Contactless Gesture Recognition for Mobile Devices

Heng-Tze Cheng*
Electrical and Computer Engineering
Carnegie Mellon University
hengtze@cmu.edu

An Mei Chen, Ashu Razdan, Elliot Buller
Office of The Chief Scientist
Qualcomm Incorporated
{anc, arazdan, ebuller}@qualcomm.com

ABSTRACT

While gesture interfaces become pervasive, most existing approaches are undesirable for mobile devices because of the high power consumption, or the inconvenience that users need to wear/hold specific sensors. In this paper, we present a contactless gesture recognition system for mobile devices using proximity sensors. A set of infrared signal feature extraction methods and a decision-tree-based gesture classifier are proposed. The system allows a user to interact with mobile devices using intuitive gestures, without touching the screen or wearing/holding any additional device. Evaluation results show that the system is low-power, and able to recognize gestures with over 98% precision in real time.

Author Keywords

Gesture recognition, proximity sensor, infrared LED

ACM Classification Keywords

H.5.2 Information Interfaces and Presentation: User Interfaces—*Input devices and strategies*

INTRODUCTION

Gesture-based interfaces provide an intuitive way for users to specify commands and interact with computers [6, 8]. As mobile phones and tablets become ubiquitous, there is an increasing need of an intuitive user interfaces for small-sized, resource-limited mobile devices.

Most existing gesture recognition systems can be classified into three types: *motion-based*, *touch-based*, and *vision-based* systems. For motion-based systems [11, 4], user cannot make gestures unless holding a mobile device or an external controller. Touch-based systems [12, 10] can accurately map the finger/pen positions and moving directions on the touchscreen to different commands. However, 3D gestures are not supported because all possible gestures are confined within the 2D screen surface. While the first two types of system require users to make contact with devices, vision-based systems [8, 14] using camera and computer vision techniques allow users to make intuitive gestures without touching the device. However, most vision-based systems are computationally expensive and power-consuming, which is undesirable for resource-limited mobile devices like tablets or mobile phones.

To solve the existing challenges, we present a contactless gesture recognition system using only two infrared proximity sensors. We propose a set of infrared feature extraction and gesture classification algorithms. Using the system as a gesture interface, a user can flip e-book pages, scroll web pages, zoom in/out, and play games on mobile devices using intuitive hand gestures, without touching, wearing, or holding any additional devices. The design also reduces the frequency of users' contact with devices, alleviating the wear and tear to screen surfaces.

The main contributions of the paper are: 1) The design and evaluation of a contactless gesture recognition system using only two proximity sensors. 2) The proposed infrared (IR) feature set and classifier for real-time gesture classification. 3) Reducing the power consumption of gesture recognition.

RELATED WORK

There has been extensive research on vision-based gesture recognition [8, 14], mostly focusing on the detection of hand trajectory. Although they can recognize complex gestures, they can be sensitive to background objects, color, and lighting. Robustness can be improved by adding color markers on the user's hand [5], with a tradeoff of the inconvenience to wear additional gears. Moreover, continuous video recording of a user can make one feel like under surveillance and pose a threat on user privacy.

Recently, SideSight [1] proposed an around-device multi-touch interface by placing ten IR sensors on the long edges of a small mobile device. Another related work, HoverFlow [3], used six IR sensors facing the user to capture IR image maps, and then classify gestures using dynamic time warping (DTW). In this work, we reduce the number of the required IR sensors to two and thus reduce the power consumption, which is mentioned as a critical issue in [1]. Even using the limited information from only two IR sensors, our system can achieve accurate gesture recognition using the proposed IR feature set and the classifier.

For motion-based system, one of the recent work uWave

*This work is done during the author's employment at Office of The Chief Scientist, Qualcomm Incorporated.

MIAA 2011, February 13, 2011, Palo Alto, CA, USA.

[4] match accelerometer data with gesture templates using DTW. 98.6% and 93.5% accuracy was achieved with and without template adaptation, respectively, for user-dependent gesture recognition. However, a user need to hold a device with accelerometer, and press a button to indicate start and end of a gesture. In this work, we eliminate these limitations with contactless gesture recognition.

Electromyogram-based (EMG-based) system [2, 13] is another novel way to recognize gesture patterns using electrical activity produced by skeletal muscles. However, a user must wear EMG sensors on the wrist at all times to perform gestures, which can be inconvenient and not suitable for mobile device interfaces.

SYSTEM DESIGN AND METHODS

Design Considerations

Our system is designed based on four design considerations: 1) *Automatically detect gesture boundaries*: A common challenge of gesture recognition is the uncertainty of when does a gesture begins or ends. We do not require a user to press a key to indicate the presence of a gesture since it would be inconvenient to do so. 2) *Recognition must be real-time*: Gesture interface must be very responsive, so no time-consuming postprocessing is allowed. 3) *False alarm needs to be minimized*: Executing a wrong command is generally worse than missing a command. 4) *No user-dependent model training process for new users*: Although supervised learning can optimize the performance for a specific user, collecting training data can be time consuming and not desirable for users.

Proximity Sensor Data Acquisition

We now describe each system component shown in Fig. 1. A proximity sensor consists of two IR LEDs and a IR receiver, which are placed underneath a plastic/glass screen surface, surrounded by optical barriers. The LEDs emit IR strobes in turns as two separate channels using time-division multiplexing. When a hand or any object is near, the receiver detects the reflection of the IR light, whose intensity increases as the object distance decreases. The light intensities of the two IR channels are sampled by the firmware at 100Hz.

Framing

Since the start and end of a gesture is not specified by the user, our program uses a moving window to scan the input IR intensity data and decide if any gesture signature is observed. The data is divided into 50% overlapping frames, each of which is 140 ms. After framing, three types of feature are extracted from each frame.

Infrared Feature Extraction

Inter-channel Time Delay

The feature measures the pair-wise time delay between the sensor data of two channels, which shows how a hand approaches the IR LEDs at different instants. This corresponds to different moving directions of hands (see Fig. 2 for example). The time delay t_D is calculated by finding the time shift n that yields maximum cross correlation value of two

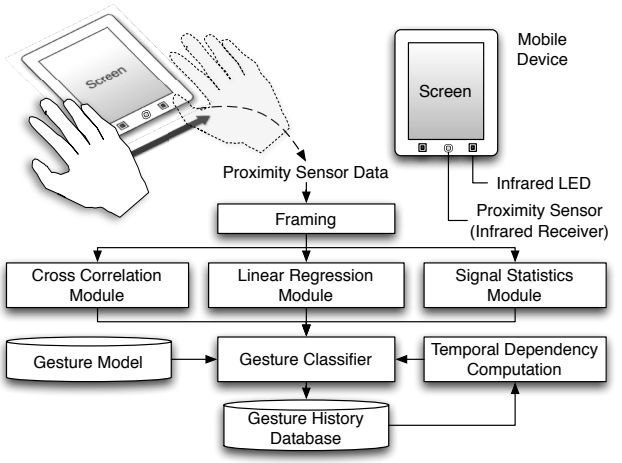


Figure 1: The architecture of the gesture recognition system.

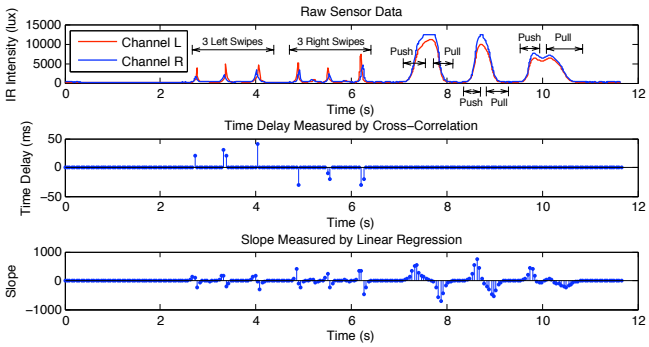


Figure 2: An example of proximity sensor data and the features.

discrete signal sequences f and g:

$$t_D = \arg\max_n \sum_{m=-\infty}^{\infty} f^*(m)g(m+n) \qquad (1)$$

Local Sum of Slopes

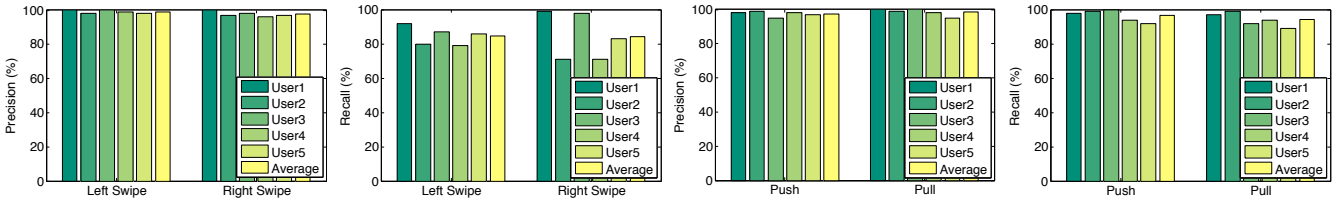
This feature estimates the local slope of the signal segment within a frame, which shows how fast the user's hand is moving toward or away from the proximity sensors. The slope is calculated by first-order linear regression, and then summed up with the slopes of the 6 previous frames. The local sum better capture the continuous trend of slopes rather than sudden changes.

Signal Statistics

The mean and variance of the raw sensor data. A high variance can be observed when a gesture is present; on the contrary, when there is no hand present or a hand hovering above, a low variance is observed.

Gesture Recognition Algorithm

After feature extraction, a decision-tree classifier shown in Fig. 3 is adopted to classify the frame as one of the gesture in the predefined gesture model, or report that no gesture is detected. We also keep a history of 7 frames to take temporal



(a) Precision of left/right swipe (b) Recall of left/right swipe (c) Precision of push/pull (d) Recall of push/pull

Figure 5: Precision and recall rate of gesture recogntion.

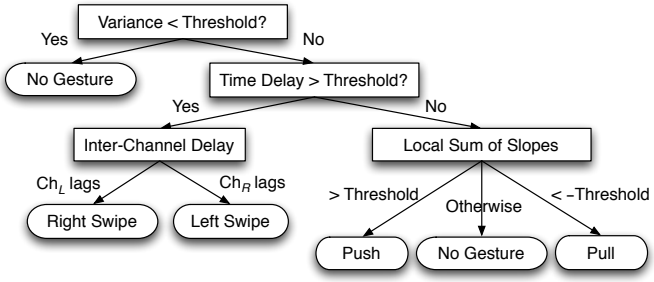
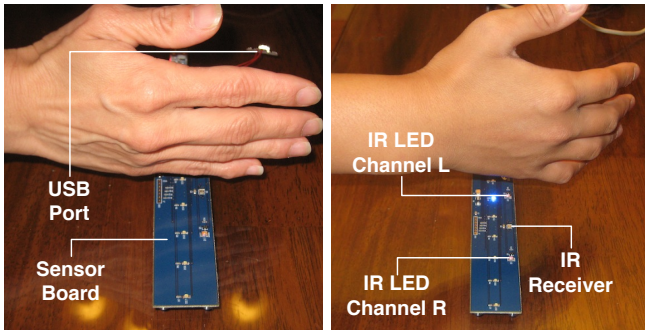


Figure 3: Illustration of the decision-tree-based gesture classifier.



Start of Gesture (Left Swipe) End of Gesture (Left Swipe)

Figure 4: A subject performed a left-swipe gesture using the prototype sensor board.

dependency between consecutive frames into consideration. For example, when a gesture is detected, the system suppress the output of the same gesture for 6 frames because it is hard for a user to make the same gesture again very quickly. Once the gesture sequence history of a user is obtained, the transition probability between gestures can also be incorporated to improve the recognition accuracy.

IMPLEMENTATION

We implemented the prototype system using Silicon Labs Si1120 infrared proximity sensor [9]. The sensor data were transmitted to a laptop through a USB serial port. The feature extraction and gesture recognition algorithm was implemented in C++. The window sizes and thresholds are empirically set through experiments to minimize the false alarm rate of the system. A picture of the prototype system and a subject performing a gesture is shown in Fig. 4.

EVALUATION

We define four essential gestures for evaluation: *left swipe*, *right swipe*, *push* (hand vertically moving vertically down toward the device), and *pull* (hand moving vertically up away from the device). The system is evaluated on a gesture dataset collected from five subjects, including four right-handed and one left-handed user. Their ages span from 20s to 40s, and one of them is female. The dataset consists of 2,000 gesture samples in total, with each user performing each of the four gesture 100 times.

Recognition Performance

We use the widely used precision/recall metric to evaluate the recognition performance:

$$precision = \frac{TP}{TP + FP} \quad (2)$$

$$recall = \frac{TP}{TP + FN} \quad (3)$$

where TP, FP, FN refer to true positive, false positive, and false negative. As shown in Fig. 5, the system achieved 98% precision in average, and is robust from user to user. The high precision implies low false alarm rate, which is ideal for gesture recognition because executing a wrong command is usually worse than missing a command. The recall rate is lower than precision because the system can miss gestures when the hand is too far from the sensor, or when a gesture is performed much slower than usual.

User and System Factors

We further design two experiments on user and system factors to evaluate the robustness and limitation of the system.

User-to-Device Distance

First, we evaluate the influence of user-to-device distance on the system performance. The distance is measured from the user's hand to the proximity sensors. As shown in Fig. 6, the system can achieve over 80% accuracy when the user's hand is within 3 inches. The effective range can be increased by increasing the power of IR LEDs, with a tradeoff of a higher power consumption. One can balance the tradeoff according to the system needs on user experience and battery life.

Speed of Gesture

Next, we evaluate the system performance when user perform gestures at different speeds. In this experiment, the user listens to a specific tempo given by an electronic metronome; the first beat "tic" indicates the start of a gesture, and the second beat "toc" indicates the end of a gesture. According to

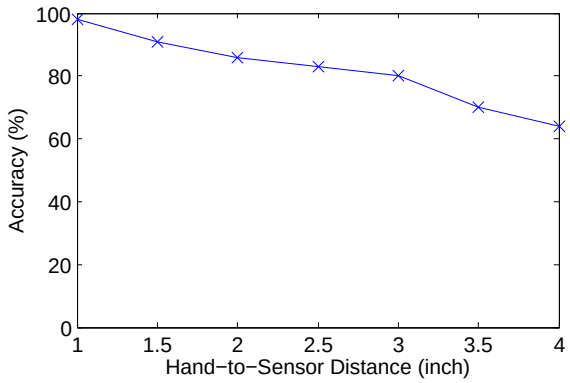


Figure 6: Recognition accuracy vs. hand-to-sensor distance.

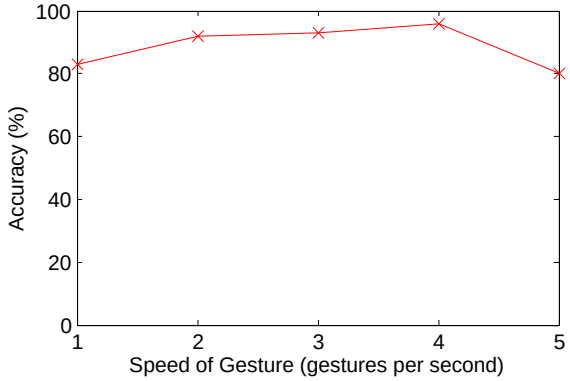


Figure 7: Recognition accuracy vs. speed of gesture.

our observation, most users naturally make gestures at the speed of 2 to 4 gestures per second. In other words, it usually take 0.5 to 0.25 seconds for general users to complete a gesture. As shown in Fig. 7, the system achieves over 90% accuracy at general gesture speeds, and also maintains a robust performance of over 80% at very slow (1 gesture per second) or very fast (5 gestures per second) gesture speeds.

Power Consumption

The system power is dominated by the power consumed by IR LED (P_{LED}) and the control chip (P_{chip}):

$$P_{LED} + P_{chip} = f_{conv} \cdot T_{prx} \cdot (I_{LED} + I_{chip}) \cdot V_{LED} \quad (4)$$

which is only 0.3 mW (idle) to 20 mW (active, when object is in proximity) [9], much lower than the 200-mW power budget for typical user interface of mobile device as reported in [7]. V, I, f_{conv}, and T_{prx} denotes voltage, current, conversion frequency, and pulse width, respectively.

CONCLUSION AND FUTURE WORK

We have presented a contactless gesture recognition system that allows users to make gesture inputs without touching, holding, or wearing any device. Using the proposed IR feature set and classifier, the system can recognize gestures with 98% precision and 88% recall rate. The low power consumption and high accuracy make the system particularly desirable for deployment on resource-limited mobile consumer devices.

Our future work is to extend the configuration to multiple sensor arrays to get more information from sensor data. Using the basic gesture set as building blocks, we can further recognize more compound 3D gestures as permutations of the simple ones. Hidden Markov model can also be incorporated to learn the gesture sequences performed by users.

REFERENCES

1. A. Butler, S. Izadi, and S. Hodges. Sidesight: multi-"touch" interaction around small devices. In *Proc. UIST*, pages 201–204, 2008.
2. J. Kim, S. Mastnik, and E. André. EMG-based hand gesture recognition for realtime biosignal interfacing. In *Proc. IUI*, pages 30–39, 2008.
3. S. Kratz and M. Rohs. Hoverflow: exploring around-device interaction with ir distance sensors. In *Proc. MobileHCI*, pages 42:1–42:4, 2009.
4. J. Liu, L. Zhong, J. Wickramasuriya, and V. Vasudevan. uWave: Accelerometer-based personalized gesture recognition and its applications. *Pervasive Mob. Comput.*, 5(6):657–675, 2009.
5. P. Mistry, P. Maes, and L. Chang. WUW - wear ur world: a wearable gestural interface. In *Proc. CHI '09*, pages 4111–4116, 2009.
6. S. Mitra and T. Acharya. Gesture recognition: A survey. *IEEE Trans. Syst., Man and Cybern.*, 37(3):311–324, 2007.
7. Y. Neuvo. Cellular phones as embedded systems. In *IEEE ISSCC*, 2004.
8. V. I. Pavlovic, R. Sharma, and T. S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *PAMI*, 19(7):677–695, 1997.
9. Silicon Labs. *Proximity/ambient light sensor with PWM output*, 2009.
10. W. C. Westerman and J. G. Elias. System and method for packing multi-touch gestures onto a hand, April 2006.
11. A. Wilson and S. Shafer. XWand: UI for intelligent spaces. In *Proc. SIGCHI conf. Human factors in comput. syst.*, pages 545–552, 2003.
12. J. O. Wobbrock, A. D. Wilson, and Y. Li. Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes. In *Proc. ACM UIST*, pages 159–168, 2007.
13. X. Zhang et al. Hand gesture recognition and virtual game control based on 3D accelerometer and EMG sensors. In *Proc. IUI*, pages 401–406, 2009.
14. M. H. Yang, N. Ahuja, and M. Tabb. Extraction of 2D motion trajectories and its application to hand gesture recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(8):1061–1074, 2002.