InSight: A Systematic Approach to Create Dynamic Human-Controller-Interactions

Roger Boldu, Haimo Zhang, Juan Pablo Forero Cortés, Sachith Muthukumarana, Suranga Nanayakkara
Augmented Human Lab, Singapore University of Technology and Design
{rboldu, haimo, juan, sachith, suranga}@ahlab.org

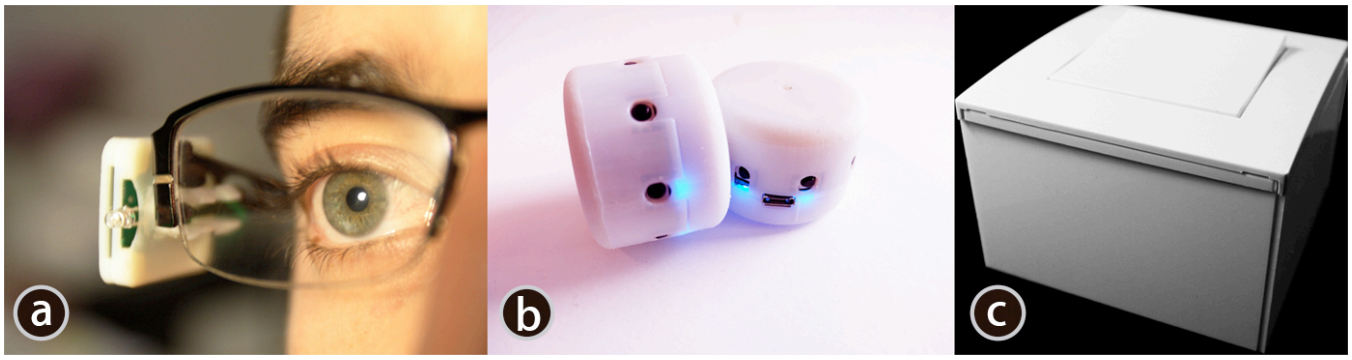

Figure 1: The InSight hardware components: a) the G-Ripple clip-on IR laser source; b) the G-Sens sensor to augment existing objects; c) example controller augmented with the hidden G-Sens sensor.

ABSTRACT
We present InSight, an intuitive technique to control smart objects with existing input devices in the environment, while simply looking at them. By leveraging the user's line of sight as a heuristic of gaze and attention, the InSight system directs input focus from input devices to the device that the user is looking at, thus creating an intuitive metaphor: you control the object you are looking at. In this paper, we contribute with technical details of the hardware and software implementation, and a discussion of single user and multi-user interaction possibilities.

CCS Concepts
• Human-centered computing~User interface management systems • Human-centered computing~Ubiquitous and mobile computing systems and tools

Author Keywords
Home Automation; Gaze Based Interactions, Sensors

INTRODUCTION & RELATED WORK
Home environments equipped with advanced automation technologies can sense the context as well as the attentional status of their users, thus providing more accurate and helpful services to the users [1,15]. Instead of conventional physical controls such as switches and knobs, virtual controls on smartphones [3] and web-based interfaces [10] are often used to control smart home appliances. They both have their own limitations. On the one hand, although virtual control interfaces are scalable and flexible due to their nature of being purely digital, they sacrifice intuitiveness and tangibility of physical controls, and require prolonged engagement with the digital interfaces, in order to accomplish micro-interactions as simple as switching off a smart lamp. On the other hand, conventional physical controls are simple and direct, but the users often need to learn a one-to-one mapping between the appliances and their respective control interfaces. For example, in a room with multiple lights and wall switches, a user needs to first learn which switch controls which light, in order to control a specific light.

Previous research on smart environments has mainly focused on inventing digital interfaces that are flexible and scalable, while less focused on re-adapting conventional physical controls in smart home applications, while retaining their intuitiveness and tangibility. The Reality Editor [4] augments smart objects with optic markers, which can be identified by an augmented reality (AR) app running on a smartphone. The AR app in turn overlays additional information and graphical user interfaces, through which a use could control the object or establish mappings between objects and controllers. It allows the user to customize the interface and behaviors of smart objects, but implies non-trivial interaction overhead

AH '17, March 16-18, 2017, Mountain View, CA, USA
 ISBN 978-1-4503-4835-5/17/03…$15.00
DOI: http://dx.doi.org/10.1145/3041164.3041195

("holster time") when establishing connections between actuators and controllers. A user has to pull out the smartphone, launch the AR app, acquire the actuator through the smartphone camera's viewfinder, and acquire the controller using the camera, while holding his/her finger on the smartphone screen. For ad-hoc connections that are used only once, this connection overhead is even larger than the actual interaction with the controller, which could be as simple as toggling a switch. Other previous research [8,13,14] used eye gaze as a heuristic of user attention, and explored the potential of smart objects that either accept voice command or utilize eye contact directly as the control signal (e.g., play/pause of a television). For such systems, alternative interaction paradigms are introduced, and the conventional physical controls associated with these devices are not leveraged. Attentive User Interfaces (AUIs) [15] is another related area of research, where gaze-aware devices carefully negotiate their request to get user attention in a socially appropriate manner, and in so doing acknowledge the fact that a user's attentional resources are scarce and need thoughtful management. AUIs however are primarily concerned with socially appropriate communications from the devices to the users [16], with rare examples of interactions initiated from the users to the devices [9].

In this paper, we are interested in this question: given the various devices and the pre-existing control interfaces in an environment, how can a user effectively and intuitively control the devices, without additional control interfaces and paradigms? We believe that this is a question worth answering, since an ideal interface should be transparent [17], which blends into the environment that the users are already familiar with.

The contributions of this paper include the technical details of the implementation of the InSight system, and a discussion of possible interaction scenarios.

INSIGHT

Inspired by gaze-based interaction [8,14] and tangible interaction [2,5], the InSight system allows for ad-hoc mapping between actuators (e.g., lights, speakers, fans, etc.) and virtually any input devices available in the environment (e.g., switches, knobs, keyboard, touch screens, etc.). A user's gaze directs the input focus to the actuator of interest, and the user relies on the existing familiarity with input devices to control the actuators.

For example, while working on a computer, a user could simply look at the lamp and use a mouse click to switch it on or off, or glace at the air-conditioner and use the scroll wheel of the same mouse to control the temperature. And when the user looks back at the computer screen, the mouse controls the cursor of the graphical user interface of the computer, as it normally would.

The above example illustrates the metaphor created by InSight: that you can use any input device to control the device you are currently looking at. The metaphor has two implications. First, it does not matter which device offers the input device. For example, the mouse is offered by the computer, but it makes sense to control the lamp, while the user is looking at it. Second, without a user looking at it, the device would not respond to user input. This makes sense when a user relies on direct visual feedback from the device to interact with it, such as a computer screen. However, eye contact might not be necessary in some scenarios, such as changing the volume of the loudspeaker when the ambient sound already indicates its status. Considering this as a trade-off between making eye contact and the necessity to acquire the control interface specifically associated with the devices (e.g., the knob on the loudspeaker, the remote control of the air-conditioner, etc.), we feel that, in many situations such as sitting in a couch in the living room, making eye contact is a smaller interaction overhead, compared to the effort of acquiring the designated controllers for the various devices.

To realize such an interaction style, there are two major challenges. First is to create a reliable and scalable method to discern a user's gaze at the devices in the environment. Second is to provide software services that dynamically create, manage, and destruct ad-hoc controller-actuator pairs, which maps between a controller's input signal and an actuator's status.

To address the first challenge, previous research [12] have sought to use eye tracking technologies [11] to discern a user's gaze direction. However, accurate and responsive gaze tracking is costly and constrained in terms of sensing range [6,7], suggesting poor scalability for deployment in a smart environment. On the other hand, wearable eye trackers, such as Tobii Pro Glasses [18], are also limited by their bulky size, and difficulty to accurately calibrate against the environment. The InSight system approximates a user's gaze as his/her head-aligned line of sight, instead of accurate eye tracking. This is achieved by simulating the user's line of sight as a directional infrared light from a clip-on emitter (Figure 1a, "G-Ripple") attached to the user's eyewear or hat. Infrared sensors (Figure 1b, "G-Sens") are retrofitted to devices, allowing the system to identify the device that the user is looking at.

To address the second challenge, the InSight software manages all G-Sens sensors as a network of devices, and provides abstractions such as communication protocols and APIs, so as to facilitate standardized and unified interaction among the user and the various devices. The software platform also allows new objects and controllers (Figure 1c) to be defined and added to the network of devices.

The overall InSight system is power-efficient, easy to deploy, scalable, and accurate to support gaze-directed interaction in a smart environment.

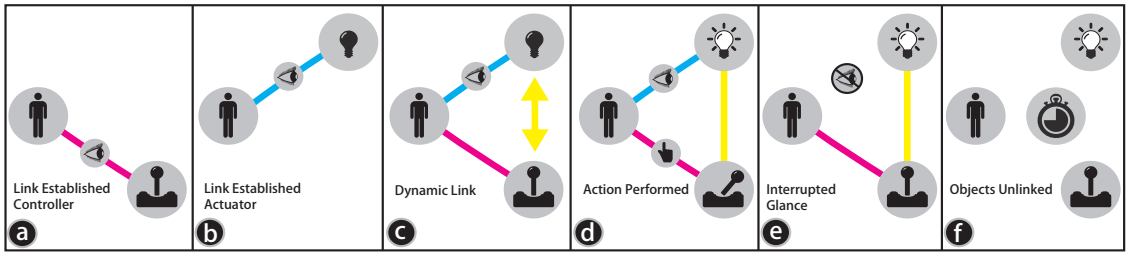

Figure 2: Operation mechanism of InSight

INSIGHT OPERATIONS

To use the InSight system, a user makes eye contact with the controller and actuator in turn (Figure 2a & 2b). A connection between the two devices of interest is established (Figure 2c), which allows the manipulation of the actuator using the controller (Figure 2d). This connection is removed when the user breaks eye contact with the actuator (Figure 2e & 2f), or the timeout of the interaction is reached.

To support this interaction, the InSight backend takes care of several tasks. G-Sens, the controller (that the user looked at), and the object (that the user is looking at) communicates with the interactions manager, which determines the mapping between the control actions and object properties and establish a link between the controller and the object. If the user breaks eye contact or releases the controller, InSight will automatically terminate the links and will make them available for a new connection.

Implementation

The InSight system is implemented as follows.

Hardware

The InSight hardwares (G-Ripple and G-Sens) are created using IR technology. Figure 3 shows the exploded view of both G-Sens and G-Ripple devices.

G-Sens: Each G-Sens sensor is comprised of a 32MHz PIC24FJ256GA406 microprocessor, eight 50°, 850nm TSOP36238 infrared receivers, and a ZigBee module (XBEE PRO 2.4GHz). The energy consumption in the low-energy mode is approximately 5.1mAh. When the user is interacting with the G-Sens device, the microprocessor would run with full power for demanding tasks, for a duration of few milliseconds, before resuming the low-energy mode. Each G-Sens device can run for approximately 50 hours on a fully charged 300mAh Li-Po battery. For firmware, each G-Sens sensor is programmed following a Manchester encoding. Each received frame is 20 bytes long, encoding information about the ID of the emitter, battery life, current time, duration of current eye contact (zero if no eye contact), number of frames, ID of the instruction and value of the instruction. The frame has a byte of CRC to detect possible transmission errors. The frames are sent at 50ms intervals. Apart from the 8 infrared receivers working simultaneously, the sensor information about the G-Sens device is sent through the ZigBee protocol, with identical frame structure as the infrared frame, at 100ms intervals. Without receiving frames from the infrared channel, a G-Sens device would enter low-energy mode to save battery.

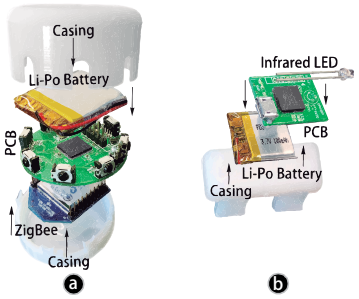

Figure 3: InSight Hardware: a) the G-Sens IR receiver; b) the G-Ripple IR laser.

G-Ripple: A G-Ripple device is worn by each user. It incorporates a 32MHz PIC24FJ256GA406 microprocessor, and an 850nm, 4°, OPV332 infrared LED-Laser Emitter. The power consumption in the low-energy mode is approximately 2mAh. During active periods with heavy processing load, the energy consumption can peak at 20mAh. At 50ms intervals, the microprocessor wakes up from the low-energy mode to perform heavy tasks for the span of a few milliseconds, before resuming the low-energy mode. This allows each G-Ripple device to run for approximately 30 hours on a fully charged 80mAh Li-Po battery. G-Ripple is programmed to transmit information about its identity in the same Manchester encoding and data frame format, as described in the G-Sens's firmware description. The frames are sent at 50ms intervals.

Software

The software backend of InSight is implemented as the following 4 main components, using the Python and C programming languages.

Interaction Manager: As the core of the system, this module identifies the objects that the user is looking at, and maps the actions performed on the active controllers to the active actuators.

G-Sens Manager: This module manages G-Sens sensors once it receives a frame from the G-Ripple device, the information is forwarded to the interaction manager module.

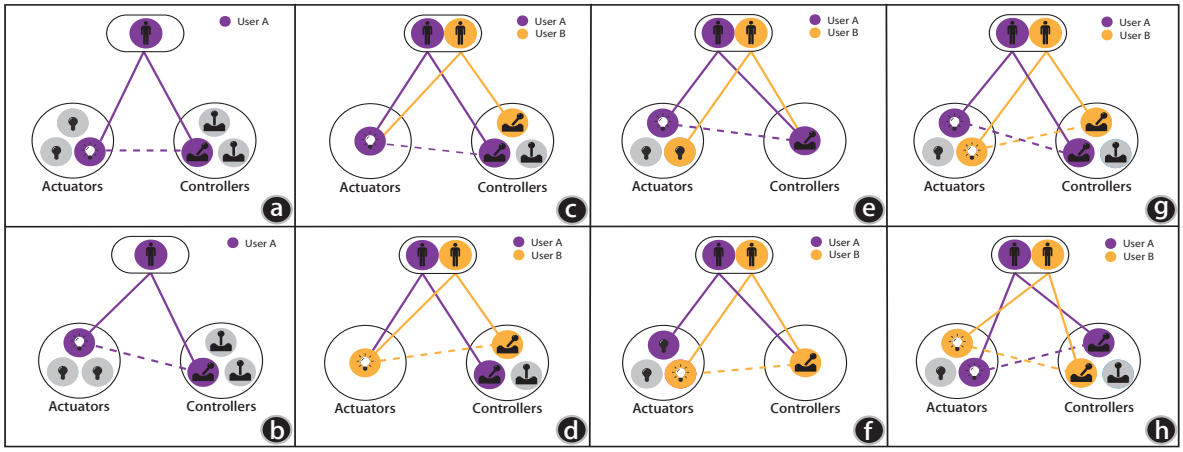

Figure 4: Interaction possibilities of InSight

Object Manager: This module manages all the different objects that can be controlled in the InSight system. Any object with network connectivity can be adapted into the system, following a template that defines the different actions that the object can perform for each control action.

Controller Manager: This module monitors actions performed on a controller. The controllers are categorized in different groups such as switches, buttons, or dials, to facilitate the mapping between control actions and device reactions.

The InSight software communicates with the devices through various protocols, including ZigBee, TTY, Bluetooth and Ethernet. It accommodates various smart home appliances, such as smart vehicles [19], smart plugs [20], and the Philips Hue smart bulb [21].

INTERACTION POSSIBILITIES

InSight allows for both single-user and multi-user interactions (Figure 4).

Single User: User A establishes a dynamic link between a controller and an actuator (Figure 4a). Only one controller/actuator pair is active at one time (Figure 4b).

Multiple users with single object: There is a specific actuator that both user A and user B attempt to use at the same time. In this scenario user A managed to create a link for the interaction before user B could. This contract stays valid until user A decides to stop looking at the object or the timeout for the interaction is reached (Figure 4c). During this time, user B is not able to connect with the object. User B will be able to interact with the object once the user A break eye contact with the object or user A's interaction with the device has timed out (Figure 4d).

Multiple users with single controller: There is a specific controller that both user A and user B are attempting to use (Figure 4e). In this scenario, user A managed to create a link for the interaction before user B could. This contract stays valid until user A decides to stop looking at the object or the timeout for the interaction is reached. Similarly to the previous case, user B is able to interact with the controller once user A has finished interacting using the controller (Figure 4f).

Multiple users with multiple controllers: Both users A and B are establishing dynamic links among different objects and controllers (Figure 4g). The dynamic links are created for each one of the users independently (Figure 4h). Once the interactions are finished, the users can switch to another set of controllers and objects.

LIMITATIONS & FUTURE DIRECTIONS

Creating ad-hoc links between existing objects and controllers presented many technological challenges. In addition to the InSight server, we would like to develop an intuitive web interface, to facilitate the addition and configuration of the different devices in the system.

In the current version, the user's gaze is approximated using infrared technology. For future research, alternative sensing options, such as ultrasonics and motion sensitive cameras, will be considered for improved accuracy, smaller form factors, and environmental robustness.

With the InSight system in place, it would be an interesting design study to find out how such a system would benefit the users, through performance-centric experiments and open-ended participatory design sessions.

CONCLUSION

In this paper, we present InSight, a robust and scalable way of controlling devices using existing controls in the environment, while the input focus follows the user's gaze. We explained the hardware components and software system that realize this interaction style. We discussed various usage scenarios considering multiple users and multiple actuators/controllers. The InSight system contributes as a way to readapt conventional physical controls in smart environment applications, which combines the flexibility and scalability of digital technologies with the familiarity, intuitiveness, and tangibility of conventional physical controls.

REFERENCES

1. A. J. Brush, Bongshin Lee, Ratul Mahajan, Sharad Agarwal, Stefan Saroiu, and Colin Dixon. 2011. Home automation in the wild: challenges and opportunities. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2115–2124. https://doi.org/10.1145/1978942.1979249
2. Luigi De Russis and Fulvio Corno. 2015. Homerules: A tangible end-user programming interface for smart homes. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, 2109–2114. https://doi.org/10.1145/2702613.2732795
3. Valentin Heun, Shunichi Kasahara, and Pattie Maes. 2013. Smarter objects: using AR technology to program physical objects and their interactions. In *CHI'13 Extended Abstracts on Human Factors in Computing Systems*, 961–966. https://doi.org/10.1145/2468356.2468528
4. Valentin Heun, Eva Stern-Rodriguez, Marc Teyssier, and Pattie Maes. 2016. Reality Editor. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (CHI EA '16), 4–4. https://doi.org/10.1145/2851581.2889431
5. Hiroshi Ishii and Brygg Ullmer. 1997. Tangible bits: towards seamless interfaces between people, bits and atoms. In *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, 234–241. https://doi.org/10.1145/258549.258715
6. Sang-won Leigh. 2013. eyeCan: Affordable and Versatile Gaze Interaction. In *Proceedings of the Adjunct Publication of the 26th Annual ACM Symposium on User Interface Software and Technology* (UIST '13 Adjunct), 79–80. https://doi.org/10.1145/2508468.2514719
7. Dongheng Li, Jason Babcock, and Derrick J. Parkhurst. 2006. openEyes: A Low-cost Head-mounted Eye-tracking Solution. In *Proceedings of the 2006 Symposium on Eye Tracking Research & Applications* (ETRA '06), 95–100. https://doi.org/10.1145/1117309.1117350
8. Päivi Majaranta. 2011. *Gaze Interaction and Applications of Eye Tracking: Advances in Assistive Technologies: Advances in Assistive Technologies*. IGI Global. Retrieved December 1, 2016 from https://books.google.com.sg/books?hl=en&lr=&id=HuWeBQAAQBAJ&oi=fnd&pg=PR1&dq=majaranta+gaze+interaction&ots=5fBaSjUaPE&sig=BjiIOP7k6vueSFmIlOXmF65rb9A
9. Aadil Mamuji, Roel Vertegaal, J. Shell, Thanh Pham, and Changuk Sohn. 2003. AuraLamp: contextual speech recognition in an eye contact sensing light appliance. In *Extended abstracts of ubicomp*. Retrieved December 1, 2016 from http://www.hml.queensu.ca/s/Mamuji-2003.pdf
10. Teddy Mantoro, Elbara E. Elnour, and others. 2011. Web-enabled smart home using wireless node infrastructure. In *Proceedings of the 9th International Conference on Advances in Mobile Computing and Multimedia*, 72–79. https://doi.org/10.1145/2095697.2095712
11. C. H Morimoto, D Koons, A Amir, and M Flickner. 2000. Pupil detection and tracking using multiple light sources. *Image and Vision Computing* 18, 4: 331–335. https://doi.org/10.1016/S0262-8856(99)00053-0
12. Ted Selker, Andrea Lockerd, and Jorge Martinez. 2001. Eye-R, a glasses-mounted eye motion detection interface. In *CHI'01 extended abstracts on Human factors in computing systems*, 179–180. https://doi.org/10.1145/634067.634176
13. Jeffrey S. Shell, Ted Selker, and Roel Vertegaal. 2003. Interacting with groups of computers. *Communications of the ACM* 46, 3: 40–46. https://doi.org/10.1145/636772.636796
14. Linda E. Sibert and Robert JK Jacob. 2000. Evaluation of eye gaze interaction. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, 281–288. https://doi.org/10.1145/332040.332445
15. Roel Vertegaal and others. 2003. Attentive user interfaces. *Communications of the ACM* 46, 3: 30–33. https://doi.org/10.1145/636772.636794
16. Roel Vertegaal, Jeffrey S. Shell, Daniel Chen, and Aadil Mamuji. 2006. Designing for augmented attention: Towards a framework for attentive user interfaces. *Computers in Human Behavior* 22, 4: 771–789. https://doi.org/10.1016/j.chb.2005.12.012
17. Mark Weiser. 1991. The computer for the 21st century. *Scientific american* 265, 3: 94–104. https://doi.org/10.1038/scientificamerican0991-94
18. 2015. Tobii Pro Glasses 2 wearable eye tracker. Retrieved December 1, 2016 from http://www.tobiipro.com/product-listing/tobii-pro-glasses-2/
19. I spy Tank White. Retrieved December 1, 2016 from http://ispytank.com/index.php/i-spy-tank-white.html
20. Wemo Switch. *Belkin*. Retrieved December 1, 2016 from http://www.belkin.com/sg/p/P-F7C027
21. Personal wireless lighting 8718291547778. *Philips*. Retrieved December 1, 2016 from http://www.philips.com.sg/c-p/8718291547778/hue-personal-wireless-lighting