User Interface Beaming

Seamless Interaction with Smart Things using Personal Wearable Computers

Simon Mayer
Department of Computer Science
ETH Zurich
Email: simon.mayer@inf.ethz.ch

Gábor Sörös
Department of Computer Science
ETH Zurich
Email: gabor.soros@inf.ethz.ch

***Abstract*—For the Internet of Things to be adopted in people's homes and at their workplaces, it is important to provide mechanisms that support them when controlling and monitoring smart things in their surroundings. We present the concept of user interface beaming, where the capabilities of different personal wearable computers are combined to allow users to conveniently interact with smart things in their environment. Smartglasses are used to select a target smart thing by means of current object recognition technologies. Then, an appropriate user interface for the target is rendered on the user's smartwatch. This interface is continuously updated to reflect state changes of the target and can be used to interact with that smart thing using different interaction modalities.**

***Keywords*—Smart Environment, Interaction, Wearable Computer, Smartphone, Smartglasses, Smartwatch**

I. Introduction

Since the advent of the Internet of Things, we are witnessing an ever increasing number of connected smart things deployed in homes and workplaces that provide advanced services to end users. We can also identify a clear trend towards wearable computing driven by the goal to reduce the time between the intention of the user and the corresponding action [8], exemplified in devices such as smartglasses, smartwatches, or connected wearable sensors. We propose a technique that combines the advantages of several such wearable devices to yield an end-to-end direct interaction system that allows users to seamlessly control smart things in their surroundings.

Personal wearable computers are always with us and are therefore ideal to simplify the interaction with our smart environment. Head-worn cameras and displays such as the ones in recent generation smartglasses offer the advantage to perceive the world from the user's viewpoint and also visualize information directly in front of the user's eye. However, it is less clear how eyewear computers can be used for interacting with smart things in the user's surroundings: The primary input modality for current devices is speech, which is cumbersome to use for many interaction tasks in daily life. Additionally, shortcomings in speech recognition algorithms may render the issuing of actuation commands even harder, especially in noisy environments. Using the built-in accelerometer that most smartglasses provide as additional input device is limited to a few selected scenarios, for instance to select an item from a list by tilting the head. While thus not being ideally suited for interaction tasks, smartglasses offer the possibility to know exactly what the user is looking at – thereby allowing the convenient *selection* of smart things or services to interact with. Smartwatches, in contrast, provide rapid and convenient *interaction* capabilities via their touch-enabled user interface and advanced gesture recognition algorithms.

In this paper, we propose to combine the advantages of head- and wrist-worn computers by allowing the selection of objects to interact with using smartglasses while interacting with them by means of a smartwatch. To this end, we present a prototype implementation that uses image recognition technologies to identify objects in the user's field of view and then renders an appropriate user interface on his/her smartwatch in a process we term *user interface beaming* (see Fig. 1). Combining different wearable computers in this way opens up new ways of interaction with everyday appliances: It allows users to discover, understand, and use interfaces of smart things in their surroundings seamlessly without relying on a dedicated remote controller. Throughout this paper, we focus on use cases in the home automation domain – the presented concept can, however, also be applied to other domains, for instance the interaction of workers with industrial robots in a "smart factory" context.

II. Synergism of Personal Wearable Computers

While wearable computing has been an active area in the research community for a long time, recently also several companies brought commercial wearable devices to the market. These devices are meant to seamlessly integrate into a body area network and enhance the sensing, processing, and actuation capabilities of the user as well as opening up new possibilities in providing personal assistance. This section presents the devices that we used in our prototype implementation of user interface beaming. Our system consists of three main components: a head-worn device for real-time object recognition, a wrist-worn device for interaction, and a pocket device that acts as mediator and provides Internet connectivity.

A. Smartglasses

Smartglasses are in our opinion particularly suitable for the recognition and selection of smart things (such as household appliances) as the viewpoint of the device's camera is very similar to the actual viewpoint of the user. For our prototype deployment, we chose the *Google Glass* device that is based on the Android operating system and has a dual-core 1.2GHz CPU as well as a camera with 720p-resolution. Its small display is positioned in front of the user's eye and is well suited for short notifications such as feedback of set values of an appliance.

Fig. 1: The concept of *user interface beaming*: (1) the user wants to interact with a device in his/her view recognized by the smartglasses; (2) the user looks at his/her smartwatch; (3) the watch downloads a semantic description of the user interface and renders the corresponding interaction primitives on the touchscreen; (4) the user controls the device using his/her smartwatch.

The Glass has direct WiFi connectivity or can be tethered to a phone via Bluetooth. While its processor allows real-time object recognition, the input of the current Glass model is limited to a slim touchpad, thereby limiting its suitability as a universal interaction device.

B. Smartwatch

In our opinion, the main advantage of a smartwatch over other on-body devices is the convenient wrist-worn touchpad for user input. In the implementation of our prototype, we relied on a *Samsung Galaxy Gear* smartwatch model. The Gear runs a custom Android operating system on a single-core 800MHz CPU and features a touch-sensitive 320x320 pixel resolution display. The Samsung Mobile SDK additionally enables gesture recognition from acceleration and tilt sensors that can be used to further enrich user input. However, the Gear's 2 Megapixel camera is located on the wristband and is therefore rather inconvenient to use. This smartwatch can communicate with selected phone models via the Bluetooth 4.0 Low Energy standard but offers no direct Internet connectivity.

C. Smartphone

Because in our case the selected smartglasses and the smartwatch devices have no direct Internet connection, a smartphone acts as a communication hub in the body area network that is created as part of our prototype. The smartphone can also act as a local store for user preferences, for instance with respect to the concrete appearance of specific device interfaces, and can cache user interface descriptions of previously recognized devices locally. In our experiments, we used a Samsung Galaxy S4 mobile phone.

III. System Overview

Our prototype system consists of three devices (cf. Fig. 2) – smartglasses, a smartwatch, and a smartphone – all of which run stand-alone Android applications. The watch is tethered to the smartphone via Bluetooth and the smartglasses can communicate with the smartphone via Bluetooth.

The software module deployed on the smartglasses runs an object classification algorithm that is based on the OpenCV-toolkit[1] and detects objects of interest in the camera frames of the device (cf. Fig. 4; details of the object classification are presented in Section IV). Whenever a smart thing (from a predefined set of objects) is recognized, the program resolves its URL using a local database and relays this information to the smartphone. The smartphone is then responsible for fetching a description of a suitable user interface for the target smart thing that is provided on the Web interface of that thing. The obtained interface description is then transmitted to the smartwatch that displays the described interface and allows the user to control the target using its touch screen and sensors (see Section V-A for details). Finally, because the Galaxy Gear does not feature direct Internet connectivity, the smartphone also acts as a mediator between the smartwatch and the target object (see Section V-B).

We have implemented several example scenarios for our proposed system:

- Controlling the volume of an audio/video system. The user can for instance use a graphical knob, a gyroscope-based orientation knob, or virtual buttons to control the volume (see Figs. 5 and 6). The audio/video system can also be switched off by shaking the smartwatch.

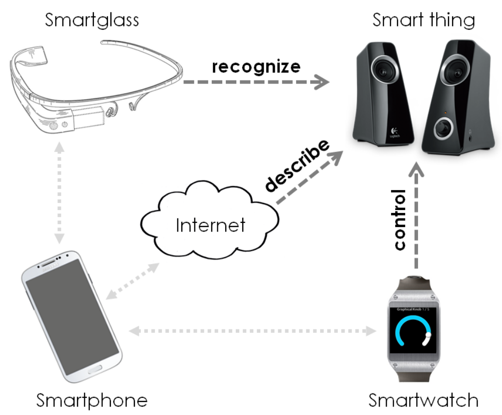

Fig. 2: System overview: the smartglasses recognize a smart thing based on its visual features; the smartphone downloads the user interface description and beams it to the smartwatch; the user can control the smart thing using the watch.

[1]See http://opencv.org/

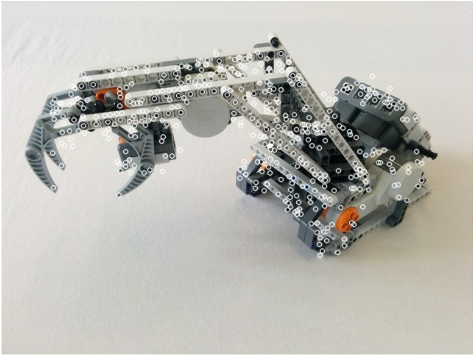

Fig. 3: Detected visual features of the smart thing.

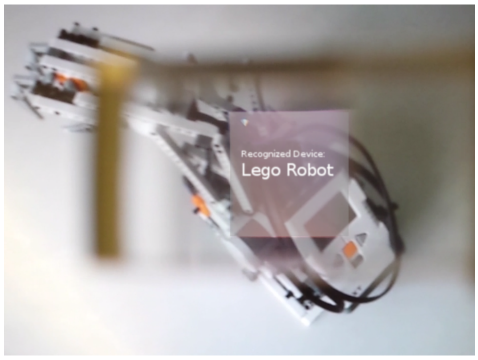

Fig. 4: The recognized smart thing seen through the display of Google Glass.

- Controlling the temperature in a room. The user can control the setpoint of a smart thermostat using interfaces similar to those for the audio/video system.
- Controlling a toy robot (cf. Fig. 3). The user can lower and lift the robot arm using graphical buttons or an orientation switch that is triggered using the gyroscope of the smartwatch.

IV. Object Recognition

Our Glass application can recognize different smart things (e.g., home appliances) from a predefined set based on their visual appearance which is described by local image features (i.e., vectors derived from patches around image corners). We experimented with various existing image feature detectors and descriptors and chose the speeded up robust features (SURF) [1] which are invariant to translation, rotation, scaling, and lighting changes (cf. Fig. 3). While other feature extraction algorithms might yield higher frame rates, SURF features are more robust to changing capturing conditions. Assuming that the user wants to interact with a smart thing in the center of his field of view, we can constrain the recognition algorithm to that part of the image, which facilitates the selection of a specific smart thing in case multiple things are visible in an image.

The extracted SURF features are quantized using the Bag of Words (BoW) [3] model and the images are classified using support vector machines (SVM), where we train a linear SVM classifier for each object in our database. In the recognition phase, we take the category that yields the highest classification score but report a match only if at least n $(n = 5)$ out of the latest $n + 1$ consecutive frames are classified to contain the same object, to improve the stability of our recognition results. For the training of our classifiers, we used 10 to 15 training images that represent different views of each smart thing and extracted 128-dimensional real-valued SURF descriptors from every image. Our prototype can robustly differentiate between eight object categories but – like other local feature-based image recognition algorithms – cannot deal with untextured objects nor differentiate between multiple object instances of the same appearance.

V. Interactor

All smart things that we consider within our scenarios (loudspeakers, smart thermostats, toy robots, etc.) contain a tiny Web server that provides access to their sensors and actuators via a REST interface. Apart from human-readable markup, a thing's Web representation also includes machine-readable descriptions that specify what an appropriate user interface for that thing should look like.[2] Given the URL of a target device, the software that is running on our smartphone is able to obtain these descriptions from the device. Because smart things may publish such descriptions in multiple different formats, an intermediary discovery service is used to obtain interface descriptions in a standard format. The interface description is then passed to the smartwatch which renders an appropriate user interface (cf. Fig. 5). This interface displays data that is provided by the target device (e.g., for sensors) and also allows the user to directly control the target device by using interaction primitives such as buttons or knobs.

Fig. 5: A graphical user interface to control the volume of an audio/video system (left: screenshot; right: our prototype).

A. *Interacting using the Galaxy Gear Smartwatch*

We have implemented user interfaces for multiple different interaction modalities: In addition to graphical interfaces that use the touch screen of a smartwatch to show information to the user and take his commands, also speech-based interfaces can be created, and the device's gyroscope and accelerometer can be used, for instance to adjust a loudspeaker's volume by tilting the device or to switch the speaker off by shaking the hand (see Fig. 6). Some of the interfaces also use the built-in vibrator of the smartwatch to give haptic feedback to the user. If multiple interfaces are suitable to control a single smart thing – such as the graphical knob, orientation knob, and buttons to control the loudspeaker volume shown in Figs. 5 and 6 – the

[2] The concrete description language that we use to convey this information is discussed in more detail in [7].

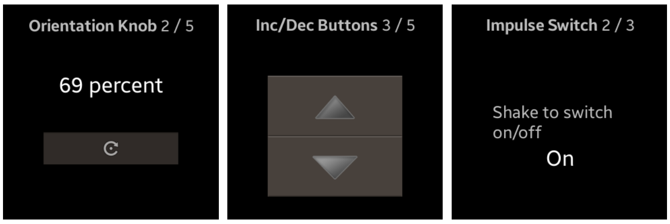

Fig. 6: Screenshots of user interfaces to control the volume of an audio/video system using the gyroscope of the smartwatch (left) and its touchscreen (center), as well as a user interface to switch the sound on/off by shaking the watch (right).

user can use a swiping gesture on the touchscreen to select his preferred interface.

When the user interacts with a rendered interface using his watch, the actions are translated into HTTP requests and sent to the target device to invoke the desired behavior (e.g., changing the volume of a loudspeaker). Likewise, the smartwatch polls the target device in regular intervals to reflect updates to its current state – for instance, if the volume has been changed by a third party – on the user interface.

B. Mediation via a Smartphone

Because the Galaxy Gear is not capable of directly connecting to the Internet, an Internet-capable device is required as a mediator to pass user commands to target devices and relay state changes to be reflected by the smartwatch. In our prototype system, whenever the Galaxy Gear wants to send an HTTP request, it formulates this call locally and then encodes it in a custom JSON-based format and sends it to the smartphone for execution, by means of the Samsung Accessory Services framework, a publish/subscribe-based message-passing middleware that is part of the Samsung Mobile SDK. Any answers that it (asynchronously) receives via that connection are inserted into a response queue locally that is regularly polled by the code responsible for managing the currently active interface. Although it was necessary to introduce the smartphone as a mediator between the watch and the target device, the system achieves round-trip times of under 200ms (i.e., the controlled device receives any command in under 100ms) and thus allows for convenient interaction with the target device.

VI. Related Work

To our knowledge, this is the first system that combines the functionality of smartglasses and smartwatches to allow convenient selection and seamless interaction with devices in the user's surroundings. It represents a further step in a line of work that started in our research group with 2D fiducial markers to identify smart things and then visualize interactions *between* them using a tablet device [5]. In a further step, we combined markerless object identification with the interaction capabilities of smartphones and tablets, to use these as universal interaction devices [6]. While, with the work presented in this paper, we demonstrate that our simple object recognition algorithms also work on wearable devices with less processing power, the major drawback of using sparse features remains: our system can only recognize a limited number of objects (about eight, depending on the number of features per object). In line with our approach, the combination of a smartwatch and a smartphone has been recently explored in [2]. The authors focus on extending the input space of the smartphone but do not consider controlling other devices. An alternative approach that specifically aims at augmenting physical things with sophisticated graphical user interfaces is presented in [4]. This system also relies on distinctive textures of the devices, but furthermore requires 3D models for augmentation. In contrast, our system requires only casually taken snapshots of the devices.

VII. Conclusion

We presented the concept and a prototype implementation of user interface beaming which combines a smartphone, smartglasses, and a smartwatch. With our system the user can recognize, understand, and control smart things in a smart environment without the need for any third-party remote control or an attached user interface. The smart things are recognized based on their visual appearance using the smartglasses, their interface descriptions are automatically downloaded, and the appropriate user interface elements are rendered on the smartwatch. Multiple interaction primitives with sensor and actuator devices have been demonstrated. Obvious application areas encompass smart homes, smart hospitals, and smart factories, as well as human-centered assistive cyber-physical environments.

Acknowledgements: This work was supported by the Swiss National Science Foundation under grant number 134631. The two authors contributed equally to this work.

References

[1] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[2] Xiang Chen, Tovi Grossman, Daniel J. Wigdor, and George Fitzmaurice. Duet: Exploring Joint Interactions on a Smart Phone and a Smart Watch. In *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems (CHI 2014), Toronto, Canada*, 2014.

[3] Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, and Cdric Bray. Visual Categorization with Bags of Keypoints. In *Workshop on Statistical Learning in Computer Vision (ECCV 2004), Prague, Czech Republic*, 2004.

[4] Valentin Heun, Shunichi Kasahara, and Pattie Maes. Smarter Objects: Using AR technology to Program Physical Objects and their Interactions. In *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems (CHI 2013), Paris, France*, 2013.

[5] Simon Mayer, Christian Beckel, Bram Scheidegger, Claude Barthels, and Gábor Sörös. Demo: Uncovering Device Whispers in Smart Homes. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia (MUM 2012), Ulm, Germany*, 2012.

[6] Simon Mayer, Markus Schalch, Marian George, and Gábor Sörös. Device Recognition for Intuitive Interaction with the Web of Things. In *Adjunct Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2013), Zurich, Switzerland*, 2013.

[7] Simon Mayer, Andreas Tschofen, Anind K. Dey, and Friedemann Mattern. User Interfaces for Smart Things - A Generative Approach with Semantic Interaction Descriptions. *ACM Transactions on Computer-Human Interaction*, 21(2), 2014.

[8] Thad Starner. Project Glass: An Extension of the Self. *IEEE Pervasive Computing*, 12(2):14–16, 2013.