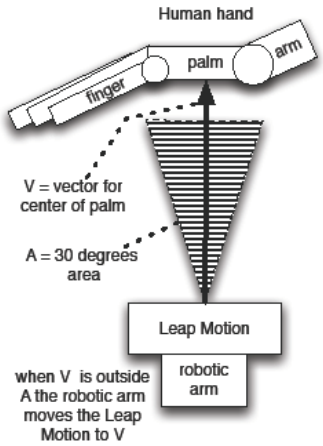

Figure 4: Overview of logic about when to move the robotic arm.

printed on a state of the art 3D printer. The robotic arm serves as a two-axis actuated mount for the controller, allowing programmable dynamic control over the orientation of the Leap Motion.

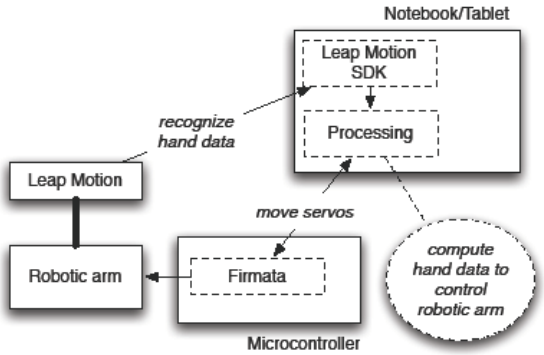

Figure 3: Overview of LeapArm system architecture

Figure 3 shows an overview of the system architecture. In our implementation we accessed the Leap Motion SDK via the Processing language.

In Processing, we use the Leap SDK to access the palm position relative to the Leap Motion and actuate the servos using the Firmata protocol. That is, we access the servos remotely from Processing without using any other code than the Firmata library on the microcontroller.

We synchronize hand data recognition with movement of the robotic arm when the vector at the center of the palm moves outside a 30 degree area (see Figure 4). We chose 30 degrees based on our own exploration and since this value still allowed complex finger gestures without having to continuously change the orientation of the controller. When V moves outside the 30 degrees area (e.g., because larger gestures are performed) the LeapArm moves its orientation towards the center of the palm.

Consequences of Actuated 3D Controllers for Gestural Interaction Design

Having provided technical details about the LeapArm system, we next discuss its consequences for gestural interaction design.

Support of divers use contexts

We have already mentioned that many new use contexts are enabled by the LeapArm system. In Figure 1 we have depicted a few exemplary desktop contexts, such as one-handed or bi-manual interaction with notebooks or touch tablets where mid-air input with the non-preferred hand (NP) is sometimes used to complement input (based on mouse, touch-pad, keyboard, touch, or pen) with the preferred hand (P). All these use contexts require the sensory spaces to be at different (absolute) positions in space. For example, when a user interacts with P via gestures "directly" with a screen (see Figure 1d) the space in front of the screen should be a sensory space and ideally the space around the rested NP should not be a sensory space in order to reduce unintended interaction.

Designing Context Transition Gestures

In our current implementation additional gestures (i.e., drag and drop) are required to move the orientation of the controller (i.e., transition from one use context to another). Figure 5 presents an example for a transition between two use contexts. Depending on the concrete application there is some potential that transition

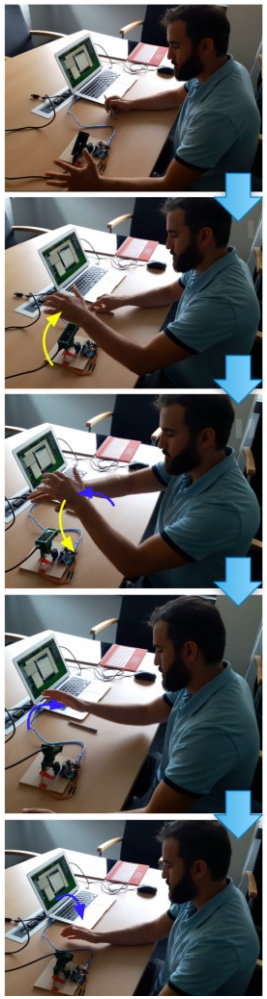

Figure 5: Example of a transition. User is in the beginning performing gestures with his left hand in a rest position. Then he moves the controller to his right hand and afterwards rests his right hand.

gestures might interfere with other large gestures. In that case alternative mechanism are required to introduce transitions, such as keyboard short cuts or complex gestures, which include hand/finger postures (e.g., pick the controller and drop it to the desired position or send the controller to a use context with a specific gesture).

Designing Gestures with Rest in Mind
The LeapArm system allows users to rest their arms while performing hand gestures. Users have some options about how they rest their arms. By designing gestures with rest in mind designers may be able to address discomfort associated with fatigue more directly. For example, gestures could be designed explicitly for rest positions, or start from rest positions, or end in rest positions.

Support for new Gesture Types
Because the sensory space is (dynamically) expanded, the LeapArm systems enables new types of gestures, which complement existing capabilities of the Leap Motion system with an additional "context" dimension (i.e., orientation of the controller). Thus, the same Leap Motion gestures can be used for different functions depending on the orientation (use context). Furthermore, large gestures (with more arm movement) that cross over multiple use contexts (such as transition gestures) become possible.

Designing for Non-Gestural Interaction
In Figure 1 we have explicitly highlighted sensory spaces and non-sensory spaces. Dynamically changing the orientation of the controller can not only be used to expand the sensory space for gestural interaction but also can expand the space for non-gestural interaction, reducing unintended interaction and allowing users to use gestures for other purposes (e.g., interpersonal non-verbal communication during video conferencing).

Collaboration
More available space for gestural interaction is desired in collaboration settings. The LeapArm system allows users to handover the controller to another user, or more precisely move the sensory space towards another user (with a gesture) who is sitting across the table. The LeapArm system could also be used to detect the orientation of multiple users and orient itself towards the (active) user. This behavior might be beneficial when the LeapArm is, for example, attached to the middle console of a car and used by both the driver and the co-driver.

Feedback
When exploring gestural interaction with the LeapArm system we experienced how the movement of the robotic arm and its continuous orientation towards the user's hand is perceived as an assuring feedback. When using the system it is clear that one is within the range of the controller. Not only for the user, but also for observers it is very visible how the system works and where the Leap arm is "looking" and which hand it is observing. Some users from our own department have argued that its behavior is very "anthropomorphic". When using the LeapArm system it feels as if it is trying to keep its gaze at your hand.

Discussion and Future Work

In our current implementation, the LeapArm system seamlessly follows a user's hand, allowing the Leap Motion controller to recognize hand, finger, and tool (e.g., pen) positions and motions. We have

implemented a finger and hand visualization application based on the Processing language to explore how the LeapArm's physical movements may influence the Leap Motion controller's performance. We have not experienced performance drops compared to a non-actuated setting; however, in our future work we aim to systematically explore the performance of the LeapArm system. One could argue that using multiple static Leap Motion controllers is an alternative to the LeapArm system with better performance. However, in our best of knowledge the Leap Motion SDK does not (yet) support multiple controllers on one PC; and for a desktop setting one would need five static controllers to span a similar space as it is possible with LeapArm.

While we have demonstrated benefits of the LeapArm system for desktop use contexts, we believe its benefits are not limited to the desktop. We imagine that the LeapArm system is also interesting for industrial settings or gaming scenarios. With regard to new use contexts, an intriguing direction of future work is the exploration of the LeapArm system when it is used in combination with a head-mounted display, such as the Occulus rift. Our concrete next step is in designing an application (i.e., a game) to explore with users in a study the performance and user experience of gestural interaction enabled by the LeapArm system.

Conclusion

We have presented the LeapArm system, which dynamically expands the sensory space for mid-air gestural interaction. We discussed both technical details of the LeapArm system and a set of conceptual consequences for interaction design that result when the optical controller is actuated. Overall, we have argued that by controlling the orientation of the Leap Motion spaces for mid-air gestural interaction and non-interaction are dynamically set, addressing some issues associated with fatigue and discomfort, and enabling new use contexts for gestural interaction.

References

[1] Adhikarla, V. K., Sodnik, J., Szolgay, P., and Jakus, G. Exploring direct 3d interaction for full horizontal parallax light field displays using leap motion controller. Sensors 15, 4 (2015), 8642-8663.

[2] Aslan, I., Krischkowsky, A., Meschtscherjakov, A., Wuchse, M., and Tscheligi, M. A leap for touch: proximity sensitive touch targets in cars. In Proc. of AUI, ACM (2015), 39-46.

[3] Aslan, I., Meneweger, T., Fuchsberger, V., and Tscheligi, M. Sharing touch interfaces: Proximity-sensitive touch targets for tablet-mediated collaboration. In Proc. of ICMI, ACM (2015), 279-286.

[4] Aslan, I., Uhl, A., Meschtscherjakov, A., and Tscheligi, M. Mid-air authentication gestures: an exploration of authentication based on palm and finger motions. In Proc. of ICMI, ACM (2014), 311-318.

[5] Freeman, D., Vennelakanti, R., and Madhvanath, S. Freehand pose-based gestural interaction: Studies and implications for interface design. In Proc. of IHCI, IEEE (2012), 1-6.

[6] Guinness, D., Jude, A., Poor, G. M., and Dover, A. Models for rested touchless gestural interaction. In Proc. of SUI, ACM (2015), 34-43.

[7] Guna, J., Jakus, G., Pogačnik, M., Tomažič, S., and Sodnik, J. An analysis of the precision and reliability of the leap motion sensor and its suitability for static and dynamic tracking. Sensors 14, 2 (2014), 3702-3720.

[8] Wachs, J. P., Kölsch, M., Stern, H., and Edan, Y. Vision-based hand-gesture applications. Commun. ACM 54, 2 (Feb. 2011), 60-71.