Touchware: a software-based technique for high-resolution multi-touch sensing devices

Xiaoling Wu

Guangzhou Institute of Advanced Technology,
Chinese Academy of Sciences,
Nansha district, Guangzhou 511458, China
E-mail: xl.wu@giat.ac.cn

Hoon Heo

Samsung Electro-Mechanics Co., LTD,
Suwon 443-743, Korea
E-mail: hoon97.heo@samsung.com

Guangcong Liu

Guangdong University of Technology,
Guangzhou 510006, China
E-mail: Liugc@gdut.edu.cn

Bangwon Lee

ATLab Inc.,
Yongin city 449-846, Korea
E-mail: banglee@atlab.ac.kr

Jianjun Li

Guangzhou Institute of Advanced Technology,
Chinese Academy of Sciences,
Nansha district, Guangzhou 511458, China
E-mail: jj.li@giat.ac.cn

Lei Shu*

College of Electronic Information and Computer,
Guangdong University of Petrochemical Technology,
525000, China
E-mail: lei.shu@lab.gdupt.edu.cn
*Corresponding author

Xiaobo Zhang

Guangdong University of Technology,
Guangzhou 510006, China
E-mail: zxb_leng@gdut.edu.cn

Sungyoung Lee

Department of Computer Engineering,
Kyung Hee University,
Yongin, Gyeonggi-do 446-701, Korea
E-mail: sylee@oslab.khu.ac.kr

Abstract: Finger pointing on touch screens is a very natural way of human computer interaction. However, for many capacitive touch sensing devices, it may suffer from the nature of direct input since the size of human fingers and the lack of sensing precision make absolute positioning on touch screen difficult, especially on multi-touch sensing devices.

Even if high-resolution/precision multi-touch devices become popular in the market, the cost is high and the positioning algorithm is not a flexible module. In this paper, we present Touchware, a software-based technique to overcome these limitations with low cost and to provide support for the development of multi-touch applications for rich input modalities. We introduce the maxima-based clustering algorithm and weight-based geometric algorithm for accurate finger positioning and first-contact-based decision with sniper for ghost pattern elimination. We evaluated the performance and show how the techniques can be successfully used for single-touch and multi-touch applications.

Keywords: multi-touch; touch screen; algorithm; absolute positioning; applications.

Reference to this paper should be made as follows: Wu, X.L., Heo, H., Liu, G.C., Lee, B.W., Li, J.J., Shu, L., Zhang, X.B. and Lee, S.Y. (2014) 'Touchware: a software-based technique for high-resolution multi-touch sensing devices', *Int. J. Ad Hoc and Ubiquitous Computing*, Vol. 17, No. 1, pp.18–30.

Biographical notes: Xiaoling Wu received her BS and MS from Harbin Inst. of Technology in 2001 and 2003, respectively. She received her PhD from Dept. of Computer Engineering at Kyung Hee University in 2008 and worked as a senior engineer in ATLab Inc. at South Korea from 2008 to 2012. She is now an Associate Professor in Guangzhou Institute of Advanced Technology, Chinese Academy of Sciences. Her research is primarily focused in touch sensing, sensor networks and IoT. She is currently a reviewer of multiple SCI/EI journals and program committee member of many international conferences.

Hoon Heo is currently a Senior Engineer in SM Development Team, Samsung Electro-Mechanics Co., Ltd. He received PhD from Computer Engineering Dept., Kyunghee Univ., Seoul, Korea, in 2005, MS from Computer Engineering Dept., Sungkyukwan Univ. in 1999 and BS from Electronics Engineering Dept., Soongsil Univ., Seoul, Korea, in 1994. His researches are mainly sensor and camera modules for mobile phone.

Guangcong Liu received his BS from South China Normal University in 1992 and his MS from Guangdong University of Technology in 2005, respectively. He was employed as a Lecturer at Guangdong University of Technology from 1992 to 2002. He is now an Associate Professor in Guangdong University of Technology and an evaluation expert in Science and Technology bureau of Guangdong province. His research interests include sensor networks and physical information fusion.

Bangwon Lee is the CEO of ATLab Inc. He received his BS of Electrical Engineering at Seoul National University in Korea in 1979, MS of Electrical Engineering at KAIST in Korea in 1981 and PhD of Electrical Engineering at University of Southern California in USA in 1991. In 1979, he joined Samsung Electronics CO. He served as an Associate Editor in Transaction on Circuits and Systems II from 1995 to 1997 and Technical Committee in Custom Integrated Circuit Conference from 1996 to 1997. His main research interests include electronics, semiconductor, touch sensing, and optics.

Jianjun Li received his BSc in Information Engineering from Xi'an University of Electronic Science and Technology, Xi'an, China. He received his MSc and PhD in Electrical and Computer from The University of Western Ontario and University of Windsor, Canada respectively. He is currently a Professor at HangZhou Dianzi University and Guangzhou Institute of Advanced Technology, Chinese Academy of Sciences. His research interests include micro-electronics, audio, video and image processing algorithms and implementation.

Lei Shu received the PhD from National University of Ireland, Galway, Ireland, in 2010. Since October 2012, he joined Guangdong University of Petrochemical Technology, China as a Full Professor. His research interests include: wireless sensor networks, multimedia communication, middleware, fault diagnosis, and security. He has published over 160 papers in related conferences, journals, and books in the area of sensor networks. He had been awarded the Globecom 2010 and ICC 2013 Best Paper Award. He has been serving as editors for a number of famous international journals. He is a member of IEEE and IEEE ComSoc.

Xiaobo Zhang received his BS from Central China Normal University and MS from Sun Yat-sen University in 2000 and 2005, respectively. He received his PhD from School of Automation at Guangdong University of Technology in 2012, where he was employed as a Lecturer. His research is primarily focused on embedded systems, wireless sensor networks, dynamic model and intelligent control. He has been serving as editors for *Journal of Chongqing University of Posts and Telecommunications* (Natural Science Edition) and *Journal of Fuzhou University* (Natural Science Edition). He is a member of ICST.

Sungyoung Lee has been the Director of the Neo-Medical Ubiquitous-Life Care Information Technology Research Center, Kyung Hee University. He received his BS of Material Science from Korea University in 1978. He received his MS of Computer Science at Fairleigh Dickinson University, USA in 1985 and received his second MS of Computer Science in Illinois Institute of Technology in 1987, where he received his PhD of Computer Science in 1991. He is currently a Professor in the Department of Computer Engineering in Kyung Hee University, Korea. His interests include ubiquitous computing, real-time embedded systems, and intelligent computing.

This paper is a revised and expanded version of a paper entitled 'A software based implementation for high resolution multi-touch applications' presented at *10th IEEE International Conference on Computer and Information Technology (CIT) 2010*, Bradford, West Yorkshire, UK, 29 June–1 July, 2010.

1 Introduction

Touch screens and finger-pointing interaction techniques have been widely and successfully used in public installations, such as ticketing machines and bank teller machines. They are also becoming popular in consumer devices (Bagchi, 2012; Goumopoulos et al., 2012; Li and Li, 2011), such as smart phones (iPhones and Android phones) and PDAs. Touch-sensing techniques enable new user interface (UI) possibilities, i.e., gestural interactions. While a traditional UI relies on the sense of sight, the touch UI is based on the sense of touch to provide user various levels of interactions with the device. Touch-sensing methods coupled with touch screen types can be classified as either resistive or capacitive. In resistive sensing (Park et al., 2010), a screen/panel is covered with two thin, transparent, conductive layers, e.g., Indium Tin Oxide (ITO) films, separated by a narrow gap. Pressing a certain point on the screen causes the two layers to be connected at that point, which is then detected by the sensing circuit. Capacitive sensing is achieved by building a screen as a collection of capacitors (channels), formed with two ITO layers and an insulation layer in between; since a finger also possesses small capacitance, placing fingers on the screen makes the capacitance slightly changed at the points of touch. The properties of capacitive sensing make it suitable for premium devices.

Although finger pointing on touch screens benefits from the nature of direct input, which makes touch screens very intuitive to use and easy to learn, especially for novice users, there are limitations in the resolution when a user interacts with them. While it is easy to select large targets by finger pointing, it is not easy to select very small objects or specify pixel accurate locations. This type of interaction can be critical in effective selection of small targets or small GUI elements (Benko et al., 2006). Touch-screen interaction can be complicated by imprecision in selection with a finger that is relatively large compared with the object, poor calibration and errors caused by the noise in the environment.

Currently, most commercially available touch screen devices are only able to detect and track a single touch on the touch screen owing to technical restrictions. However, with the recent emergence of many multi-touch prototype devices (Chen et al., 2011, Chang et al., 2009; Han, 2005; Alex et al., 2008; Joern, 2007), research on multi-finger, multi-hand and multi-user interactions has increased (Dietz and Leigh, 2001; Wu et al., 2010). In addition to dealing with the same issues as the single-touch devices, the fundamental technology of multi-touch sensitive devices often tends to be weaker to noise.

In this paper, we present Touchware, a software-based technique to overcome these limitations of touch screens and to provide support for the development of applications for rich input modalities. It is designed for high-resolution/precision single or multi-touch applications on a touch screen with high flexibilities and low cost even in a noisy environment. We introduce in detail the maxima-based clustering algorithm to detect and distinguish single/multiple touches, weight-based geometric centre (GC) algorithm to calculate an interpolated coordinate for each finger touch and first-contact-based decision with sniper for ghost pattern removal. We also prove the relationship between the user-defined resolution and the normalised capacitance values.

The novelty of Touchware lies on the following aspects:

- Touchware has better architecture compared with conventional capacitive touch screen solution with more flexibility, easy upgradability and unique user programmable features and low cost.
- Touchware can provide various resolutions without limit by simply regulating the nP_value ranges.
- Touchware can remove ghost pattern (Yang and Kwon, 2011) in diamond pattern with simple algorithm, which has low costs and high flexibilities.
- The relationship between the user-defined resolution and the normalised capacitance values is proved, which can be the guidance of touch screen design.

The rest of the paper is organised as follows. Section 2 introduces related work, Section 3 presents Touchware architecture and Section 4 thoroughly explains the proposed clustering and positioning algorithms. In Section 5, evaluations and implementation of the proposed method are presented. We conclude with a summary in Section 6.

2 Related work

The display devices that sense the human fingers' touch and support multi-touch input have been constructed in several ways. Many of these techniques apply spatially segmented arrays of antennas to sense local changes in electrical capacity, a technique that is employed in the iPhone (http://www.apple.com/iphone). Some other option can be used to detect the fingers by optical sensors (Han, 2005). A touch-sensing scheme for capacitive touch screen devices is proposed in Yang and Kwon (2011) to maximise the responsiveness of touch sensing for realisation of high-speed/resolution touch screens. A touch controller is proposed in Park et al. (2010) for on-cell capacitive touch screen panel systems, which adopts the differential sensing method to enhance the dynamic range of sensing voltage and to be robust to display noise. Two-finger and two-handed interactions for the activation of menus and widgets have been explored by researchers, among which the most related existing work is DiamondTouch (Dietz and Leigh, 2001), which is front-projected and uses antennas embedded in the touch surface. It produces row/column data in which ghost pattern exists. DiamondTouch removes ghost pattern by connecting a separate receiver to the user capacitively through the user's chair. When a user touches the surface, the antennas near the touch point couple a small signal through the user's body to the receiver. By this technique, DiamondTouch supports multi-touch and multi-user applications. However, connecting an extra receiver to the user through chair has disadvantages such as extra cost, restrictions of the surroundings and limitation of applications (large tabletop). Touchware is different and much more flexible since the ghost pattern is solved by simple judgement through time-based decision and internal characteristics of the ITO hardware. In this point of view, Touchware is superior in terms of low costs and flexibilities to support various multi-touch applications independent of the surroundings (e.g., chair). Moreover, Touchware provides various resolutions without limit by regulating the nP_value ranges. However, DiamondTouch has maximum resolution of 2500 × 1500 on a tabletop surface.

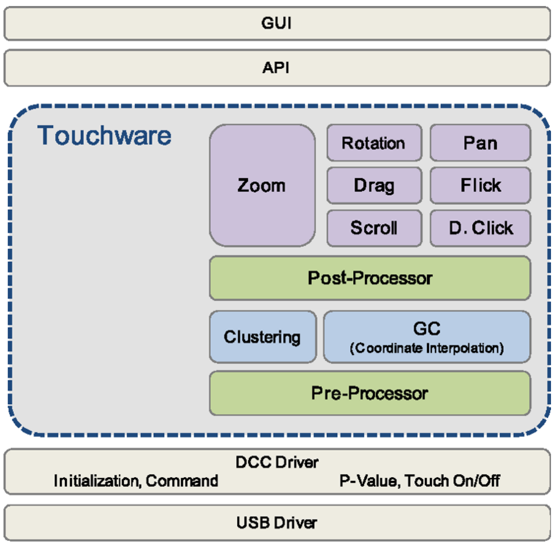
3 Touchware architecture

Unlike conventional capacitive touch screen solution including embedded Microprocessor Unit (MPU) (http://www.apple.com/iphone), our proposed Touchware is located in system MPU, which makes hardware size thinner and provides easy upgradability and optimised resolution for each gesture-based application. Conventional capacitive touch module including MPU generates coordinate values and pre-defined gestures only. Even though the embedded MPU is made easy to use, it is difficult to change MPU firmware. Conversely, Touchware is implemented in the main phone MPU as a form of middleware of mobile OS. Thus, upgrading of Touchware is done as a part of Mobile OS. Touchware not only has Mouseware like features such as changeable tapping speed, Count Per Inch (CPI) magnification, smart pointing and tracing, but also has new features such as new gestures, proximity sensing, grey-data-based feature and programmable resolution. Comparing that conventional capacitive module produces coordinate values and capacitive module with Touchware generates actual capacitance values. Those multi-level capacitance values are like grey-level of image. In addition, Touchware will contribute industry segmentation that each party of module maker, assembly maker, set maker, chip maker and OS provider can play a role independently. Conventional capacitive touch module is a solution that includes capacitive touch panel, capacitive sensor and MPU. Consequently, few companies can provide the solution. It is important to notice that the most intelligent part of the conventional capacitive touch mode is firmware in MPU. Touchware resides in mobile OS, so that the capacitive touch panel makers and sensor chip makers can work independently. Besides that, mobile phone manufacturers can differentiate quickly phone features in API software of mobile OS. Eventually, Touchware contributes to make a common platform in hardware and software.

User programmable features/functions can be added in Touchware. Capacitive module with Touchware produces capacitance value, whereas conventional module with embedded MPU produces coordinate value or gesture value. The sensitivity can be increased or decreased by redefining touch areas, such as proximity sensing, Z-movement (approaching or separating from surface). For example, Z-movement is used for acceleration of a mobile car racing game and to adjust speaker volume control.

Figure 1 shows Touchware architecture. Touchware is located between device driver (USB driver and digital contact controller (DCC) driver) and API and GUI. From bottom to up, Touchware has pre-processor, clustering and GC block, post-processor and various applications such as zoom, rotation, flick and scroll.

Figure 1 Touchware architecture (see online version for colours)



To better understand how each layer/block works, we present Touchware functional flow chart in Figure 2.

P-value (e.g., capacitance value) is defined as the measured capacitance from touch panel or touch screen at current instant, whereas nP_value is defined as the normalised P_value. The useful value required at the touch technology is not the directly measured capacitance but the variation of capacitance is due to finger touch. Thus, nP_value is calculated by subtracting offset from P-value, as shown in equation (1).

$$\text{nP\_value} = \text{P\_value} - \text{ref\_value} \quad (1)$$

where ref_value (reference P-value, the offset mentioned earlier) should be determined at touch-off status before calculating nP-value. The nP_value calculation from P-value is done in pre-processing block. The ref_value is a kind of calibrated value, which should never reflect the variation of finger touch. It has the following functions:

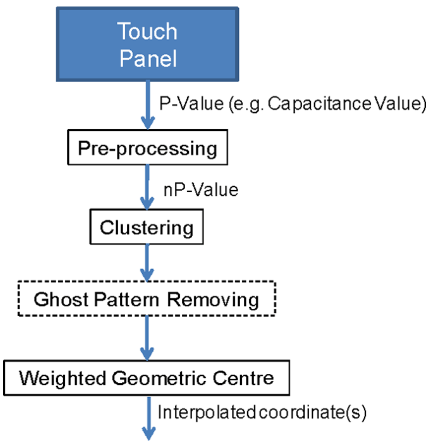
- remove noise by filtering or thresholding
- calibrate non-linearity from manufacturing tolerance and any changes by environmental changes.

Clustering block detects number of touches and distinguishes each touch by proposed maxima (including global maxima and local maxima) based clustering scheme.

Depending on patterns, ghost pattern (symmetric pattern) may occur. Diamond pattern generates ghost patterns, but signal coupling pattern and triangular pattern do not. Ghost pattern removing block, which is application dependent (optional), may be needed after clustering block to differentiate actual pattern from ghost pattern.

Weighted geometric centre (GC) block calculates interpolated coordinates for each cluster to have a singular precise point to represent the position of each finger touch.

Figure 2 Functional flow chart (see online version for colours)



4 Proposed algorithms

4.1 Maxima-based clustering

4.1.1 Diamond patterns with 2-layer

The 2-layer ITO board with diamond patterns consists of overlapping vertical and horizontal arrays of channels, which are represented by rows and columns on two different layers shown in Figure 3. The hardware periodically produces frames of data (P_values) that measure the proximity or touch of the user's finger(s) to each channel.

Figure 3 Actual touches on 2-layer 2D ITO (see online version for colours)

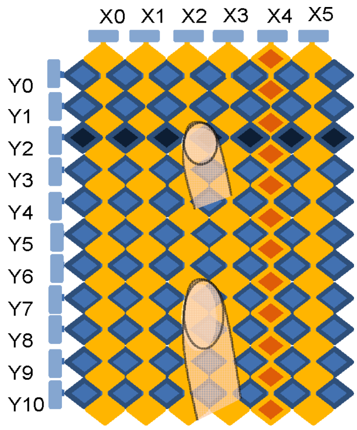


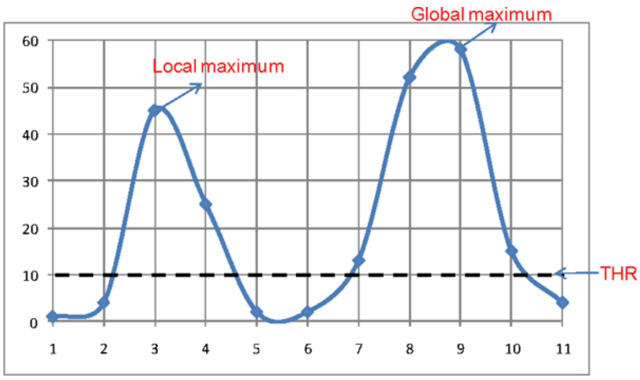
Figure 4 gives an example of nP_value row/column data (Xi, Yj) of a 6×11 2-layer ITO board and marks the calculated two clusters corresponding to the actual touches on the board shown in Figure 3. The ITO board has diamond pattern with six channels in *x*-axis and 11 channels in *y*-axis. The Capacitive touch Elements (CELs), which is a notation for physical capacitive touch element), on the same X or Y channel generate the same X nP_value or Y nP_value.

Figure 4 Maxima in an nP_value array (see online version for colours)

(0,1)	(6,1)	(137,1)	(84,1)	(9,1)	(4,1)
(0,4)	(6,4)	(137,4)	(84,4)	(9,4)	(4,4)
(0,45)	(6,45)	(137,45)	(84,45)	(9,45)	(4,45)
(0,25)	(6,25)	(137,25)	(84,25)	(9,25)	(4,25)
(0,2)	(6,2)	(137,2)	(84,2)	(9,2)	(4,2)
(0,2)	(6,2)	(137,2)	(84,2)	(9,2)	(4,2)
(0,13)	(6,13)	(137,13)	(84,13)	(9,13)	(4,13)
(0,52)	(6,52)	(137,52)	(84,52)	(9,52)	(4,52)
(0,58)	(6,58)	(137,58)	(84,58)	(9,58)	(4,58)
(0,15)	(6,15)	(137,15)	(84,15)	(9,15)	(4,15)
(0,4)	(6,4)	(137,4)	(84,4)	(9,4)	(4,4)

The basic idea of maxima-based clustering is to search maxima, which includes global maxima and local maxima in each *x*-/*y*-axis, respectively, and group the adjacent nP_values, which are larger than a certain threshold together with the found maxima to form the clusters as shown in Figure 5. In case of the sample nP_values shown in Figure 4, for Y nP_value, the local_maximum is Y2 with value 45 and global maximum is Y8 with value 58; for X nP_value array, there is only one maximum (global maximum), which is X2 with value 137.

Figure 5 Searching for maxima with THR (see online version for colours)



Take the above-mentioned Y nP_value array with size = 1 as an example: [1, 4, 45, 25, 2, 2, 13, 52, 58, 15, 4], maxima-based clustering algorithm follows the steps shown here:

Step 1: Set the values that are below THR to 0 to discard the local maxima that are below THR. In this example, Set THR = 10, the array becomes [0, 0, 45, 25, 0, 0, 13, 52, 58, 15, 0].

The factors to decide THR include:

- minimum touch distance (distance of finger touches and CEL resolution)
- usually, 1~2 gaps between touches.

Bigger values of THR make cluster size smaller; however, accuracy of interpolation for GC is limited. Smaller values make cluster size bigger, and help to find better GC and detect approaching fingers (weak touch), however GC is easily affected by noise, which results in GC jerking.

THR determines the cluster boundary/size. In this example, THR is set to be 10, the array becomes [0, 0, <u>45, 25</u>, 0, 0, <u>13, 52, 58, 15</u>, 0], so cluster size is 2 and 4, respectively, as shown in Figure 5. However, if THR = 20, the array becomes [0, 0, <u>45, 25</u>, 0, 0, 0, <u>52, 58</u>, 0, 0], cluster size is 2 and 2, respectively; If THR = 30, [0, 0, 45, 0, 0, 0, 0, 52, 58, 0, 0], cluster size is 1 and 2, respectively.

To deal with boundary problem, simply add a zero prior to and a zero coming after the nP_value array, i.e., the nP_value array becomes nP[13] = [0, 0, 0, 45, 25, 0, 0, 13, 52, 58, 15, 0, 0] with size 13.

Maxima detection computes the first and second derivation and the process is explained in detail in steps 2 and 3, respectively.

Step 2: Calculate (nP[i + 1] – nP[i])/abs(nP[i + 1] – nP[i]), i = 0, …, 12, to get a new array named diffa[12] = [0, 0, 1, –1, –1, 0, 1, 1, 1, –1, –1, 0].

Step 3: Use diffa[i + 1] to minus diffa[i] we can get another new array diffb [11] = [0, 1, –2, 0, 1, 1, 0, 0, –2, 0, 1].

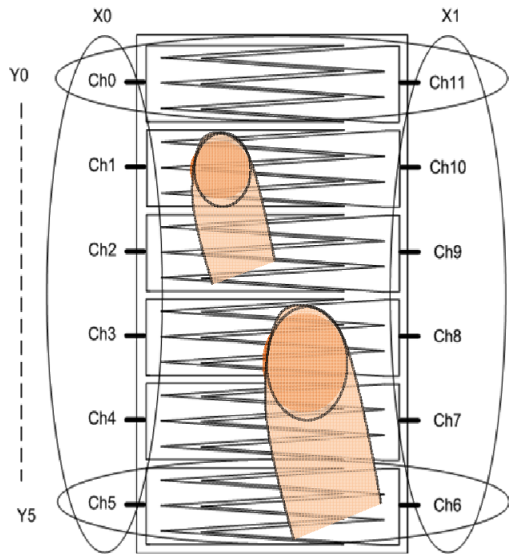
Step 4: Store the nPvalue with the index, which has value −2. In this case, nP[2] = 45 and nP[8] = 58. Two maxima are found. We can get number of clusters, which equals to the number of maxima.

Step 5: Group the adjacent nP_values together with each maximum to form the clusters.

4.1.2 Triangular pattern with one-layer

The triangular pattern with one-layer can have saw tooth pattern (Figure 6) or right angled triangle (RAT) pattern. To get nP_values for clustering with one-layer triangular pattern, we have to convert the nP_value of each channel to nP_value of each axis.

Figure 6 1-layer 2D ITO with two touches (see online version for colours)



The nP_value of *y*-axis can be calculated as follows:

Y0(nP_value) = Ch0 + Ch11

Y1(nP_value) = Ch1 + Ch10

……

Y5(nP_value) = Ch5 + Ch6

The nP_value of *x*-axis can be calculated with the following rule:

X0 can be 1 or combination of (Ch0, Ch1, …, Ch5)

X1 can be 1 or combination of (Ch6, Ch7, …, Ch11).

For example, define a threshold TH_X. If (nP_value (Chi) > TH_X), where Chi = (Ch0 ~ Ch5), nP_value Chi can be a candidate of X0. The converted nP_values have a format shown in Figure 7.

Take the nP_values in Figure 8 as an example, for *y*-axis, local maximum is Y1 with value 35 + 22 = 57 and global maximum is Y3 with value 51 + 56 = 107. The two clusters can be found following the same maxima-based clustering algorithm described in the previous section. For the upper cluster, X0 = 35, X1 = 22; for the lower cluster, X0 = 51 + 23 = 74, X1 = 56 + 24 = 80.

Figure 7 Converted nP_values

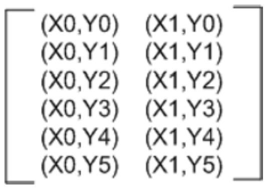
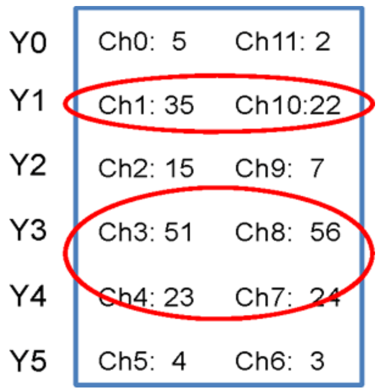


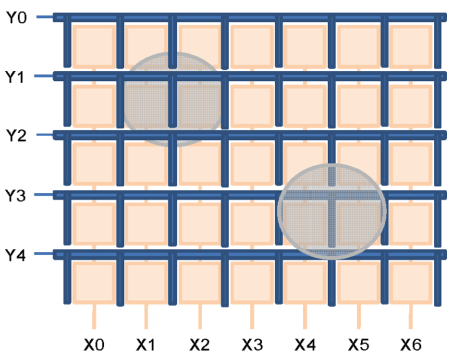
Figure 8 Clustering in 1-layer 2D ITO with TH_X = 20 and TH_Y = 40 (see online version for colours)



4.1.3 *Signal-Coupling with 2-layer and square pattern with one-layer*

In case of Signal-Coupling with 2-layer shown in Figure 9, since coupled value (equivalently capacitance) is reduced at the time finger touches ($\mathbf{C_C}^{NT} > \mathbf{C_C}^{T}$) shown in Figure 10, nP-values are reversed compared with nP-values of the 2-layer diamond pattern. Generate 'constant – nP_value' and the same equation as diamond pattern can be used. In case of single-layer square pattern shown in Figure 11, each square represents a sensing channel, which generates an nP-value. The basic one-dimensional maximum-based clustering has to be changed to two-dimensional. After finding the maxima, which are separated by at least one square, the nP-value slope in two-dimensional directions can be determined, which can group together the centre maximum as one cluster after thresholding.

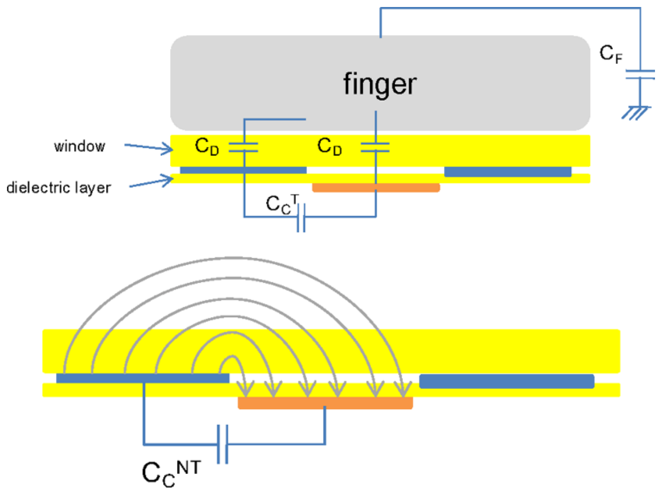
Figure 9 Signal-Coupling with 2-layer (see online version for colours)



4.2 *Weight-based GC algorithm*

A finger touch on a 2-layer diamond ITO board will probably span two or even more rows and columns. The nP_value (normalised P_value), which represents received signal strength, is used to calculate a geometric centre for the touch, obtaining positioning more precise than the physical row and column spacing.

Figure 10 Fringing capacitance between sensing pads when finger exists/does not exist (see online version for colours)



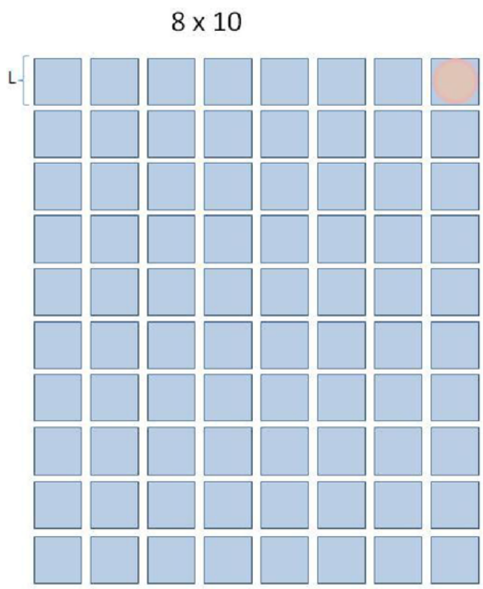
CD: Sensing pad to finger.

CCT: Fringing capacitance between sensing pads when finger exists.

CF: Finger capacitance.

$C_C{}^{NT}$: Fringing capacitance between sensing pads when finger does not exist.

Figure 11 Single layer square pattern (see online version for colours)



We propose weight-based GC algorithm, which is applied to each cluster posterior to maxima-based clustering. With the same nP_values, different kinds of touches are possible. Figure 12 shows an example with same nP_value 10 in two

X channels and two Y channels but with three different kinds of touch patterns. It can be big touch areas on two-diagonal CELs or small areas on four CELs. Whatever patterns are real, GC algorithm produces the same correct results. Figure 14 shows the measured CELs in physical coordinate corresponding to the actual touched CELs shown in Figure 13. The rectangle-shaped CELs in Figure 14 are not real touched but produce the same nP_values as in the same channel owing to the characteristics of 2-layer row/column pattern. Take the following nP_values as an example:

$$\begin{array}{l}(X0,Y0),(X1,Y0),(X2,Y0)\ (6,4),(8,4),(7,4) \\ (X0,Y1),(X1,Y1),(X2,Y1) = (6,5),(8,5),(7,5) \\ (X0,Y2),(X1,Y2),(X2,Y2)\ (6,3),(8,3),(7,3)\end{array} \quad (2)$$

Figure 12 Possible touches with the same nP_values (see online version for colours)

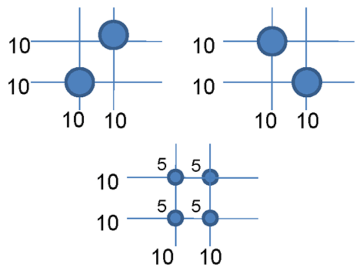


Figure 13 Touched CELs in physical coordinate (see online version for colours)

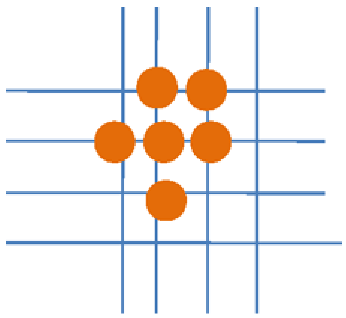
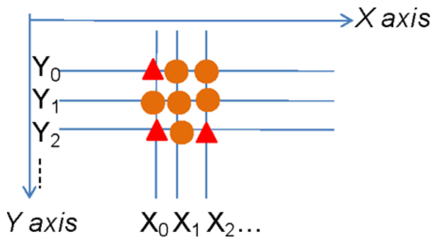


Figure 14 Measured CELs in physical coordinate (see online version for colours)



First, convert nP_Value to weight value, which is calculated as follows:

$w_x = \sum X_i$, $i = 0, 1, \ldots$, WIDTH,

$w_y = \sum Y_j$, $j = 0, 1, \ldots$, HEIGHT

where WIDTH and HEIGHT are number of channels in x- and y-axis, respectively.

In this example, the weight value is calculated as follows:

$w_x = X_0 + X_1 + X_2 = 6 + 8 + 7 = 21$,

$w_y = Y_0 + Y_1 + Y_2 = 4 + 5 + 3 = 12$.

Let GC coordinate be denoted as (Xc, Yc), and each CEL has physical coordinate (x_i, y_i), then

$X_c = \sum(X_i \cdot x_i / w_x)$,

$Y_c = \sum(Y_j \cdot y_j / w_y)$,

where w_x is the weight value for x-axis, and w_y is the weight value for y-axis.

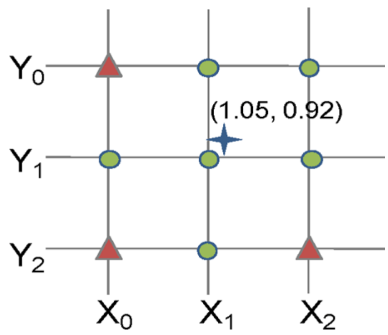
In this example,

$X_C = (6 \cdot 0 + 8 \cdot 1 + 7 \cdot 2)/21 = 22/21 = 1.05$,

$Y_C = (4 \cdot 0 + 5 \cdot 1 + 3 \cdot 2)/12 = 11/12 = 0.92$.

So, $(X_C, Y_C) = (1.05, 0.92)$, shown in Figure 15, is marked as a 4-point star.

Figure 15 Calculated GC (see online version for colours)



User can define any resolution of GC and the GC coordinate result can be calculated by the following equations:

$X_{co} = (X_C$ Resolution_X)/(WIDTH – 1),

$Y_{co} = (Y_C$ Resolution_Y)/(HEIGHT – 1);

where Resolution_X and Resolution_Y are user-defined resolution, which can be a very high value such as several hundreds. For example, if user-defined resolution for x- and y-axis is 5 and 9, respectively, the resulted coordinates will be (2.62, 2.29) and (4.71, 4.13), shown in Figure 16; if user-defined resolution for x- and y-axis is 255 and 900, respectively, the resulted coordinates will be (133.62, 116.79) and (471, 413).

The relationship between the maximum allowed resolution and nP_value range can be proved as follows:

Figure 17 shows the process of finger moving from one channel to the adjacent channel, where CH1 represents channel 1, CH2 represents channel 2 and the circle represents the finger touch area. Assume that the maximum sampled value of one channel is 100. When the finger moves from channel 1 to channel 2, as the nP_value is always an integer and the finger size is fixed, its variation as finger moves follows the rule as shown in Table 1. The user-defined resolution between two channels must be less than or equal to maximum nP_value. Therefore, the resolution of the whole touch screen should not exceed maximum nP_value of one channel multiplied by number of channels in x- (or y-) axis.

Figure 16 Calculated GC by user defined resolution 6×6 and 10×10 (see online version for colours)

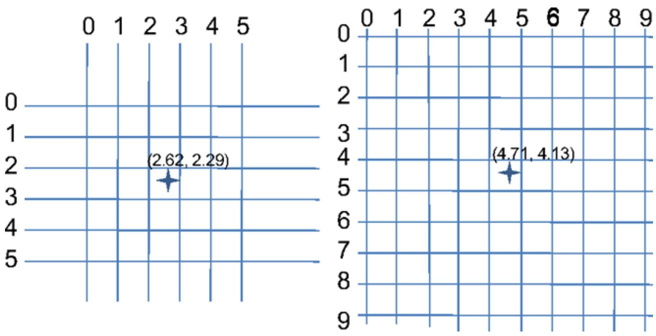


Figure 17 Finger moves between two adjacent channels

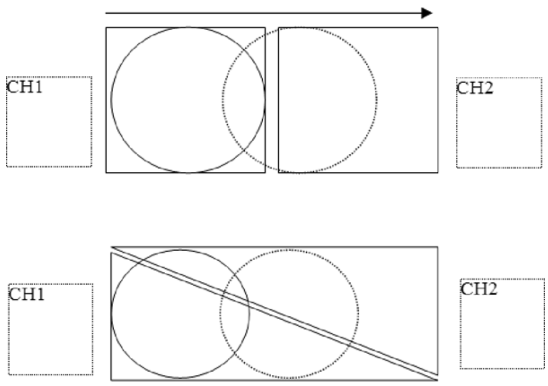


Table 1 The nP_value variation as finger moves

Finger locations	*The nP_values of channel 1*	*The nP_values of channel 2*
Leftmost	100	0
……	99	1
	98	2
	……	……
	1	99
Rightmost	0	100

In case the two clusters in triangular pattern with one-layer shown in Figure 18 follow the sample nP_values in Figure 8, the weight-based GC can be calculated as follows:

- For cluster 1,

$w_{1x} = 35 + 22 = 57,$

$X_{1c} = (35\ 0 + 22 \cdot 1)/57 = 22/57 = 0.39$

$w_{1y} = 57,$

$Y_{1c} = 57 \cdot 1/\ 57 = 1$

- For cluster 2,

$w_{2x} = 74 + 80 = 154,$

$X_{2c} = (74 \cdot 0 + 80 \cdot 1)/154 = 80/154 = 0.52$

$w_{2y} = 107 + 47 = 154,$

$Y_{2c} = (107 \cdot 3 + 47 \cdot 4)/154 = 509/154 = 3.31$

The calculated two GCs (0.39, 1) and (0.52, 3.31) for two clusters in triangular pattern with one-layer are shown in Figure 19. Figure 20 presents the flow chart for the whole process of maxima-based clustering and weight-based GC using nP_values as input and THR to decide cluster boundary.

Figure 18 Clustered nP_values in triangular pattern with 1-layer (see online version for colours)

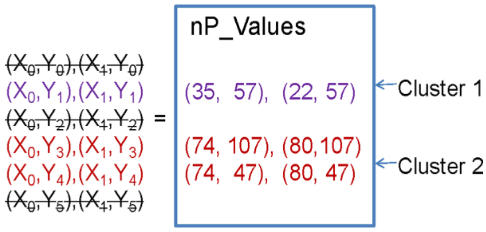


Figure 19 GC for each cluster in triangular pattern with 1-layer (see online version for colours)

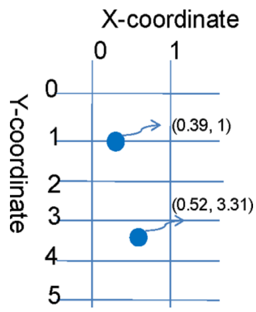
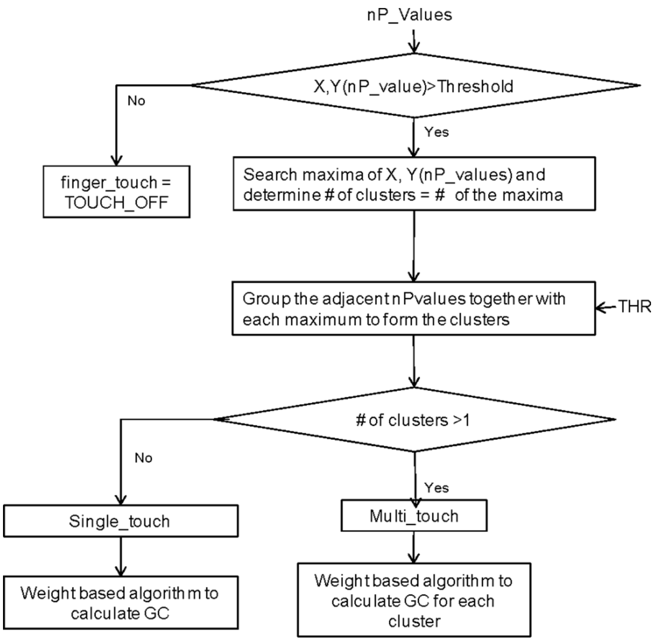


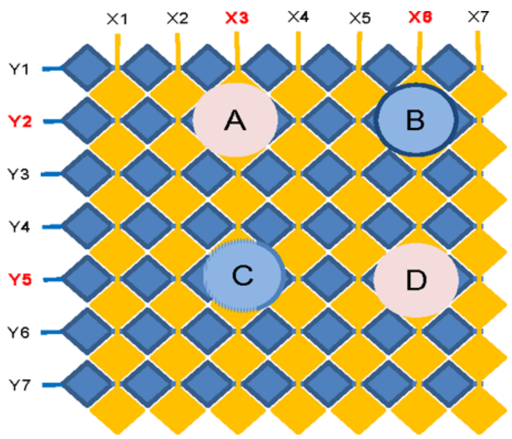
Figure 20 Clustering and weight based GC flow chart



4.3 *Ghost pattern removal*

Because of the characteristics of diamond pattern that each channel gives the same nP_values, the two touches in diagonal two positions, for example case 1 (X3, Y2) and (X6, Y5), and case 2 (X3, Y5) and (X6, Y2) shown in Figure 21 result in nP_values that generate symmetrical patterns, four clusters A, B, C and D. Thus, it is very difficult to directly distinguish case 1 or case 2 as actual touches from the available nP_values. We name the fake diagonal touched positions as ghost pattern (Yang and Kwon, 2011) and the actual touched positions as real pattern.

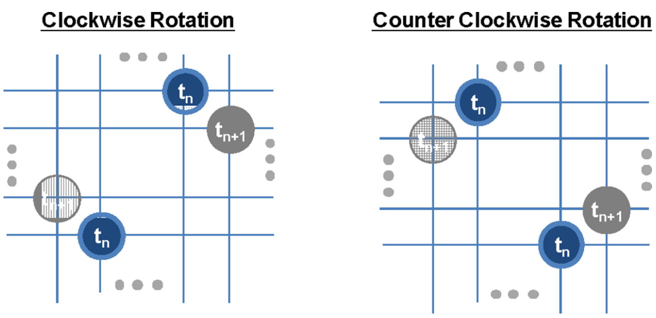
Figure 21 Ghost pattern with diamond ITO (see online version for colours)



In some multi-touch applications such as rotating a picture, it can hardly distinguish the direction as clockwise or counter clockwise as shown in Figure 22 when ghost pattern exists.

Touchware implements a function to remove ghost pattern. It firstly decides the real pattern by first-contact-based decision if there is a certain time difference between two touches. Otherwise, the proposed algorithm named sniper to remove ghost pattern is employed. The flow chart of the whole process is shown in Figure 23.

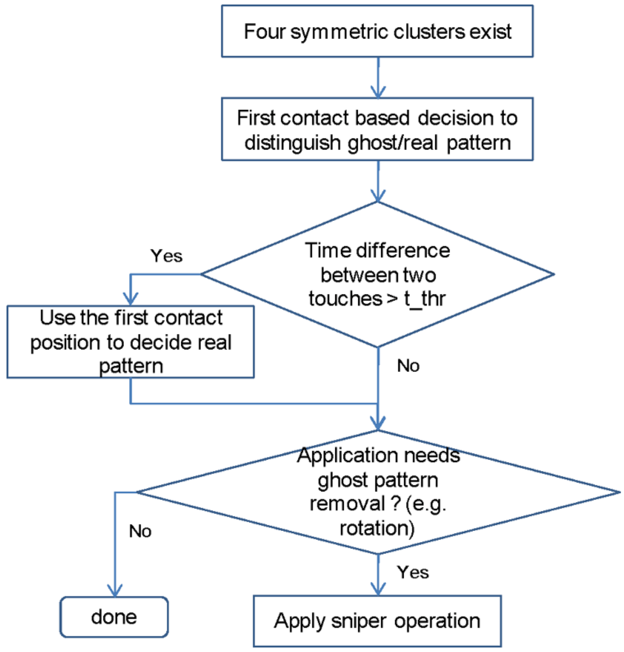
Figure 22 Problem caused by ghost pattern in rotation (see online version for colours)



The main idea of sniper is based on the nP_value difference owing to the difference of the resistance of on-chip resistor and ITO resistance between sensing pad and touch location. Experiment has been done on a 6×11 ITO board with X sensing line at the top and Y sensing line on the left. We use an 8 mm circular standard PCB finger moving straight from location Y0–Y10 at X4 sequentially as shown in Figure 24.

Figure 25 shows the experiment result of *x*-axis nP_value vs. different touch positions in case of different ITO resistance. At position Y0, the X nP_value is obviously larger and at position Y10 the X nP_value is smaller and |max.npValue – min.npValue| is approximately 100. *Y*-axis nP_value shows similar characteristics. Thus, based on the X and Y nP_values, sniper algorithm can be used to distinguish ghost/real pattern. The flow chart of sniper function based on the example touch pattern (Figure 21) is shown in Figure 26.

Figure 23 Ghost pattern removal flow chart (see online version for colours)



Take a 6×6 nP_values presented in Table 2 as an example, which was measured with an ITO board with X and Y sensing line located at top and left side. The corresponding possible touches can be A and D or B and C similar to Figure 21. Assume that two fingers touch simultaneously; sniper should be executed as follows,

$X(A) = 60 > X(B) = 44,$

$Y(A) = 43 > Y(C) = 37,$

Since nP_value X(A) > X(B) and Y(A) > Y(C), we can decide A and D as real patterns.

Table 2 6-by-6 nP_values with two diagonal touches

	X0	*X1*	*X2*	*X3*	*X4*	*X5*
Y0	(0,0)	(60,0)	(0,0)	(0,0)	(44,0)	(0,0)
Y1	(0,43)	*(60,43)*	(0,43)	(0,43)	*(44,43)*	(0,43)
Y2	(0,0)	(60,0)	(0,0)	(0,0)	(44,0)	(0,0)
Y3	(0,0)	(60,0)	(0,0)	(0,0)	(44,0)	(0,0)
Y4	(0,37)	*(60,37)*	(0,37)	(0,37)	*(44,37)*	(0,37)
Y6	(0,0)	(60,0)	(0,0)	(0,0)	(44,0)	(0,0)

Figure 24 6-by-11 ITO board with X sensing line at the top and Y sensing line on the left (see online version for colours)

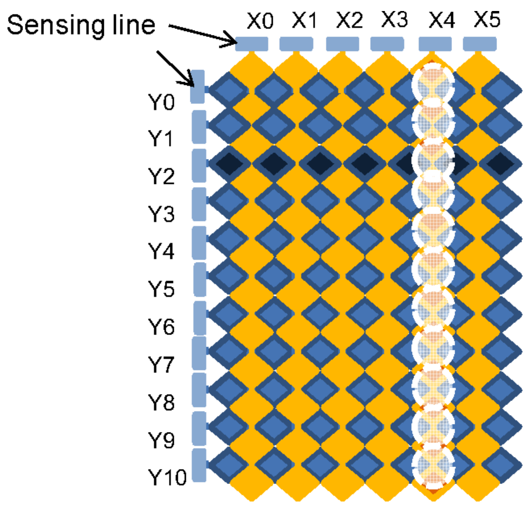


Figure 25 Experiment result of X axis nP_value vs. different finger positions (see online version for colours)

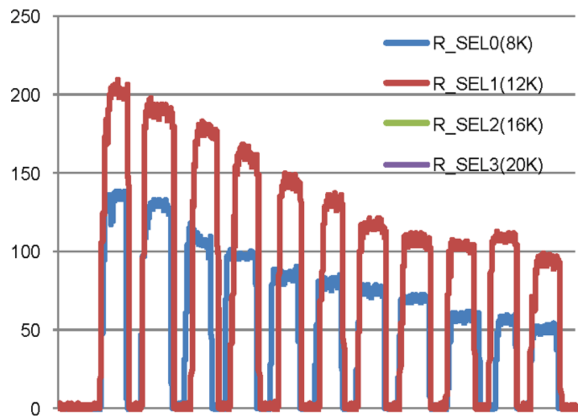
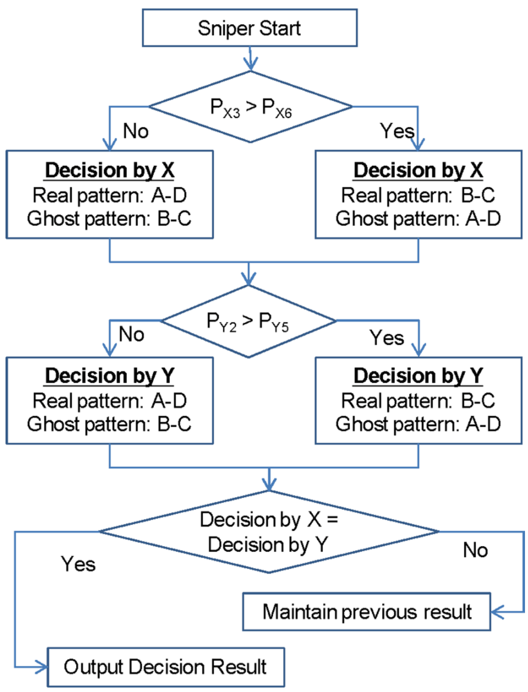


Figure 26 Flow chart for sniper (see online version for colours)



5 Evaluations and implementations

We implemented Touchware in H/W prototype with the above-mentioned touch panels, i.e., diamond pattern (6 channels on *x*-axis and 8 channels on *y*-axis), single-layer triangular pattern (6 channels on each side, totally 12 channels) and single-layer square patterns (6 channels on *x*-axis and 8 channels on *y*-axis, totally 48 channels). First of all, a tuning system is used to help tune the target touch board with various parameter adjustments that determine the performance of the target touch board. It consists of a sample target touch board, a USB I/F board, which transmits data between the PC and the target touch board, and a PC where the tuning program runs. Tuning system allows the developer to view all necessary parameters and transfers the desired values of parameters to the chip attached on or connected to the target touch board. The application board to develop typically includes a chip, a touch screen panel and the application system board, which interfaces with the chip. The chip uses Time-domain Multiplexing (TDM) architecture, which shares one pulse generation unit to measure capacitance variations of all sensor input channels to achieve a cost-effective design and adopts active pulse-pass architecture to improve noise immunity by automatically calibrating offset capacitance and gradual environmental capacitance changes of sensor input channels. It also contains two different low-pass filters and a down sampler to suppress various noises. Its wide dynamic range compensates for offset capacitance in the sensor input channels. The capacitance acquired by human touch is represented as 10-bit digital values.

The maximum reporting rate, jitter tolerance and signal-to-noise ratio (SNR) are 160 Hz, ±0.2 mm and 12 dB, respectively, in the implementation. The resolution can be maximum 1024 between two adjacent channels, when we evaluate a 3.5-in wide ITO panel with full nP_value 1024 in one channel. The total code memory and data memory size including algorithms and gestures are less than 12 KB and 500 B, respectively, with 8-bit MCU, which saves a lot of cost when selecting proper MCU. Pure hardware-based solutions in capacitive touch screens can hardly achieve multi-touch and smooth finger trace. As compared with hardware-based solutions with minimum software, Touchware-based solution saves nearly half of the cost to achieve the same performance. Moreover, it can eliminate the ghost points for diamond pattern, which, to our best knowledge, cannot be realised by hardware-based solutions.

The performance of Touchware is shown with four gesture-based applications (flick, scroll, zoom and rotation) and a handwriting function. The main UI consists of tapping icons with corresponding GC coordinates displayed on the upper right side. Zooming is activated by tapping the ‘Zoom’ button on the main menu. Figure 27 shows the function of picture zooming in/out with two fingers. Figure 28 shows an example of rotation application, which allows a user to rotate a picture with any angle that is more comfortable for the user to view. Actual touches are displayed with cross marks. Notice that to produce a correct rotation direction ghost pattern should be detected and eliminated. Especially, when touches are crossing quadrants, ghost patterns easily cause a wrong decision. Touchware successfully removed ghost points for 'rotation' function to work perfectly with any user-desired angle. The real simultaneous multi-touch (maximum five fingers) algorithm is implemented with a single-layer square pattern shown in Figure 29. And, Figure 30 shows the android-based demo set with Touchware embedded to generate coordinates.

Figure 27 Zoom (see online version for colours)

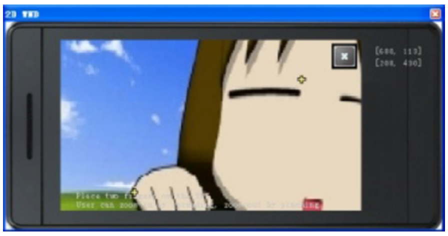


Figure 28 Rotation (see online version for colours)

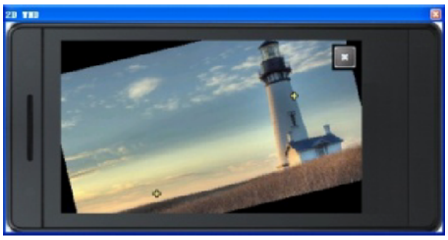


Figure 29 The real multi-touch pad and its user interface (see online version for colours)

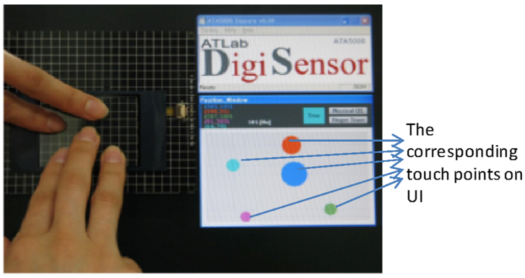
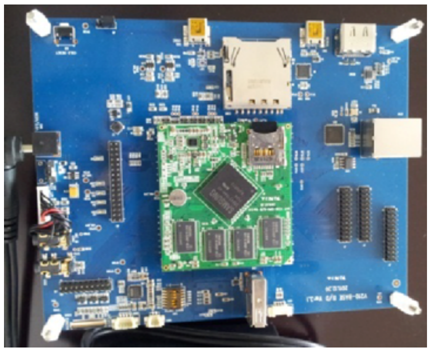


Figure 30 Android based demo set with Touchware embedded for coordinates: (a) front side and (b) back side (see online version for colours)

(a)

Figure 30 Android based demo set with Touchware embedded for coordinates: (a) front side and (b) back side (see online version for colours) (continued)



(b)

6 Conclusions

This work presented Touchware, a software-based technique to support the development of high-resolution/precision single-/multi-touch applications on a capacitive touch-sensing device with low cost. Three algorithms were introduced, which are maxima-based clustering algorithm to detect and distinguish single/multiple touches, weight-based geometric centre algorithm to calculate a precise interpolated (x, y) coordinate for each finger touch and a first contact-based decision with sniper scheme for ghost pattern removal. We also proved the relationship between the user-defined resolution and the normalised capacitance values. We showed how the techniques were successfully used for single-touch applications, such as click, scroll and flick, and multi-touch applications such as zoom in/out and rotation. More applications can also be easily embedded.

Acknowledgements

This research was supported by the MSIP (Ministry of Science, ICT&Future Planning), Korea, under the ITRC (Information Technology Research Center) support programme supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2013-(H0301-13-2001)), and 2013 Guangdong Province Petrochemical Equipment Fault Diagnosis Key Laboratory Open Fund.

References

Alex, B., Shahram, I. and Steve, H. (2008) 'Side Sight: multi-touch interaction around small devices', *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology* (UIST '08), ACM, New York, NY, USA, pp.201–204.

Bagchi, S. (2012) 'The software architecture for designing interactive mobile computing applications', *Journal of Ad Hoc and Ubiquitous Computing (IJAHUC)*, Vol. 10, No. 4, pp.230–240.

Benko, H., Wilson, A.D. and Baudisch, P. (2006) 'Precise selection techniques for multi-touch screens', *Proc of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2006*, ACM Press, New York, NY, USA, pp.1263–1272.

Chang, W.Y., Fang, T.H., Yeh, S.H. and Lin, Y.C. (2009) 'Flexible electronics sensors for tactile multi-touching', *Sensors*, Vol. 9, No. 2, pp.1188–1203.

Chen, Y.L., Liang, W.Y., Chiang, C.Y., Hsieh, T.J., Lee, D.C., Yuan, S.M. and Chang, Y.L. (2011) 'Vision-based finger detection, tracking, and event identification techniques for multi-touch sensing and display systems', *Sensors*, Vol. 11, No. 7, pp.6868–6892.

Dietz, P. and Leigh, D. (2001) 'DiamondTouch: a multi-user touch technology', *Proc. of UIST 2001*, ACM Press, New York, NY, USA, pp.219–226.

Goumopoulos, C., Kameas, A.D. and Berg, E. (2012) 'A framework for developing Pervasive Awareness Systems in smart environments', *Journal of Ad Hoc and Ubiquitous Computing (IJAHUC)*, Vol. 9, No. 3, pp.142–158.

Han, J.Y. (2005) 'Low-cost multi-touch sensing through frustrated total internal reflection', *Proc. of UIST'05*, ACM Press, New York, NY, USA, pp.115–118.

Joern, L. (2007) 'Two-finger input with a standard touch screen', *ACM UIST'07*, 7–10, October, 2007, Newport, Rhode Island, USA, ACM Press, New York, NY, USA.

Li, C.L. and Li, L.Y. (2011) 'Utility-based approach for utilising services of mobile devices in mobile grid', *Journal of Ad Hoc and Ubiquitous Computing (IJAHUC)*, Vol. 7, No. 4, pp.235–245.

Park, Y.S., Bae, J.W., Kim, E.S. and Park, T.J. (2010) 'Maximizing responsiveness of touch sensing via charge multiplexing in touchscreen devices', *IEEE Trans. on Consumer Electronics*, Vol. 56, No. 3, August, pp.1905–1910.

Wu, X.L., Lee, B.W., Joung, C.Y. and Jang, S.E. (2010) 'Touchware: a software based implementation for high resolution multi-touch applications', *Proc. of 10th IEEE International Conference on Computer and Information Technology (CIT) 2010*, 29 June–1 July, 2010, Bradford, West Yorkshire, UK, pp.1703–1710.

Yang, I.S. and Kwon, O.K. (2011) 'A touch controller using differential sensing method for on-cell capacitive touch screen panel systems', *IEEE Trans. on Consumer Electronics*, Vol. 57, No. 3, August, pp.1027–1032.

Website

http://www.apple.com/iphone