Estimating 3D Finger Angle on Commodity Touchscreens

Robert Xiao **Julia Schwarz** **Chris Harrison**
Qeexo, Co.
2483 Old Middlefield Way, Suite 202, Mountain View, California 94043
info@qeexo.com

ABSTRACT

We describe a novel approach for estimating the pitch and yaw of fingers relative to a touchscreen's surface, offering two additional, analog degrees of freedom for interactive functions. Further, we show that our approach can be achieved on off-the-shelf consumer touchscreen devices: a smartphone and smartwatch. We validate our technique though a user study on both devices and conclude with several demo applications that illustrate the value and immediate feasibility of our approach.

Author Keywords
Touchscreen input; mobile devices; rich touch; capacitive sensing; finger orientation.

ACM Classification Keywords
H.5.2. Information interfaces and presentation (e.g., HCI): User Interfaces: Input devices and strategies.

INTRODUCTION

Today's touch interfaces are primarily driven by the 2D location of touch events. As we describe in Related Work, there are many other dimensions of touch that can be captured and utilized interactively. By increasing the richness of touch input, users can do more in the same small space, potentially enabling richer applications. In this work, we describe a new method that estimates a finger's angle relative to the screen. The angular vector is described using two angles – *altitude* and *azimuth* – more colloquially referred to as *pitch* and *yaw*. Our approach works in tandem with conventional multitouch finger tracking, offering two additional analog degrees of freedom for a single touch point.

In this work, we present a new algorithm for simultaneously estimating finger pitch and yaw across a wide range of poses, from flat to perpendicular. Uniquely, it uses only data provided by commodity, off-the-shelf touch devices, requiring no additional hardware or sensors. We prototyped our solution on two platforms – a smartphone and a smartwatch – each fully self-contained and operating in real-time. We quantified the accuracy of our technique through a user study, and explored the feasibility of our approach through example applications and interactions (see Video Figure).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
ITS '15, November 15-18, 2015, Funchal, Portugal
© 2015 ACM. ISBN 978-1-4503-3899-8/15/11···$15.00
DOI: http://dx.doi.org/10.1145/2817721.2817737

RELATED WORK

There has been significant research into enhancing interaction on touch devices. One approach is to use conventional touch data in combination with spatial or temporal sequences, for example, tap-and-hold and multi-finger chording gestures [2,11]. More experimental are efforts that capture e.g., pressure [12], shear forces [8], shape of the hands [4], rolling motions of stationary fingers [14], and what part of the finger touched the screen [7].

In addition to X/Y finger tracking, most modern touchscreens fit an ellipse to finger contact areas, obtaining ellipse axes and orientation [1]. The ellipse size is often used as a proxy for "pressure", and can also trigger different interaction functions via thresholds [3]. Ellipse orientation is similar to yaw, though true finger direction is ambiguous (orientation is not directional, thus offering two possible vectors). More importantly, as we will show, the ellipse orientation varies with finger pitch. Thus, prior work using ellipse orientation for yaw (e.g., [16,17]) required fingers to lie nearly flat on the screen surface. We improve upon these results (see Video Figure, where yaw is tracked even when fingers are *perpendicular*), and further add finger pitch tracking, which no touchscreens report today.

It is important to note that, while prior systems have achieved pitch and yaw tracking, all use *special hardware* that is impractical for mobile devices. For example, PointPose [10] used a depth camera mounted obliquely to the touchscreen to capture finger "rotation and tilt". Similarly, KinectTouch [5] also uses a depth camera, but it is mounted above the display. Z-Touch [15] uses a series of multiplexed infrared line lasers to create a shallow-field depth sensing touchscreen, capable of recovering finger angle. Using a commercial-grade fingerprint scanner, Holz and Baudisch [9] were able to estimate pitch and yaw based on the fingerprint patch that was visible, which was used to improve

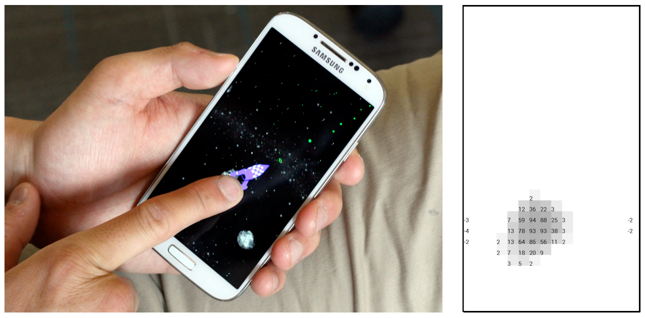


Figure 1. Left: A ship is piloted through 3D space using the finger's vector. Right: 16x28 capacitive image from the smartphone's touchscreen.

targeting accuracy. Zaliva [18] uses neural networks to estimate finger pose, though no hardware or results are described. Finally, and most similar to our work, is Angle-Pose [13], which used a 4x6 grid of capacitance-to-ground sensing electrodes (offering high SNR) and a particle filter to estimate a finger's 3D pose. This setup is used to evaluate how pose information can improve 2D targeting accuracy, but the pitch/yaw estimates themselves were not evaluated.

In addition to being the first approach that can run on commodity devices, we are also the first to consider how pitch/yaw can be valuable at smartwatch scales, and also evaluate our approach's performance. Further, we bring new interactions to light through our example applications.

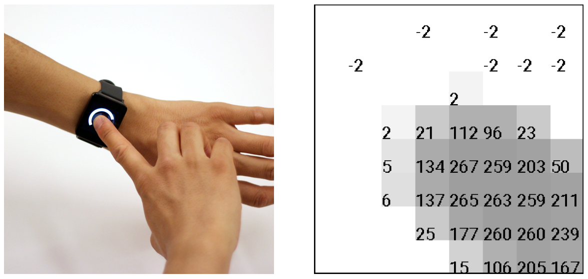
IMPLEMENTATION

Projected capacitive sensing is the most pervasive touchscreen technology today. It works by sensing disturbances in a projected electric field caused by a proximate capacitive object (e.g., a fleshy finger). Field strength diminishes significantly with increasing distance. However, importantly, the sensitivity is not zero - nearby finger parts will still produce weak signals. This produces a characteristic "comet" artifact for non-perpendicular fingers (Figures 1 and 2).

More specifically, our implementation uses the *capacitive image* (Figures 1 and 2, right) provided by the touchscreen. Put simply, this is the capacitance measured at each point of a touch sensor's capacitive grid; the data is used internally by the touchscreen controller to detect touch events. The capacitive image is also provided as a debugging feature by many touchscreen controllers. We gained access to the capacitive image on our proof-of-concept devices (both of which run Android) by modifying the Linux kernel. Of note, although the capacitive image is not directly accessible in Apple iOS and Windows Phone, the underlying hardware almost certainly supports it.

Proof-of-Concept Platforms

We selected two popular devices to represent smartphone and smartwatch form factors. For the phone, we chose a Samsung Galaxy S4 (Figure 1), which features a 16x28 capacitive touch grid over top of a 5.0" 1080x1920 pixel screen. On this device, we can poll the capacitive image 25 times per second. For the smartwatch, we chose an LG 'G' smartwatch (Figure 2), which has an 8x8 capacitive grid on top of a 1.65" 280x280 pixel screen. Due to the smaller image size, we can poll the capacitive image at 60 FPS.



			-2		-2		-2
	-2				-2	-2	-2
				2			
		2	21	112	96	23	
		5	134	267	259	203	50
		6	137	265	263	259	211
			25	177	260	260	239
				15	106	205	167

Figure 2. Left: Yaw controlling volume level. Right: 8x8 capacitive image from the smartwatch's touchscreen.

Feature Extraction

We use the direction and magnitude of the "comet" shape to estimate the pitch and yaw of a finger. Starting from each identified touch point, we flood-fill within the capacitive image to find the set of non-zero pixels sensing the finger B (the touch blob). We discard all grid points that have a capacitance delta (compared with background) below a noise threshold (3 pF in our implementation).

For each remaining point in the touch blob, let x and y denote its position and let z denote the capacitance deviation at that point. We derive seven datasets:

- The power-transformed datasets $S_i = \{(x, y, z^i) | x, y, z \in B\}$ for $i = 0, 1, 2$
- The thresholded power-transformed datasets $T_i = \{(x, y, z^i) | x, y, z \in B, z \geq 30\}$ for $i = 0, 1, 2$
- The log-transformed dataset $U = \{(x, y, \ln z) | x, y, z \in B\}$

Note that $S_1 = B$. We then fit an ellipsoid to each dataset by computing the centroid $(\bar{x}, \bar{y})$ where:

$$\bar{x} = \frac{\sum_{x,y,z} xz}{\sum_{x,y,z} z}, \bar{y} = \frac{\sum_{x,y,z} yz}{\sum_{x,y,z} z}$$

and the central image moments $\mu_{02}, \mu_{20}, \mu_{11}$ where:

$$\mu_{ij} = \sum_{x,y,z} (x\text{-}\bar{x})^i (y\text{-}\bar{y})^j z$$

From these moments, we compute the ellipsoid orientation:

$$\theta = \text{-}\frac{1}{2}\tan^{-1}\frac{2\mu_{11}}{\mu_{20}\text{-}\mu_{02}}$$

and the eigenvalues:

$$\lambda_{\pm} = \frac{\mu_{20} + \mu_{02} \pm \sqrt{4\mu_{11} + (\mu_{20}\text{-}\mu_{02})^2}}{2}$$

and finally the ellipsoid's eccentricity:

$$\varepsilon = \sqrt{1 - \lambda_{-}/\lambda_{+}}$$

We combine these parameters with the distance and angle between the touch point and the ellipsoid centroid to obtain a set of 6 features for each of our 7 datasets. We use these 42 features to compute the pitch and yaw.

Pitch Estimation

The electric field strength decreases as the square of the distance, reducing the measured capacitance. Due to the differing dielectric constants of glass and air, the decrease does not perfectly fit an inverse-square law. Thus, to produce reliable estimates of finger pitch, we used Weka [6] to train a Gaussian process regression that takes our 42 features as input and produces an estimate of the pitch angle.

Yaw Estimation

When a user's finger is low to the screen, we see the characteristic "comet" shape, in which the major axis of the ellipsoid of the sensed touch pixels extends *parallel* to the finger's yaw angle (Figure 3, left two panels). However, when the user's finger is sufficiently vertical (pitches near 90º), the fingertip's contact area with the screen forms a small, flattened ellipsoid, with a major axis *perpendicular* to the finger's yaw angle (Figure 3, far right). In between, for angles between 45º and 60º, the ellipsoid is nearly a perfect circle (Figure 3, center right), allowing for accurate pitch,

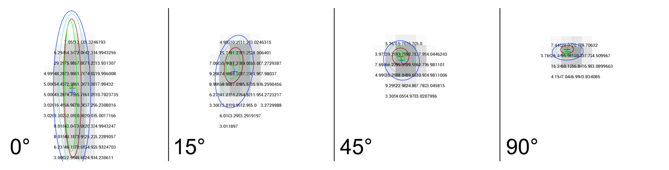


Figure 3. A finger at varying pitches, but constant yaw (0°). Note how the S_i ellipsoids vary; in particular, how they are vertically elongated below 45°, near circular at 45°, and horizontally elongated above 45°.

but poor yaw estimation. To account for these differences, we implemented a simple heuristic: the yaw angle is simply the orientation of the S_1 ellipsoid, with a 90º correction applied if the pitch exceeds 50º.

USER STUDY

Prior to running our study, the three authors used the procedure described below to collect data, which was used to train the pitch regression model. Thus, the study (purposely) contains *no* per-user training or calibration. Also, pitch and yaw estimates were computed *live*, for a real time result (i.e., no post hoc corrections or algorithm tweaks).

We recruited 10 participants (2 female, mean age 35.2) to participate in a 20-minute user study. Participants were asked to replicate a series of pitch and yaw positions. Specifically, we tested 7 finger pitches from 0° (flat on the screen) to 90° (perpendicular) in 15° increments, and tested 9 finger yaw from -60° to +60° in 15° increments (a range we found comfortable to replicate in piloting). Each of these 63 combinations was tested on both our phone and watch prototypes. For each combination of device and pitch angle (randomized), users performed all yaw angles in sequence from -60 to 60 to streamline collection.

In each trial, users placed their fingertip in the center of the display. The requested yaw was displayed on screen using an arrow. However, we found no reliable way to illustrate a requested pitch. In response, we laser-cut plastic wedges for all of our tested angles. Participants placed the appropriate wedge under their finger to establish the correct finger pitch. Once satisfied, the participant removed the wedge (to avoid capacitive interference) and then the experimenter triggered the data collection function.

Two participants had long nails, which precluded the touchscreen from detecting their input at 90º pitches. This inability to sense is not related to our method, but rather the touchscreen hardware and sensitivity settings. Minus these two participants' 90° pitch blocks, we obtained 1224 pitch/yaw trials in total. There were no significant performance differences between users.

RESULTS

As seen by the relatively flat lines in Figure 4 left, the angular accuracy of both pitch and yaw across the full range of yaws we tested varies little. This is also true of pitch when analyzed across the full range of pitches (Figure 4, right, solid lines). However, yaw accuracy varies significantly with changing pitch, as seen by the "hump" in the dashed lines in Figure 4 right. This is due to the appearance of fingers at intermediate pitches (around 45°) as circular patches on the capacitive image (Figure 3). This circular shape is useful in inferring that the finger is at an intermediate pitch, leading to high pitch accuracies. However, it provides little data for inferring yaw, and as such, error is high (Figure 4, right, dashed lines).

The smartphone generally had superior accuracy to the smartwatch. We believe this was chiefly due to the larger touchscreen better capturing the full extent of fingers' capacitive signal (i.e., useful data simply extends off the smartwatch screen, as seen in Figure 2, right), The phone had a mean angular pitch error of 9.7° (SD=8.9°), while the watch had 14.5° (SD=12.6°). Mean angular yaw error was 26.8° for the phone and 31.7° for the watch. However, as noted previously, estimating yaw when pitches are around 50° is near impossible, leading to misleading averages, so the latter averages can be misleading. If we look only at mean angular yaw error for pitches 30° or less, the phone and watch achieve 10.6° (SD=18.9°) and 13.1° (SD=20.1°) mean angular yaw error respectively. Similar results are found when the finger is mostly perpendicular.

EXAMPLE APPLICATIONS

Holz and Baudisch [9] found that finger pitch systematically affected targeting accuracy, and note that if pitch was known, targeting accuracy could be significantly increased. Our pitch estimation approach could be readily deployed onto today's devices (e.g., a software update) to enable such accuracy gains. We also believe pitch and yaw are also exciting first-class input dimensions, used to trigger interactive functionality directly. This is especially true on smartwatches, where small screens make spatial gestures (e.g., pigtails) and multi-finger chording cumbersome. To more concretely demonstrate these interactive possibilities, we created several demo applications (see also Video Figure).

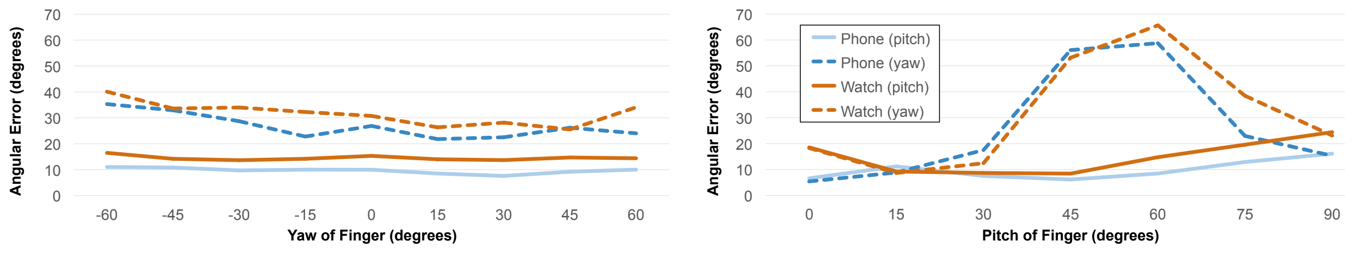


Figure 4. Accuracy of pitch (solid lines) and yaw (dashed lines) for smartphone (blue lines) and smartwatch (orange lines) prototypes. Summarized here is angular accuracy with respect to the range of yaws (left) and pitches (right) evaluated.

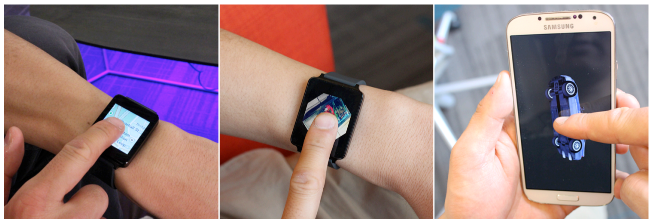


Figure 5. Left: map panning and zooming. Center: pan and rotate images. Right: 3D Object manipulation

Twist to Set Value
Setting analog values on contemporary touch interfaces generally means manipulating (through finger translation) a small slider or “spinner” widget. By tracking yaw, “twist” sensitive controls (demonstrated in [16,17] using orientation) can be created, which provide relative or absolute, in-place, analog input. As a demo, we created a volume control for our smartwatch (Figure 2). This interaction could also be used to set a kitchen timer, modify screen brightness, aid in color selection, and other analog manipulation tasks.

Pan and Zoom
Today’s smartwatch interfaces generally lack mapping applications that can pan *and* zoom. With such limited screen space, two-finger pinch to zoom is cumbersome; for example, the Apple Watch instead moves zooming functionality to a mechanical thumb wheel. With pitch and yaw tracking, users can translate the map with one finger panning and zoom with one-finger twists (Figure 5, left).

Pan and Rotate
We also created a photo album viewer for our smartwatch prototype. Photos can be navigated with left and right swipes, while relative pitch movement controls the image crop. In a direct manipulation manner, users can twist the image with a single finger to rotate it (Figure 5, center).

3D Manipulation
Pitch and yaw lends itself well to 3D object manipulation. We created a demo where users can rotate an object *on all three* rotational axes without translating the finger (Figure 5, right). Relative pitch motions along the horizontal and vertical axes control rotation about the Y- and X-axis respectively. Yaw naturally controls rotation about the Z-axis.

Vector into 3D space
In contemporary touch interfaces, a fingertip pressed against a screen is thought of as a point on a 2D plane. However, thinking more holistically about the *entire* finger, it is also a vector pointing into 3D space. As a demonstration of this, we created a simple game: a spaceship is rendered on the tip of the finger and tracks with the finger’s 3D pointing direction; the goal is to shoot and avoid asteroids (Figure 1).

CONCLUSIONS AND FUTURE WORK
In this work, we described and evaluated a new method of tracking finger angle using unmodified off-the-shelf touchscreens. We provided a number of example applications to demonstrate interactions made possible using finger angle information. In the future, hover-sensing touchscreens could be used to further extend sensing above the display plane, enabling improved accuracy across a wider range of pitch angles, while future improvements in touchscreen accuracy and sensor density could improve angle estimations and enable more fine-grained interactions.

REFERENCES

1. Android API. http://developer.android.com/reference/android/view/MotionEvent.html#AXIS_ORIENTATION
2. Apple, Inc. Multi-touch gesture dictionary (2008). US Patent 20070177803.
3. Boring, S., Ledo, D., Chen, X., Marquardt, N., Tang, A. and Greenberg, S. The fat thumb: using the thumb's contact size for single-handed mobile interaction. In *Proc. MobileHCI '12*. 207-208.
4. Cao, X., A. Wilson, R. Balakrishnan, K. Hinckley, S. Hudson. ShapeTouch: Leveraging contact shape on interactive surfaces. In *Proc. ITS ’08*. 129–136.
5. Dippon, A and Klinker, G. KinectTouch: accuracy test for a very low-cost 2.5D multitouch tracking system. In *Proc. ITS '11*. 49-52.
6. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. H. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1), 2009.
7. Harrison, C., Schwarz, J. and Hudson S. E. TapSense: Enhancing Finger Interaction on Touch Surfaces. In *Proc. UIST '11*. 627-636.
8. Heo, S. and Lee, G. Force gestures: augmented touch screen gestures using normal and tangential force. In *Proc. UIST’11*. 621-626.
9. Holz, C. and Baudisch, P. The generalized perceived input point model and how to double touch accuracy by extracting fingerprints. In *Proc. CHI '10*. 581-590.
10. Kratz, S., Chiu, P and Back, M. PointPose: finger pose estimation for touch input on mobile devices using a depth sensor. In *Proc. ITS '13*. 223-230.
11. Lepinski, J.G., Grossman, T., and Fitzmaurice, G. The design and evaluation of multitouch marking menus. In *Proc. CHI '10*. 2233-2242.
12. Ramos, G., Boulos, M., and Balakrishnan, R. Pressure widgets. In *Proc. CHI '04*. 487-494.
13. Rogers, S., Williamson, J., Stewart, C., and Murray-Smith, R. AnglePose: robust, precise capacitive touch tracking via 3d orientation estimation. In *Proc. CHI '11*. 2575-2584
14. Roudaut, A., Lecolinet, E. and Guiard, Y. MicroRolls: expanding touch-screen input vocabulary by distinguishing rolls vs. slides of the thumb. In *Proc. CHI '09*. 927-936.
15. Takeoka, Y., Miyaki, T., and Rekimoto, J. Z-touch: an infrastructure for 3d gesture interaction in the proximity of tabletop surfaces. In *Proc. ITS '10*. 91-94.
16. Wang, F. and Ren, X. Empirical evaluation for finger input properties in multi-touch interaction. In *Proc. CHI '09*. 1063-1072.
17. Wang, F., Cao, X., Ren, X. and Irani, P. Detecting and leveraging finger orientation for interaction with direct-touch surfaces. In *Proc. UIST '09*. 23-32.
18. Zaliva, V. 3D finger posture detection and gesture recognition on touch surfaces. In *Proc. ICARCV '12*. 359-364.