FarOut Touch: Extending the Range of ad hoc Touch Sensing with Depth Cameras

Vivian Shen
Carnegie Mellon University
Pittsburgh, PA, USA
vhshen@cmu.edu

James Spann
University of Rochester
Rochester, NY, USA
jspann2@cs.rochester.edu

Chris Harrison
Carnegie Mellon University
Pittsburgh, PA, USA
chris.harrison@cs.cmu.edu

ABSTRACT

The ability to co-opt everyday surfaces for touch interactivity has been an area of HCI research for several decades. Ideally, a sensor operating in a device (such as a smart speaker) would be able to enable a whole room with touch sensing capabilities. Such a system could allow for software-defined light switches on walls, gestural input on countertops, and in general, more digitally flexible environments. While advances in depth sensors and computer vision have led to step-function improvements in the past, progress has slowed in recent years. We surveyed the literature and found that the very best ad hoc touch sensing systems are able to operate at ranges up to around 1.5 m. This limited range means that sensors must be carefully positioned in an environment to enable specific surfaces for interaction. In this research, we set ourselves the goal of doubling the sensing range of the current state of the art system. To achieve this goal, we leveraged an interesting finger "denting" phenomena and adopted a marginal gains philosophy when developing our full stack. When put together, these many small improvements compound and yield a significant stride in performance. At 3 m range, our system offers a spatial accuracy of 0.98 cm with a touch segmentation accuracy of 96.1%, in line with prior systems operating at less than half the range. While more work remains to be done to achieve true room-scale ubiquity, we believe our system constitutes a useful advance over prior work.

CCS CONCEPTS

• **Human-centered computing** → **Ubiquitous and mobile computing systems and tools**.

KEYWORDS

Depth sensing, on-world computing, projection, ubiquitous computing, finger tracking.

ACM Reference Format:
Vivian Shen, James Spann, and Chris Harrison. 2021. FarOut Touch: Extending the Range of ad hoc Touch Sensing with Depth Cameras. In *Proceedings of the 9th ACM Symposium on Spatial User Interaction.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/1122445.1122456

SUI '21, Nov 08–10, 2021, Virtual

ACM ISBN 978-1-4503-XXXX-X/21/10…$15.00
https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Computers with touchscreens have become near-ubiquitous in recent years, allowing for more screen real estate, the removal of mechanical buttons, and entirely new physical designs. Touch input's popularity is also driven by the ease-of-use of direct manipulation interfaces. Today, touch experiences are generally constrained to devices with integrated touch pads and screens, but there is a growing body of work on "ad hoc" touch interfaces that appropriate existing (i.e., unmodified) surfaces in the environment for interactive control. For example, double tapping on a wall in your house could toggle the lights on or off; sliding one's fingers up or down would act like a dimmer control, while horizontal swiping gestures could move between lighting presets (e.g., full illumination, theatre, cool color temperature).

Unlike traditional mechanical switches, such software-defined virtual controls would offer tremendous flexibility and opportunities of personalization. Consider an ordinary coffee table located between a user's couch and television – a sensor operating in the TV (or set top box) could appropriate the table such that it becomes a large trackpad that controls a cursor for a complex interface (e.g., Netflix, Apple TV). As another example, a smart speaker in your kitchen could appropriate countertops for touch interaction. Small circling motions could control speaker volume without having to issue voice commands, while swiping gestures could trigger the device to repeat or advance steps in a recipe. Importantly, this vision of ad hoc touch sensing does not require users to replace all the surfaces in their house and run new wires, but instead relies on a distributed array of more advanced versions of existing consumer objects containing the requisite ad hoc touch sensing capabilities.

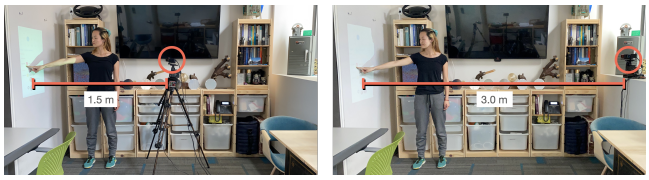

Figure 1: Depth-camera-based touch sensing systems in prior work are able to operate at ranges up to around 1.5 m. This limited range generally means they must be either ceiling mounted or placed awkwardly in the middle of a room (left image). Our system – FarOut Touch – is able to operate at ranges up to 3 m (right image), and thus offers much greater flexibility in placement and allow such systems to be considerably less obtrusive.

Perhaps the most promising and practical avenue to achieve this vision is through the use of depth cameras. Unlike conventional RGB cameras that operate in "pixel space" coordinates, depth cameras capture the 3D geometry of scenes in real-world coordinates (e.g., millimeters), allowing for absolute tracking irrespective of surface orientation to the camera. Conventional RGB cameras also struggle to differentiate between a finger touching a surface perpendicular to the camera vs. a finger hovering slightly above it (i.e., "click" detection), whereas depth cameras can readily detect the distance between a finger and a surface. While stereo RGB camera setups can be used to estimate depth, they fail on homogeneous surfaces such as painted walls (to overcome this, some products use active projection, but at this point we would consider it to be a type of depth camera). A final benefit of depth cameras is robustness to variation in illumination.

For these key reasons, depth cameras have become the most popular sensor for ad hoc, remotely-sensed, touch tracking applications. Microsoft's release of the Kinect sensor in 2010 was a watershed moment when the technology became plausible for consumer uses (a 4GB Xbox 360 with a bundled Kinect cost $299 USD). Since then, depth cameras have increased in resolution, and decreased in size and cost. For example, although STMicroelectronics' VL53L5 can only capture an 8×8 depth map, it measures a mere 5.4×3.0×1.4 mm and costs under $10, underscoring the tremendous progress made in miniaturization and the high likelihood of e.g. surface mount depth cameras in the near future.

As we will discuss in Related Work, dozens of systems have employed depth cameras for ad hoc touch tracking. An overview of key systems is provided in Table 1, which shows limited progress over the past decade. Despite employing different depth camera technologies and having totally different software pipelines, a commonality is limited range, never exceeding 1.8 m in testing, and with most systems operating below 1.5 m. This would preclude all of the example applications we described above, and in general is a major impediment to achieving the vision of lightweight ubiquitous touch sensing.

This research was initiated with the following goals: 1) To make a step function improvement in sensing range – doubling the range of the next best system; 2) To achieve comparable spatial accuracy to prior work (no more than 2 cm tracking error at >95% click segmentation); 3) To use only the depth data feed, and not RGB or infrared imagery, so as to help preserve user privacy. This paper details our approach and the technical changes and innovations we made to achieve this goal.

2 RELATED WORK

Our work was inspired by previous explorations and advances in ad hoc touch sensing, especially systems employing cameras to achieve remote sensing, and thus not requiring any surface instrumentation. We also include a discussion on prior work that sought to improve the quality of depth camera imagery, as this is the fundamental technology on which we build our system.

2.1 Non-Camera-Based ad hoc Touch Sensing

Researchers have considered innumerable ways to capture touch events on everyday objects and surfaces. Due to the computational complexity of computer vision, early systems tended to employ passive acoustic methods. For example, PingPongPlus [28] used four contact microphones to capture and triangulate ping pong ball bounces on an instrumented table. Similarly, the Interactive Window project [41, 42] used contact microphones on a large sheet of glass to localize finger taps, while Toffee [70] used contact microphones in the back of devices (e.g., smartphones) to temporarily appropriate surfaces for touch interaction. Later work by Pham et al. [43–45] showed that continuous tracking of fingers was possible with passive acoustics. ScratchInput [24] took a different approach, calculating not absolute spatial position, but rather listening for patterns of sounds to co-opt large everyday surfaces for gestural inputs. Active acoustic methods are also possible – these powered systems typically emit ultrasonic chirps, which are captured by external microphones that can triangulate the origin [51, 75].

Electrical methods are comparatively rarer, as they most often require deeper instrumentation of the host surface, and thus are not truly ad hoc. However, electrodes can at least be hidden behind paint or wallpaper, as in the case of LivingWall [5], Wall++ [77] and Electrick [76]. Everyday conductive objects can be made touch sensitive through capacitive sensing, such as fruit, door handles, and metal furniture [11]. The more advanced capacitive sensing approach in Touche [48] moves beyond binary touch, and allows for detection of basic gestures and grasps. We note that the latter two systems are more object-scale than room-scale, and it is not immediately clear how to expand the techniques to larger interactive areas.

Finally, non-camera optical systems are also possible, but require technology to be placed onto the surface as opposed to at a distance. For example, Paradiso et al. [54] used a spinning LIDAR attached to a wall to create a curtain of touch interactivity. In a similar fashion, SurfaceSight [36] used a spinning LIDAR in the base of IoT devices (e.g., smart speakers, thermostats) to detect user touch and object events in a thin plane above a host surface (e.g., kitchen countertops, walls). SideSight [6] was functionally similar, but used discrete infrared proximity sensors ringing a small mobile device.

2.2 Camera-Based ad hoc Touch Sensing

The hardware most commonly associated with computer vision and sensing systems are RGB (i.e., visible light) cameras, and ad hoc touch sensing systems are no exception. DigitalDesk [60] introduced the idea of subtracting successive RGB frames to detect motion of the hand and fingers. LightWidgets [15] runs a skin detection algorithm using the hue and saturation values of the RGB image. Also using color, Sugita et al. [55] and Marshall et al. [39] detect touches by tracking the color change of fingernails when a finger is pressed to a surface. TouchLight [61] and Bonfire [33] both use multiple RGB cameras to create virtual interactive spaces, digitizing finger inputs using computer vision.

Thermal cameras use infrared radiation to track heat rather than visible light, and are sometimes used instead of RGB cameras for touch detection. These cameras can track the heat transfer from a fingertip to a host surface by measuring the residual heat and using background subtraction, as demonstrated by Heatwave [37], ThermoTablet [29], Iwai et al. [30], and Funk et al. [18]. Rather than use a static background, Thermal Touch [35] and Abdelrahman et

Paper / System Name	Year	Reported Sensor Range	Sensor Used / Technology Type	Click Accuracy	Precision Error
Wilson [64]	2010	150 cm	Kinect v1 / Structured Light	-	15.0 mm
OmniTouch [23]	2011	20-100 cm	Custom short-range PrimeSense (similar to Kinect v1) / Structured Light	96.5%	16.2 mm
Ying Yin [73]	2012	120 cm	Kinect v1 / Structured Light	-	10.0 mm
Extended Multitouch [40]	2012	120 cm	Kinect v1 / Structured Light	-	8.0 mm
WorldKit [67]	2013	~75 cm	Kinect v1 / Structured Light	-	-
Haubner et al. [25]	2013	188*cm	Kinect v1 / Structured Light	-	0.7 - 12.5 mm
Thermal Touch [35]	2014	30 cm	Optris PI 200 / Thermal Camera	~75.0%	7.8 mm
Seggerman and Perez [49]	2014	~91 cm	Kinect v1 / Structured Light	-	-
Cadena et al. [7]	2016	120 cm	Kinect v2 / Time-of-Flight	-	9.4 mm
DIRECT [68]	2016	160 cm	Kinect v2 / Time-of-Flight	99.3%	4.8 mm
Fischbach et al. [16]	2016	90 cm	Kinect v2 / Time-of-Flight	-	-
Xiao et al. [69]	2017	90 cm	Asus Xtion / Structured Light	-	~10.0 mm
MRTouch [71]	2018	100 cm	HoloLens / Time-of-Flight	96.5%	~6.0 mm
FarOut Touch (our system)	2021	**300 cm**	Kinect v2 / Time-of-Flight	96.1%	9.8 mm

Table 1: A survey of remote-sensing, ad hoc touch tracking systems. *At this range, the authors note their system is unable to detect fingers, and instead whole hands must be used. Dash means not reported.

al. [1] segment the entire image with temperature thresholds in order to find the most likely fingertip locations.

Rather than just passively tracking infrared radiation, some active IR cameras use short wavelengths of infrared light to illuminate an area and generate imagery. For example, Tomasi et al. [57] created an IR camera system by projecting a keyboard onto a surface along with infrared line lasers, tracking touches when they broke the beam. This approach was mirrored by Z-touch [56] with multiple layers of these line lasers laid directly on top of a surface. Rather than tracking the finger itself, Wilson's PlayAnywhere system [62] uses IR illumination to create and track hand shadows, the shape of which can be used to trigger touch events.

More closely related to our technique are ad hoc touch sensing systems that use depth cameras, which we summarize in Table 1, starting with pioneering work by Wilson and Benko in the 2000s [3, 63–65]. Since the 2010 launch of Microsoft's Kinect depth camera (which used structured light), uses in HCI have exploded. Early on, Wilson [64] demonstrated a single Kinect v1 camera could provide conventional touch events by using a per-pixel depth threshold determined from a histogram of a static scene. Both the latter work and LightSpace [65] use a multi-depth camera setup that required calibration before they could run. Similarly, Extended Multitouch [40] and Seggerman et al. [49] segmented the image by thresholding depth slices close to the surface. Omnitouch [23] and Yin [73] introduced the concept of using depth map gradients to segment finger input, which does not require background subtraction and can work on-the-go and in dynamic scenes. Worldkit [67] and Haubner et al. [25] operated by finding connected (user) components on a background-subtracted surface image, although the latter was unable to segment finger touches and focused instead on larger hand and arm inputs.

Since 2016, ad hoc touch sensing systems have tended to use the updated time-of-flight-based Kinect v2 (which we also use) and depth cameras from other vendors such as Intel. Fischbach et al. [16], Cadena et al. [7] and DIRECT [68] both use the Kinect v2, but follow the previous literature by background subtracting a live image before blob tracking to determine finger interaction locations. DIRECT and Cadena et al., however, utilize both the infrared and the depth camera streams to create a superior composited image. MRTouch [71] is a direct extension of DIRECT's depth+IR image pipeline, but uses HoloLens' (Kinect-v2-like) ToF camera instead. Finally, Desktopography [69] utilizes Xtion depth cameras and an extension of Omnitouch's finger tracking algorithm to create mixed reality interfaces on uninstrumented tabletops.

2.3 Improving Depth Camera Imaging

There are many types of depth cameras on the market today, most of which are built around stereography, structured light, or time-of-flight approaches ([4, 47] provide an excellent explanation of these different technologies). In this work, we use a Microsoft Kinect v2, an older, but still capable time-of-flight depth camera. Using multiple modulated infrared signals, it calculates the distance to objects by comparing the phase of reflected light. This is an extraordinary sensing feat for a consumer device, and tradeoffs were made in the design to achieve a price point of under $200. Like all cameras – depth or otherwise – there is noise in the signal that degrades the captured image.

For this reason, researchers have pursued techniques to maximize image quality. For instance, classical approaches for super resolution [8, 27] use multiple slightly-shifted images to capture different quantizations of sub-pixel features, which can then be composited and resolved. Even when holding a smartphone still, there is sufficient hand shake to perform multi-frame super resolution to great effect [66]. Single-frame super resolution is also possible using machine learning techniques, which learn the relationships between high resolution images and their low resolution counterparts [17], though this tends to be domain-specific (i.e., one cannot use a single-frame, deep-learning-based super resolver trained on RGB images of urban scenes and then apply it to fingers in a depth map). Researchers have investigated super resolution techniques specifically for depth imagery, such as [20] and [32].

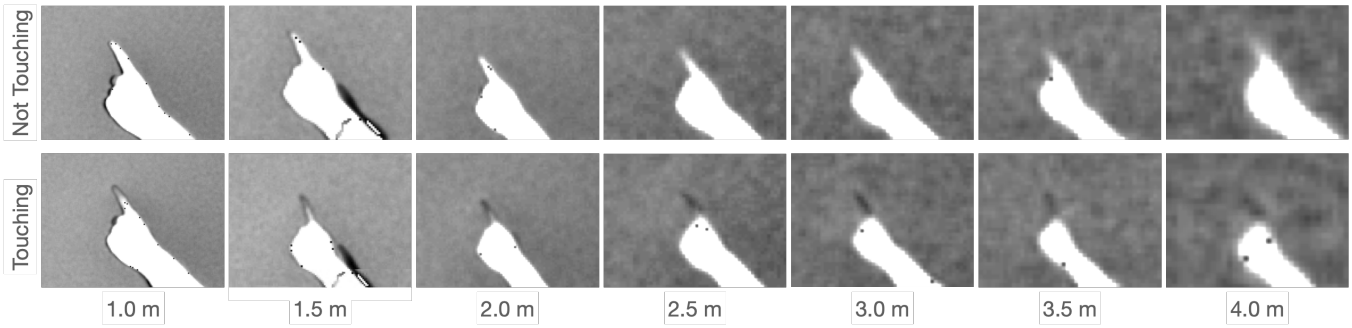

Figure 2: A characteristic "dent" appears in the depth map of a host surface when a finger is touching (bottom row), due primarily to multipath reflections. The effect is most prominent in the 2.5 - 3.0 m range. The effect disappears when the finger is lifted 1 cm or more above the surface (top row), offering a strong and characteristic signal on which to build a touch detector.

More specific to time-of-flight depth cameras and the Kinect v2, we have work by Lawin et al. [38] that improved depth map quality by computing multiple depth hypotheses per pixel and then ranking them using kernel density estimation (which helps to resolve specific phase unwarping ambiguities). Even still, there can be holes in the depth map, and so many researchers have looked at leveraging data from the Kinect v2's RGB camera to perform infilling, especially around object boundaries and other discontinuities [9, 46, 52, 72]. Using SLAM, KinectFusion [31] was able to create higher-than-native-resolution meshes by fusing depth data from multiple viewpoints (essentially another form of super resolution). Even temperature changes can affect the Kinect v2's depth readings, as discovered by Wasenmuller et al. [59]; Corti et al. [12] found that an external fan helped to stabilize measurements.

3 EXPLORATIONS

As our research was goal driven (as opposed to starting with a technical insight), we considered a wide range of possible technical avenues, often drawing inspiration from prior work in the literature. We also explored several orthogonal fields that employ imaging (e.g., biology and astronomy), looking for advances we might bring to the domain of depth sensing. This spurred numerous technical investigations with working prototypes, but the results fell short of our stated goal. Our failure to identify a singular technical breakthrough (e.g., deep learning, multi-spectral IR imaging, single frame super resolution) that would offer a significant improvement in sensing range was discouraging, and served to reinforce why progress in sensing range has been slow, despite being a well-trodden research area. Nonetheless, our explorations uncovered an interesting depth camera sensing phenomena that we could leverage for longer ranges. Additionally, we found that there were many small optimizations and improvements that while not particularly novel or impactful alone, had potential to compound.

Figure 3: We tested the robustness of the denting effect in different environments and lighting conditions, on different materials and surfaces, and at different sensor distances. Shown here are eight example scenes, with zoomed depth maps showing touch and no-touch conditions.

3.1 Finger "Denting" Phenomena

There was one effect we discovered that held promise: we found that small geometric features (such as fingers) would create negative shadow-like regions in larger planar surfaces (such as walls), manifesting as a characteristic "dent" in the Kinect v2 depth image. This can be seen in Figure 2 as a darker region (bottom row), registering (incorrectly) as farther away than the wall itself. At first, we suspected this might be due to occlusion boundaries and signal averaging, but this is not the case. Foremost, the effect disappears when the finger is lifted from the surface (despite the fact the occlusion boundary remains; Figure 2, top row). Second, it manifests most significantly at particular ranges, and especially beyond 1.5 m. Third, by varying the distance and reflectivity of objects placed between the sensor and wall, we could vary the strength of this "denting" effect, and concluded that multipath reflections of the Kinect v2's modulated infrared light was the main cause (see [2, 19, 38, 50, 59] for a detailed description of how multi-frequency time-of-flight depth sensors operate).

Put simply, when the sensor is placed close to a wall, the finger is almost exclusively imaged by line-of-sight infrared. However, as the sensor moves back, the floors and ceiling (and potentially walls, furniture and other objects) become secondary (and longer) paths from which scattered infrared light can illuminate otherwise "empty" occlusion boundaries on the finger. The signal at the corresponding pixel in the Kinect is combined, incorporating correct and incorrectly far values, creating the dented effect. When the finger is lifted from the wall by just a few centimeters, the significantly

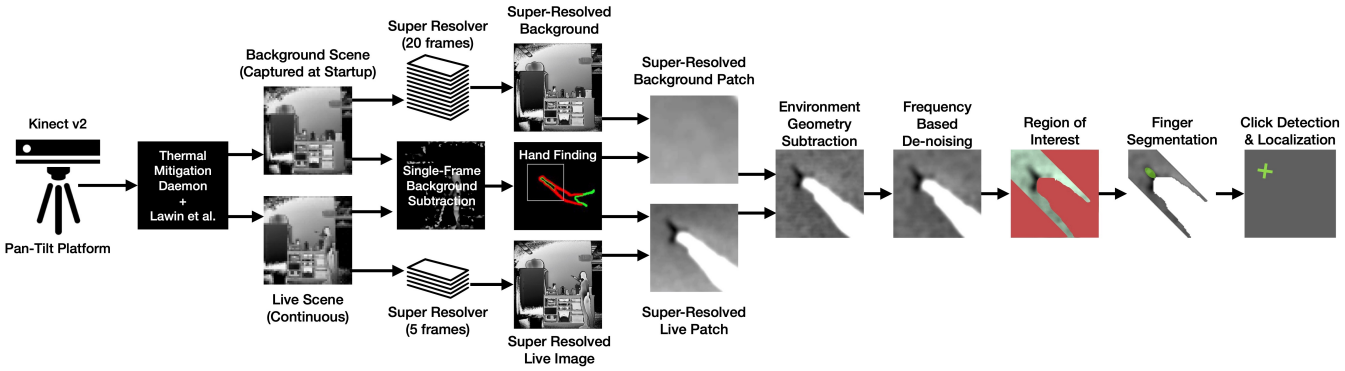

Figure 4: Architectural overview of our FarOut Touch pipeline.

nearer finger readings dominate any incorrect far readings when combined, and so the finger is seen as exclusively closer to the sensor. We found that at ranges beyond roughly 3.5 m, the effect is less pronounced due to depth noise and low spatial resolution (i.e., at this range, the finger is only a few pixels in size). However, in the range of 1.5 m (where most previous systems stop) to 3.5 m, this unique signal can be used to detect touch events. We confirmed the robustness of this signal across many rooms, distances, and lighting conditions (see Figure 3 for eight example scenes).

3.2 Marginal Gains Approach

In our survey of prior work, we found many interesting ideas implemented in isolation, but no system brought them together in a unified way. Further, while some systems were very advanced in specific ways, other aspects were left unoptimized. Thus, as we developed our novel system, we sought to optimize each aspect of our full stack, from sensor to event handling. This is akin to a marginal gains approach [10, 21] popularized by Sir Dave Brailsford (in the field of professional cycling). This philosophy emphasizes making lots of small improvements that not only add together, but compound (i.e., exponentially) [22]. In other words, instead of a singular big advance, one can make five or ten small improvements to achieve a similar gain. We decided to embrace this marginal gains philosophy, which required us to scrutinize every component and process that makes up a touch sensing stack, looking for gains no matter how modest. When possible, we drew together good ideas from the literature, on top of which we integrated our own novel advances. In the next section, we describe our final stack and the various improvements we discovered and implemented, concluding with a discussion of the relative benefit of each improvement.

4 IMPLEMENTATION

We now describe the key pieces of our system, especially those where we were able to incorporate improvements as part of our marginal gains strategy. An overview of our pipeline architecture can be found in Figure 4.

4.1 Sensing Apparatus

While there are many depth cameras available today, we chose to use a Kinect v2 for several important reasons. First and foremost, this sensor is very popular in the research community and widely employed in ad hoc touch tracking literature. This allows us to more directly and fairly compare touch accuracies with the most competitive systems (for instance, we found no touch tracking systems using the Kinect v2's successor device, the Kinect Azure). Second, the Kinect v2 is a good middle-of-the-road sensor, with a depth image resolution of 512×424 pixels and a 92.7° diagonal field-of-view. Third, the Kinect v2 uses a "time-of-flight" approach (i.e., phase unwrapping of multiple modulated infrared signals), which is becoming increasingly practical and popular (e.g., the LIDAR sensors in newer Apple mobile hardware are also time-of-flight based). For detailed performance parameters of the Kinect v2, please see [34, 59].

We mount our Kinect v2 sensor to a motorized pan/tilt head (Figure 5), which allows us to programmatically rotate the sensor on the X and Y axes. This rig consists of two Vexta PK245-01BA-C4 stepper motors with steps as small as 0.00625 degrees. The stepper motors are controlled by an Arduino, which is tethered to a computer over USB for programmatic control. This whole apparatus sits on a tripod for ease of movement and deployability.

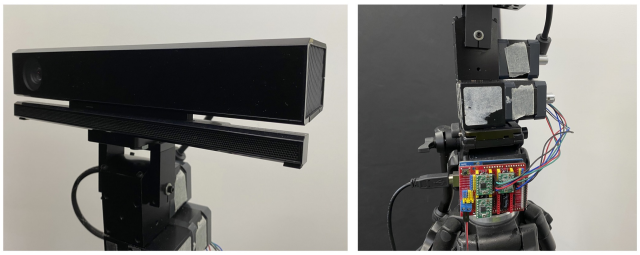

Figure 5: A Kinect v2 (left) mounted to a motorized pan/tilt rig (right).

4.2 Privacy

As discussed in our Introduction, we envision room-scale touch sensing in contexts such as homes, schools, medical facilities, and offices. In these settings, users usually do not want to be surveilled. As such, there is an important push towards technologies that more innately preserve user privacy. Conventional visible light cameras

are particularly privacy invasive, followed closely behind by infrared cameras. In both cases, a user can be easily recognized by their face, and even text on documents can be seen. Although depth maps can still possibly reveal identity and activity, they are comparatively less invasive, and for this reason we build our pipeline using only depth data (and not the Kinect's RGB or infrared camera feeds, which have been previously used to boost touch accuracy, e.g., DIRECT [68]). In the future, our pipeline could run on depth sensing hardware that is only capable of capturing depth maps.

4.3 Mitigating Thermal Drift

Like all cameras, noise on the image sensor varies with ambient temperature. Wasenmuller and Stricker [59] found that the Kinect v2's depth measurements drift as the sensor warms up (i.e., from a "cold start") – growing monotonically to a not insignificant 20 mm mean error after 16 minutes of use, after which the integrated fan activates and moderates the device's temperature. Figure 6 shows an example of this variability over time. Corti et al. [12] were successful at stabilizing this signal by using a second, external fan, though this is not particularly practical. Instead, we wrote a simple daemon that keeps the Kinect running, and thus at a much more consistent thermal state. Our later pipeline pulls depth frames from this daemon instead of directly from the Kinect SDK. We acknowledge that keeping the Kinect running constantly can be power consumptive, but it should be considered an implementation proxy for a more refined thermal management strategy. Although the code is proprietary, we suspect the Kinect v2's fan logic was designed to protect the sensor from overheating, but not provide the most accurate depth estimates for finger tracking (a task never intended for the Kinect).

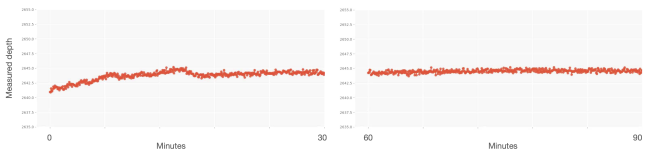

Figure 6: Depth value of a single pixel over time. Left: Values from the first 30 minutes after start up. Right: 30 minutes of values after one hour, which are more stable.

4.4 Improved Depth Map Quality

Although the Kinect v2 for Windows SDK is mature software at this point, researchers have been able to make advances in the fundamental algorithms to improve multi-phase-based time-of-flight depth estimation. Notable among these efforts is the approach by Lawin et al. [38] to resolving phase unwrapping ambiguities using spatial kernel density estimation, selecting the best depth estimate by considering the distribution of hypotheses in a local neighbourhood. This process not only extends the Kinect's sensing range, but yields reliable depth estimates in areas previously too ambiguous. This difference is readily seen in Figure 7, which offers side-by-side output from the Kinect SDK vs. libfreenect's implementation of Lawin et al.'s approach [38]. Given these superior results, we opted to adopt this approach in our pipeline.

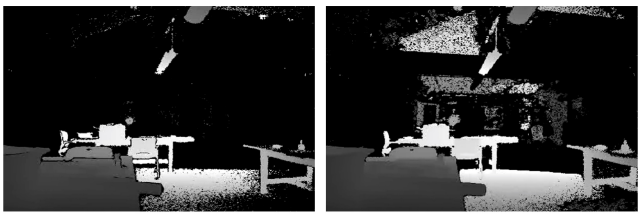

Figure 7: Left: Standard depth map from Kinect v2. Right: Depth map using the Lawin et al. [38] approach.

4.5 Temporal Averaging

Perhaps the most common technique for reducing noise across practically all sensor domains is to increase exposure time. This is akin to averaging, which suppresses transients [13]. In the case of uncorrelated Gaussian noise, like what we face, the mean converges to the true depth value with more frames (averaging n samples reduces the noise power by a factor of n). To take advantage of this benefit, we maintain rolling per-pixel averages using the last five frames of data from the Kinect v2. The improvement is readily apparent in Figure 8. This, of course, does not come for free – there is a cost in terms of latency. The Kinect v2 runs at 30 FPS, and so five frames constitutes 167 ms. However, we found this delay to be acceptable for discrete touch events, such as pressing a button.

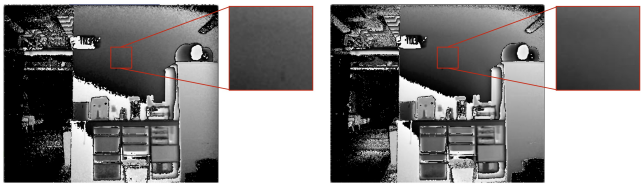

Figure 8: Left: Single frame depth map with portion of wall enlarged for detail. Right: Temporally averaged depth map of the same scene with equivalent callout. Note the noise reduction.

4.6 Geometric Super Resolution

Each pixel in the Kinect v2 has a field of view of roughly 0.14°. As previously noted, our Kinect v2 sits on a motorized pan-tilt rig controlled by an Arduino, which we use to tilt the Kinect back and forth diagonally by 0.125° with a period of 310ms. This means geometric features, such as bumps and edges, translate across roughly two depth pixels, offering different quantizations of sub-pixel geometry. With this multi-frame data, we perform geometric super resolution. Specifically, each incoming depth frame is enlarged by 2x on each axis using linear interpolation. We then use OpenCV's implementation of parametric image alignment using enhanced correlation coefficient (ECC) maximization [14] to align all frames (with translation only) to a canonical frame captured at startup. We then stack the aligned frames, compute an average image, and perform sharpening. Figures 9 and 11 offer comparisons between native and super-resolved output. We use geometric super resolution twice: once at startup to compute a 20-frame super-resolved

Figure 9: Example patch showing the improvement from single frame (left) to five-frame super resolution (right).

background image (without the user present), and continuously during operation using our five-frame rolling window.

4.7 Hand Finding

As we will discuss in the next section, we use a characteristic signal to detect finger touches on the environment. However, similar signals can manifest due to noise, and thus it is useful as a preprocessing step to identify a region of interest where touches may occur. As an added benefit, we can also eliminate unnecessary computation in areas where there will be no touches. For this, we follow a similar approach to DIRECT [68], where we identify the wrist and forearm contours of a user operating close to a wall using predefined distance thresholds (3-12 cm for the wrist and 8-25 cm for the forearm). Figure 10 offers an illustrative example of a user touching a wall; the wrist contour is shown in red, while the forearm contour is shown in green. Note that we cannot simply look at the depth slice between e.g., 0-3 cm to directly find the finger (as done in prior work), as this is too noisy and many finger-like blobs exist. Instead, we extrapolate a probable hand location by finding the vector from the centroid of the forearm contour to the centroid of the wrist contour, and extend it beyond the wrist, cropping an 80x80 pixel patch (Figure 10, right, white square).

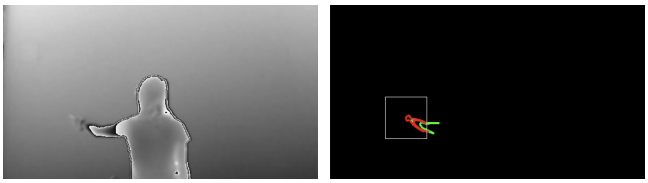

Figure 10: An example of hand finding. Left: Live super-resolved image. Right: Arm and wrist contours and the probable hand patch.

4.8 Environment Geometry Subtraction

Depth cameras operating in devices such as smart speakers and televisions should not assume they will be neatly aligned with walls, tables and other surfaces. Surfaces at angles with respect to the sensor appear as gradients in the depth map, and complex object geometries can obscure the hands and fingers. To account for this, we simply subtract our previously computed super-resolved background, which normalizes the image. For instance, previously angled walls now appear as flat regions in the depth map (Figure 12, A vs. B). When a user is present in the scene, the remaining geometry is essentially the user's body and noise.

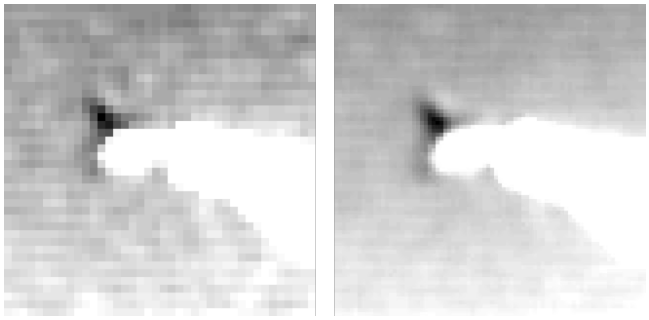

Figure 11: A user's arm with the index finger touching a wall. Left: Single frame. Right: Five-frame super-resolved output.

4.9 Frequency-Based Denoising

Early in our investigations, we saw that noise in the depth map manifested at particular spatial frequencies. This could be an artifact of the physical sensor or the Kinect v2's internal filtering and processing pipeline – as a proprietary device, it is unclear. Nonetheless, it exists, and we can attempt to account for it to improve our signal fidelity. However, this must be done with care and precision, as finger blobs can also appear as small "dimples" in the depth map, similar to noise. We tried both wavelet decomposition [53] and DCT filtering [74] approaches, finding the latter easier to tune. Figure 12C shows the input with DCT filtering using a sigma of 1.5 with a window size of 20x20. These parameters differ significantly from those recommended in the paper because our small hand patch has a very limited range of frequencies. While the finger blob can become partially obscured, noise is also decreased, leading to an overall net gain in tracking performance.

4.10 Finger Region of Interest

Now with noise suppressed, we narrow our finger search area even further. First, we compute an arm-aligned vector using the centroid of the arm and wrist contours (see previous Hand Finding section). We then create a rectangular inclusion region aligned with this vector, spanning the hand patch, that encompasses the arm and the likely location of any fingers. We then exclude depth values that are greater than 1 cm above the surface. An example of the resulting inclusion region can be seen in Figure 12D (green-tinted

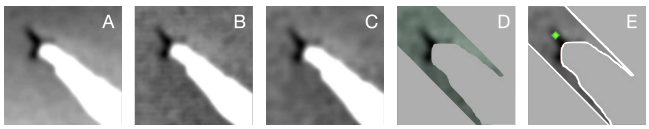

Figure 12: An example hand patch passing through the end of our processing pipeline. A) The cropped hand patch taken from the super-resolved live image. B) Environment geometry subtraction. C) Frequency-based denoising. D) Finger region of interest. E) Click Detection and Localization.

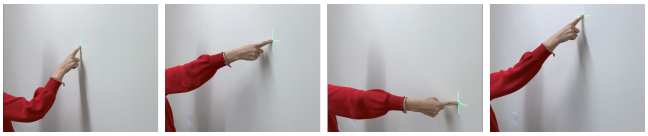

Figure 13: Examples of our system localizing a user's touch input, visualized by projecting a green crosshair at the click location. See also Video Figure.

area; the gray area represents all excluded pixels). We note that this basic approach does mean that the finger has to project from the wrist along roughly the same axis, and we do not support touches where the finger is angled significantly from the wrist and arms.

4.11 "Click" Detection and Touch Localization

As mentioned in our exploration section, we discovered an effect where fingers touching a wall manifest not as an outward bump (as one would expect since they are closer to the camera than the wall), but rather as a "dent" into the wall. It is this characteristic signal that we take advantage of for click detection. This signal is strongest between 1.5 and 3.5 m – the interaction range beyond prior systems' capabilities – and thus the focus of our research. However, for distances below 1.5 m, we recommend utilizing existing techniques, which provide sufficient accuracy (summarized in Table 1). For this reason, our evaluation uses 3.0 m as a benchmark.

Once the region of interest has been found, we must determine whether a click has actually occurred and where it is located. To do so, we first find the farthest 50 pixels in the hand patch and run blob detection. Blobs smaller than 6x6 pixels are filtered, and then we select the blob that is most distant from the wrist contour centroid. If we do not find any candidate "dark" blobs, then the finger is assumed to not be touching the surface. If we do find an appropriate blob, then we consider the centroid of this blob as the finger click location (green dot in Figure 12E). To add further stability, we use a five-frame rolling window of candidate touch blobs, and compute the mean and median locations. If these two values separate, indicating high variance in predictions (i.e. our algorithm is finding competing "dents" around the hand), then we do not trigger a click event. Only when there high consistency is a touch event triggered. Figure 13 shows the detected click positions projected onto a wall.

4.12 Performance and Latency

Our full pipeline, developed in C++, runs at 30 FPS on an Intel Core i7-8759H @ 2.20 GHz laptop, which is the native framerate of the Kinect v2. We found a mean motion-to-photo latency of 233 ms, roughly a third of which comes from the Kinect transmitting depth frames over USB to the laptop and double-buffered graphics going to the digital projector.

5 EVALUATION

There are two key factors we wished to measure in order to compare with prior work in the ad hoc touch tracking literature. First is spatial touch accuracy – put simply, if a user touches a target, how close can we resolve the touch coordinate? Second is touch segmentation accuracy – that is, if a user performs a touch, do we correctly capture the event or miss it?

5.1 Setup

We recruited 10 participants (3 female, mean age = 21.2), who were all right-handed and completed the study using their right hand. The study took approximately 30 minutes and participants were compensated $20 for their time. After a brief orientation, participants were asked to stand in front of a plain wall in a typical office measuring 4.0×2.9×3.1 m and containing commonplace furniture (desks, chairs, cabinets, etc.). Tape was placed on the ground 30 cm away from the wall, and users were asked to stand behind this during the course of the study. This was included to ensure that a user's other limbs would not touch the wall during the experiment, generating an unintended touch event. Our pan-tilt Kinect v2 apparatus sat on a tripod towards the back of the room at a distance of 3 m from the wall. As noted before, this range was of key interest to us, as prior work has extensively studied accuracies sub 1.5 m and we recommend using existing techniques at those ranges. Although input is the focus of our work, we found that projecting color-coded crosshairs onto the wall greatly expedited our study procedure and afforded us flexibility in design. Thus, just to the side of the Kinect, we placed a BenQ DLP HD 1080p Projector (W1070), which provided a projection area of 226.6×127.4 cm on the wall. The Kinect and projector were calibrated using a four-point perspective transform (this was sufficient for our study, but we note that more advanced calibrations are possible, such as the projection mapping used in [69]).

5.2 Data Capture Procedure

In order to capture data to evaluate touch precision and segmentation accuracy, we had users touch a series of crosshairs projected onto the wall at roughly chest height, arranged in a 6×4 grid (24 cm horizontal interval, 18 cm vertical interval). Crosshairs appeared one at a time in a random order; one round of data collection consisted of clicking all 24 crosshairs. We asked participants to hold their finger to the crosshair until the system made an audible beep, after which they could move to the next target. Importantly, no live visual feedback was shown to users, as they might attempt to accommodate any system inaccuracies and "steer" visual feedback onto the requested targets. Participants were asked to touch the crosshair center as accurately as possible, as we wanted to minimise user error in our calculations. Participants were also asked to stand roughly facing the wall, with their touching finger roughly inline with the wrist, and their arm naturally off to one side so as not to occlude the arm from the Kinect v2. Participants could move around as they saw fit to reach the targets. If the system did not detect the click event, the experimenter flagged the trial manually as a miss, and the participant was allowed to try again. In total, each participant completed three rounds of data collection, resulting in 72 click trials per participant, and thus 720 total trials were collected from our ten participants.

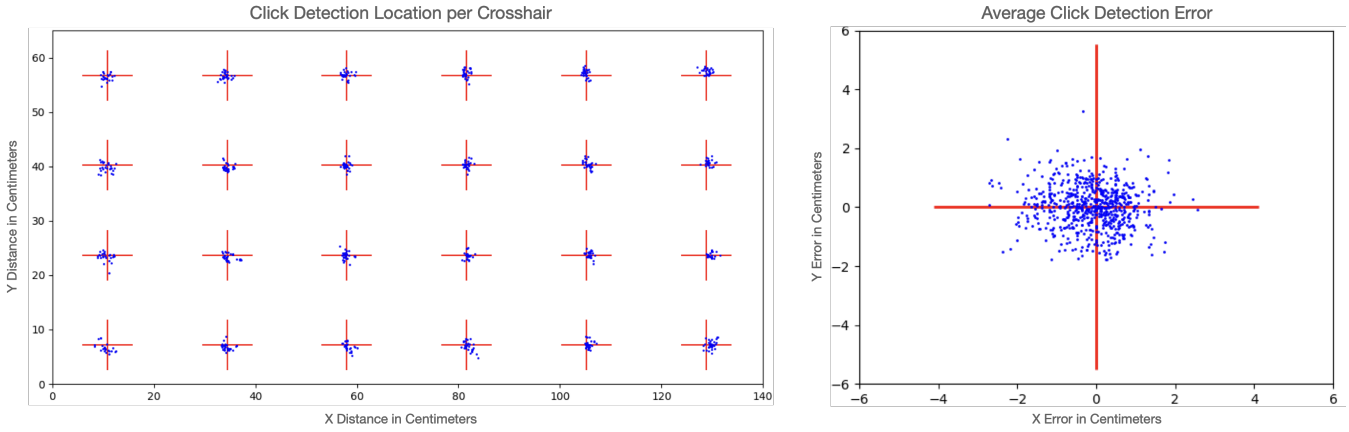

Figure 14: Results from our user study. Left: Crosshairs and detected click locations. Right: Data from all crosshairs combined.

5.3 Results

To make our results maximally comparable to prior work, we followed the procedure in OmniTouch [23], DIRECT [68] and MRTouch [71]. This consists of a single normalization step, where a mean X/Y offset is applied to all users' data post hoc to correct for any systematic offset (we note this approach is more conservative than per-user post hoc offsets as used in e.g., [26, 58]). In our data, we found a mean offset across our ten participants of 1.18 mm to the left and 10.08 mm down. With this correction applied, our mean euclidean error is 9.81 mm (SD=2.77). These results can be seen in Figure 14, left, with the crosshairs visualized at their real world coordinates and the detected click locations plotted as blue dots. On the right is a visualization of all 720 click trials overlaid onto a unified crosshair.

Across our 720 click trials, the system correctly detected a click on the first attempt in 698 trials (96.9%). In 15 trials, the user had to click again (i.e., twice), and in 7 trials three clicks were needed to advance. This results in an overall mean segmentation accuracy of 96.1% (i.e., 720/749 attempts).

6 DISCUSSION

It is difficult to measure the exact quantitative contribution of every improvement described above, as they inter-depend and interact in complex ways (i.e., one cannot simply A/B test different parts of the pipeline). Nonetheless, from our experience in developing the system, we can speak broadly to the relative value of each contribution.

The most important element in the success of our pipeline was utilizing the finger denting effect (Section 3.1), which helps us accurately localize the finger tip and detect touch events. Without this, the pipeline would not function at all. Second most impactful was multi-frame super resolution (Section 4.6) that increased depth map resolution, allowing us to more accurately resolve the location of the fingertip (a weakness in many prior systems, which had to extrapolate finger tip position).

Next most useful were techniques that narrowed our search field to a small region, where we could more extensively search for finger inputs and disregard finger-like objects and noise. This includes the hand finding stage (Section 4.7), which narrows our image to just a square around the hand, and then a finger region of interest (Section 4.10). It is inside this very small region where we run our click detection and touch localization steps (Section 4.11), which would fail if run on the full depth map.

To further decrease the possibility of false positives, our system includes two denoising steps, which were helpful, but not critical. Environment geometry subtraction (Section 4.8) helps to reduce scene complexity by removing the background (which might contain finger-like objects and complex geometry in general) and also "flatting" surface gradients (due to non-perpendicular placement of the camera). In other words, the later stages of our pipeline can essentially treat the entire scene as flat, with only the user's body appearing as a significant deformation in the depth map. Secondarily, frequency-based denoising (Section 4.9) [74] helps to smooth noise particular to the Kinect v2. However, we found this step was only useful in particularly noisy scenes.

Finally, and what we found to be least useful, are steps in our pipeline that improve the quality and stability of the depth map. This includes mitigating thermal drift (Section 4.3), kernel density estimation (Section 4.4) [38], and temporal averaging (Section 4.5). All three of these methods have been separately proven to help make Kinect v2 depth readings more accurate. However, in practice, our pipeline was not particularly sensitive to more-global variations and drift in depth estimation.

7 LIMITATIONS AND FUTURE WORK

First and foremost are limitations that are inherent to the techniques we chose to use. In particular, FarOut Touch relies on the finger denting effect that we discovered through our exploration and uses this to localize touches at longer ranges than prior work. The effect is enhanced or reduced by the room geometry surrounding the touch surface — the more surfaces the IR signal has to bounce off, the more pronounced the effect becomes. This means that our system is less effective in wide open spaces, although appears to perform well in natural living and working areas. In order to increase the effectiveness of our pipeline in wide open areas, one solution

could be to add more objects or include small special purpose reflectors that could greatly enhance multipath. Additionally, as our pipeline relies on small changes in depth, it becomes unreliable on deformable surfaces or those with large gradient changes, such as furniture. For this reason, we recommend using our pipeline on flat, rigid surfaces such as walls, tables, and countertops.

We also found that the finger denting effect is significantly weaker beyond 3.5 meters, and we suggest this as an upper bound in range. Another limiting factor was the sensor hardware itself, and the amount of noise present in the depth map. At longer ranges, the noise in contemporary depth sensors makes it nearly impossible to segment the finger when touching a surface (i.e, sensor noise is >1 cm, while the thickness of a finger is ~1 cm). Additionally, our pipeline requires line-of-sight to a touching finger, yet is easy for the body or the arm to occlude the view — an issue faced by all camera-based ad hoc touch tracking systems.

There are also multiple system-level limitations that exist because of the architecture of our pipeline. We combat potential errant click detection by requiring five frames of convergence to trigger a click event, which means that in our 30 FPS pipeline, users must keep their finger in one spot for about 167 milliseconds. This contributes to our system's 233 ms motion-to-photo latency, which is longer than any generally accepted threshold for a "real time" system. Although this latency may be acceptable for "click" events, interactive swiping and dragging actions would be unacceptably delayed. These issues could be rectified if the noise in the depth map were reduced, rendering multi-frame averaging and convergence unnecessary.

Another limitation of our present system is that it is currently configured to detect only one finger touch at a time and does not support multitouch interactions. We note this is not an inherent limitation, and our technique could be extended to process multiple finger contacts. Additionally, although our low cost system consists of only one depth camera mounted on a pan-tilt rig, it is still computationally expensive to run. We are utilizing a laptop with a built in GPU to handle the pipeline, which means that our system would require considerable re-engineering before it could run on a commodity IoT device, such as a smart speaker. Fortunately, these devices continue to grow in compute power and dedicated depth-processing ASICs (like that seen in the Microsoft HoloLens) could enable realtime performance on consumer devices in the future.

8 CONCLUSION

We have presented FarOut Touch, a new ad hoc touch tracking pipeline that focuses on extending the range of touch sensing. Our design, based off an interesting finger "denting" signal we identified, followed the marginal gains philosophy of making many incremental improvements that compound to make a significant system contribution. In our study, we show that FarOut Touch offers competitive touch accuracy at a distance of 3 m – less than 1 cm mean euclidean error with a touch detection rate of 96.1%. Our evaluated range of 3 m is roughly twice as far as any prior work using depth sensors. While constituting a useful stride towards practical room-scale, ad hoc touch sensing, to fully realize this vision probably requires on the order of 5 m sensing range. From our experience deeply considering each step of the pipeline, we believe that improvements to sensing hardware will likely yield the most value in the future.

REFERENCES

[1] Yomna Abdelrahman, Alireza Sahami Shirazi, Niels Henze, and Albrecht Schmidt. 2015. Investigation of material properties for thermal imaging-based interaction. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. 15–18.

[2] Cyrus S Bamji, Patrick O'Connor, Tamer Elkhatib, Swati Mehta, Barry Thompson, Lawrence A Prather, Dane Snow, Onur Can Akkaya, Andy Daniel, Andrew D Payne, et al. 2014. A 0.13 μm CMOS system-on-chip for a 512× 424 time-of-flight image sensor with multi-frequency photo-demodulation up to 130 MHz and 2 GS/s ADC. *IEEE Journal of Solid-State Circuits* 50, 1 (2014), 303–319.

[3] Hrvoje Benko and Andrew D Wilson. 2009. DepthTouch: using depthsensing camera to enable freehand interactions on and above the interactive surface. In *In Proceedings of the IEEE Workshop on Tabletops and Interactive Surfaces*. Citeseer.

[4] Timo Breuer, Christoph Bodensteiner, and Michael Arens. 2014. Low-cost commodity depth sensor comparison and accuracy analysis. In *Electro-Optical Remote Sensing, Photonic Technologies, and Applications VIII; and Military Applications in Hyperspectral Imaging and High Spatial Resolution Sensing II*, Vol. 9250. International Society for Optics and Photonics, 92500G.

[5] Leah Buechley, David Mellis, Hannah Perner-Wilson, Emily Lovell, and Bonifaz Kaufmann. 2010. Living wall: programmable wallpaper for interactive spaces. In *Proceedings of the 18th ACM international conference on Multimedia*. 1401–1402.

[6] Alex Butler, Shahram Izadi, and Steve Hodges. 2008. SideSight: multi-" touch" interaction around small devices. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*. 201–204.

[7] Arturo Cadena, Rubén Carvajal, Bruno Guamán, Roger Granda, Enrique Peláez, and Katherine Chiluiza. 2016. Fingertip detection approach on depth image sequences for interactive projection system. In *2016 IEEE Ecuador Technical Chapters Meeting (ETCM)*. 1–6. https://doi.org/10.1109/ETCM.2016.7750827

[8] David Capel. 2004. Image mosaicing. In *Image Mosaicing and super-resolution*. Springer, 47–79.

[9] Chongyu Chen, Jianfei Cai, Jianmin Zheng, Tat-Jen Cham, and Guangming Shi. 2013. A color-guided, region-adaptive and depth-selective unified framework for Kinect depth recovery. In *2013 IEEE 15th International Workshop on Multimedia Signal Processing (MMSP)*. IEEE, 007–012.

[10] James Clear. 2018. *Atomic habits: Tiny changes, remarkable results: An easy & proven way to build good habits & break bad ones*. Avery.

[11] Beginner's Mind Collective and David Shaw. 2012. Makey Makey: improvising tangible and nature-based user interfaces. In *Proceedings of the sixth international conference on tangible, embedded and embodied interaction*. 367–370.

[12] Andrea Corti, Silvio Giancola, Giacomo Mainetti, and Remo Sala. 2016. A metrological characterization of the Kinect V2 time-of-flight camera. *Robotics and Autonomous Systems* 75 (2016), 584–594.

[13] Kyis Essmaeel, Luigi Gallo, Ernesto Damiani, Giuseppe De Pietro, and Albert Dipanda. 2012. Temporal Denoising of Kinect Depth Data. *8th International Conference on Signal Image Technology and Internet Based Systems, SITIS 2012r*, 47–52. https://doi.org/10.1109/SITIS.2012.18

[14] Georgios D Evangelidis and Emmanouil Z Psarakis. 2008. Parametric image alignment using enhanced correlation coefficient maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 10 (2008), 1858–1865.

[15] Jerry Alan Fails and Dan Olsen Jr. 2002. Light widgets: interacting in everyday spaces. In *Proceedings of the 7th international conference on Intelligent user interfaces*. 63–69.

[16] Martin Fischbach, Hendrik Striepe, Marc Erich Latoschik, and Birgit Lugrin. 2016. A low-cost, variable, interactive surface for mixed-reality tabletop games. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology*. 297–298.

[17] William T Freeman, Egon C Pasztor, and Owen T Carmichael. 2000. Learning low-level vision. *International journal of computer vision* 40, 1 (2000), 25–47.

[18] Markus Funk, Stefan Schneegass, Michael Behringer, Niels Henze, and Albrecht Schmidt. 2015. An interactive curtain for media usage in the shower. In *Proceedings of the 4th International Symposium on Pervasive Displays*. 225–231.

[19] Peter Fürsattel, Simon Placht, Michael Balda, Christian Schaller, Hannes Hofmann, Andreas Maier, and Christian Riess. 2015. A comparative error analysis of current time-of-flight sensors. *IEEE Transactions on Computational Imaging* 2, 1 (2015), 27–41.

[20] Chunle Guo, Chongyi Li, Jichang Guo, Runmin Cong, Huazhu Fu, and Ping Han. 2018. Hierarchical features driven residual learning for depth map super-resolution. *IEEE Transactions on Image Processing* 28, 5 (2018), 2545–2557.

[21] David Hall, Derek James, and Nick Marsden. 2012. Marginal gains: Olympic lessons in high performance for organisations. *HR Bulletin: Research and Practice* 7, 2 (2012), 9–13.

[22] EJHBR Harrell. 2015. How 1% performance improvements led to Olympic gold. *Harvard Business Review* 30 (2015).

[23] Chris Harrison, Hrvoje Benko, and Andrew D Wilson. 2011. OmniTouch: wearable multitouch interaction everywhere. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. 441–450.

[24] Chris Harrison and Scott E Hudson. 2008. Scratch input: creating large, inexpensive, unpowered and mobile finger input surfaces. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*. 205–208.

[25] Nadia Haubner, Ulrich Schwanecke, Ralf Dörner, Simon Lehmann, and Johannes Luderschmidt. 2013. Detecting interaction above digital tabletops using a single depth camera. *Machine vision and applications* 24, 8 (2013), 1575–1587.

[26] Christian Holz and Patrick Baudisch. 2010. The generalized perceived input point model and how to double touch accuracy by extracting fingerprints. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 581–590.

[27] Michal Irani and Shmuel Peleg. 1991. Improving resolution by image registration. *CVGIP: Graphical models and image processing* 53, 3 (1991), 231–239.

[28] Hiroshi Ishii, Craig Wisneski, Ben Orbanes, Julian andChun, and Joe Paradiso. 1999. PingPongPlus: design of an athletic-tangible interface for computer-supported cooperative play. In *Proceedings of the 1999 CHI Conference on Human Factors in Computing Systems*. 394–401.

[29] Daisuke Iwai and Kosuke Sato. 2005. Heat sensation in image creation with thermal vision. In *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*. 213–216.

[30] Daisuke Iwai and Kosuke Sato. 2011. Document search support by making physical documents transparent in projection-based mixed reality. *Virtual reality* 15, 2-3 (2011), 147–160.

[31] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. 2011. KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. 559–568.

[32] Zhongyu Jiang, Huanjing Yue, Yu-Kun Lai, Jingyu Yang, Yonghong Hou, and Chunping Hou. 2021. Deep edge map guided depth super resolution. *Signal Processing: Image Communication* 90 (2021), 116040.

[33] Shaun K Kane, Daniel Avrahami, Jacob O Wobbrock, Beverly Harrison, Adam D Rea, Matthai Philipose, and Anthony LaMarca. 2009. Bonfire: a nomadic system for hybrid laptop-tabletop interaction. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*. 129–138.

[34] Kourosh Khoshelham and Sander Oude Elberink. 2012. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors* 12, 2 (2012), 1437–1454.

[35] Daniel Kurz. 2014. Thermal touch: Thermography-enabled everywhere touch interfaces for mobile augmented reality applications. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 9–16.

[36] Gierad Laput and Chris Harrison. 2019. SurfaceSight: a new spin on touch, user, and object sensing for IoT experiences. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–12.

[37] Eric Larson, Gabe Cohn, Sidhant Gupta, Xiaofeng Ren, Beverly Harrison, Dieter Fox, and Shwetak Patel. 2011. Heatwave: thermal imaging for surface user interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2565–2574.

[38] Felix Järemo Lawin, Per-Erik Forssén, and Hannes Ovrén. 2016. Efficient Multi-Frequency Phase Unwrapping using Kernel Density Estimation. In *European Conference on Computer Vision (ECCV)*. Springer International Publishing AG, Amsterdam. VR Projects: Learnable Camera Motion Models, 2014-5928, Energy Models for Computational Cameras, 2014-6227.

[39] Joe Marshall, Tony Pridmore, Mike Pound, Steve Benford, and Boriana Koleva. 2008. Pressing the flesh: Sensing multiple touch and finger pressure on arbitrary surfaces. In *International Conference on Pervasive Computing*. Springer, 38–55.

[40] Sundar Murugappan, Niklas Elmqvist, and Karthik Ramani. 2012. Extended multitouch: recovering touch posture and differentiating users using a depth camera. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*. 487–496.

[41] J. A. Paradiso, Che King Leo, N. Checka, and Kaijen Hsiao. 2002. Passive acoustic sensing for tracking knocks atop large interactive displays. In *SENSORS, 2002 IEEE*, Vol. 1. 521–527 vol.1. https://doi.org/10.1109/ICSENS.2002.1037150

[42] Joseph A Paradiso and Che King Leo. 2005. Tracking and characterizing knocks atop large interactive displays. *Sensor Review* (2005).

[43] DT Pham, M Al-Kutubi, Z Ji, M Yang, Z Wang, and S Catheline. 2005. Tangible acoustic interface approaches. In *Proceedings of IPROMS 2005 Virtual Conference*. Citeseer, 497–502.

[44] DT Pham, Z Wang, Z Ji, M Yang, M Al-Kutubi, and Stefan Catheline. 2005. Acoustic pattern registration for a new type of human-computer interface. In *Proc. Virtual Conf., IPROMs*.

[45] Duc Truong Pham, Ze Ji, Ming Yang, Zuobin Wang, and Mostafa Al-Kutubi. 2007. A novel human-computer interface based on passive acoustic localisation. In *International Conference on Human-Computer Interaction*. Springer, 901–909.

[46] Fei Qi, Junyu Han, Pengjin Wang, Guangming Shi, and Fu Li. 2013. Structure guided fusion for depth map inpainting. *Pattern Recognition Letters* 34, 1 (2013), 70–76.

[47] Hamed Sarbolandi, Damien Lefloch, and Andreas Kolb. 2015. Kinect range sensing: Structured-light versus Time-of-Flight Kinect. *Computer vision and image understanding* 139 (2015), 1–20.

[48] Munehiko Sato, Ivan Poupyrev, and Chris Harrison. 2012. Touché: enhancing touch interaction on humans, screens, liquids, and everyday objects. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 483–492.

[49] Katie Seggerman and Andy Perez. 2014. Table Top Computer Interface Using A Depth Sensor.

[50] John Sell and Patrick O'Connor. 2014. The xbox one system on a chip and kinect sensor. *IEEE Micro* 34, 2 (2014), 44–53.

[51] Adam Smith, Hari Balakrishnan, Michel Goraczko, and Nissanka Priyantha. 2004. Tracking moving devices with the cricket location system. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services*. 190–202.

[52] Wanbin Song, Anh Vu Le, Seokmin Yun, Seung-Won Jung, and Chee Sun Won. 2017. Depth completion for kinect v2 sensor. *Multimedia Tools and Applications* 76, 3 (2017), 4357–4380.

[53] Jean-Luc Starck, Jalal Fadili, and Fionn Murtagh. 2007. The undecimated wavelet decomposition and its reconstruction. *IEEE transactions on image processing* 16, 2 (2007), 297–309.

[54] Joshua Strickon and Joseph Paradiso. 1998. Tracking hands above large interactive surfaces with a low-cost scanning laser rangefinder. In *CHI 98 Conference Summary on Human Factors in Computing Systems*. 231–232.

[55] Naoki Sugita, Daisuke Iwai, and Kosuke Sato. 2008. Touch sensing by image analysis of fingernail. In *2008 SICE Annual Conference*. IEEE, 1520–1525.

[56] Yoshiki Takeoka, Takashi Miyaki, and Jun Rekimoto. 2010. Z-touch: an infrastructure for 3d gesture interaction in the proximity of tabletop surfaces. In *ACM International Conference on Interactive Tabletops and Surfaces*. 91–94.

[57] Carlo Tomasi, Abbas Rafii, and Ilhami Torunoglu. 2003. Full-size projection keyboard for handheld devices. *Commun. ACM* 46, 7 (2003), 70–75.

[58] Feng Wang and Xiangshi Ren. 2009. Empirical evaluation for finger input properties in multi-touch interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1063–1072.

[59] Oliver Wasenmüller and Didier Stricker. 2016. Comparison of kinect v1 and v2 depth images in terms of accuracy and precision. In *Asian Conference on Computer Vision*. Springer, 34–45.

[60] Pierre Wellner. 1993. Interacting with paper on the DigitalDesk. *Commun. ACM* 36, 7 (1993), 87–96.

[61] Andrew D Wilson. 2004. TouchLight: an imaging touch screen and display for gesture-based interaction. In *Proceedings of the 6th international conference on Multimodal interfaces*. 69–76.

[62] Andrew D Wilson. 2005. PlayAnywhere: a compact interactive tabletop projection-vision system. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*. 83–92.

[63] Andrew D Wilson. 2007. Depth-sensing video cameras for 3d tangible tabletop interaction. In *Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP'07)*. IEEE, 201–204.

[64] Andrew D Wilson. 2010. Using a depth camera as a touch sensor. In *ACM international conference on interactive tabletops and surfaces*. 69–72.

[65] Andrew D Wilson and Hrvoje Benko. 2010. Combining multiple depth cameras and projectors for interactions on, above and between surfaces. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. 273–282.

[66] Bartlomiej Wronski, Ignacio Garcia-Dorado, Manfred Ernst, Damien Kelly, Michael Krainin, Chia-Kai Liang, Marc Levoy, and Peyman Milanfar. 2019. Handheld multi-frame super-resolution. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–18.

[67] Robert Xiao, Chris Harrison, and Scott E Hudson. 2013. WorldKit: rapid and easy creation of ad-hoc interactive applications on everyday surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 879–888.

[68] Robert Xiao, Scott Hudson, and Chris Harrison. 2016. Direct: Making touch tracking on ordinary surfaces practical with hybrid depth-infrared sensing. In *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces*. 85–94.

[69] Robert Xiao, Scott Hudson, and Chris Harrison. 2017. Supporting responsive cohabitation between virtual interfaces and physical objects on everyday surfaces. *Proceedings of the ACM on Human-Computer Interaction* 1, EICS (2017), 1–17.

[70] Robert Xiao, Greg Lew, James Marsanico, Divya Hariharan, Scott Hudson, and Chris Harrison. 2014. Toffee: enabling ad hoc, around-device interaction with acoustic time-of-arrival correlation. In *Proceedings of the 16th international conference on Human-computer interaction with mobile devices & services*. 67–76.

[71] Robert Xiao, Julia Schwarz, Nick Throm, Andrew D Wilson, and Hrvoje Benko. 2018. MRTouch: Adding touch input to head-mounted mixed reality. *IEEE transactions on visualization and computer graphics* 24, 4 (2018), 1653–1660.

[72] Jingyu Yang, Xinchen Ye, Kun Li, Chunping Hou, and Yao Wang. 2014. Color-guided depth recovery from RGB-D data using an adaptive autoregressive model. *IEEE transactions on image processing* 23, 8 (2014), 3443–3458.

[73] Ying Yin. 2012. A hierarchical approach to continuous gesture analysis for natural multi-modal interaction. In *Poster in Proceedings of the 14th ACM international conference on Multimodal interaction*. 357–360. https://groups.csail.mit.edu/mug/projects/gesture_kinect/docs/fingertip-tracking.pdf

[74] Guoshen Yu and Guillermo Sapiro. 2011. DCT image denoising: a simple and effective image denoising algorithm. *Image Processing On Line* 1 (2011), 292–296.

[75] Cheng Zhang, Qiuyue Xue, Anandghan Waghmare, Ruichen Meng, Sumeet Jain, Yizeng Han, Xinyu Li, Kenneth Cunefare, Thomas Ploetz, Thad Starner, et al. 2018. Fingerping: Recognizing fine-grained hand poses using active acoustic on-body sensing. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–10.

[76] Yang Zhang, Gierad Laput, and Chris Harrison. 2017. Electrick: Low-cost touch sensing using electric field tomography. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 1–14.

[77] Yang Zhang, Chouchang Yang, Scott E Hudson, Chris Harrison, and Alanson Sample. 2018. Wall++ room-scale interactive and context-aware sensing. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–15.