Learning Instead of Markers: Flexible Recognition of Mobile Devices on Interactive Surfaces

Philipp Mock[1](✉), Jörg Edelmann[2], and Wolfgang Rosenstiel[1]

[1] University of Tübingen, Sand 13, 72076 Tübingen, Germany
{philipp.mock,wolfgang.rosenstiel}@uni-tuebingen.de
[2] Knowledge Media Research Center, Schleichstr. 6, 72076 Tübingen, Germany
j.edelmann@iwm-kmrc.de

Abstract. We propose an approach for recognition of mobile devices on interactive surfaces that do not support optical markers. Our system only requires an axis aligned bounding box of the object placed on the touchscreen in combination with position data from the mobile devices integrated inertia measurement unit (IMU). We put special emphasis on maximum flexibility in terms of compatibility with varying multi-touch sensor techniques and different kinds of mobile devices. A new device can be added to the system with a short training phase, during which the device is moved across the interactive surface. A device model is automatically created from the recorded data using support vector machines. Different devices of the same size are identified by analyzing their IMU data streams for transitions into a horizontally resting state. The system has been tested in a museum environment.

Keywords: Human computer interaction · Tabletop interaction · Markerless object tracking · Machine learning

1 Introduction

Large interactive tabletop installations are frequently used in museums or other kinds of exhibitions as a visitor information system. In many cases, a visitors guide for mobile devices accompanies these installations. Using mobile devices and a tabletop in conjunction offers several opportunities for the design of information systems. For example, such a combination can be used to allow visitors to create a selection of favorite artworks on their smart phone while visiting the exhibition. This individual selection can later be transferred to the tabletop in order to access further information that cannot be presented adequately on the small screen of a mobile device. The other way round, a visitor can select an individual tour on a tabletop at the entrance of a museum or trade show and then transfer it to an app on a smart phone that will guide him or her through the exhibition. To realize this, the devices need to communicate with each other.

N. Streitz and P. Markopoulos (Eds.): DAPI 2015, LNCS 9189, pp. 325–336, 2015.
DOI: 10.1007/978-3-319-20804-6_30

While the data can simply be sent via UDP socket connection or Bluetooth, the problem of assigning individual data streams to a specific device is not trivial.

The easiest way of achieving a correct assignment is to use the tabletop's touch sensor for the recognition. That way, the device can simply be placed on the interactive surface to establish the communication. If an optical sensor is used, optical markers attached to mobile devices are a fast and reliable way to realize this (e.g., camera based [5], Microsoft PixelSense). However, many sensor types do not support these markers. Particularly touch frames, which sense active fingers from the side and which are frequently used for large screen diagonals, do not support this kind of interaction (e.g., PQ Labs G4 [11]). As a consequence, another way of sensing mobile devices has to be found for these cases.

In this work, we present a recognition system that does not rely on optical markers and has minimum requirements regarding the touch sensor. Our tracking software is designed to simply be run in the background to provide object recognition for any given application.

As the amount of information that an interactive screen provides for a touch interaction varies heavily depending on the actual sensor configuration, we focus on using only the most basic features (x- and y-position and size in the form of an axis aligned bounding box) in conjunction with accelerometer data from the mobile device.

2 Related Work

The most common way of enabling object recognition on an optical screen are fiducial markers: An optically individual marker (similar to a QR code) is placed on an object and the optical sensor uses computer vision and object recognition algorithms to detect and track an active marker (e.g., [3,5,7,10]). This is possible for installations that sense interactions from below the screen with sufficient resolution.

When optical markers are impractical or the sensing technique does not support markers, the communication between mobile devices and a large screen installation has to established in other ways. One way to do this is to use the built-in flashlight of a mobile device [2,13]: combined with an optical touch sensor, short messages can be encoded with a series of automatically generated short light pulses. These can be used as an authentication scheme to establish bidirectional communication between a large screen device and a mobile phone or tablet.

Miyaoku et al. [8] proposed a light signal marker method called "C-Blink". Here, the LCD of a mobile device is used to emit so called hue-difference-blinks that a large screen installation can detect using a screen-side sensor.

Built-in IMUs have also been successfully used to implement communication between mobile and tabletop devices. For instance, Schmidt et al. enable pick-and-drop-style transfer of objects between a mobile phone and an interactive surface using a combination of accelerometer and multi-touch sensor data [12]. Phone touch events are discriminated from finger touch events by the surface and device identities are determined using data from an external accelerometer attached to the mobile device.

Albarelli et al. recognize appear and stop gestures of mobile devices on an interactive screen by using supervised machine learning methods [1]. For each gesture, a model is trained with a large data set containing positive and negative examples optical touchscreen combined with acceleration and velocity measurements obtained from mobile devices. The authors could show that using such an approach can be a robust alternative to using optical markers.

3 Contribution

The main contribution of this work is a lightweight tracking software for markerless detection of mobile devices, which can be used with any type of interactive surface that can detect the size of an object on the screen. Similar to [1], we use a machine learning approach to discriminate objects on the screen (e.g., a mobile phone or a tablet) from regular finger touch interactions. A trained object is described with the axis aligned bounding box only to ensure compatibility with all common touch sensing hardware. The identity of mobile devices is eventually determined with IMU data: a device is assigned to a touch event with correct dimensions, if a simultaneous transition to a horizontally resting state is detected with the integrated IMU.

For our tracking software, we put special emphasis on ease of use and maximum flexibility. Accordingly, we describe how a new device can be added to the system by moving it across the screen's surface for a short period of time. A model is then automatically created from recorded interaction data. The resulting device model inherently reflects characteristics like sensor noise, since real observations from the touch sensor are used as training data. Our software was designed to be run in the background and distribute touch and object events via TUIO protocol [6].

The proposed system was developed to enable an intuitive way of establishing the communication between two devices for a visitor information system in a

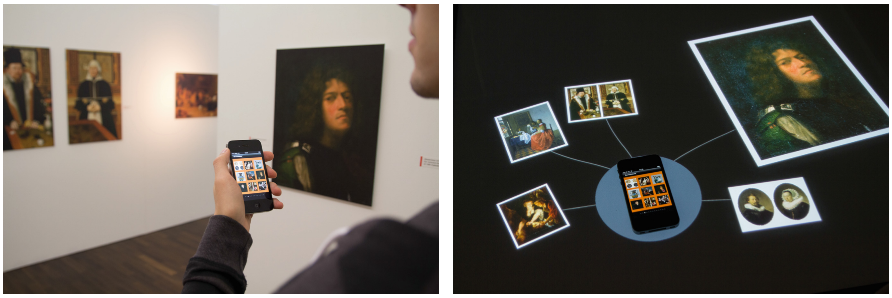

Fig. 1. Practical evaluation of the system in an art museum environment: Users can select artworks on their mobile while visiting the exhibition. The mobile device can be placed on the tabletop to access more detailed information about an artwork.

museum environment. The visitor information system uses mobile devices in conjunction with a large touchscreen. Tablets and mobile phones can be used within the exhibition as a multimedia guide. During the visit, artworks can be tagged to create a collection of personal favorites on the mobile device. Later on, visitors can access additional information for these favorites on the tabletop after exiting the exhibition. The object recognition system was evaluated in an experimental exhibition that uses reproductions of the original artworks and which is used for several museum and visitor research studies (see Fig. 1).

4 Detecting Object Shapes

The possibilities of recognizing objects on the surface of an interactive screen largely depend on the used multi-touch technology. Regarding optical touch sensing, most technologies can be separated into approaches that detect fingers and objects from below the screen (e.g., with a camera) and approaches that use senders and emitters which are placed around the screen and thus detect the visual hull of an object from the sides (e.g., [4,9]). Depending on the hardware, a system of the latter kind provides the convex hull, an object aligned bounding box or just an axis aligned bounding box for an active object.

We are using an 80" tabletop device equipped with a PQ Labs G4 multi-touch screen [11], which is another example for the visual hull technique, for a visitor information system in an art museum. Apart from position on the screen, the G4 touch frame provides an axis aligned bounding box that correlates to the number of sender/emitter-pairs that have been blocked by an object.

As the number of features that describe an object is very low (width and height of the axis aligned bounding box) and we know the dimensions of the original device, simple trigonometry can be used to verify whether the size of a bounding box matches the dimensions of a mobile device. However, if we assume that the mobile device is a perfect rectangle, we potentially induce errors due to device characteristics like rounded corners. If the sensor furthermore suffers from noise, this also has to be modeled separately. In order to avoid the parameter tuning that would be necessary to fit a parametrized model to the exact shape of a device while considering the sensor noise of the interactive surface, we chose a machine learning approach instead. That is, recording real measurements of a device on the tabletop and using this data to train a support vector machine (SVM).

SVM classifiers are binary large margin classifiers that optimize the decision boundary between data samples of two categories so that the distance between the separating hyperplane and all data points is maximized. This generally reduces the generalization error of the classifier. The objective of distinguishing a certain active mobile device from any other input can be considered an anomaly detection problem. We implemented this by using a one-class SVM, which determines the hyperplane that separates samples from one class from the origin with maximum margin. A one-class SVM is trained with data from a single class. The necessary labeled data can be obtained with a dedicated training phase during which the

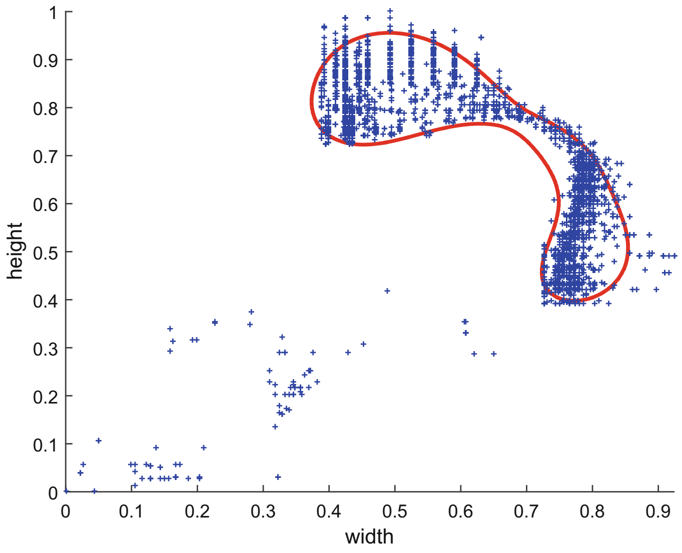

Fig. 2. Normalized measurements of width and height of an iPod on the screen's surface. The decision boundary of the one-class SVM classifier is depicted in red. Note that the data set contains highly noisy measurements and unintentional input.

mobile device is moved across the screen's surface. Figure 2 illustrates a data set that was obtained from such a training phase. As the number of features (width and height) is low compared to the number of samples (several thousand), we use a radial basis function (RBF) kernel for our one-class SVM classifiers. The SVM parameters ν and γ, which control the number of outliers in the training data and the standard deviation of the RBF, have to be optimized per training data set using grid search. Figure 3 shows the results for an example data set. In this case, $\nu = 0.04$ and $\gamma = 28$ gave the best results.

Using this approach, a new device can be added as follows: Apart from the default tracking mode, our object recognition application can also be operated in training mode. To add his or her device, a user switches to training and is then requested to place the device on the screen. After moving the mobile device around for 30 s, a device model is automatically created from the recorded data, which can be henceforth used for tracking. In order to avoid having multiple models for nearly identical sized devices, a combined model can be created, if the performances of two models show comparable classification performance for samples of the respective other class. Still, having multiple models for similar sized objects is not a problem, because device identity is ultimately determined using accelerometer data.

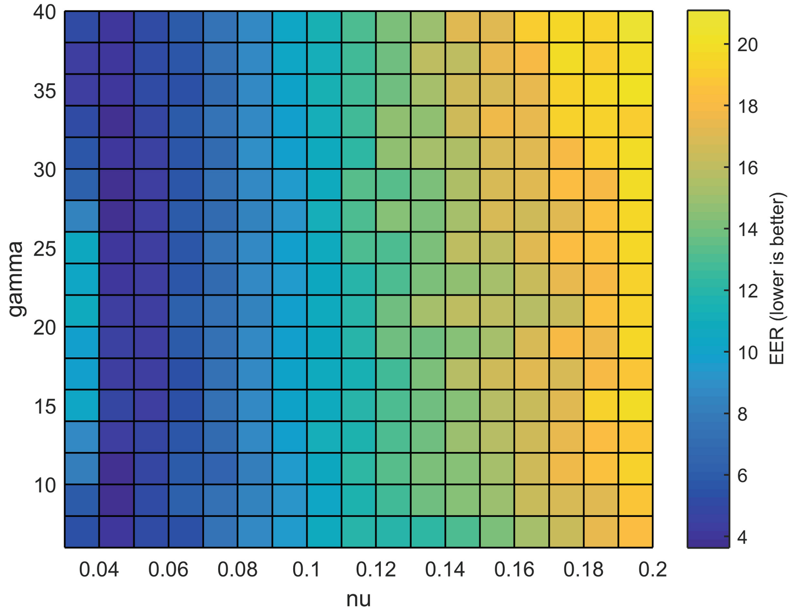

Fig. 3. Grid search results for an example training data set of iPod Touches including a significant number of noisy samples. The EER values apply to recorded regular finger touches.

5 Communication with Mobile Devices

When regarding only the sizes of objects, two devices with equal dimensions cannot be kept apart. Furthermore, a palm or multiple fingers can randomly cause patterns that are falsely classified as a trained object. To identify a particular device and to avoid false positives from unintentional input, the IMU of a mobile device is included into the decision process. Transitions from any inclination to a horizontally resting state can be observed directly from the accelerometer signal.

One way to integrate IMU data into the system is to model appear and stop gestures of mobile devices with combined data from the touch sensor and the IMU, as described in [1]. That is, synchronizing the clock of all involved devices and modeling causalities of observed touches and measurements from the position sensors. The resulting models reflect mutual relations between IMU and touch data for each of the modeled gestures. While this can result in very good detection rates, it requires a relatively high amount of samples for each of the modeled gestures, so that overfitting of the training data is avoided. For example, a model that matches the behavioral patterns of only a few users perfectly could produce substantial errors for unknown users who use the system in a different manner.

As our software was designed to be used in various different museum and visitor research application scenarios with a broad variety of devices, it is rather inconvenient to retrain gestures whenever a new device is added to the system.

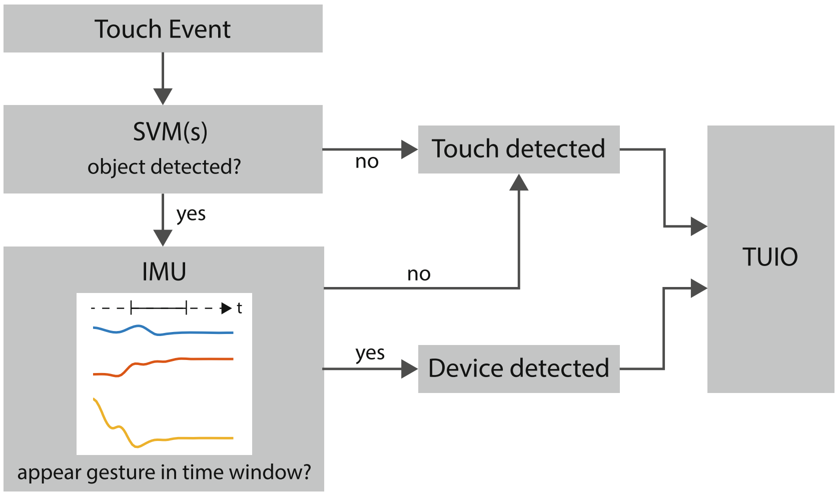

Fig. 4. The device recognition process including SVM classifiers trained with objects sizes and a subsequent analysis of accelerometer data. Figure 5 shows a more detailed example of accelerometer data during an appear event including the applied time windows.

This is why we opted for a simpler yet more flexible approach instead: Device dimensions are determined with the above object detection based on one-class SVM classifiers. Whenever a device SVM gives a positive result (i.e., an object of the expected size has been detected), the IMU data of all nearby mobile devices (sent via UDP) is analyzed for transitions to a resting state. The overall classification process is illustrated in Fig. 4.

A transition is characterized by a predetermined number of measurements corresponding to motion followed by a number of measurements that correspond to a resting device. To be more precise, we require all acceleration values to be within a threshold of 0.05 around the expected values (Ax and $Ay \approx 0$, $Az \approx -1$) at the end of the action and at least of them to be out of the range beforehand (compare Fig. 5). Eventually, a detected touch is associated to a specific device, if such a transition is detected within a short time window after a positive result of the corresponding SVM.

The timespan between a positive SVM classification and a detected IMU gesture depends on several factors: network latency, refresh rate of the accelerometer data and the way a device is placed on the surface. The first two factors add up to a overall system latency. For an IMU refresh rate of 10 Hz every measurement that is read before a device is accepted adds 100 ms to the system latency that is induced by the UDP network. We got the best results using a window of five consecutive measurements (2 in motion, 3 horizontally resting).

Lastly, the actual gesture influences the required duration because the velocity and direction of motion determine the time it takes before the IMU values level out at the expected resting state. Recorded data revealed that the difference

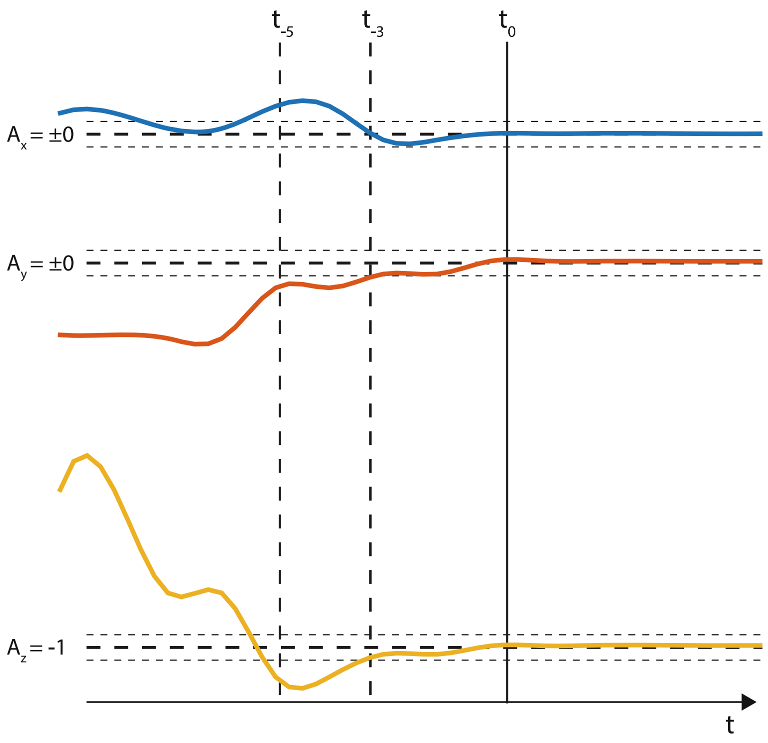

Fig. 5. Visualization of the accelerometer data of a device when being placed on the tabletop. The device is recognized at t_0. The time window starts at t_{-5}, a resting state is detected from t_{-3} on.

between a slow and a very fast gesture can be up to 200 ms. This interval also has to be added to the overall time window. Taking these factors into account, we found that a time window of 700 ms yielded the best results.

A contact area that has been associated with a certain device remains linked to it until it disappears. This means, that the SVM does not need to correctly identify a device on a permanent basis. Still, a problem of most touch sensors is that finger and object tracking is not perfect especially for rapid movements. As a consequence, a recognized touch can disappear for one or more consecutive frames. When it reappears, it usually receives a new identifier, resulting in sequential *TOUCH UP* end *TOUCH DOWN* events.

A stability timer for disappearing objects can fix this issue. That is, whenever a touch, which is associated with a mobile device, disappears and a new touch appears at the approximate same position, the system analyzes the IMU data of the respective device: If the device remained on the surface, the IMU data will not show an appear gesture. Consequently, if A_z remains within a threshold of 0.05 around -1, the new touch is assigned to the device. This induces a short delay for remove events as we have to wait for IMU data, but depending on the used touch sensor it potentially increases the object tracking stability significantly.

6 Results

In order to evaluate the object shape SVM classifiers, we recorded size measurements of iPod Touches and iPad Minis that were moved on the screen's surface and regular finger interactions during normal use of the visitor information application. Each of the resulting data sets contains between 1500 to 2000 samples. The device models where trained with samples from a 30 s training phase.

We found that the optimal model parameters vary for different training data sets. The determining factor is how clean the initial data sets are. That is, how many unintentional touches are included in the recorded data. The more noisy samples are included, the more samples have to be considered as outliers. This can directly be adjusted with the ν parameter. In our evaluations, the average equal error rate (EER) for different iPod models is 6.11 compared to regular finger touches. iPad Minis are easier to distinguish from iPods due to the major difference in terms of screen diagonals. Accordingly, average EER is 2.9 for this case. Performance of the iPad Mini models was uniformly good: averaged EER values are 0.41 for fingers and 0.89 for iPods. Figure 6 illustrates the performance of an iPod model compared to finger and iPad interactions in a receiver operating characteristic (ROC) curve for $\gamma = 20$ and $\gamma = 10$. We advise against using devices that are much smaller than an iPod Touch or a regular smart phone. This is because small objects are likely to be confused with fingers, which drastically increases the likelihood of false positive classifications.

The recognition rates of the IMU based device identification are harder to evaluate during real-world usage, because studies in the experimental exhibition usually require participants to put their mobile device on the tabletop only once or twice. As we could not create any meaningful statistics from this sparse data, we asked members of our department to place mobile devices on the tabletop in close succession to create a large number of samples within a short period of time. That way, we recorded a total of 440 appear events from 7 participants. Altogether, 99.09 % of these gestures were recognized correctly from the IMU data streams.

In order to estimate the system's robustness with regard to IMU data from other devices, we furthermore evaluated data streams from normal usage of a mobile visitor information application. The critical factor for misclassifications due to collisions with other data streams are false positive recognitions of devices that are currently not interacting with the tabletop. Accordingly, the test persons were asked to visit the experimental exhibition and use their mobile devices to retrieve information about the exhibits in form of audio, text and illustrations.

For one device, an average of 1.05 transitions into a resting state were detected per minute. The probability p of a collision can be calculated using a hypergeometric distribution as $p = \binom{N-n_{active}}{k} / \binom{N}{k}$. N is the total number of time steps and n_{active} is the number of time steps during which transitions are accepted due to a positive result of an SVM. k is the number of detected transitions during the considered time frame. When assuming one device being placed on the tabletop per minute, a time window of 700 ms and the above 1.05 transitions per minute, this gives a collision likelihood of 1.17 %. The value increases with additional devices or more frequent device interactions.

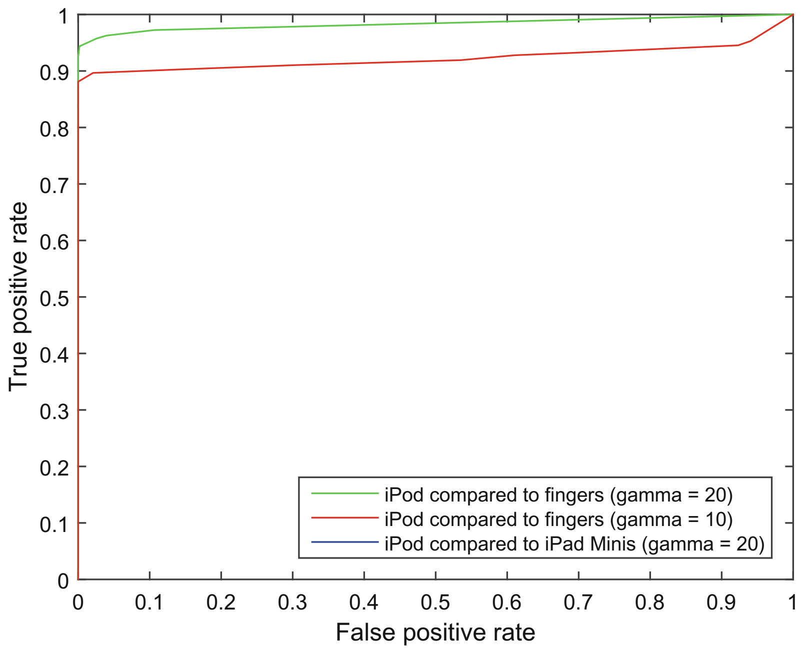

Fig. 6. ROC curve of an iPod model compared to finger interactions and iPad Minis with different kernel parameters. For $\gamma = 20$, EER is below 5 compared to both categories.

7 Discussion and Future Work

Our results reveal that basic sensor data (axis aligned bounding box and position) is sufficient to implement recognition of mobile devices on an interactive surface. By keeping sensor requirements at a minimum, we ensure compatibility with a broad range of touch sensors. We could show that data from a 30 s training phase is enough to add a new device model to the recognition system. As a proof of concept, the approach was applied to a visitor information system in a museum environment.

We identify devices with similar dimensions by incorporating position data from the mobile's integrated IMU. The gesture models for appear events have been simplified in order to achieve high flexibility in terms of using different types of mobile devices. The achieved classification accuracies qualify our approach for a variety of application scenarios where infrequent misclassifications are tolerable (e.g., a visitor information system or a multi-media presentation platform for meeting rooms).

Although we got satisfactory results in our testings, there are some limitations to the system's performance. First, the number of simultaneously used mobile devices has a strong impact on the reliability of the tracking, as the approach we propose is probabilistic. We found that misclassifications are unlikely for a small number of devices. Still, the likelihood of confusing two devices increases with every additional data stream that is transmitted to the tabletop. This means that the approach will perform poorly in application scenarios with a large amount of simultaneously active mobile devices. In this case, a proximity sensor could reduce the number of devices that communicate with the tabletop at a time. If the facility, for example, provides an indoor positioning system, the communication could be disabled for all units that are not in direct proximity to the interactive surface.

Apart from that, it is possible to intentionally evoke tracking errors by laying down the mobile at a very slow pace or by exactly timing two gestures with different devices to one another. However, we do not see this as a major drawback, as we did not observe any occasions where this happened by accident.

It should also be noted that a user has to pick up his or her device and place it down again in case the tabletop did not recognize it in the first place. This is a problem of all gesture based solutions for object recognition.

In the future, we intend to explore the possibilities of using an approach similar to the one presented in this work to reduce the amount of unintentional input on a touchscreen. More precisely, we want to validate whether an interaction model trained from a large data set of regular finger touches can be used to make a tabletop more robust against clothes or other objects accidentally touching the screen.

Acknowledgements. Philipp Mock was a doctoral student of the LEAD Graduate School [GSC1028], funded by the Excellence Initiative of the German federal and state governments.

References

1. Albarelli, A., Bergamasco, F., Celentano, A., Cosmo, L., Torsello, A.: Using multiple sensors for reliable markerless identification through supervised learning. Mach. Vision Appl. **24**(7), 1539–1554 (2013)
2. Hesselmann, T., Henze, N., Boll, S.: Flashlight: Optical communication between mobile phones and interactive tabletops. In: Proceedings of the ITS 2010, pp. 135–138. ACM (2010)
3. Hodges, S., Izadi, S., Butler, A., Rrustemi, A., Buxton, B.: Thinsight: Versatile multi-touch sensing for thin form-factor displays. In: Proceedings of the UIST 2007, pp. 259–268. ACM (2007)
4. Jovanovic, N., Korst, J., Pronk, V.: Object detection in flatland. In: Proceedings of the ADVCOMP 2009, pp. 95–100. IEEE Computer Society (2009)
5. Kaltenbrunner, M., Bencina, R.: Reactivision: a computer-vision framework for table-based tangible interaction. In: Proceedings of the TEI 2007, pp. 69–74. ACM (2007)
6. Kaltenbrunner, M., Bovermann, T., Bencina, R., Costanza, E.: Tuio: a protocol for table-top tangible user interfaces. In: Proceedings of the 2nd Interactive Sonification Workshop (2005)

7. Marquardt, N., Kiemer, J., Greenberg, S.: What caused that touch?: Expressive interaction with a surface through fiduciary-tagged gloves. In: Proceedings of the ITS 2010, pp. 139–142. ACM (2010)
8. Miyaoku, K., Higashino, S., Tonomura, Y.: C-blink: a hue-difference-based light signal marker for large screen interaction via any mobile terminal. In: Proceedings of the UIST 2004, pp. 147–156. ACM (2004)
9. Moeller, J., Kerne, A.: Zerotouch: an optical multi-touch and free-air interaction architecture. In: Proceedings of the CHI 2012, pp. 2165–2174. ACM (2012)
10. MultiTouch Ltd., Multitaction codice. http://www.multitaction.com/products/software/codice/
11. PQ Labs. G4 multi-touch screen. http://multitouch.com/product_g4.html
12. Schmidt, D., Chehimi, F., Rukzio, E., Gellersen, H.: Phonetouch: a technique for direct phone interaction on surfaces. In: Proceedings of the UIST 2010, pp. 13–16. ACM (2010)
13. Schöning, J., Rohs, M., Krüger, A.: Using mobile phones to spontaneously authenticate and interact with multi-touch surfaces. In: Proceedings of the Workshop on Designing Multitouch Interaction Techniques for Coupled Public and Private Displays (2008)