Tracking Mobile Phones on Interactive Tabletops

Florian Echtler, Gudrun Klinker, PhD
I16 - Fachbereich Augmented Reality
Technische Universität München
Fakultät für Informatik
Boltzmannstr. 3, D-85747 Garching
{echtler|klinker}@in.tum.de

Abstract: The number of „interactive surface" systems, especially tabletop interfaces, in public is increasing. As more and more casual users interact with such systems, they may wish to use existing computing infrastructure, such as their mobile phone, in conjunction with the new interface. In this paper, we present a method to reliably detect and track unmodified mobile phones that are placed upon an interactive tabletop. Range detection and data exchange is done via Bluetooth, while the location of the phone is tracked through its shadow on the surface.

1 Introduction

With interactive surfaces, a novel and interesting field of research has emerged in recent years. These devices are particularly suited for the exploration of new user interface paradigms, especially those that require little to no prior knowledge of the system. Such interfaces are therefore well adapted to being deployed in public scenarios, where many casual users will be using the device for relatively short periods of time, sometimes even concurrently.

Nowadays, most people who would use an interactive surface can be expected to also carry a mobile phone with them. Current models have many advanced features, such as Bluetooth, camera and music functions, which users actively employ to create and share media. It is therefore reasonable to assume that, given the opportunity, users would want to exchange media with an interactive tabletop device.

2 Related Work

Interaction with physical devices on display surfaces is a concept that has been extensively examined, e.g. by Ishii [IU97] and Greenberg [GF01]. Most of these systems rely on an optical tracking modality, often a rear-mounted camera.

Along with the increasing interest in tabletop interfaces, the integration of mobile phones into such systems has also been investigated by several researchers. One widely publicized

example is the Surface system from Microsoft [Mic], which recently has been deployed in a mobile operator's store to provide information about mobile phones placed on top. Our work has been inspired by this system. However, the Surface relies on special tags (either optical or RFID) that have to be attached to the phones in order to be recognized. In contrast, our setup aims to support unmodified phones.

Tracking of devices via Bluetooth has also been investigated, e.g. by Hallberg et al. [HNS03] and Castano et al. [CSE04]. These systems rely on measuring the signal strength of an established Bluetooth link.

3 Hardware Setup

The central element of our system is an FTIR-based multi-touch table, similar to the system by Han [Han05]. The table is also equipped with an infrared shadow tracker which is able to assign finger contacts to a certain hand (see also [EHK08]). In this scenario, it will be used to determine the location of devices on the surface. While the additional hardware increases total complexity, it is nevertheless necessary as a means to track objects on the surface. As mobile phones are likely to have a hard plastic shell which doesn't show up on the FTIR screen, a second tracking mode is needed. While a soft silicone layer might also be used to this end, the shadow tracking has the added benefit of differentiation between a phone and, e.g., the user's palm, which also generates an FTIR response in addition to a shadow. An overview of the system is given in Figure 1.

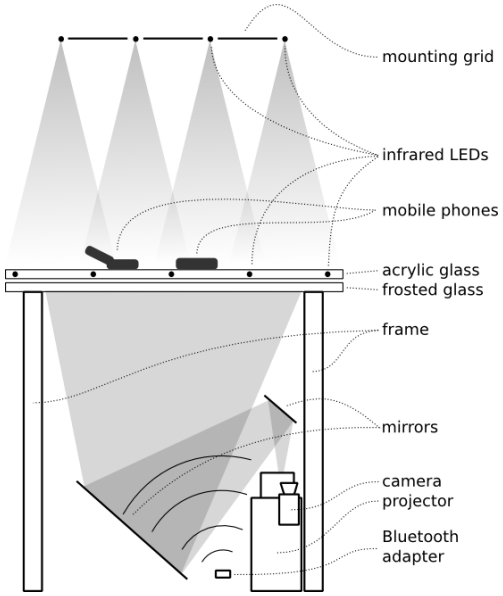


Figure 1: Hardware Setup

Note that although the primary mode of interaction in this scenario happens through the shadow tracker, the touch screen is still functional. As the rear-mounted camera generates 60 frames per second, it is possible to alternate between the shadow tracker and the FTIR surface every other frame while still providing a smooth user experience. For a more detailed description of the hardware design and the various issues which had to be solved, see our previous paper [EHK08].

In order to perform proximity detection on the mobile phones and exchange data with them, the integrated computer is equipped with a Bluetooth adapter. As proximity sensing is performed via the Received Signal Strenght Indicator (RSSI), this information should be available with low latency. Therefore, we selected a Broadcom USB adapter that supports the „inquiry with RSSI" feature which was introduced in version 1.2 of the Bluetooth Core Specification [Blu]. This allows us to have the adapter continuously run inquiry scans while at the same time getting RSSI data on all discoverable devices within radio range. One scan cycle takes about 1 second as opposed to older Bluetooth devices that don't have this feature. These have to issue a time-consuming connection request (up to 11 seconds) for every single RSSI measurement. Moreover, this connection requires a pairing between the mobile device and the adapter, for which the user has to enter a PIN code. In contrast, our system is able to function without explicit user interaction.

4 Software Design

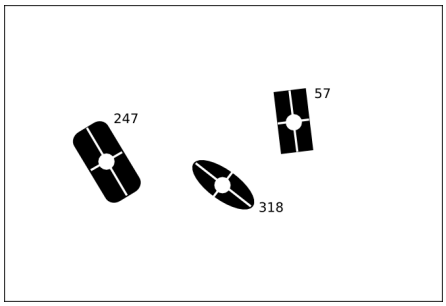
Our setup has two data sources: shadow location data from the optical tracking system as well as RSSI values from the Bluetooth adapter. The optical tracking layer processes the raw image data from the infrared camera and provides a high-level abstraction. After background subtraction, thresholding and segmentation, every blob is analyzed with respect to its size, location of centroid and major/minor principal axis. To determine this data, the blobs' central moments of first and second order are calculated. The blob is also assigned an numerical tracking identifier. This identifier stays with the blob as it moves over the surface by calculating the motion vector of the blob from the previous frame and matching it with the blob which is closest to the predicted location in the next frame. This data is then transmitted via a network interface to applications or higher-level trackers (see also Figure 2(a)).

Our tracking software is composed of two threads. The first one continuously collects RSSI data from Bluetooth devices within reception range, while the second one receives and processes the optical tracking data. The first step in this thread is to differentiate between shadows that are really cast by mobile phones and those cast by other objects, such as the users' hands.

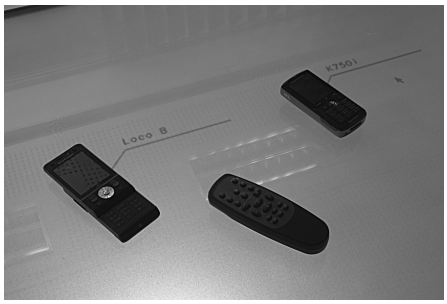
One obvious and easily applied criterion is blob size. There are upper and lower bounds on the surface area which a mobile phone is able to cover, as it is usually roughly pocket-sized. For our setup, we have determined these bounds to be at 1000 and 10000 pixels, respectively. While the range may seem large, it was chosen to account for, e.g., the dif-

ference between open and closed clamshell phones. In practice, these values have proven to be sufficient to include every phone we have placed on the surface.

The second criterion which we examine is blob motion. Before a phone can be reliably recognized, it should remain motionless on the table for one second, as this is approximately the duration of one inquiry scan cycle. Our software examines the blob position for the last 30 frames and calculates its standard deviation. If it falls below a threshold of 2 pixels, the blob is considered a candidate for being from a mobile phone.



(a) Data provided by the shadow tracker: tracking ID, location of centroid, size of shadow area and major/minor principal axis. Roundness is implied by size/axis ratio.



(b) Two mobile phones are automatically annotated with their Bluetooth names. One non-Bluetooth object on the table surface is ignored.

Figure 2: Shadow tracker data and resulting name/location assignment

The next step is to correlate these candidates with the proximity data from the Bluetooth thread. The RSSI measurements are usually returned in dBm.[1] The values typically range between -40 for close proximity and -90 at the limits of reception range. Obviously, these values are dependent on the mobile phone as well as the Bluetooth adapter in use. As our adapter is mounted at a distance of approximately 80 cm below the tabletop, a phone lying on the surface generates RSSI values of about -60 dBm. Therefore, we use a proximity threshold of -65 dBm to determine whether a phone is on or near the table surface. Although the distance to a mobile phone which is carried in the pocket of a person standing beside the table is about the same, the RSSI values for such phones are significantly lower. This is due to the non-uniform reception pattern of the dipole antenna which is employed in almost all Bluetooth dongles. Such an antenna usually exhibits several distinct lobes with high reception sensitivity. In our case, the antenna is oriented so that the main lobe points straight upwards, thereby favoring phones located on the surface and not those beside the table.

Finally, the list of phone candidate blobs and Bluetooth devices can be compared. In an ideal case, there is one unassigned blob and one newly detected device in range, which makes the assignment trivial. In this case, the optical characteristics (size and length of

[1] Even though this value might not reflect the true received signal power, but rather some internal measure, we can accept this measurement as-is, as we are currently relying on an experimentally determined threshold to decide between the inside- and outside-range cases.

major and minor axis) of the blob are also stored along with the Bluetooth data. This can be used for later identification of devices if ambiguities arise. For example, a candidate blob can appear without suitable Bluetooth devices in range. This can occur when the discoverable mode of a phone has a fixed timeout. In this case, the blob features are compared with the list of previously recognized devices. The best-fitting match according to a squared-error measure is then used to match the blob with Bluetooth data. Also, if several blobs and Bluetooth devices appear simultaneously, the blob features are compared with previous matches to resolve this ambiguity.

We have written a proof-of-concept application which annotates every Bluetooth device on the table surface with the name that was retrieved during the inquiry scan (see also Figure 2(b)). Non-Bluetooth devices are ignored. This application is easily extensible to support, e.g., a drop-down menu which is triggered by tapping on the name. This menu can then offer additional interaction possibilities, like browsing through the phone contents. In a drag-and-drop-based environment, the phone could also be used as drop target, e.g. by adding a ”halo” which indicates the drop-sensitive area (see Figure 3).

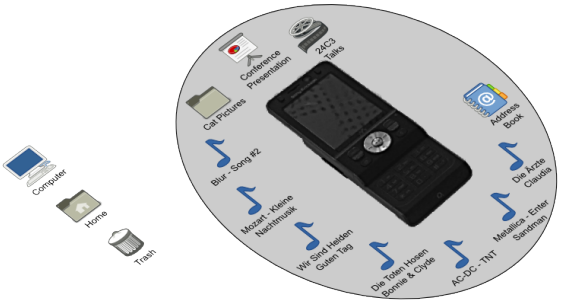


Figure 3: Mockup of a mobile phone media browser

5 Conclusion and Future Work

In this paper, we have presented a method to track and identify unmodified mobile phones which are placed on an interactive tabletop. We have implemented a prototypical application to demonstrate the feasibility of our approach and are planning to develop a mobile phone media browsing application. One shortcoming of the current system is that it is unable to differentiate between several objects which are placed on the surface in a short timeframe ($<$ 1 sec.). Therefore, we are investigating possible improvements to our tracking approach, such as using several Bluetooth adapters in parallel. Although RSSI data is very noisy, a least-squares optimizer could be employed to provide at least a rough position estimate as opposed to the current binary within/outside range decision. This would allow the tracker to correctly distinguish between multiple phones and non-Bluetooth devices which are simultaneously placed on the surface.

References

[Blu] Bluetooth SIG. Core Specification 2.1 + EDR. `http://www.bluetooth.com/Bluetooth/Technology/Building/Specifications/`.

[CSE04] J.G. Castano, M. Svensson, and M. Ekstrom. Local positioning for wireless sensors based on Bluetooth. *Radio and Wireless Conference, 2004 IEEE*, pages 195–198, 19-22 Sept. 2004.

[EHK08] F. Echtler, M. Huber, and G. Klinker. Shadow Tracking on Multi-Touch Tables. In *AVI '08: Proceedings of the ACM Conference on Advanced Visual Interfaces*, pages 388–391, 2008.

[GF01] S. Greenberg and C. Fitchett. Phidgets: easy development of physical interfaces through physical widgets. In *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*, pages 209–218, 2001.

[Han05] J.Y. Han. Low-cost multi-touch sensing through frustrated total internal reflection. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 115–118, New York, NY, USA, 2005. ACM Press.

[HNS03] J. Hallberg, M. Nilsson, and K. Synnes. Positioning with Bluetooth. *Telecommunications, 2003. ICT 2003. 10th International Conference on*, 2:954–958, 23 Feb.-1 March 2003.

[IU97] H. Ishii and B. Ullmer. Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms. In *CHI '97: Proceedings of the Conference on Human Factors in Computing Systems*, pages 234–241, 1997.

[Mic] Microsoft. Surface. `http://www.microsoft.com/surface/`.