Hand Contact Shape Recognition for Posture-Based Tabletop Widgets and Interaction

Fabrice Matulic[1,2] **Daniel Vogel**[1] **Raimund Dachselt**[2]
[1]Cheriton School of Computer Science
University of Waterloo, ON, Canada
{fabrice.matulic, dvogel}@uwaterloo.ca

[2]Interactive Media Lab Dresden
Technische Universität Dresden, Germany
dachselt@acm.org

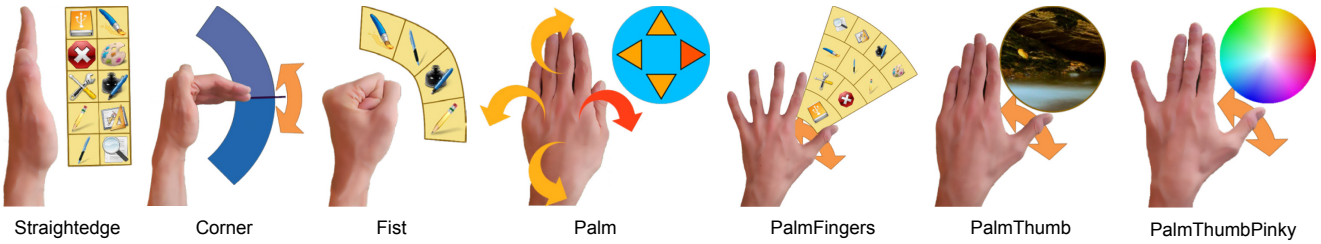

Figure 1. Examples of widgets triggered by Hand Contact Postures: Straightedge - rectangle menu; Corner - radial gauge; Fist - arc menu; PalmFingers - fan menu; PalmThumbPinky - colour wheel; PalmThumb - magic lens; Palm - directional pad.

ABSTRACT

Tabletop interaction can be enriched by considering whole hands as input instead of only fingertips. We describe a generalised, reproducible computer vision algorithm to recognise hand contact shapes, with support for arm rejection, as well as dynamic properties like finger movement and hover. A controlled experiment shows the algorithm can detect seven different contact shapes with roughly 91% average accuracy. The effect of long sleeves and non-user specific templates is also explored. The algorithm is used to trigger, parameterise, and dynamically control menu and tool widgets, and the usability of a subset of these are qualitatively evaluated in a realistic application. Based on our findings, we formulate a number of design recommendations for hand shape-based interaction.

Author Keywords

interactive tabletops; shape recognition; multi-touch; pen+touch; bimanual input; palm rejection

INTRODUCTION

Most multi-touch input uses one or more fingers for interaction (e.g [1, 4, 10]) with whole hand contacts considered undesirable and filtered out [25]. However, when the surface is large enough, intentional hand contact shapes can be used as first-class input for mode switching [31], tool selection [12] and other types of interaction techniques [16, 24, 29, 30, 32]. The contact shape can be extracted using the raw capacitive signal [33] or infrared cameras mounted in or beneath the surface [24, 32, 35], depending on the available sensing technology. Previous shape-based interaction techniques typically use coarse features such as the overall contact ellipse [29] or treat contact shapes as end effectors for physics-based object manipulation [32]. However, ellipses do not capture the full range of hand contact shapes and using contact geometry for manipulation does not translate well to interface widget control.

Our objective is to leverage hand contact shapes to trigger different widgets "in-place" [15] and support dynamic adaptation and parameterisation according to hand placement and finger movements. The goal is for those widgets to function as quick-access tools summoned by the non-dominant hand for bimanual interaction with pen and touch input [14, 19]. A challenge is to make these explicit widget-triggering postures robust while still supporting relaxed arm poses, such as when the forearm rests on the surface.

In this paper, we make the following contributions:

- An easily reproducible algorithm using standard computer vision functions to recognise typical hand shapes, with arm-rejection and hover filtering, using the raw grey image provided by tabletop sensors;
- Interaction techniques demonstrating how hand shapes can locally trigger and parameterise menus and tools with dynamic control based on hand and finger motion;
- Evaluations of posture detection accuracy and realistic usage within a pen and touch application;
- A discussion of design guidelines and expanded interactions leveraging hover detection and estimation of local hand-on-surface pressure.

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.
ISS '17, October 17–20, 2017, Brighton, United Kingdom
© 2017 Association for Computing Machinery.
ACM ISBN 978-1-4503-4691-7/17/10…$15.00
https://doi.org/10.1145/3132272.3134126

RELATED WORK

Previous work has explored interactions on multi-touch tabletops and tablets, but recognition methods have been a secondary consideration. Wu and Balakrishnan propose a set of multi-finger and whole-hand techniques [30]. Rock & Rails are a set of hand gestures to constrain 2D manipulation tasks [29]. Koura et al. use forearm contact for various interactions [16]. TouchTools enable virtual tool manipulation based on the contact pattern of hands and fingers when mimicking the use of a physical tool [12]. MultiTouch Menus utilise the base of the palm and the five fingers to support advanced menu triggering [1], a concept further developed in HandMarks where grids of menu icons appear between fingers [26]. Koura et al. do not provide details about their recognition method, but the remaining works all use standard touch points and/or contact ellipses with their recognition algorithm tailored to specific interactions. Without using the full contour of the hand contact shape, generalisation is difficult and the capability for accurate widget placement and dynamic parameter control is limited.

The full hand contact shape has been used for user identification [24], recognising fingers and their orientations [33, 35] and analysing hand occlusion [27], but these techniques do not generalise to recognition for interaction. ShapeTouch uses hand contour geometry to mimic close-to-physical object manipulations [32], but the technique does not recognise or classify hand shapes. Moreover, the works above have not formally evaluated hand contact recognition accuracy, and it is not known how usable whole-hand contacts would be in real application contexts. Finally, Le et al. explore four categories of contact postures when users lean on the tabletop, including hands, elbows and forearms [18]. The work elicits and recognises arm lean postures, but it does not look at hand contact shapes in detail.

HAND CONTACT SHAPE RECOGNITION

Our goal is to detect hand contact shapes suitable for posture-triggered widgets that may be dynamically positioned and adjusted based on simple geometric features. To support bimanual work with arm-resting, the algorithm should also be able to cope with shape traces left by those limbs.

For speed and reproducibility, we use a pipeline of standard computer vision functions available in OpenCV. Our implementation uses C# with the Emgu OpenCV wrapper.

Calibration

In its full version, our algorithm requires an estimated hand size and examples of hand contact shapes for all of the supported postures. In a one-time calibration step, each posture is formed twice in three positions at the bottom of the surface. The outstretched hand posture is formed with the wrist aligned with the bottom bezel to estimate hand length. All other postures are formed such that the top of the posture is aligned with the top of the outstretched hand.

All images are processed using the steps below to create template contours for matching. Later, we evaluate when this calibration is per-user, and when it is based on generic templates obtained from different users.

Capture

Hand contact shapes can be captured using a camera mounted above the surface (e.g. [5]), but the most practical method is to capture from beneath. This can be a high resolution image obtained from infrared cameras inside a direct illumination tabletop (e.g. original Microsoft Surface) or a coarse resolution image from the raw capacitive signal [33]. For our prototype implementation we use a Samsung SUR40 with Microsoft PixelSense as it exposes a highly sensitive raw input image as part of its API, which allows easy prototyping of interaction techniques based on arbitrary contact shapes [18]. The greyscale image has a resolution of 960×540 px (24 dpi) with fidelity between a conventional camera and a raw capacitive signal, suggesting our approach could be adapted for either, provided signal strength can be tuned and raw touch data can be accessed for low-level processing [14].

Pre-processing

After a greyscale image is captured, we apply a global threshold to create a binary image. From that image, we extract the connected component contours and group them according to size and shape. Smaller elliptical contours are fed to a finger and pen input processor (to handle them as standard touch and pen events) and contours exceeding thresholds corresponding to the minimum area of a hand contact are saved. Each contour polygon is simplified using the Douglas-Peucker algorithm.

Hand Isolation and Forearm Rejection

Comfortably forming a hand contact posture typically requires resting the forearm on the surface. Because large shapes are also considered valid input in addition to smaller finger touches, we cannot discard large blobs altogether to "reject" arms and undesired contact shapes. We filter out the forearm to isolate the hand in the following way: First, the longest diagonal of the shape contour is determined with

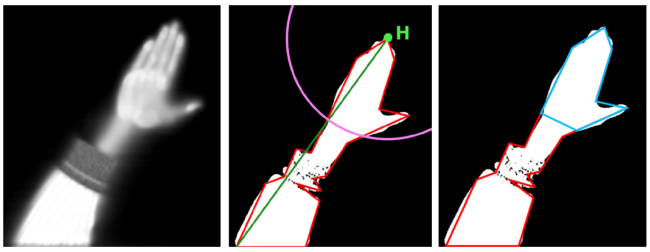

Figure 2. Hand contour segmentation process. Left: Original raw touch image of the hand. Middle: Polygon contour (red) with longest diagonal (green) and cut-off circle (pink) centred on the extremity H. Right: Extracted hand contour (blue).

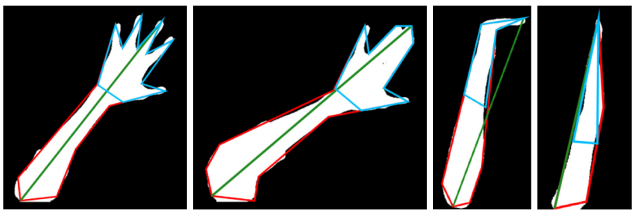

Figure 3. Examples of arm and hand shapes with detected contours.

the end H corresponding to the top of the hand, determined using either frequency of convexity defects (indicating the likelihood of fingers) or a simple heuristic to select the end farthest away from orthogonal bezels. We then perform a Boolean AND operation between the shape contour and a circle centred at H with radius equal to the hand length obtained in the calibration step. The result of that operation is the isolated hand contour (Figure 2). Examples of different hand postures extracted using that process are shown in Figure 3.

Hover Filtering

If the touch digitiser is sufficiently sensitive, hands and arms can be sensed even when slightly above the surface, which can sometimes lead to relaxed hand poses being falsely recognised as explicit contact. We exploit how objects above a touch surface appear blurred from capacitive signal degradation [14], camera focus, or light diffusion [22, 23], depending on the sensing technology. We quantify blur (and therefore hover) by calculating a sharpness index using local gradients along detected shape contours. Specifically, we compute the Laplacian of the original grey image masked by shape component contours rendered with a 20-pixel thick stroke. From those gradients, we determine a single measure of sharpness by calculating the standard deviation within the contour mask. The sharpness index is used to discard hands that are not completely touching the surface and support transitions between casual resting poses – distinguishing between fingers lightly curled when resting and explicit menu triggers when hands are fully in contact. Further below, we show how blurriness can also be used to support explicit interactions.

Posture Matching

From the previous operations, we obtain an array of hand contours. Each contour is classified as a specific hand posture in two steps, where the second step is optional if the features of the first method are sufficient.

Convexity and Angle Matching

The convex hull and its convexity defects are computed for each contour using techniques similar to mid-air hand and fingertip tracking [8]. We use convexity defects along with inter-finger angles as features to perform a first matching of the hand shape. These generally suffice to identify open hand postures with spread fingers, but postures such as closed fists and vertical hand edges that exhibit no defects, or only a small number of defects, remain ambiguous.

Moment Matching

To distinguish between shapes with similar convexity characteristics, we perform an additional matching step based on contour similarity. There are numerous methods to measure similarity between shapes and polygons. We use the OpenCV `matchShape` function based on Hu Moments [21]. They are invariant to rotation, reflection and scale, making them suitable for matching left and right hand postures of different sizes and orientations. The shapes are matched against the templates obtained in the calibration.

EVALUATION: ALGORITHM ROBUSTNESS

In this evaluation, we measure algorithm recognition accuracy with a representative set of hand postures.

Hand Postures

Epps et al. [6] report which gestures people would adopt for a range of atomic tabletop tasks, many of which were used in research prototypes [20, 29, 30]. We use this as a starting place for the seven postures we evaluate (see also Figure 1):

- *Straightedge*: a hand edge (appearing in previous work as vertical hand [6, 30], karate chop [20, 30] and rail [29]).
- *Corner*: a hand edge with flexed fingers (appearing in previous work as corner-shaped hand [30], L-shaped hand [32], curved hand [6] and curved rail [29]).
- *Fist*: an upright clenched fist (appearing in previous work as fist [6] and rock [29]).
- *Palm*: a flat palm with joined fingers (appearing in previous work as flat hand and horizontal hand [30]).
- *PalmFingers*: a flat palm with all spread fingers (appearing in previous work as a spread hand [6] and used for user identification in [24]).
- *PalmThumb:* a flat palm with abducted thumb (similar to postures used for grasping gestures [8], but we are not aware of it used for 2D shape-based input on tabletops).
- *PalmThumbPinky:* a flat palm with abducted thumb and little finger (a new posture that we introduce).

Participants and Apparatus

We recruited 15 volunteers (12 males and 3 females, average age 28.2 years old). All were right-handed and used mobile touch devices on a daily basis. Eight participants also had some experience with digital tabletops. Those participants also completed a second study described later to evaluate qualitative aspects of widgets enabled by these postures. In both studies, we used the Samsung SUR40 tabletop and the algorithm described above.

Protocol and Task

Each participant first completed the calibration step explained in the previous section. Then, they formed each of the seven postures twice in all regions of a 3×2 grid dividing the tabletop surface. An indication of the hand posture to mimic was shown as an icon in a corner of the target area. The participants were instructed to form the posture in the designated region in a comfortable manner and they could rest their arm on the surface if they wished. The grid and emphasis on comfort allowed us to better capture different hand orientations and resting arm patterns.

Pilot tests showed long sleeves could confuse the algorithm (example shown in Figure 4). Ten people wore shirts with long sleeves, therefore we asked all participants to first perform the calibration and testing steps with their sleeves rolled up. Then, the participants with long sleeves also performed the testing step with sleeves rolled down.

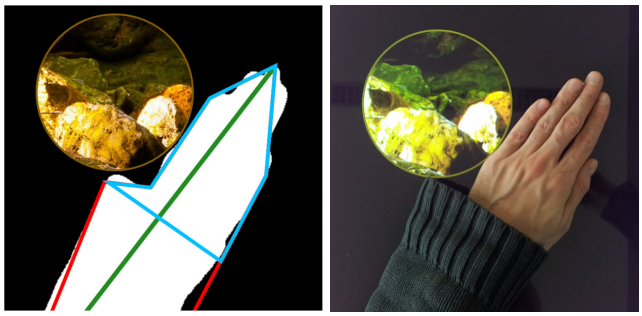

Figure 4. Example of an erroneously detected PalmThumb shape because of a long sleeve introducing a convexity defect

Note that the algorithm did not run during this capture phase, only raw images were collected for later analysis. For all participants, we gathered 42 raw images for calibration (7 postures × 3 regions × 2 repetitions) and 84 raw images for testing (7 postures × 6 regions × 2 repetitions). For participants wearing long sleeves, we collected an additional 84 raw images.

Results

	Per-user templates	*Other-user templates*	*Per-user templates w/ sleeves*	*Other-user templates w/ sleeves*
Corner	**89.9**	80.5	80.6	70
Palm	**99.2**	95	95.6	90
PalmFingers	**96.7**	92.5	93.3	89.4
PalmPinkyThumb	**100**	100	100	100
PalmThumb	**98.3**	96.7	98.3	94.4
Straightedge	**75.8**	54.7	65	56.7
Fist	**75.8**	68.2	52. 8	43.6
AVERAGE	**90.8**	83.9	83.7	77.7

Table 1. Recognition accuracy (in %)

	C	*P*	*PF*	*PPT*	*PT*	*S*	*F*	*nvsd*
Corner	**70**	0.2	0.2	8.5	8.1	9.8	2.4	0.7
Palm	0	**90**	0	0	4.4	0	5.6	0
PalmFingers	0	0	**89.4**	0	0	0	0	10.6
PalmPinkyThumb	0	0	0	**100**	0	0	0	0
PalmThumb	1.1	0	0.6	3.9	**94.4**	0	0	0
Straightedge	11.7	0	0	3.3	0	**56.7**	26.1	2.2
Fist	11.1	9.3	0	5.6	8.3	21.2	**43.6**	0.9

Table 2: Confusion matrix for the worst condition (long sleeves with other user templates). Shape names have been abbreviated in the column labels but are listed in the same order. nvsd=no valid shape detected.

We processed images from the test sets against different choices of calibration data. A match success was registered if the correct shape had been recognised and a failure in all other cases (no or wrongly recognised shape). Table 1 shows the recognition accuracies under different conditions and Table 2 displays the confusion matrix between the shapes for the worst case, that is, long sleeves with templates from other users.

Per-User Templates

Using templates from the same participant, the algorithm achieves a mean accuracy of 91%. All palm-based postures are above 96%, while *Fist* and *Straightedge* are less robust at 76%. Corners have a recognition accuracy of 90%.

Other-User Templates

To evaluate the feasibility of using generic hand shape templates for matching, we calculated accuracy rates for each participant using combined templates from all other participants. This reduces mean accuracy to 84%. All palm-based postures are above 92% while *Fist* and *Straightedge* are again the least robust at 55% and 86% respectively. We attribute those differences to user-specific wrist orientations and differences in applied pressure on finger contacts when executing those poses. Depending on the size of the hand and the way fingers are clenched, for some users, a *Straightedge* can be mistakenly recognised as a *Fist* and vice versa (see confusion matrix in Table 2).

Impact of Sleeves

To evaluate the effect of users wearing long sleeves, we calculated accuracy rates for each participant, again using per-user templates and combined templates from all other participants. The average accuracy for long sleeves and per-user templates falls by approximately 7% for both template conditions, but this is primarily due to non-palm postures. For other-user templates, the accuracy drop is similar. In this worst case, *Fist* has a score of 43.6% and thus is not properly detected more than half of the time.

Summary

Overall, per-user templates with no sleeves are preferred, but other-user templates and sleeves are still practical for palm-based postures. All palm-based postures accuracy rates are above 90% regardless of template and sleeve conditions. This suggests that those postures can likely be used with a set of generic templates in application contexts, where individual calibration is not possible or too cumbersome, such as tabletops available in public environments.

SHAPE-BASED WIDGETS

We now explain how detected hand postures can form the basis for locally triggered widgets, whose position, shape, and parameters can adapt to the contour of the hand. Moreover, the convexity defects, the spikes and other geometrical properties of the shape obtained in the detection phase can also function as serviceable support points for the positioning and orientation of user interface elements. In particular, the region between the index finger and the thumb is a good candidate for widget placement or anchoring, as it is the largest space within the convex hull of the hand contour and also lends itself to a pinching metaphor, according to which users can virtually "hold" objects between their two fingers. Furthermore, the thumb can serve as a moving

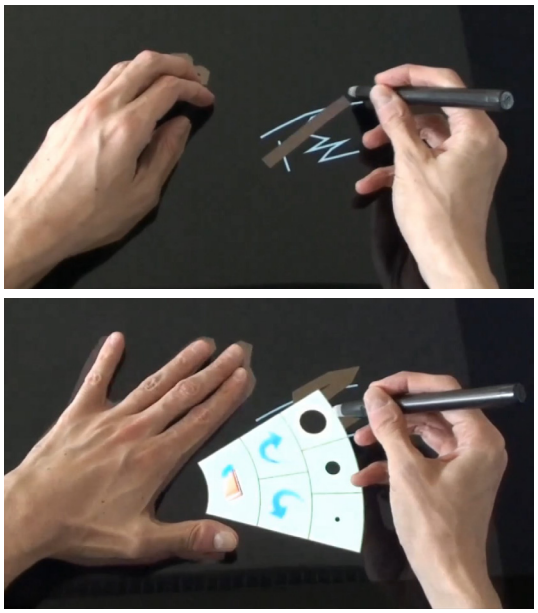

Figure 5. Transitioning from a relaxed pose of the non-dominant hand on the tabletop surface to a menu-triggering posture in a bimanual pen and touch context.

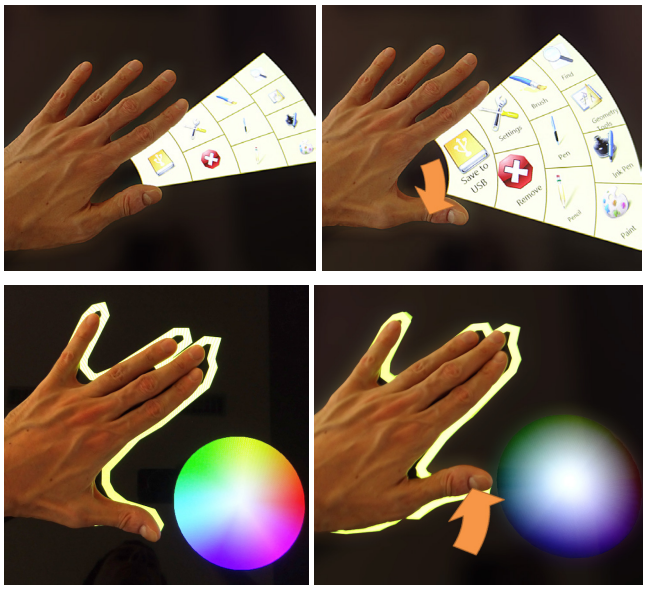

Figure 6. Using the thumb to control a widget parameter. Top: displaying further information about menu items. Bottom: controlling the saturation value of a colour wheel. Visual feedback for the selected colour is a thickly stroked hand contour.

"needle" to control the size or an attribute of the widget. The position of such a needle can easily be determined from the outmost spike formed by the thumb in the detected polygon.

We believe our hand shape polygons with identified hull defects theoretically allow finer widget arrangement compared to coarse ellipses [29] or techniques solely based on finger input [1, 4, 10, 26] and provide robust positioning anchors that are not readily available from continuous interaction paradigms based on physical models [32]. We provide examples of how those anchors can be used with concrete widgets below.

Posture-Widget Mappings

The above postures can be mapped to different types of widgets depending on their shape and space created around them. Examples of such associations are shown in Figure 1. Depending on the application context and the role of the hands in the target tasks, some mappings can be more judicious than others. For instance, postures with abducted index finger and thumb and hand edges with flexed fingers lend themselves to widgets that can adapt or be modulated by finger movements. The widgets associated with *PalmFingers*, *PalmThumb*, *PalmThumbPinky,* and *Corner* illustrate examples of such components (Figure 1 and Figure 6). Respectively, these are: a fan-shaped menu with adjustable opening angle, where additional items can be displayed beyond a particular threshold; a magic lens with controllable size or zoom level; an HSV colour wheel with modifiable saturation value; and a radial gauge. In the first three cases, the position or angle of the thumb adjusts a parameter of the widget, while in the fourth example the position of the gauge needle is determined by the angle formed by the flexed fingers with respect to the edge of the palm. We believe those interactions are intuitive and easy to execute as they are mapped to natural movements of the hand and leverage spatial memory for menu selection [26].

Application Context

We see the primary application context of our shape-based widgets to be bimanual work settings derived from traditional tabletop or desktop activities, where the non-dominant hand (NDH) lies on the surface while the dominant hand (DH) executes the main task. Classic examples of this bimanual division of labour in desktop work are drawing and writing [11]. Thus, we would like our detectable shapes and their associated menus to be actionable by switching from casual hand-resting poses with relaxed (lightly bent) fingers to specific taut postures near the user's workspace (Figure 5). Those transitions may involve very different postures or only small motions of the NDH, where sometimes even a slight lateral movement of a single finger suffices. We believe such a trigger mechanism is an interesting alternative to other more active menu-calling methods involving dragging gestures [15, 17, 19, 34]. Dragging is potentially problematic on friction-prone tabletop surfaces such as glass panes, so interactions with a low drag demand are desirable. Furthermore, we feel that the DH momentarily leaving the workspace to select an item from a menu activated or "held" by the NDH is an intuitive, even if not necessarily efficient, gesture. We liken it to the painter holding a palette in their NDH, while the brush held by the DH from time to time moves away from the canvas to dip into the colours of the palette before returning to the main painting task. In contrast to NDH chords [2, 9, 28], we also believe hand shape contains more information to be exploited for precise positioning and richer interaction.

EVALUATION: USABILITY IN CONTEXT

The formal analysis in the first evaluation did not consider errors emerging in the context of a real task, where the arm rests on the surface and switches between various menu-triggering postures. In pilot tests, we found that some shapes are more prone to involuntarily activation during casual interactions, especially those with few or no convexity defects, like *Palm*, *Straightedge,* and *Fist*. Furthermore, the dominant hand pose when writing can sometimes be misrecognised as a *Corner* posture. These observations, combined with lower recognition accuracies for non-palm-based shapes, suggest some postures may be unreliable without additional filtering or input data if users may freely rest their arms (potentially with long sleeves) and hands.

Therefore, we focus on the most accurately recognised *PalmThumb*, *PalmFingers* and *PalmThumbPinky* postures for further exploration and evaluation in a real application context where arm-resting is permitted. The same 15 participants took part in this second. Before running the actual study, we let participants test a demo interface featuring all supported postures and associated widgets shown in Figure 1. The goal of this study was to assess the general suitability and viability of selected hand shapes as menu triggers. We logged all actions and recorded possible mistakes or errors through observation along with qualitative feedback collected after the completion of the tasks.

Task

We chose a pen and touch drawing task, a classic example of bimanual tabletop work. The task we asked participants to execute was to roughly reproduce a series of four coloured sketches appearing in different areas of the workspace using a minimal toolset consisting of two menus: a colour wheel and a three-level fan menu to select stroke widths, undo, redo actions and a delete mode (Figure 5 bottom). Menu items could be selected either using finger touch or the pen. Visual feedback of selected ink or mode properties was provided as coloured contours of the hand and arm while touching the surface. The stroke colour and thickness reflected the currently selected menu settings, as illustrated in Figure 6 (bottom). Delete mode, when engaged, was indicated by a dashed contour. The sketches to reproduce were such that, to complete them, at least five menu activations and six item selections were needed, either to change the colour or the thickness of the stroke.

Protocol and Design

We conducted two different series of sketching tasks under different shape-matching schemes and posture-widget associations. Specifically, for one series (series A) we associated the colour wheel with *PalmThumb* and the tool menu with a relaxed version of *PalmThumbPinky*, that is, we allowed the ring finger to also be slightly abducted for the menu to be triggered. The shapes were recognised by the full algorithm described above, i.e. using the two shape-matching methods and with the user-specific templates obtained from the first part of the study. For the other series (series B), the colour wheel was to be triggered with *PalmFingers* and the tool menu with relaxed *PalmThumbPinky*. Since those postures involve at least two convexity defects, detection, here, was handled only using the method based on angle and defects matching. We were curious to see if those features would suffice to ensure a smooth experience with no or few false positives. Note that in both cases, users could switch between the two different postures with a simple abduction/adduction movement of a finger only: the little finger for a switch between *PalmThumb* and relaxed *PalmThumbPinky* and the index finger to change between relaxed *PalmThumbPinky* and *PalmFingers*. Moving the thumb was also possible in both cases to change the brightness of the current colour selection in the wheel.

Participants performed the tasks with both series. The order of the series and their association with each drawing model were counterbalanced.

Results

Overall, participants could easily trigger and use the colour wheel and the tool menu. Table 3 summarises the average number of menu invocations, menu item selections and errors that participants made in the two series.

	Series A	**Series B**
Avg. Menu Invocations	23.9 (SD=3)	25.1 (SD=3.5)
Avg. Item Selections	78.4 (SD=26.8)	72 (SD=15.8)
Avg. Invocation Errors	3.5 (SD=1.5)	2.7 (SD=1.7)

Table 3. Average menu invocations, item selections and invocation errors for the two sketching series

Errors

The menu invocation errors represent instances where users did not obtain the expected menu or a menu wrongly appeared due to misrecognitions or hand postures not being correctly executed (as observed by the study supervisor). In other words, errors include both false negatives and positives as they often occurred together. Indeed, we observed that most mistakes happened when a particular finger was not joined tightly enough to the others and thus, with the noise in the raw image data, caused the recognition algorithm to oscillate around the dividing threshold between the two menu signatures, which made the widgets appear in rapid successions. Participants were quick to react when they noticed that behaviour, and adjusted their finger position accordingly so that the desired menu was displayed in a stable manner. For most participants, this adjustment was only a matter of adopting the right habit, but, despite our efforts to relax some of the postures to allow for more flexibility, for three people, it was more difficult to keep fingers together or to spread them sufficiently wide apart in order to form the correct hand shape that would generate the required number of convexity defects. Conversely, some participants were much suppler and could extend their thumb very far, enabling wider ranges of angle-based adaptations between the index finger and the thumb.

Usage Pattern

Because the sketches were designed to require relatively frequent tool or colour changes, all participants mostly kept a menu hand posture active at all times and only four of them occasionally relaxed their hand or collapsed their fingers to make menus disappear.

Regarding the preferred switching method using the little finger or the index finger, 11 participants preferred the latter and 4 the former. Switching between the two types of menu using the index finger was generally perceived as being easier and more intuitive but people also remarked that it sometimes affected interaction with widgets that appeared within the index-thumb area. Conversely, changing menus with the little finger was deemed slightly more difficult but it had the advantage of being clearly separated from the menu display zone.

Regarding the finger motion itself, the majority of participants performed switches by abducting and adducting the required finger, but two participants found out they could also change menus by simply lifting the required finger. This gesture caused a convexity defect to disappear and thus trigger the other menu. Of course, keeping a finger raised while the rest of the hand is pressed flat on the surface for an extended period of time causes discomfort and thus, such poses are only viable for temporary activations. Continuing with the subject of muscular strain, four participants said that it was a little tiring to keep the palm down with all fingers spread for an extended period. This is not required by the interaction design, however, since menus can be triggered only when they are needed and the hand relaxed when not (Figure 5).

An interesting behaviour that we observed with three participants is that they would summon the colour wheel in the vicinity of the model sketch to be able to closely compare and match the colour that needed to be selected. While this type of operation is certainly possible with any movable or in-place menu, we believe that our widget-calling techniques are particularly suited for local placements and toolglass-like interactions [3], as users immediately know where and how they can expect widgets to appear when they place their hand at the desired location on the display.

Recall

Another comment that we received from our participants is that remembering which menu is activated by which finger position can be difficult when not familiar with the postures. Three people expressed that they would rather have had two markedly different postures, for example, a *Palm* and a *StraightEdge* to make the switching motion more obvious and thus cognitively more explicit. Posture-Tool associations are a matter of design choice, however, and widgets can be mapped to different shapes depending on user preferences and application context. If desired, very different postures can be chosen so that clearer separations exist between shape triggers in order to make them easier to remember, but at the cost of additional physical effort.

DISCUSSION

Our results can be summarised as design recommendations:

- Palm-based whole-hand postures are more robust as shape-based triggers than postures that generate a smaller or thinner handprint. They are less user-dependent and less prone to misdetections due to long sleeves and improperly rejected arm contact shapes. Therefore, we recommend they be considered first for hand shape-based interaction vocabularies. Careful consideration for possible dexterity and fatigue issues should be given.
- The more convexity defects the hand shape exhibits, the easier it is to identify using only convexity and angle matching, relying less on user-specific shape templates. Thus, postures such as *PalmFingers* and *PalmThumbPinky* are preferred when users cannot calibrate, such as public contexts.
- Due to differences in finger articulation skills, it is wise to relax posture requirements for a single widget trigger, especially one or two non-thumb finger abductions.
- Fingers, especially thumbs, can be used as moving needles to adapt or set widget parameters. The achievable scope is however very user-specific, as it depends on the extent of the joints' flexion and extension ranges. We suggest using either discrete switches or continuous mappings across short articulation ranges (similar to thumb controls used on gripped tablets [7]). When using such designs, potential conflicts with other finger-based postures should also be taken into consideration.

ADVANCED INTERACTIONS USING HOVER DETECTION

With minor refinements to our sharpness index algorithm, we can expand hand contact shape interaction to support explicit hover-based input.

Hover Detection for Proximity

Discrete approximations of hand hover can be obtained by using the sharpness index as an estimate of height. Ideally, we would like to have a hover state for each of our detected hand pose, but we found that it was not possible to obtain clear contours and convexity defects while the hand is above the surface (Figure 7 left). In practice, we can only define a general hover state by finding the right sharpness index threshold. This hover state can be used to trigger a

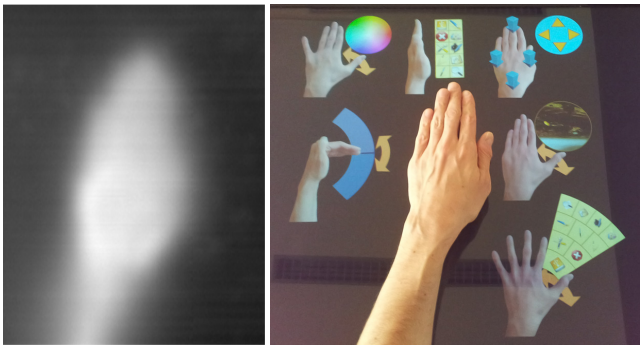

Figure 7. Left: hovering hand blurry image. Right: help guide revealed with a hand held for a moment above the surface.

help guide for novices revealing which hand shape summons which widget (Figure 7 right).

Hover Detection for Shear and Tilt

When a palm is tilted or shear forces applied, the blurriness of the hand shape has a distinct pattern with sharp discontinuities on one side and dim contours on the other. The difference between gradients applied to each side of the hand contour can be used to detect tilt direction.

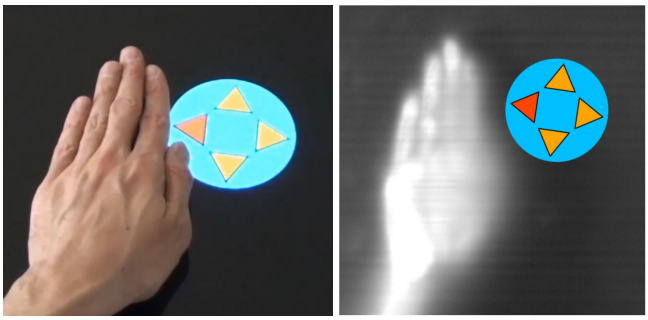

Figure 8. Tilting the palm to control a four-way direction widget. The portion of the hand pressing hard on the surface produces a sharp contour in that area of the contact shape, while the opposite region appears blurry.

Our algorithm can be extended to compute multiple local sharpness indices around the shape contour to estimate the amount of pressure exerted at those regions. This theoretically offers more flexibility for generic shapes than techniques to detect finger angles based on ellipses [33] or fingers [4, 13]. Probably the best posture to demonstrate this capability is *Palm*. In this pose, four zones corresponding to tilting or applied pressure can be defined for left, right, top and bottom directions. By measuring the differences between opposing indices, the tilt direction of the hand can be determined and used to control a navigation widget (Figure 8).

CONCLUSION AND FUTURE WORK

We described a generalised, reproducible computer vision algorithm to recognise hand contact shapes where arms can rest on the surface. We showed how hand postures can enable a variety of control menu and tool widgets well suited to bimanual tasks, where the non-dominant hand summons widgets and triggers modes to set the context for dominant hand input. Our evaluation of the detection accuracy and the performance of selected widgets in a realistic application provided us with valuable insights into contact shape-based interaction. Our main finding is that palm-based postures with spread fingers are most reliable, but user-specific articulatory issues are important to consider.

As future work, the accuracy of non-palm shapes could perhaps be improved by including texture features to further differentiate similar contour geometries. Our algorithm does not directly recognise hand poses when fingers are detected as separate blobs, such as during a grabbing motion. By tuning the hover and binarisation thresholds and training on those poses, we believe our method can be extended, which would further expand the range of the interaction vocabulary. To further increase the overall recognition precision and support the robust classification of even more postures, a machine-learning approach might be worth considering, especially as powerful deep-learning toolkits are becoming available. Finally, the potential for proximity, pressure, and tilt detection also warrant more detailed investigations.

We hope our work provides a practical tool, ideas, and insights for researchers and designers interested in hand contact shape-based interaction.

REFERENCES

1. Bailly, G., Demeure, A., Lecolinet, E. and Nigay, L. MultiTouch menu (MTM). In *Proc. IHM 2008*, 165-168.
2. Bailly, G., Müller, J. and Lecolinet, E. Design and evaluation of finger-count interaction: Combining multitouch gestures and menus. *Int. Jour. of Human-Computer Studies*, *70*, 10 (2012), 673-689.
3. Bier, E. A., Stone, M. C., Pier, K., Buxton, W. and DeRose, T. D. Toolglass and magic lenses: the see-through interface. In *Proc. SIGGRAPH '93*, 73-80.
4. Brandl, P., Leitner, J., Seifried, T., Haller, M., Doray, B. and To, P. Occlusion-aware menu design for digital tabletops. In *Proc. CHI EA 2009*, 3223-3228.
5. Chen, X. A., Schwarz, J., Harrison, C., Mankoff, J. and Hudson, S. E. Air+touch: interweaving touch & in-air gestures. In *Proc. UIST 2014*, 519-525.
6. Epps, J., Lichman, S. and Wu, M. A study of hand shape use in tabletop gesture interaction. In *Proc. CHI EA 2006*, 748-753.
7. Foucault, C., Micaux, M., Bonnet, D. and Beaudouin-Lafon, M. SPad: a bimanual interaction technique for productivity applications on multi-touch tablets. In *Proc. CHI EA 2014*, 1879-1884.
8. Frati, V. and Prattichizzo, D. Using Kinect for hand tracking and rendering in wearable haptics. In *Proc. WHC 2011*, 317-321.
9. Ghomi, E., Huot, S., Bau, O., Beaudouin-Lafon, M. and Mackay, W. E. Arpège: learning multitouch chord gestures vocabularies. In *Proc. ITS 2013*, 209-218.
10. Gu, J., Han, J. and Lee, G. HandCall: Calling a Tool by a Hand Gesture on the Tabletop. In *Proc. DIS 2012*.
11. Guiard, Y. Asymmetric division of labor in human skilled bimanual action: the kinematic chain as a model. *Journal of Motor Behavior*, *19*, 4 (1987), 486-517.
12. Harrison, C., Xiao, R., Schwarz, J. and Hudson, S. E. TouchTools: leveraging familiarity and skill with physical tools to augment touch interaction. In *Proc. CHI 2014*, 2913-2916.
13. Heo, S. and Lee, G. Indirect shear force estimation for multi-point shear force operations. In *CHI 2013*, 281-284.

14. Hinckley, K., Heo, S., Pahud, M., Holz, C., Benko, H., Sellen, A., Banks, R., O'Hara, K., Smyth, G. and Buxton, W. Pre-Touch Sensing for Mobile Interaction. In *Proc. CHI 2016*, 2869-2881.
15. Knies, R. In-Place: Interacting with Large Displays. *Reporting on research by Pahud, M., Hinckley, K., and Buxton, B. TechNet Inside Microsoft Research Blog Post.* 04.10.2012. https://blogs.technet.microsoft.com/inside_microsoft_research/2012/10/04/in-place-interacting-with-large-displays/
16. Koura, S., Suo, S., Kimura, A., Shibata, F. and Tamura, H. Amazing forearm as an innovative interaction device and data storage on tabletop display. In *Proc. ITS 2012*, 383-386.
17. Kurtenbach, G. P. The design and evaluation of marking menus. *PhD Thesis*, University of Toronto, 1993.
18. Le, K.-D., Paknezhad, M., Woźniak, P. W., Azh, M., Kasparavičiūtė, G., Fjeld, M., Zhao, S. and Brown, M. S. Towards Leaning Aware Interaction with Multitouch Tabletops. In *Proc. NordiCHI '16*, 1-4.
19. Luo, Y. and Vogel, D. Pin-and-Cross: A Unimanual Multitouch Technique Combining Static Touches with Crossing Selection. In *Proc. UIST 2015*, 323-332.
20. Matulic, F. and Norrie, M. Pen and Touch Gestural Environment for Document Editing on Interactive Tabletops. In *Proc. ITS 2013*, 41-50.
21. Ming-Kuei, H. Visual pattern recognition by moment invariants. *IEEE Trans. Inf. Theory*, *8*, 2 (1962), 179-187.
22. Moghaddam, A. B., Svendsen, J., Tory, M. and Albu, A. B. Integrating touch and near touch interactions for information visualizations. In *Proc. CHI EA 2011*, 2347-2352.
23. Pyryeskin, D., Hancock, M. and Hoey, J. Extending interactions into hoverspace using reflected light. In *Proc. ITS 2011*, 262-263.
24. Schmidt, D., Chong, M. K. and Gellersen, H. HandsDown: hand-contour-based user identification for interactive surfaces. In *Proc. NordiCHI 2010*, 432-441.
25. Schwarz, J., Xiao, R., Mankoff, J., Hudson, S. E. and Harrison, C. Probabilistic palm rejection using spatiotemporal touch features and iterative classification. In *Proc. CHI 2014*, 2009-2012.
26. Uddin, M. S., Gutwin, C. and Lafreniere, B. HandMark Menus: Rapid Command Selection and Large Command Sets on Multi-Touch Displays. In *Proc. CHI 2016*, 5836-5848.
27. Vogel, D. and Casiez, G. Hand occlusion on a multi-touch tabletop. In *Proc. CHI 2012*, 2307-2316.
28. Westerman, W. Hand tracking, finger identification, and chordic manipulation on a multi-touch surface. *PhD Thesis*, University of Delaware, 1999.
29. Wigdor, D., Benko, H., Pella, J., Lombardo, J. and Williams, S. Rock & rails: extending multi-touch interactions with shape gestures to enable precise spatial manipulations. In *Proc. CHI 2011*, 1581-1590.
30. Wu, M. and Balakrishnan, R. Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. In *Proc. UIST 2003*, 193-202.
31. Wu, M., Chia, S., Ryall, K., Forlines, C. and Balakrishnan, R. Gesture registration, relaxation, and reuse for multi-point direct-touch surfaces. In *Proc. TABLETOP 2006*, 185-192.
32. Xiang, C., Wilson, A. D., Balakrishnan, R., Hinckley, K. and Hudson, S. E. ShapeTouch: Leveraging contact shape on interactive surfaces. In *Proc. TABLETOP 2008*, 129-136.
33. Xiao, R., Schwarz, J. and Harrison, C. Estimating 3D Finger Angle on Commodity Touchscreens. In *Proc. ITS 2015*, 47-50.
34. Yoshikawa, T., Shizuki, B. and Tanaka, J. HandyWidgets: local widgets pulled-out from hands. In *Proc. ITS 2012*, 197-200.
35. Zhang, Z., Zhang, F., Chen, H., Liu, J., Wang, H. and Dai, G. Left and right hand distinction for multi-touch tabletop interactions. In *Proc. IUI 2014*, 47-56.