SpareEye: Enhancing the Safety of Inattentionally Blind Smartphone Users

Klaus-Tycho Foerster
Distributed Computing Group
ETH Zurich, Switzerland
foklaus@ethz.ch

Alex Gross
Distributed Computing Group
ETH Zurich, Switzerland
grossal@student.ethz.ch

Nino Hail
Distributed Computing Group
ETH Zurich, Switzerland
hailn@student.ethz.ch

Jara Uitto
Distributed Computing Group
ETH Zurich, Switzerland
juitto@ethz.ch

Roger Wattenhofer
Distributed Computing Group
ETH Zurich, Switzerland
wattenhofer@ethz.ch

ABSTRACT

Using mobile phones while walking for activities that require continuous focus on the screen, such as texting, has become more and more popular in the last years. To avoid colliding with obstacles, such as lampposts and pedestrians, focus has to be taken off the screen in regular intervals. In this paper we introduce SpareEye, an Android application that warns the smartphone user from obstacles in her way. We use only the camera of the phone and no special hardware, ensuring that it requires minimal effort from the user to use the application during everyday life. Experimental results show that we can detect obstacles with high accuracy, with only some false positives and few false negatives.

Keywords

Obstacle avoidance; visual distraction; mobile devices.

Categories and Subject Descriptors

C.3 [**Special Purpose and Application-Based Systems**]: Real-time and embedded systems.

1. INTRODUCTION

Mobile phones are no longer exclusively used as telephones, but for a multitude of different applications. Whether it be web browsing, sending short messages, using Twitter or checking Facebook, these activities require the user to focus on the screen of their phone instead of their surroundings. If one looks around at a crowded place, very often one will see people that are only paying attention to their phone while walking – and only look up from time to time to check that they will not crash into anything. It has been shown that using a phone, even for simple activities, causes inattentional blindness [6]. While this can lead to amusing scenes for the bystanders and to popular youtube videos, severe injuries can occur for the people that are tripping, falling down, or colliding with someone while focussing only on their smartphones. According to a recent study, the number of pedestrians in the US who visited the emergency room in 2008 due to phone-related accidents increased fourfold compared to 2006 [9].

MUM '14, November 25–28 2014, Melbourne, VIC, Australia
Copyright © 2014 ACM 978-1-4503-3304-7/14/11 ... $15.00.
http://dx.doi.org/10.1145/2677972.2677973

Most of the previous work tackled the problem by equipping the user with special hardware that detects obstacles. As an example, CrashAlert uses a Microsoft Kinect to get a depth map from the area in front of the user [5]. However, all approaches with extra hardware suffer from same the drawback: the user has to take special hardware with her that she normally would not carry along, requiring extra effort and monetary investment.

Why not use something that the people that are focussing on their smartphone already have and are using – the smartphone? In this paper we introduce SpareEye, an app that uses the camera of a smartphone to provide users with more safety for texting while walking. SpareEye detects obstacles coming towards the phone and notifies the user so that she can avoid them. We provide an algorithm that first separates obstacles in an input image from background and estimates their relative distance to the user. If the distance to an obstacle decreases significantly between consecutive frames, the user is given an auditory and a tactile warning. The application is designed to run in the background while the user is focusing on other tasks.

We did a field test with 21 participants, where we show that our system works in a real life environment and can be used to improve the safety of people focussing on their smartphones while walking.

2. RELATED WORK

A common approach to detect obstacles is to use technology that can detect the depth of field in front of the user. In [15], the authors use an array of custom-built ultrasound sensors combined with a vibration jacket and an ARM9 based embedded system to help visually impaired people to detect and avoid obstacles. They note that they can detect obstacles in front of the user well, but not drop-offs like staircases. Samsung recently acknowledged the importance of obstacle detection for visually impaired users and announced a cover for one smartphone model that uses ultrasound to detect obstacles up to two meters away [14]. Earlier projects for mobile obstacle detection via ultrasound were already started decades before, e.g., in 1991 [7].

The release of the Microsoft Kinect as an infrared-laser with a CMOS sensors for depth-information has spawned projects with similar goals, like [2] for visually impaired people and CrashAlert to enhance peripheral alertness when using mobile devices while walking [5]. In [5], a Kinect is attached under a tablet and connected to a laptop, which then sends information via bluetooth to the phone. Their application assists the user in avoiding obstacles while focussing on the screen by informing about the obstacles on the display. Hincapié-Ramos and Irani note in [5] that "*technological support for safer walking and multi-tasking is at large unexplored*", and that "*CrashAlert is one of the first explorations of a safety-aware*" walking user interface. By practical evaluation, they show that CrashAlert leads to safer walking.

With head-mounted displays, one can display upcoming obstacles as virtual objects within the display. In [11], the authors use edge detection to display obstacle information on a pair of glasses, intended to use for patients with tunnel vision. With the upcoming availability of Google Glass [4], more research might be conducted in this area for private everyday use, as used, e.g., by aviators [13].

The difference between our work and the works mentioned above is that they use dedicated hardware, whereas we do not.

Another approach is to enhance the interface of applications correlated to the task the user is doing in the background, in this context also known as walking user interfaces (WUIs), and to study the effects of walking on pressure-based interaction [19]. A naïve way is to display the image of the rear-facing camera in the background of the application, but this requires modified versions of each desired application. Furthermore, it does not work well in practice according to [10], due to people not being able to process too much information while multitasking.

In [8], the authors study dynamically changing button sizes as the user moves and suggest user-specific personalization after their evaluation. Their work relates to ours with the goal of simplifying the use of smartphone apps while walking. However, they only focus on enhancing the user interface, and do not investigate the detection of obstacles. A different method is to design a user interface that can be used without looking at the screen of the phone at all [1, 18].

WalkSafe [17] uses the camera of a smartphone to warn the user from incoming danger. However, it is only applied to a very specific scenario, the detection of cars when crossing a road, it requires the camera to point towards the car. The authors use machine learning to differentiate cars from empty roads and try to warn the user if danger is imminent. Our solution is designed to work in a more general setting.

In [12], a smartphone-based pixel scanning scheme is used to detect obstacles. The authors divide the image into three regions (center, left, and right) and calculate the safe depth for each region. The main difference to our work is that they do not track obstacles over time, but just look at one image. As such, they only warn when an obstacle is deemed close enough. Our approach anticipates the obstacles movement vector: e.g., the obstacle might move out of the image, even though it is close, or the object will move quickly towards the user, even though it is far away.

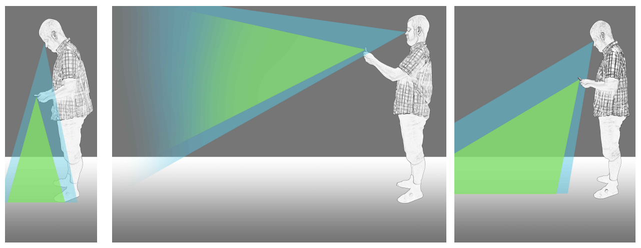
(a) 0° (b) 90° (c) 45°

Figure 1: The assumed way for a person to hold the phone is shown in Figure 1(c). Then, the field of view provides us enough information about the path in front while leaving out far away details, such as landscape, that we are not interested in.

We would like to note that obstacle detection for blind persons differs from obstacle detection for inattentionally blind users. The first such electronic aids have been available since the 1960s [16]. We refer to [3] for a comparative survey of portable obstacle detection systems for blind people.

3. FIELD OF VIEW

In general, we do not make any assumptions regarding the environment where our application is used, besides that there has to be a proper video stream available. E.g., we can not do much if it is too dark to see the obstacles or if the vision is otherwise obscured.

Besides this "sanity" assumption, our solution requires that the user holds the phone in an angle of approximately 45 degrees (where 90 degrees indicates that the screen is parallel to the body of the user) and that the camera is pointing towards the destination to where the user is walking. We point out that it is rather straightforward to train a user to hold the phone like this. In addition, the accelerometer can be used to warn the user if the angle of the phone differs too much from the required 45 degrees. The desired way to hold the phone is illustrated in Figure 1(c).

For our purposes, holding the phone incorrectly yields two difficulties. First, if the phone is pointed towards the ground (see Figure 1(a)), we can not see the objects coming towards the user early enough to give a warning. Second, if the phone was pointed towards the horizon (see Figure 1(b)), the image would contain a lot far away objects and the recognizing task becomes a lot harder. Therefore, it is crucial for our approach that the field of view extends only and at least up to a few meters.

4. OBSTACLE TRACKING

The general idea of our approach is to first detect the background of an image. The second step is to detect areas in the image that significantly differ from the background and estimate the distance from the user to each area. If an area is continuously and quickly getting closer to the user, the user is warned of a possible collision. Our tests were performed using a Samsung Galaxy Nexus, which manages around four frames per second while running our application. For simplicity, we assume in the description of our algorithm that

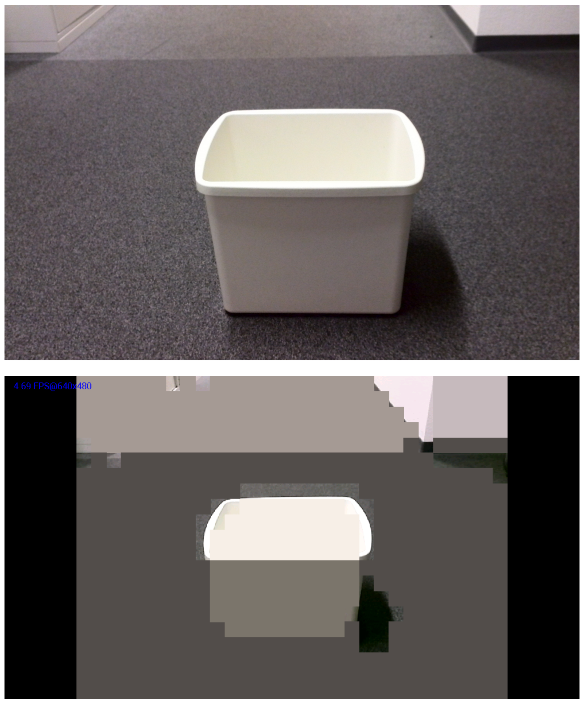

Figure 2: The input image on the top is transformed into super pixels and the result is shown on the bottom image. The areas are depicted as a set of super pixels colored by the average color value in the corresponding area. If a super pixel belongs to an area with less than 13 super pixels, the pixels within the super pixel are drawn as in the original image.

the time delay between two successive frames is a constant.

The basic building blocks of our algorithm are blobs in an image. A blob corresponds to an area in an image where the color deviation is relatively small. To find blobs from the image, we partition the input image into small disjoint super pixels of equal size to provide efficient processing. For our purposes, we observed that choosing a super pixel count of approximately 600 (20×30) super pixels provides good detection rate while preserving a reasonable frame rate. The color values of the super pixel correspond to the average values of the color values of the pixels within the super pixel.

To decide whether two adjacent super pixels s_1 and s_2 belong to the same blob, we compare the color values of s_1 and s_2 and if the values differ for less than a threshold, we consider the super pixels similar. Similar super pixels are then merged into a blob. To avoid detecting blobs from the noise in the image, we first use a Gaussian filter to blur the image and in addition, we discard all blobs with less than 13 super pixels in them. Dividing an image into areas is illustrated in Figure 2.

4.1 Obstacles and Background

As the first step towards detecting objects that the user might run into, we investigate which areas in the image correspond to obstacles that the user might collide with. The first observation is, that since we only have vision towards one direction, we ignore objects coming from sides and focus on objects in front of the user. We focus on the case where the object is initially relatively far from the user and we can detect it coming towards the user. In other words, if the user walks into an object that is right next to her when starting the application, or if the user walks into an object right after turning, we do not consider this as a failure of the system.

From the previous assumptions, we get into the most crucial assumption in our work. We assume that if an area in the image touches the bottom line of the picture, i.e., is next to the user, we consider it as part of the background. As depicted in Figure 2(a), in a usual setting, the nearby floor area is considered as a large background area and an anomaly, the white trash bin, is tracked. We show in our experiments that we are able to detect obstacles with high accuracy and that false positives occurred by patterns of the floor or noise in the image are relatively rare.

4.2 Tracking Areas

One of the most important attributes of a blob is the center of mass that is the average of the coordinates of the pixels contained in the blob. We use this attribute to detect the movement of an area in successive images. The basic idea is that with a high enough frame rate in the video stream, an object does not move much between successive images and therefore, we can identify same areas in successive pictures. Another assumption required here is that the lightning conditions do not change much between two successive images. From a practical perspective, this means that after turning on or off some lights in a room, the system needs a few frames to adapt and therefore the change in the lightning conditions might cause a false alarm.

In more detail, for two successive images I_1 and I_2, we compute the areas in the image and the centers of mass for each area. Then, we compare the average color values of each area in I_1 to each area in I_2 and whenever there is a difference of at most Δ units in the average values for each color channel, we add the pair of areas in a set of candidate matches. Then, if the distance is at most d, these areas are considered a valid match which indicates that they are the same area. According to our empirical tests, setting d to approximately one fourth of the screen width provides reliable matchings. In the case of multiple possible matches, each area is paired at most once in a greedy manner, which indicates that a new area is paired to the first applicable existing area that is found. The tracking and pairing of the areas is depicted in Figure 3.

4.3 Warning

The final step of the tracking process is to decide whether or not a tracked obstacle will collide with the user. As mentioned before, we are only interested in obstacles that the user is approaching or vice versa. Therefore, we measure the distance from each object o that is tracked in the image to the low edge of the image. Then we can measure the difference in the distances of object o in successive images. A high difference in the distances indicates that either object o is approaching really quickly or it is already quite close and even a relatively slow approaching speed results in a significant difference in the distance.

In addition, we do not have to care about objects passing the user without colliding. In other words, if the walking path of the user does not cross the path or location of the obstacle, no warning is needed. We model the movement of

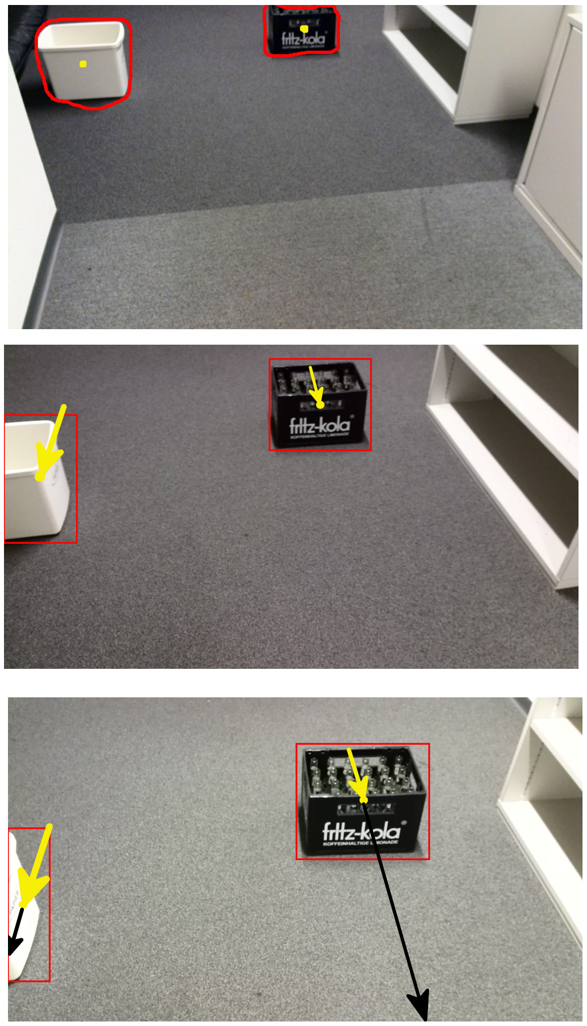

Figure 3: The yellow arrows depict the movement of the centers of masses of the objects and the black arrows correspond to the point where the object will go out of the field of view. The tracked objects are denoted by a red rectangle and their centers of mass are shown with a yellow circle. Since the movement of the thrash bin is going left of the picture, we consider this object as harmless. The general direction of the fritz-kola case is towards the user and therefore a warning is thrown.

the objects as the average movement speed within the last four frames. We assume that an obstacle collides with the user if it crosses the x-axis in the image. From the direction of the movement, we can first derive if the obstacle will hit the user by maintaining its direction. Next, from the speed, we can derive how quickly the user will hit the obstacle. An obstacle is considered dangerous and a warning is given if the obstacle is calculated to collide within the next two seconds.

When a need for a warning is detected, the application plays a loud siren sound and vibrates. This makes our application suitable for both visually impaired and hearing impaired users.

5. EVALUATION

We performed an evaluation of SpareEye to show how well it can detect real-life obstacles in an everyday setting. We chose a university cafeteria, where the ground was flat, various tables and chairs, large plants, trashcans, and pillars were present as obstacles, requiring multiple turns to reach the destination (cf. Figure 4). The participants (18 male, 3 female, mean age of 29.4 years) all started the walk on the same spot and had to walk to a defined endpoint about 25 meters away. No other people were present while testing.

Due to the nature of a testing setup, the participants knew that obstacles would be present in their path. Since we wanted the participants to walk without checking for obstacles in their peripheral vision/looking up from the screen, we blindfolded them. As such they could not avoid crashes by paying more attention to their surroundings, which they would not do in a everyday situation [6].

When SpareEye warned them, they were allowed to lift the blindfold, assess the situation, and continue walking blindfolded. The participants were instructed to hold the phone roughly in an angle of 45 degrees, as described in Figure 1, and not to put their finger over the lens of the camera.

5.1 Results

We begin listing our results by emphasizing that due to the nature of our application, our main goal was to minimize the number of false negatives, i.e., that the user bumped into an object without getting a warning. The high sensitivity of the app yielded a rather high count of false positives, i.e., that the user was warned without actual danger being present. On average, the app threw a false warning every 1.3 minutes. However, this is a relatively long time taking into account that the effort needed from the user is only to check what is right in front of her.

In total, SpareEye warned 103 times in 21 experiments, with 87 true positives ($avg = 4.1$, $sd = 1.8$, $med = 4$) and 16 false positives ($avg = 0.8$, $sd = 0.8$, $med = 1$). In 6 cases ($avg = 0.3$, $sd = 0.5$, $med = 0$), SpareEye failed to warn from an obstacle, meaning the participants bumped into something. The main cause for false positives were changing light patterns on the floor, where SpareEye recognized these large spots as dangerous objects. The false negatives were induced by the application classifying objects as part of the ground, and therefore being not dangerous. This happened when the participants were walking towards an object at a very narrow angle, i.e., the object was almost parallel to the user.

Each walk lasted between 26 and 97 seconds ($avg = 62.2s$, $sd = 21.9s$, $med = 58s$). Most participants chose to walk along roughly the same route, which can be seen in Figure 4. All participants also seemed to trust the app: None of them stopped or tried to lift the blindfold without receiving a warning first or walking into an obstacle. Furthermore, they did not seem to walk in a cautious manner or tried to detect obstacles with their feet or hands, despite being blindfolded.

All participants were asked after the walk if they text while walking (12 yes, 9 no), and if yes, approximately how often they take their focus off the screen usually ($avg = 1.8s$, $sd = 1.4s$, $med = 1s$). We also asked these participants on a 5-point scale from one (never) to five (always/definitely) if they would use the system ($avg = 3.4$, $sd = 1.1$, $med = 4$) and recommend it to others ($avg = 3.8$, $sd = 0.6$, $med = 4$).

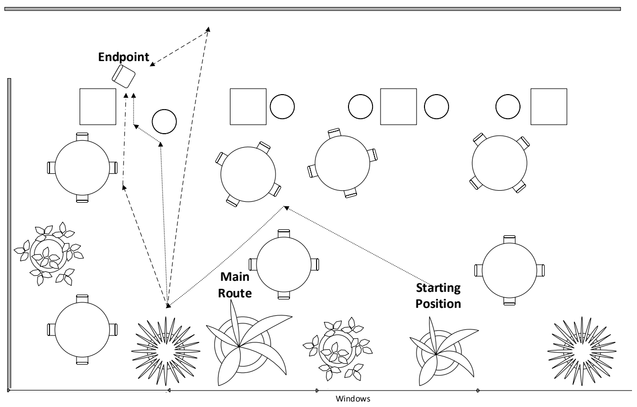

Figure 4: Most participants chose to walk roughly along the route labeled as main route, a few went along the dashed lines.

A few participants noted that they would like the app to be 100% reliable before using it.

6. CONCLUSION

In this paper we presented SpareEye, a smartphone app that can warn users about upcoming obstacles. We perform this task without any additional hardware, i.e., using only the video stream of the smartphone. Our main contribution is a proof of concept that one can use the video stream of a smartphone to estimate the change in the distances of obstacles in the field of view without certainty of the real distances to those obstacles.

Our experiments show that we can reliably detect dangerous obstacles in a real world scenario. The experiments were conducted in a university cafeteria, where the setting was designed s.t. the users would walk into obstacles by simply walking along a straight line. Our method was able to detect obstacles with a small false negative rate, i.e., the app only rarely left an obstacle unnoticed.

Our results combined with the observations from [5] indicate that modern smartphones could provide visual feedback about dangerous situations to users that are focusing on the device and not paying attention to the surroundings. By running the warning application in the background, the users could feel safer while using their phones without a negative effect on their user experience.

7. REFERENCES

[1] N. Banovic, K. Yatani, and K. N. Truong. Escape-keyboard: A sight-free one-handed text entry method for mobile touch-screen devices. *IJMHCI*, 5(3):42–61, 2013.

[2] D. Bernabei, F. Ganovelli, M. D. Benedetto, M. Dellepiane, and R. Scopigno. A low-cost time-critical obstacle avoidance system for the visually impaired. In *IPIN*, 2011.

[3] D. Dakopoulos and N. Bourbakis. Wearable obstacle avoidance electronic travel aids for blind: A survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 40(1):25–35, Jan 2010.

[4] Google. The Glass Explorer Program. http://www.google.com/glass/start/how-to-get-one/, 2014. Last accessed: 2014-10-22.

[5] J. D. Hincapié-Ramos and P. Irani. Crashalert: enhancing peripheral alertness for eyes-busy mobile interaction while walking. In *CHI*, pages 3385–3388. ACM, 2013.

[6] I. E. Hyman, S. M. Boss, B. M. Wise, K. E. McKenzie, and J. M. Caggiano. Did you see the unicycling clown? Inattentional blindness while walking and talking on a cell phone. *Applied Cognitive Psychology*, 24(5):597–607, 2010.

[7] T. Ifukube, T. Sasaki, and C. Peng. A blind mobility aid modeled after echolocation of bats. *IEEE Trans Biomed Eng*, 38(5):461–5, 1991.

[8] S. K. Kane, J. O. Wobbrock, and I. E. Smith. Getting off the treadmill: evaluating walking user interfaces for mobile devices in public spaces. In *MobileHCI*, pages 109–118. ACM, 2008.

[9] S. M. Lopresti-Goodman, A. Rivera, and C. Dressel. Practicing safe text: the impact of texting on walking behavior. *Applied Cognitive Psychology*, 26(4):644–648, 2012.

[10] E. Ophir, C. Nass, and A. D. Wagner. Cognitive control in media multitaskers. *Proceedings of the National Academy of Sciences*, 106(37):15583–15587, Sept. 2009.

[11] E. Peli, G. Luo, A. Bowers, and N. Rensing. Applications of augmented-vision head-mounted systems in vision rehabilitation. *Journal of the Society for Information Display*, 15(12):1037–1045, 2007.

[12] E. Peng, P. Peursum, L. Li, and S. Venkatesh. A smartphone-based obstacle sensor for the visually impaired. In *UbiComp*, pages 590–604, 2010.

[13] C. Rash. *Helmet Mounted Displays: Design Issues for Rotary-wing Aircraft.* Press Monographs. SPIE Press, 1999.

[14] Samsung Mobile Press. Samsung Expands Galaxy Core Advance Experience with Specialized Usability Accessories. www.samsungmobilepress.com/2014/03/14/Samsung-Expands-Galaxy-Core-Advance-Experience-with-Specialized-Usability-Accessories-1, March 2014. Last accessed: 2014-10-22.

[15] B.-S. Shin and C.-S. Lim. Obstacle detection and avoidance system for visually impaired people. In *HAID*, pages 78–85. Springer-Verlag, 2007.

[16] J. Terven, J. Salas, and B. Raducanu. Computer vision systems for visually impaired people. *Computer*, 99(PrePrints):1, 2013.

[17] T. Wang, G. Cardone, A. Corradi, L. Torresani, and A. T. Campbell. Walksafe: a pedestrian safety app for mobile phone users who walk and talk while crossing roads. In *HotMobile*, pages 5:1–5:6. ACM, 2012.

[18] Y. Wang, C. Yu, J. Liu, and Y. Shi. Understanding performance of eyes-free, absolute position control on touchable mobile phones. In *MobileHCI*, pages 79–88. ACM, 2013.

[19] G. Wilson, S. A. Brewster, and M. Halvey. The effects of walking and control method on pressure-based interaction. In *CHI*, pages 2275–2280. ACM, 2011.