PolarTrack: Optical Outside-In Device Tracking that Exploits Display Polarization

Roman Rädle[1] **Hans-Christian Jetter**[2] **Jonathan Fischer**[3] **Inti Gabriel**[3] **Clemens N. Klokmose**[1]
Harald Reiterer[3] **Christian Holz**[4]
[1]Aarhus University [2]University of Applied Sciences Upper Austria [3]University of Konstanz
[4]Microsoft Research
{roman.raedle,clemens}@cc.au.dk, hans-christian.jetter@fh-hagenberg.at,
{jonathan.fischer,inti.gabriel,harald.reiterer}@uni-konstanz.de, cholz@microsoft.com

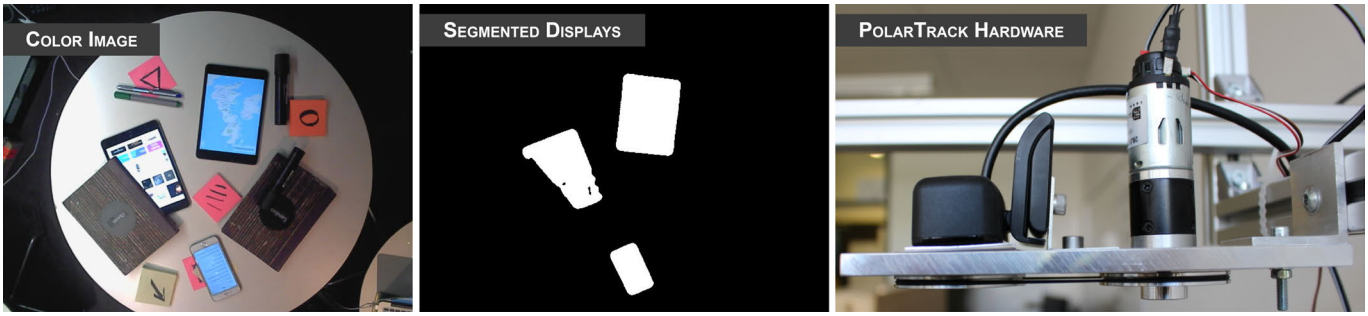

Figure 1. PolarTrack is an optical tracking system for mobile devices that combines an off-the-shelf RGB camera with a rotating linear polarization filter mounted in front of the lens. PolarTrack exploits the use of polarized light in current displays to segment device screens in the camera feed from the background by detecting periodical changes of display brightness while the linear polarizer rotates.

ABSTRACT

PolarTrack is a novel camera-based approach to detecting and tracking mobile devices inside the capture volume. In PolarTrack, a polarization filter continuously rotates in front of an off-the-shelf color camera, which causes the displays of observed devices to periodically blink in the camera feed. The periodic blinking results from the physical characteristics of current displays, which shine polarized light either through an LC overlay to produce images or through a polarizer to reduce light reflections on OLED displays. PolarTrack runs a simple detection algorithm on the camera feed to segment displays and track their locations and orientations, which makes PolarTrack particularly suitable as a tracking system for cross-device interaction with mobile devices. Our evaluation of PolarTrack's tracking quality and comparison with state-of-the-art camera-based multi-device tracking showed a better tracking accuracy and precision with similar tracking reliability. PolarTrack works as standalone multi-device tracking but is also compatible with existing camera-based tracking systems and can complement them to compensate for their limitations.

CHI 2018, April 21–26, 2018, Montreal, QC, Canada

ISBN 978-1-4503-5620-6/18/04…$15.00

DOI: https://doi.org/10.1145/3173574.3174071

ACM Classification Keywords

H.5.2. Information Interfaces and Presentation (e.g. HCI): User Interfaces; Input devices and strategies

Author Keywords

display polarization; cross-device; multi-device; spatial interaction; tracking technology.

INTRODUCTION

Locating and tracking devices in 3D-space is an essential component of systems that enable *proxemic* [1, 5, 22] or *spatially-aware cross-device interactions* [13, 24, 25, 30]. While many different approaches for sensing device locations and orientations have been proposed [6, 10, 16, 22, 24, 31], each solution has specific advantages and limitations. An ideal tracking technology continuously tracks devices with sub-centimeter accuracy without requiring markers, additional software or hardware on the tracked devices, needs no or only a minimal setup of tracking hardware in the environment, and is computationally inexpensive. Our work is a step towards such a tracking system by leveraging rotating polarizing filters in front of the camera for optical device segmentation.

We present PolarTrack, a low-cost approach for detecting and tracking mobile device displays using an off-the-shelf RGB camera and a linear polarizing filter that rotates in front of its lens. By exploiting basic laws of physics, PolarTrack can reliably differentiate between physical objects and mobile displays in a video stream using a computationally inexpensive

algorithm. It introduces a new approach to optical device tracking that works with high precision, accuracy, and reliability as a stand-alone solution, but could also greatly improve existing optical tracking approaches.

PolarTrack's contributions include:

- a novel approach for display segmentation in a camera feed based on periodical changes of display brightness resulting from a linear polarizer rotating in front of a camera lens
- a camera-based tracking system that segments displays from the background with minimal processing,
- an inexpensive algorithm for device detection and tracking,
- a technical evaluation of PolarTrack's tracking quality, showing higher accuracy compared to a state-of-the-art open source system [24], and
- a low-cost approach to building a PolarTrack system around a Kinect V2 camera from off-the-shelf components.

We conclude this paper with a discussion of our approach and highlight how PolarTrack can complement existing tracking solutions to address their limitations.

RELATED WORK

Research on cross-device interaction is popular in HCI with use cases such as collaborative photo-sharing [19], brainstorming [20], around the table collaboration [24], content curation [2], map navigation [6, 24], annexing devices to extend display space or building large tiled displays [8, 9, 24, 26], sensemaking for individuals [7] or groups [30, 32], active reading [3], and visual analytics [23]. Beyond sensing presence and approximate distances, recent research increasingly focuses on spatially-aware systems [13, 19, 20, 22, 24, 30] that sense absolute device locations and orientations for seamless cross-device interactions, e.g., flicking content between devices or cross-device "pick, drag, and drop" [24, 25].

Our work mainly draws from two topics in HCI research: First, it is related to technologies for camera-based tracking of device locations and orientations. Second, it exploits the principles of light polarization, an approach that has been pursued in HCI before, although not for tracking purposes.

Camera-based Sensing of Mobile Devices

Camera-based sensing often follows one of the following two strategies: *inside-out* and *outside-in* tracking.

Inside-out tracking of mobile devices typically uses the front-facing camera to visually detect features in the environment that can serve as external points of reference. For example, Li & Kobbelt use a single marker (e.g., on a room's ceiling) that is detected by multiple devices to determine their relative positions to each other to form a tiled display [16]. Similarly, Grubert et al. use an individual's head as a shared point of reference for spatial registration of multiple devices for ad hoc multiple display environments [6].

Outside-in tracking (such as PolarTrack) uses one or multiple cameras to continuously observe a capture volume in which the tracked devices must be visible. It typically uses computer vision algorithms to detect markers or to segment devices from their surroundings in order to compute their absolute spatial location and orientation. One of the key challenges of outside-in approaches is therefore to accurately and reliably identify which and where devices are present in the capture volume, even if there are many other non-device objects present.

Many commercial (e.g., OptiTrack or VICON motion capturing systems) or open source solutions (e.g., ARToolKit) use fiducial markers attached to users or devices. These markers carry an easily identifiable optical pattern or configuration in the IR or visible range. For developers, marker tracking is comparably easy to implement and is therefore also popular in HCI research (e.g., [21, 30]), but it has the disadvantage of requiring users to attach markers to devices which does not fit well into ad hoc cross-device interaction scenarios. The same is true for active markers or beacons, for example, those for controller tracking with the VIVE virtual reality headset. While they offer precise tracking and great responsiveness, the VIVE beacons are too large for most mobile device tracking scenarios.

Since mobile devices alone offer few distinct features in the visual range to clearly segment them from their surroundings, some approaches therefore combine different sensing approaches and targets: GroupTogether uses ceiling mounted depth cameras to identify and track users and radio triangulation to identify their handheld devices [22]. Wu et al.'s EagleSense uses depth images of the shape of users holding devices to track people and devices in interactive spaces [31].

However, in some scenarios, devices need to be tracked even when they are not held by users, e.g., when they are put on a table. Rädle et al.'s HuddleLamp uses both the IR and RGB images from a depth camera in a desk lamp to track multiple mobile devices on a table [24]. Devices are initially recognized in the RGB image by fiducial markers flashed on their screens and from then on, the rectangular shape of their physical screens is tracked in the IR image. This is possible because mobile screens generally have a much lower IR reflectance than other everyday objects, so that their shapes become clearly visible in a binary IR image after thresholding, regardless of what they display or ambient visible light. Limitations of this approach arise from the noisy and very low resolution IR image (especially in comparison to state-of-the-art RGB cameras), a circular blind region in the center of the camera (the so-called Sauron's eye), and the computationally expensive tracking of devices necessary to avoid false positives for non-screen rectangular objects with low IR reflectance [24]. Furthermore, HuddleLamp's tracking fails if mobile displays are positioned too close together, so that there is no visible bezel in the low-resolution IR image. However, HuddleLamp's approach of using simple optical properties (e.g., IR reflectance) instead of sophisticated computer vision to differentiate screens from other objects has been very influential for PolarTrack and also drove us to use HuddleLamp as a benchmark in our evaluation.

CrossMotion implements the opposite approach and continuously compares motion in a Kinect V2 depth feed with the inertial motion reported by mobile devices [29]. After initial calibration of coordinate systems, CrossMotion analyzes the 3D optical flow in the depth images and matches it with the

mobile inertial and magnetometer data. The main benefit of CrossMotion is a single-camera tracking system that is independent of line-of-sight, making it robust to occlusion and not limited by the number of devices.

Uses of Polarizing Filters in HCI

Previous uses of polarization in HCI were primarily aimed at selectively controlling the visibility of screen content for different cameras or users by using parallel or orthogonal linear polarization between light source and viewer. One of the earliest examples for the use of such polarization filters is Tang & Minneman's VideoDraw system for video-based remote collaborative drawing [28]. Nearly orthogonal linear polarizing filters were used in front of cameras and screens to solve the problem of video feedback, i.e., to make local screen content invisible in a camera image that should only transmit images of users and their interactions in front of the screen to the remote location. Similarly, Küchler & Kunz used linear polarizing filters in Collaboard to segment a human who is standing in front of a screen from the surrounding screen content and to use this for an embodied representation at the remote location [15].

Further applications include single display privacyware [27], in which specific regions of a single screen become visible or invisible to different users by either using frame interleaving (in combination with synchronized shutter glasses) or, alternatively, orthogonal polarization. Lissermann et al.'s Permulin uses the former and displays two different user-specific images on the same physical 3D screen for improving tabletop collaboration [18]. Karnik et al.'s PiVOT uses the latter and combines polarized and unpolarized light sources with modified liquid crystal (LC) displays to create personalized view overlays for tabletops [11]. In subsequent work, Karnik et al. present MUSTARD, a multi-user see-through AR display that uses polarized content and modified LC panels to show user-specific information [12]. The principles of polarization are also exploited in the polarizing adjustment layer that Lindlbauer et al. use to create a dual-sided see-through display with controllable transparency [17].

Finally, polarization has also been used by Koike et al. to create transparent 2D markers for LCD tabletop systems [14]. They selectively cancel image blocking between a LC display and camera by using two half-wave plates from which markers can be constructed.

In summary, polarization has been used as a method to selectively control visibility of screen content for different observers but it has not been used to detect and segment mobile screens from their surroundings or other objects in a video stream before. To explain how PolarTrack exploits polarization for this purpose, we describe its implementation and briefly summarize the underlying optical principles in the following.

POLARTRACK'S IMPLEMENTATION

Our implementation consists of three parts: (1) the polarization characteristics of current display technologies that PolarTrack exploits, (2) PolarTrack's hardware setup, and (3) our software processing pipeline to detect and track devices over time.

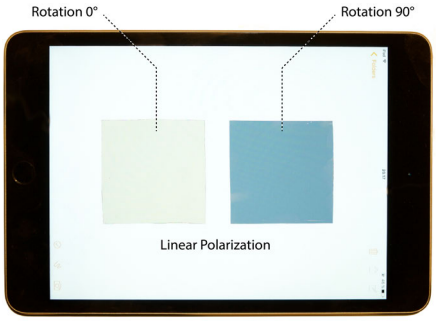

Figure 2. An Apple iPad Mini with two linearly polarized filters put side-by-side on top of the screen. The filters are rotated 90° to each other.

Polarization characteristics of current displays

A common characteristic of current display technologies is their luminous emittance of polarized light. The reason is that each display integrates a polarizing layer or has a polarizing film on top of the screen. The polarizer only allows light in a particular polarized direction (i.e., linear, circular, or elliptical polarization) to pass and blocks light of other directions respectively. Displays either integrate the polarizer due to a technical requirement (i.e., for liquid crystal displays) or add it to improve "contrast and robustness" [4] and to reduce light reflections on the screen. For example, it reduces reflections caused by external light sources such as sunlight or room illumination. Examples for displays that have a polarizer include liquid crystal displays (LCD), active matrix organic light-emitting diode displays or derivatives (e.g., OLED, AMOLED or Super AMOLED displays), and even future display technologies such as bendable displays that only replace glass with a polyethylene (plastic) substrate (i.e. POLED). In consequence, the light emissive elements (pixels) of a display emit polarized light. The polarization of display pixels is noticeable when two linear polarization filters are put side-by-side on top of a display and rotated 90° to each other (see Figure 2).

However, sunlight, light from room illumination, or light reflected from everyday objects (e.g., tables, walls, or cups) is usually unpolarized. Figure 3 illustrates polarized light emitted by a display and unpolarized light from the background environment. Both pictures show the same scene and contain a table, everyday objects, and a display. They are taken with the same camera (Depth Senz3D) and with a linear polarization filter in front of the camera. In the first picture, the polarization filter is at 0° rotation and in the second picture the filter is at 90° rotation. As a result, they show a change in the light intensity in those areas in the pictures where the display is while the surrounding background stays (almost) unaffected. It is noteworthy that this effect is visible for all state-of-the-art display technologies. In the case of LCDs which typically contain linear polarizers, a rotating linear polarization filter results in visible changes of brightness with almost complete disappearance of the screen image in orthogonal positions. For more recent technologies, with non-linear polarizers, a rotating linear polarizer may not entirely block the emitted light, but still results in visible changes in both brightness and color spectrum. In PolarTrack, we exploit the changing

Figure 3. Both pictures show an image of the same scene taken by a camera with a polarization filter. The rotation of the polarization filter is 0° in the first picture and 90° in the second picture.

light intensity to detect displays in a RGB image stream and segment them from the background.

PolarTrack's hardware setup

For tracking steady displays in a capture volume, it is sufficient to take two consecutive images at perpendicular polarizer positions. For example, at angles of 0° and 90°, 90° and 180°, 180° and 270°, or 270° and 360°. However, displays may be oriented in arbitrary directions or may even be subject to rotation during tracking. In PolarTrack, the polarizer is therefore continuously rotated while capturing images to compensate for both.

We built custom hardware that assembles an RGB camera, a polarizer, and a gear motor with an encoder (see Figure 1 PolarTrack Hardware). The camera was mounted on top of an aluminum plate. For our experiments, we used a Creative Senz3D RGB-D. This is the same camera model that was used for HuddleLamp [24]. We opted for this camera to allow comparison to HuddleLamp's hybrid sensing. The plate integrated a ball bearing with low friction. We used a single-row groove ball bearing with an inner diameter of 65 mm, an outside diameter of 100 mm, and a thickness of 18 mm. A aluminum ring was put in the opening of the ball bearing that fixates the polarizer and provides a groove. A rubber belt was set into the groove to transfer the movements from the motor to the ball bearing, effectively rotating the polarizer. We used a 12V gear motor 0.2 kg-cm/1080 RPM 3.7:1 DC with an encoder. With four possible perpendicular polarizer positions ($\alpha + 90°$) for every rotation and 18 rotations per second, the chosen motor allows for cameras up to a maximum of 72 fps.

Pre-processing pipeline for display detection

Our display segmentation algorithm relies on light intensity or color changes between two consecutive RGB images. Therefore, the first step in our display detection processing pipeline subtracts the current image I_t from the previous image I_{t-1}:

$$I^D(u) = |I_{t-1}(u) - I_t(u)|$$

where u defines an image pixel of an image at frame t. It results in a differential image I^D with absolute differences of pixels between the two input images, and for the R, G, and B color channels. Lastly, the image is converted from RGB color space to a grayscale image I^{DG}.

Rolling Shutter Effect

Ideally, this differential grayscale image contains pixel values with zero difference for the background and non-zero pixel values where a change happened, either through change of light intensity or color changes of pixels. However, due to the camera's rolling shutter and the continuous rotation of the polarization filter, the camera might have captured the pixels of an image at different times even though all of them are considered to be part of frame t. Consequently, this small time difference results in an image where pixels are captured with different polarizer angles, effectively resulting in a rolling shutter effect in I^{DG} (see top row of images in Figure 4)

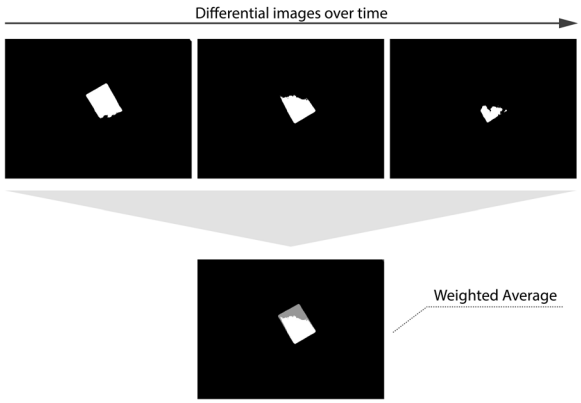

Figure 4. A sliding window approach averages the n (i.e. $n = 3$) past differential images with weights to compensate for rolling shutter effects.

While it is possible to synchronize the camera shutter with the motor rotation to capture all pixels of one image at the exact same polarizer angle, it would require special hardware. For example, a camera that accepts a generator lock (genlock) signal and captures images only when the motor pauses its rotation, or a Geneva drive that transforms continuous rotations into nearly discrete steps. Instead of adding complexity to the hardware, we opted for a software-based solution with a naïve weighted sliding window to compensate for the rolling shutter effect and to get a binary image sufficient for further processing to track the displays. The naïve weighted sliding window approach keeps a history of I^{DG} images and sums them up with a certain weight for each preceding image, and is defined as follows:

$$I^W(u) = \sum_{i=0}^{n-1} I^{DG}(u)_{t-i} \cdot \omega_i$$

where u defines the image pixel, n the size of the sliding window, and ω the weight each differential grayscale image I^{DG} is *considered* in the weighted image I^W. The weight is defined as $\omega_i = 0.5^{i+1}$. The sliding window considers n past differential grayscale images, eventually compensating for the rolling shutter effect by filling in image differences for past image frames from $t-1$ to $t-(n-1)$ (see bottom image in Figure 4).

Binary Image for Multi-Device Tracking Pipeline

The last step before in the pre-processing pipeline is the binary thresholding. The binary image is required as input for HuddleLamp's [24] tracking pipeline to identify and track multiple

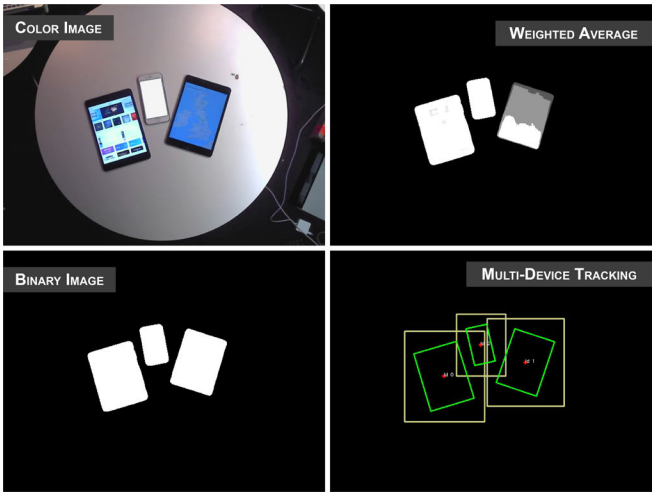

Figure 5. The PolarTrack processing steps: the color input image, the weighted average of differential images, binarization of the weighted average, and the output of the multi-device tracking.

mobile device over time. It is defined as:

$$I^B(x,y) = \begin{cases} 1 & if\ I^W(u) > \tau \\ 0 & otherwise \end{cases}$$

where I_W is the weighted image, u the image pixel, and τ a the threshold to decide if a pixel is set to 0 or 1. PolarTrack uses the binary image I^B as input for an unmodified version of HuddleLamp's open source multi-device tracking pipeline[1].

EVALUATION AND DISCUSSION

To test the technical feasibility of PolarTrack, we evaluated its tracking quality with the tracking benchmark tool used for existing mobile-device tracking systems [24][2]. The tracking benchmark tool allows for testing *accuracy*, *precision*, and *reliability* of the processed input. Further, we evaluated the tracking under different lighting conditions, and different devices and types of occlusion, like in HuddleLamp's technical evaluation.

For the evaluation, we positioned the camera to track a table surface with an area of 875 mm × 645 mm. The table was partitioned in a 3 × 5 grid with 3 rows and 5 columns resulting in 15 sampling points. The distance between two neighboring points in the grid was exactly 100 mm. The center point of the grid (i.e., the point in the 2nd row and 3rd column) was centered to the camera's view. To sample data for a condition, the center of a tablet was positioned at each grid point while the benchmarking tool captured its location and orientation for exactly 10 seconds. Then it was moved to the next grid point to capture data and so on.

Precision is defined as the mean SD of all samples at one grid point. A highly precise tracking has no deviation between the different samples. There are two components that factor into accuracy: (i) the mean and (ii) SD of the variance from the expected distance (i.e., 100 mm). The reliability is the percentage of frames in which the device was successfully detected.

[1]Huddle Engine on GitHub – https://github.com/huddlelamp/huddle-engine (last accessed September 18th, 2017)

[2]HuddleLamp's Tracking Benchmark Tool on GitHub – https://github.com/huddlelamp/tracking-benchmark (last accessed September 18th, 2017)

Device	Light (lux)	Precision mm and degrees Mean (of all SD points) X	Y	α	Reliab. % SD	Accuracy mm Mean X	Y	SD X	Y
iPad (3rd gen.)	10	.26	.28	.58	99.9	1.83	1.69	.92	.42
	200	.37	.68	.75	100.0	.76	.41	.49	.28
	380	.29	.38	.45	100.0	1.03	1.52	.69	1.01
iPad 2	10	.20	.31	.40	100.0	1.47	1.74	.98	.89
	200	.27	.34	.71	100.0	.90	.71	.56	.55
	380	.23	.32	.58	100.0	1.29	1.65	.61	1.00
[24]	1600	.78	1.05	.29	100.0	2.10	2.24	1.47	1.67

Table 1. Evaluation results of PolarTrack under different lighting conditions and with different device types, and compared to HuddleLamp's best performed tracking quality [24].

The evaluation was split into two parts. First, we evaluated the tracking quality of PolarTrack under different lighting conditions ranging from 20 lux to 380 lux and with two different device types (i.e., an Apple iPad 3rd generation and an Apple iPad 2). Second, we evaluated the tracking quality with three different types of occlusions: 1-finger, 1-hand, and 2-hands occlusion (Figure 6).

Figure 6. PolarTrack's tracking quality was evaluated with 1-finger, 1-hand, and 2-hands occlusion.

Lighting Conditions and Device Types

The results for the different lighting conditions and device types show a tracking quality with sub-centimeter precision and high accuracy. The reliability of the tracking is 100% when devices are not occluded. PolarTrack achieves better precision and accuracy when compared to HuddleLamp's best reported tracking quality [24] (compare last row in Table 1 to all other rows in the table). The reason for this is that HuddleLamp uses a relatively small IR confidence image (i.e., 283 × 159 pixels) for the device tracking, and only compensates with a much more precise tracking when devices are reliably tracked in the RGB image. PolarTracker uses a higher image resolution with 640 × 480 pixels to detect and track devices in the camera feed. The higher deviation in the angles α are eventually caused by the naïve weighted sliding window approach, which cause the devices to slightly jitter in the binary image. This can be solved with more sophisticated approaches that compensate for this jitter with a temporal component.

Unreliable tracking with occlusions

While the replication of the technical evaluation, as presented in HuddleLamp [24], results in higher precision, accuracy, and reliability for 1-finger and 2-hands occlusion, it returns unreliable results if devices are occluded with a hand (see Table 2). PolarTrack only reveals the parts of the screen that

Occlusion	Precision mm and degrees Mean (of all SD points)			Reliab. % SD	Accuracy mm Mean		SD	
	X	Y	α		X	Y	X	Y
1-finger	.28	.43	.53	100.0	1.11	.52	.53	.37
1-hand	No Reliable Tracking							
2-hands	.23	.31	.72	100.0	.95	.71	.52	.31

Table 2. Evaluation results of PolarTrack with 1-finger, 1-hand, and 2-hands occlusion. The 1-hand occlusion did not give reliable tracking results.

are not covered by non-polarized objects (e.g., hands, books, or paper). In the limitations section, we mention how this limitation can be overcome by combining it with HuddleLamp's IR confidence image tracking.

Difference from RGB-D Multi-Device Tracking

PolarTrack is a computationally *cheap* approach to segmenting displays from the background. While we use HuddleLamp's tracking pipeline [24], we replace their display segmentation step with PolarTrack's display segmentation. Unlike HuddleLamp, PolarTrack does not require processing of both an IR confidence image and an RGB image to segment devices from the background, and an additional step to fuse data from the two separate image processing steps.

Thus, PolarTrack's core difference is its alternative approach to segmenting displays from the background. However, RGB+D based multi-device tracking approaches could benefit from PolarTrack's display segmentation to (i) increase precision and accuracy due to higher resolution color images compared to lower resolution depth images and (ii) improve reliability e.g., when HuddleLamp's hybrid sensing fails due to "Sauron's eye effect" [24] (i.e. displays absorb IR light when displays are orthogonal to IR emitting LED).

Also, as discussed in the next section, PolarTrack requires minimal hardware ($5 motor and $2 polarization filter, e.g., from sun glasses) and adds on to any existing RGB camera.

Polarization at Tilted Angles

PolarTrack's display segmentation works with devices at tilted angles. While the intensity of light emitted by the display and captured by the camera decreases with tilted angles, the polarization remains the same even when devices are almost tangential to the camera (e.g., approx. 90°). Therefore, the light fluctuation of tilted displays and between two consecutive images is considerably lower. However, PolarTrack's algorithm detects periodically *oscillating* pixel intensities. Even if these oscillations have a small bit depth, PolarTrack's algorithm will reliably detect such pixels.

LOW-COST FABRICATION OF A POLARTRACK ADD-ON

In addition to our previous apparatus, we built a clip-on mount to attach PolarTrack to commodity depth-cameras, such as Kinect. As shown in Figure 7, we cut a gear construction from 4 mm acrylic with two covering plates to hold all gears in place with screws. Two stabilizing struts clip into the bottom two holes in the back plate to rest the assembly on the microphone bar of the Kinect. The half-bent top strut clamps the assembly to the Kinect. We added a small blob of hot glue to the top mount to increase the robustness of this spring-loaded clamp. An Arduino Pro Micro controls the Solarbotics Gear Motor 10 (260 rpm, 14.5 mA at 3.3 V, 2.4 in*oz torque) that is mounted against the Kinect body and that drives one of the small gears to rotate the large gear at 60 rpm. Two other small gears add rotation stability and smoothness. We cut out a piece from an off-the-shelf set of "clip-on 3D glasses" ($1.75 on Amazon) with a circular polarization filter, flipped it accordingly to expose the linear filter, and taped it onto the large gear.

Our low-cost assembly is small enough to fit on top of the Kinect's RGB camera without obstructing the infrared depth sensor. Processing both camera feeds simultaneously allows us to get continuous depth tracking while identifying display pixels from the RGB feed as described above.

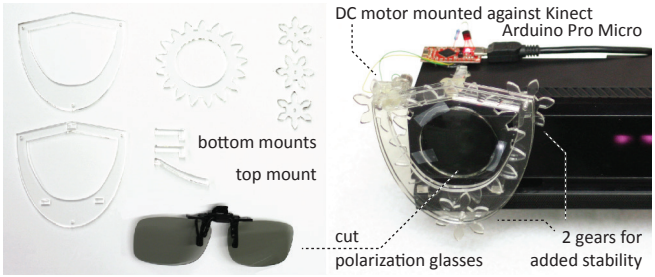

Figure 7. A low-cost implementation of PolarTrack's hardware setup: We laser-cut a gear system and mounting parts, on which we attached a piece cut from low-cost polarized "clip-on 3D glasses". An Arduino powers a small DC motor that presses against the Kinect body and drives one gear for constant rotation of the polarization filter.

LIMITATIONS & FUTURE WORK

The technical evaluation of PolarTrack showed promising results. However, it also revealed limitations by this approach, which provides opportunities for future work.

The amount of light emitted by a display plays an important role for the functioning of PolarTrack. It relies on changes in the light intensity between two consecutive images as a result of the rotation of the polarizer in front of the camera lens. However, the display does not emit any (or only minimal) light for black pixels, which leads to no changes in the differential image. In opposition, pixels that change color due to dynamic content of screens (e.g., a video playing) or movement of other objects result in visible changes in the differential image, which could potentially lead to false positives.

This limitation can be overcome with a hybrid sensing approach similar to HuddleLamp [24]. In HuddleLamp, Rädle et al. show a hybrid approach that combines robust and computationally cheap IR confidence image tracking with less reliable but more precise RGB tracking. To compensate for occlusion, HuddleLamp uses a fallback where it recovers occluded parts from previous frames and fills it with information from the depth image. PolarTrack could complement the IR confidence tracking with a more precise tracking using the RGB video feed, and thereby overcome the unreliable tracking when devices are occluded. Alternatively, PolarTrack could complement room-based tracking systems for people tracking (e.g., EagleSense [31]) to allow for low-cost *people-to-device* and *device-to-device* interactions.

While PolarTrack is most closely related to other camera-based tracking systems, related projects have implemented cross-device tracking through radio (e.g., GroupTogether [22]) or audio and inertial sensors (e.g., Tracko [10]) for locating devices. PolarTrack may be suitable to be integrated with these systems. For example, adding PolarTrack could potentially enable tracking of devices without additional device instrumentation [22] or improve tracking precision (mean error of 13.5 cm within 1 m) [10].

Cameras with automatic exposure can decrease captured light intensities when lighting conditions frequently change. For example, the camera does not receive enough light from the display when the lighting conditions change from bright to dark because the camera may have adjusted to a very short exposure time. In contrast, the device contours in the camera image decrease when the lighting condition changes from dark to bright. However, PolarTrack shares this limitation with all visual tracking systems, as light intensities emitted by displays are inferior to sunlight. In our evaluation, the camera was configured in fixed exposure mode and the experiment environment was indoors (i.e., no interference from sunlight). A camera with high-dynamic range (e.g., Logitech BRIO 4K Ultra HD webcam with HDR) could solve the exposure problem and will thus resolve comparably minimal fluctuations in brightness as caused by the rotating polarization filter. Future work will include experiments with HDR cameras and testing PolarTrack's suitability for outdoor use.

This work utilized a relatively small tracking area, but PolarTrack could be extended to track devices in larger areas such as rooms by stitching together multiple camera images. This, however, remains subject to future work.

Lastly, PolarTrack only tracks devices at angles that are approximately orthogonal to the camera. This limitation is inherited from HuddleLamp's multi-device tracking pipeline. Future systems could integrate devices' inertial measurement units (IMU) or their gyroscope to infer device orientation in 3D space, similar to CrossMotion [29]. A more expensive alternative is image or sensor fusion of multiple cameras [22].

CONCLUSION

We have presented PolarTrack, a novel camera-based approach to detect and track mobile devices inside the capture volume by exploiting display polarization. We described the physical characteristic of current display technologies that result in blinking displays in a camera feed when a linear polarization filter rotates in front of a color camera. PolarTrack exploits this characteristic and uses a computationally inexpensive algorithm to segment displays from the background using differential images and a naïve weighted sliding window approach. We show how open source multi-device tracking pipelines can use the resulting binary images for device detection and multi-device tracking. We evaluated PolarTrack's tracking quality, showing higher accuracy compared to a state-of-the-art open source system. Lastly, and to make PolarTrack reproducible, we provide a low-cost approach to building a PolarTrack system around a Kinect V2 camera from off-the-shelf components.

ACKNOWLEDGMENTS

This work has been funded by the Aarhus University Research Foundation and the Innovation Fund Denmark (CIBIS 1311-00001B). We thank Lindsay Reynolds for her valuable input and Peter Friis-Nielsen for technical assistance.

REFERENCES

1. Till Ballendat, Nicolai Marquardt, and Saul Greenberg. 2010. Proxemic Interaction: Designing for a Proximity and Orientation-Aware Environment. In *ACM International Conference on Interactive Tabletops and Surfaces - ITS '10*. ACM Press, New York, New York, USA, 121. DOI: http://dx.doi.org/10.1145/1936652.1936676

2. Frederik Brudy, Steven Houben, Nicolai Marquardt, and Yvonne Rogers. 2016. CurationSpace: Cross-Device Content Curation Using Instrumental Interaction. In *Proceedings of the 2016 ACM on Interactive Surfaces and Spaces (ISS '16)*. ACM, New York, NY, USA, 159–168. DOI:http://dx.doi.org/10.1145/2992154.2992175

3. Nicholas Chen, François Guimbretière, and Abigail Sellen. 2013. Graduate student use of a multi-slate reading system. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*. ACM Press, New York, New York, USA, 1799. DOI:http://dx.doi.org/10.1145/2470654.2466237

4. R. Cok. 2004. OLED display with circular polarizer. (Sept. 30 2004). https://www.google.com/patents/US20040189196 US Patent App. 10/817,536.

5. Saul Greenberg, Nicolai Marquardt, Till Ballendat, Robert Diaz-Marino, and Miaosen Wang. 2011. Proxemic Interactions: The New Ubicomp? *interactions* 18, 1 (jan 2011), 42. DOI: http://dx.doi.org/10.1145/1897239.1897250

6. Jens Grubert and Matthias Kranz. 2017. HeadPhones: Ad Hoc Mobile Multi-Display Environments Through Head Tracking. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 3966–3971. DOI: http://dx.doi.org/10.1145/3025453.3025533

7. Peter Hamilton and Daniel J. Wigdor. 2014. Conductor: enabling and understanding cross-device interaction. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14 (CHI '14)*. ACM Press, New York, New York, USA, 2773–2782. DOI:http://dx.doi.org/10.1145/2556288.2557170

8. Ken Hinckley, Gonzalo Ramos, Francois Guimbretiere, Patrick Baudisch, and Marc Smith. 2004. Stitching: pen gestures that span multiple displays. In *Proceedings of the working conference on Advanced visual interfaces - AVI '04*. ACM Press, New York, New York, USA, 23. DOI:http://dx.doi.org/10.1145/989863.989866

9. Da-Yuan Huang, Chien-Pang Lin, Yi-Ping Hung, Tzu-Wen Chang, Neng-Hao Yu, Min-Lun Tsai, and

Mike Y. Chen. 2012. MagMobile: enhancing social interactions with rapid view-stitching games of mobile devices. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia - MUM '12 (MUM '12)*. ACM Press, New York, New York, USA, 1. DOI: http://dx.doi.org/10.1145/2406367.2406440

10. Haojian Jin, Christian Holz, and Kasper Hornbæk. 2015. Tracko: Ad-hoc Mobile 3D Tracking Using Bluetooth Low Energy and Inaudible Signals for Cross-Device Interaction. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. ACM, New York, NY, USA, 147–156. DOI: http://dx.doi.org/10.1145/2807442.2807475
11. Abhijit Karnik, Diego Martinez Plasencia, Walterio Mayol-Cuevas, and Sriram Subramanian. 2012a. PiVOT: Personalized View-overlays for Tabletops. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST '12)*. ACM, New York, NY, USA, 271–280. DOI: http://dx.doi.org/10.1145/2380116.2380151
12. Abhijit Karnik, Walterio Mayol-Cuevas, and Sriram Subramanian. 2012b. MUSTARD: A Multi User See Through AR Display. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 2541–2550. DOI: http://dx.doi.org/10.1145/2207676.2208641
13. Ulrike Kister, Konstantin Klamka, Christian Tominski, and Raimund Dachselt. 2017. GraSp: Combining Spatially-aware Mobile Devices and a Display Wall for Graph Visualization and Interaction. *Computer Graphics Forum, 36* 36 (6 2017), 503–514. https://doi.org/10.1111/cgf.13206
14. Hideki Koike, Wataru Nishikawa, and Kentaro Fukuchi. 2009. Transparent 2-D Markers on an LCD Tabletop System. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 163–172. DOI: http://dx.doi.org/10.1145/1518701.1518728
15. Martin Kuechler and Andreas M. Kunz. 2010. Collaboard: A Remote Collaboration Groupware Device Featuring an Embodiment-enriched Shared Workspace. In *Proceedings of the 16th ACM International Conference on Supporting Group Work (GROUP '10)*. ACM, New York, NY, USA, 211–214. DOI: http://dx.doi.org/10.1145/1880071.1880107
16. Ming Li and Leif Kobbelt. 2012. Dynamic tiling display: building an interactive display surface using multiple mobile devices. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia - MUM '12*. ACM Press, New York, New York, USA, 1. DOI:http://dx.doi.org/10.1145/2406367.2406397
17. David Lindlbauer, Toru Aoki, Robert Walter, Yuji Uema, Anita Höchtl, Michael Haller, Masahiko Inami, and Jörg Müller. 2014. Tracs: Transparency-control for See-through Displays. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 657–661. DOI: http://dx.doi.org/10.1145/2642918.2647350
18. Roman Lissermann, Jochen Huber, Martin Schmitz, Jürgen Steimle, and Max Mühlhäuser. 2014. Permulin: Mixed-focus Collaboration on Multi-view Tabletops. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 3191–3200. DOI: http://dx.doi.org/10.1145/2556288.2557405
19. Andrés Lucero, Jussi Holopainen, and Tero Jokela. 2011. Pass-them-around: collaborative use of mobile phones for photo sharing. In *Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11*, Vol. 54. ACM Press, New York, New York, USA, 1787. DOI:http://dx.doi.org/10.1145/1978942.1979201
20. Andrés Lucero, Jaakko Keränen, and Hannu Korhonen. 2010. Collaborative use of mobile phones for brainstorming. In *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services - MobileHCI '10*. ACM Press, New York, New York, USA, 337. DOI: http://dx.doi.org/10.1145/1851600.1851659
21. Nicolai Marquardt, Robert Diaz-Marino, Sebastian Boring, and Saul Greenberg. 2011. The proximity toolkit: prototyping proxemic interactions in ubiquitous computing ecologies. In *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST '11*. ACM Press, New York, New York, USA, 315. DOI: http://dx.doi.org/10.1145/2047196.2047238
22. Nicolai Marquardt, Ken Hinckley, and Saul Greenberg. 2012. Cross-device interaction via micro-mobility and f-formations. In *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*. ACM Press, New York, New York, USA, 13. DOI:http://dx.doi.org/10.1145/2380116.2380121
23. Thomas Plank, Hans-Christian Jetter, Roman Rädle, Clemens N. Klokmose, Thomas Luger, and Harald Reiterer. 2017. Is Two Enough?: ! Studying Benefits, Barriers, and Biases of Multi-Tablet Use for Collaborative Visualization. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 4548–4560. DOI: http://dx.doi.org/10.1145/3025453.3025537
24. Roman Rädle, Hans-Christian Jetter, Nicolai Marquardt, Harald Reiterer, and Yvonne Rogers. 2014. HuddleLamp: Spatially-Aware Mobile Displays for Ad-hoc Around-the-Table Collaboration. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces - ITS '14*. ACM Press, New York, New York, USA, 45–54. DOI: http://dx.doi.org/10.1145/2669485.2669500

25. Roman Rädle, Hans-Christian Jetter, Mario Schreiner, Zhihao Lu, Harald Reiterer, and Yvonne Rogers. 2015. Spatially-aware or Spatially-agnostic?: Elicitation and Evaluation of User-Defined Cross-Device Interactions. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. ACM Press, New York, New York, USA, 3913–3922. DOI: http://dx.doi.org/10.1145/2702123.2702287

26. Mario Schreiner, Roman Rädle, Hans-Christian Jetter, and Harald Reiterer. 2015. Connichiwa: A Framework for Cross-Device Web Applications. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA '15*. ACM Press, New York, New York, USA, 2163–2168. DOI:http://dx.doi.org/10.1145/2702613.2732909

27. Garth B. D. Shoemaker and Kori M. Inkpen. 2001. Single Display Privacyware: Augmenting Public Displays with Private Information. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '01)*. ACM, New York, NY, USA, 522–529. DOI: http://dx.doi.org/10.1145/365024.365349

28. John C. Tang and Scott L. Minneman. 1990. VideoDraw: A Video Interface for Collaborative Drawing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '90)*. ACM, New York, NY, USA, 313–320. DOI: http://dx.doi.org/10.1145/97243.97302

29. Andrew D. Wilson and Hrvoje Benko. 2014. CrossMotion: Fusing Device and Image Motion for User Identification, Tracking and Device Association. In *Proceedings of the 16th International Conference on Multimodal Interaction (ICMI '14)*. ACM, New York, NY, USA, 216–223. DOI: http://dx.doi.org/10.1145/2663204.2663270

30. Pawel Wozniak, Nitesh Goyal, Przemyslaw Kucharski, Lars Lischke, Sven Mayer, and Morten Fjeld. 2016. RAMPARTS: Supporting Sensemaking with Spatially-Aware Mobile Interactions. *Proceedings of the SIGCHI Conference on Human Factors in Computing System* 2016 (2016). DOI: http://dx.doi.org/10.1145/2858036.2858491

31. Chi-Jui Wu, Steven Houben, and Nicolai Marquardt. 2017. EagleSense: Tracking People and Devices in Interactive Spaces Using Real-Time Top-View Depth-Sensing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 3929–3942. DOI: http://dx.doi.org/10.1145/3025453.3025562

32. Johannes Zagermann, Ulrike Pfeil, Roman Rädle, Hans-Christian Jetter, Clemens Klokmose, and Harald Reiterer. 2016. When Tablets meet Tabletops: The Effect of Tabletop Size on Around-the-Table Collaboration with Personal Tablets. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. ACM Press, New York, New York, USA, 5470–5481. DOI: http://dx.doi.org/10.1145/2858036.2858224