# Touch Technologies

*Sara Kilcher*
MSc Computer Science Student
Distributed Systems Group
ETH Zrich
Raemistrasse 101
8092 Zurich
sakilche@student.ethz

**ABSTRACT**
Touch has become a standard input method for smartphones over the last few years. Additionally, capacitive touchscreens have found their way into many other everyday devices, such as tablets, laptops, gaming devices and even glass-ceramic stoves. However, all of these application scenarios are limited to flat screens.

In this report, we will investigate three different approaches that go beyond the scope of traditional touchscreens. Using the 3rd dimension, they create entirely new application scenarios.

**ACM Classification:** H5.1 [Information interfaces and presentation]: Multimedia Information Systems. - Artificial, augmented, and virtual realities. H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

**General terms:** Design, Human Factors

**Keywords:** On-demand interfaces, finger tracking, on-body computing, appropriated surfaces, object classification, biosensing, bio-acoustics, multitouch, stereoscopic display, 3d user interfaces

## SKIN AS TOUCHSCREEN

The first two papers, OmniTouch [1] and Skinput [2] both try to appropriate skin as a touchscreen. To understand the motivation behind this, we look at current smart-phone trends.

Mobile phones are getting increasingly similar to computers. Their processing power and capabilities almost match standard PC's. At the same time mobile phone screens and therefore the interactive surface area is extremely small in comparison. A really cumbersome input interface with small buttons and keyboards that are error-prone is the result. This makes users unable to take advantage of the full potential of mobile phones. The bottleneck that is created between the user and the device cannot be mitigated by simply increasing the screen size, as this would significantly decrease the desired mobility.

A straightforward approach to solve this issue would be to use available surfaces in the user's environment such as tables. This easily allows for a larger input area while still keeping the mobile device small. There is however a downside to this approach. It is not suitable for situations where no table-like surfaces are available. For example when jogging or riding a bus people would still be forced to use the cumbersome phone touchscreen.

Yet, there is one surface that is available in every situation. Ones own skin. Using the skin as an input device has the additional benefit that it allows humans to exploit their good sense of proprioception. This makes it possible to interact with the surface even without visual cues. Both OmniTouch and Skinput are attempting to appropriate the skin for input using two different approaches.

## SKINPUT

Skinput is a direct follow-up of the paper *Enabling always-available input with muscle-computer interface* [3]. This paper attempted to recognize predefined gestures by measuring muscle activity using electromyography. The muscle computer interface reached an average accuracy of 79% for classifying four different gestures (please refer to the paper, section *Results, Part A* for details). In Skinput a higher accuracy is targeted, focusing on recognizing touch events as opposed to gestures. Using five fingers in a very similar setup, Skinput reaches an average accuracy of 87.7%.

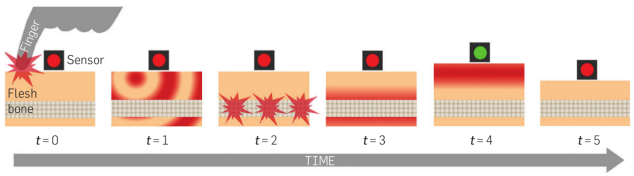
### Hardware

When a finger touches the skin somewhere on the hand or the arm, two types of waves are generated: longitudinal and transverse waves. The longitudinal waves (figure[1] 1a) cause bones to vibrate, which in turn creates new longitudinal waves along the length of the bone. At the same time, transverse waves (figure 1b) radiate out from the impact location, similar to the ripples that appear when throwing a pebble into water. These waves can be picked up by bio-acoustic sensors (note that the sensor cannot distinguish between the two wave types directly).

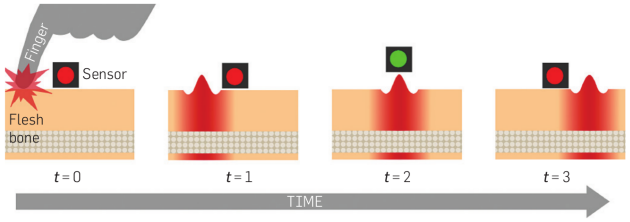
The Skinput hardware therefore consists of an array of ten bio-acoustic, mechanical vibration sensors, strapped to an armband. They are spatially distributed, and each of them is tuned for a specific frequency (Hz). Optionally, a pico-projector can be added to the setup.

A pico-projector has two advantages. On one hand, it is very small and lightweight, which makes it possible to strap it to the upper arm. Directly fixing the device to the person allows for a certain degree of mobility, while ideally keeping the

[1] All figures are either taken from the original papers or self made.




(a) Longitudinal waves




(b) Transverse waves

Figure 1: Waves emitted by finger impact

projected image steady on the lower arm. On the other hand, the chosen pico-projector uses laser technology. This ensures that the projected image is sharp regardless of the distance to the surface.

### Click Detection

The first step in detecting when a click event is triggered is to aggregate all of the sensors waves. An exponential average is calculated and analyzed in realtime by a computation unit. When this average exceeds the trigger threshold, it is considered a potential click, and the start timestamp is saved. If the average subsequently falls below the closing threshold (within 100 to 700 ms), the click is confirmed and the end timestamp is recorded. This can be observed in figure 2.

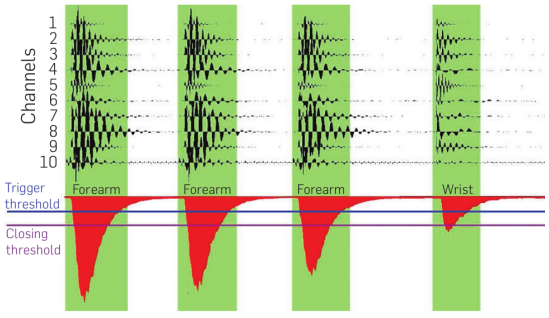



Figure 2: Exponential average with thresholds

Though simple, this method works extremely well, and the error rate was found to be negligible according to the authors.

### Click Localization

Once a click is detected, several features are calculated from the sensor signals measured between the start and end timestamp. The most important features of a total of 186 features are:

- For each sensor the average amplitude is calculated. Every average amplitude ratio between pairs of sensors is taken as one feature. This results in $\binom{10}{2} = 45$ features.
- Additionally, a Fast Fourier Transfom (FFT) is performed for every signal. The lowest 10 values are then used as features. This produces $10 * 10 = 100$ more features.
- The remaining 41 features were not found to have a significant impact, and are not further discussed.

The actual localization is being performed by a multiclass Support Vector Machine (SVM). In a training phase, each user has to tap every desired touch location several times.

After the training is completed, the system is ready to be used by this user. New click events can now be classified automatically by the SVM.

Note that only the previously trained locations can be recognized. Behaviour for new locations is undefined.

### Interface Design

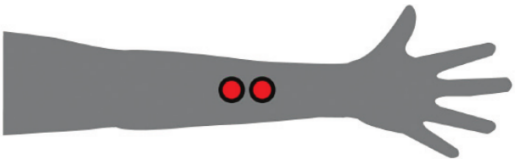
Using the presented components, simple interfaces can be designed. A few concrete examples are listed.

- Several buttons can be displayed in predefined locations. This allows for navigating through menus or lists and selecting the desired item.
- Another possibility would be to exploit proprioception and use an interface without a projector. Tapping a finger with the thumb could serve as trigger for a certain action, e.g. changing volume of a multimedia application.
- Besides the use cases mentioned by the authors, acoustic feedback could be given. For example incoming text or voice messages could be signaled. Controlled by finger taps these could then be displayed or deleted. In the projector-less case, the messages could be played instead.

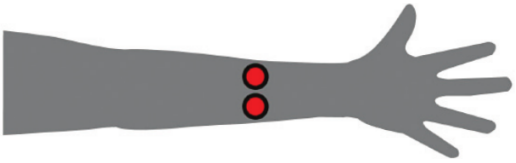
### Evaluation

Different button arrangements were analysed in a user study with 13 participants. On average 87.6% of the touch events were classified correctly. The experiment was setup so that each participant trained the device. This training data was then used to classify the later touches. Results from individual experiments ranged from 81.5% to 95.5%. A different number of input locations was used (5 to 10).

- Obviously, increasing the number of locations will result in more classification errors. However, performance can be increased by choosing locations wisely.
- On the arm, buttons arranged next to each other longitudinally (figure 3a) generally outperform a lateral arrangement (figure 3b).
- The distance from the click locations to the sensor armband also influences accuracy. This is assumed to be largely due to decreasing wave strength over distance and distortion of the signal caused by the joints. This was confirmed by an experiment. An accuracy loss of 7.2% was observed when moving the sensor from below the elbow to just above it.
- Although not mentioned by the authors, changing the arrangement of bones, joints and muscles by moving or rotating the arm has a great influence on accurate classification. This may be attributed to the same phenomenon that lets expert violinists create wonderful sounds with their instruments. Depending on the tension of the strings, different wavelengths are produced.



(a) Two longitudinally adjacent buttons



(b) Two laterally adjacent buttons

Figure 3: Button placements

### Conclusion

Using a novel approach, the authors have shown that it is possible to turn the skin into a usable touchscreen. Limitations include a relatively low number of locations that can be distinguished, as well as having only predefined locations available. Furthermore, the current need to retrain the system every time a user wants to interact with it is considered too cumbersome for practical applications. Hopefully this can be reduced in the future.

### OMNITOUCH

OmniTouch also tries to achieve capturing touch events on human skin to create an omnipresent interface. However, it extends the usable surface area to walls, tables and other flat surfaces surrounding the user. Also, different hardware is used to capture input, as described in the following section.

### Hardware

As shown in figure 4, a laser pico-projector and a short-range depth camera are fixed on the users shoulder. The projector and camera need to be calibrated in a common reference coordinate system, which is required for the projective texturing explained in section *Surface Segmentation and Tracking*. These devices are connected to a standard computer that is used for all the calculations.


(a) Hardware (b) Usage

Figure 4: The setup

### Finger Segmentation

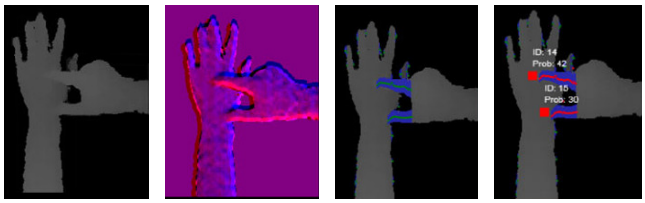
Each image from the depth camera (figure 5a) is passed through an image processing pipeline, where the following steps are performed:

1. The depth derivatives in x and y directions are calculated using a 5x5 sliding window, similar to a Sobel edge detector. The result of this step can be seen in figure 5b.

2. This image is searched for finger slices. The algorithm achieves this by looking at individual pixel columns and searching for a predefined pattern. This pattern consists of five parts, namely a relatively flat region at the beginning, middle and end of the potential finger slice and positive respectively negative derivatives in between. The order is very important to ensure that concavities do not get recognized (figure 5c). Also the lengths of the smooth region in the middle must be within predefined bounds. This ensures keeping real finger slices and discarding wrong slices e.g. from the hand. Note that this vertical search only allows detection of horizontal fingers.

3. With a greedy approach the slices are grouped into finger paths. Slices that cannot be grouped and groups that are too short or too long are discarded. Therefore, fingers that are bent may be ignored because they are too short. The remaining groups are assumed to be fingers, as seen in figure 5d.

4. In a last step, for each finger the fingertip is calculated as the middlepoint of the leftmost slice. Temporal smoothing (using a Kalman filter) is applied to ensure that the fingertip position does not flicker in the presence of noise.




(a) Depth image (b) Derivatives (c) Slices (d) Results

Figure 5: The finger segmentation pipeline

### Finger Click Detection

Based on the location of the finger from the previous steps, a flood fill operation is started from the midpoint of the detected finger. From there, the operation expands in all directions but right. Depending on whether the finger is touching the surface or not, the flood results in the images seen in figure 6. If not touching, only the finger itself is filled. But if the finger is close enough, the display surface itself will be flooded, resulting in a much larger area. Since the average area of a finger is known, the area size can be used as an indicator of whether the finger is touching or not. Therefore a simple threshold is set that triggers a click event when reached. This would not work, if the algorithm would expand in all directions, since in this case also the hand would be flooded.

For simple cases this approach works very well. Since it is applied on each frame independently, a dragging motion can be recognized as a series of continous clicks with moving fingertip.

Figure 6: The finger segmentation pipeline

### Surface Segmentation and Tracking

To be able to correctly display the interface, additional information about the location, orientation and movement of the display surfaces is required. This information can then be used to generate a simplified 3D model of the scene the camera is currently looking at. It is assumed that all of the surfaces are approximately planar. Using this model, projective texturing can be applied to display interfaces distortion free and in the correct location of the real world scene. This makes the interface appear glued to the surfaces, even when they are moved or rotated.

To actually segment and localize the surfaces, the following steps are performed.

**Components** First a connected components analysis is perfomed on the 2D depth image. Each large enough component is a potential interface.

**Z-axis rotation** The next step is to calculate the rotation angle around the Z-axis. A covariance analysis is performed on all the pixels belonging to the component. Computing the first two moments provides the major axes as well as the desired Z-axis rotation.

**Lock point** Finally, a lock point is required to determine the surfaces location in 3D space. Simply using the average of all pixels coordinates would be one possibility. But this has the main drawback that it is not stable in case of occlusions, even small ones. Therefore, the authors move the point 10 cm along the major axis from the previous step.

**Remaining rotations** The remaining rotations can be easily calculated by aggregating over the surface normals extracted from the depth image.

Using this technique makes it also possible to display and use several interfaces simultaniously. An example might be a drawing application where the color palette is displayed on the users hand and he can draw on the table.

Instead of using planar surfaces only, projection could be improved when using a more general 3D model. This model could be created with a similar technique like the one that is used in *Steerable augmented reality with the beamatron* [4] or iLamps paper [5]. Of course it would have to be adapted in such a way that it can cope with movement. Possibly in combination with the current surface tracking approach.

### Projected Interface

Up to this point it is not clear yet what constitutes a suitable interface. There are three different approaches on how to select an interface size.

1. Always display the smallest conceivable interface. This approach limits the size of the interface to the size of the smallest targeted surface, which is the hand. Projecting on walls or tables doesn't increase the interface size. This does not exploit the full potential of the bigger surfaces.

2. A second possibility consists of letting the user choose the interface size and location for each surface he wants to use. This approach is flexible but requires additional user input.

3. The last explored option tries to classify each surface into one of the five categories hand, arm, notepad, wall, or table. Depending on the classification, a suitably sized interface is displayed. Classification works using assumptions about surface normals and component sizes of the different categories. This approach works automatically, but it might be difficult to extend to further surface classes. Also, classification errors might occur.

### Evaluation

A user study was performed with 12 participants. We list the key insights gained.

- Finger clicks were generally detected very accurately, with an error rate of only 3.5% overall. Most of the errors were due to single clicks being identified as two or more clicks. Adding a simple timeout would decrease the error rate to 1.1%.
- Click location accuracy was examined and compared to traditional touchscreens. To identify 95% of the touch events correctly, button diameters on a touchscreen need to be at least 15 mm. For OmniTouch, the required diameter in the hand setting was found to be 22.5 mm. Even better was the performance of the wall setting with a required diameter of only 16.2 mm, reaching nearly the levels of a touchscreen.
- Dragging accuracy was also evaluated. Users were asked to trace projected lines and circles from start to endpoint. Instant visual feedback was given. The resulting average deviation from the given line was only 6.3 mm.

### Conclusion

OmniTouch has presented a proof-of-concept system and application to extend mobile interfaces to environmental surfaces. Since the system is very modular, the individual parts can be easily improved. For example finger detection might be extended in the future to be independent of direction. Also, the assumption about planar surfaces could be dropped in the future. Despite all of this, the main menhir that is still blocking the road to a commercial product remains the hardware. Carrying the proposed setup with the projector, depth camera and computation unit on the shoulders is not feasable. Only time will tell if the system can be miniaturized with sufficient quality to make it successful as a mass product.

## COMPARISON SKIN AS TOUCHSCREEN

In this section we will compare both of the presented papers and analyze their similarities and differences, as well as strenghths and weaknesses.

**Environment** No instrumentation of the environment is required, allowing the devices to be used in mobile settings. Despite that, OmniTouch manages to turn the environment into an interface, by letting the users appropriate surfaces such as tables and walls.

**Projection** Both papers use a pico-projector with laser technology to display sharp images on uneven surfaces. The disadvantage is a low projection brightness and therefore the systems cannot be used in bright light. However, the projector is optional in Skinput. This makes it possible to use the system in eyes-free settings such as driving.

**Input** Different sensors are used to get input events from the user. While OmniTouch relies on a depth camera, Skinput is using bio-acousting sensors. This also results in their different approach to recognizing input. Since the depth camera provides an image, an image processing pipeline is used for recognition. In contrast, machine learning is the chosen method for classifying input received via the bio-acoustic sensors. The choice of sensor also influences possible interaction gestures. The bio-acoustic sensors only react to *impacts*, making dragging and long-press touches impossible to recognize. The depth camera has no such limitations.

**Calibration** OmniTouch requires calibrating the projector and the camera in a unified 3D space. After this, any person can directly use the system. In contrast, Skinput requires an extensive amount of calibration. Each user has to calibrate the system for each input location. Additionally, recalibration is often required after arm movement.

**Potential** We are of the oppinion that the potential of OmniTouch is higher than that of Skinput. This has several reasons. Firstly, OmniTouch already displays a significantly higher accuracy. Secondly, the algorithms of OmniTouch can be easily improved further and made more robust. Each of the used algorithms is mostly independent and can be improved or replaced with state-of-the-art techniques. Improving one part improves the overall result. Doing the same in Skinput seems highly unlikely, since the entire classification is performed automatically in the SVM. The SVM may be tweaked, or replaced, but it will never be able to recognize classes for wich it hasn't been trained. Thirdly, users of Skinput have to hold their arms reasonably stable, while OmniTouch is much more motion-tolerant.

## TOUCHEO

Toucheo [6] tries to merge current consumer trends: Multitouch interaction and stereoscopic displays. In the last few years touchscreens as well as stereoscopic screens have become increasingly popular.

There are multiple reasons for attempting to combine these technologies. On one hand, mulitouch gestures allow for an easy and intuitive touch interface. On the other hand, interaction with virtual 3D objects (on a stereoscopic screen) is often hard and cumbersome with 2D interfaces. Especially 3D rotations are cognitively demanding.

The result of this paper should allow users to perform difficult tasks, such as a 3D docking task, easily and without much training.

### Hardware

The system consists of a commercially available touchscreen and stereoscopic display. Actually, both of them support stereoscopic images, but for the touchscreen model this functionality is disabled. These two screens are combined as seen in figure 7 and schematically in figure 8.

Figure 7: A user interacting with Toucheo

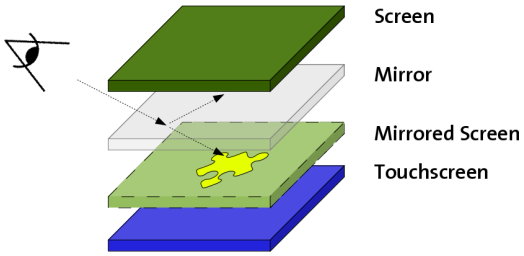



Figure 8: The screen gets mirrored

The user is looking down onto a touchscreen. Above it, the stereoscopic screen is hanging upside down. A semi-transparent mirror is mounted in between. This creates the illusion of a screen floating between the mirror and the touchscreen. When showing objects on the top screen and turning the rest of it completely dark, only the displayed object remains visible (figure 9). When a user interacts with the system, his hands rest on the touchscreen. The displayed 3D objects will automatically be above the users hands. Therefore, occlusion and depth collision are not an issue.

The technique of combining mirrors with screens has been been known for some time, see for example *Augmented Reality with Back-Projection Systems using Transflective Surfaces* [7].

Another advantage of this setup is that the user has a direct line of sight of both the touchscreen and the virtual 3D objects. Compared to a setup where both are shown side-by-side, this allows a faster switch between the two.

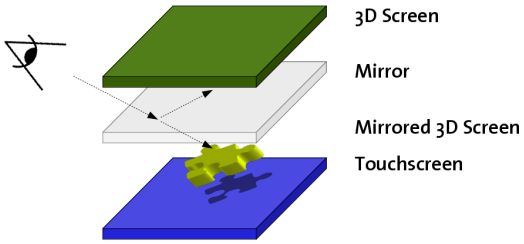



Figure 9: Only the bright object remains

### Interface

The authors designed an interface that enables users to transform virtual 3D objects with a total of 9 + 1 degrees of freedom (DOF): 3 for rotating, 3 for translating and 3 for scaling, with an additional pseudo-DOF for uniform scaling (see figure 10). Designing such an interface is quite challenging, especially since typical touchscreen interaction only provides 3+1 DOF (figure 11).

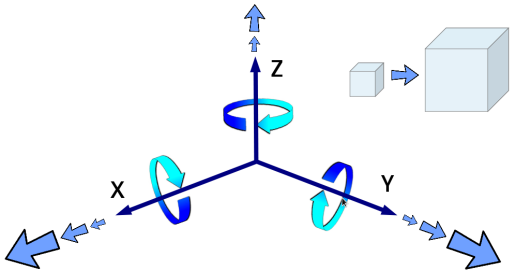



Figure 10: All 9+1 DOF

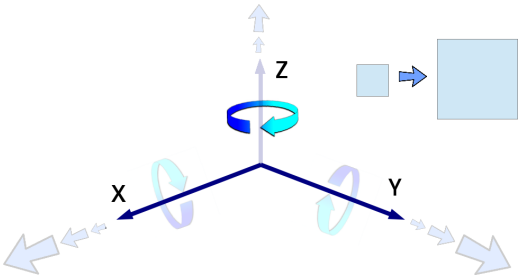
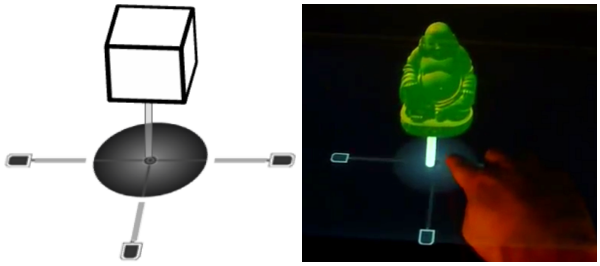



Figure 11: The 3+1 DOF of a typical smartphone

In the following it is explained how the final interface looks like. To create a visual connection between the virtual 3D object and its transformation widget (figure 12a) on the touchscreen, a virtual ray is used (see figure 12b).

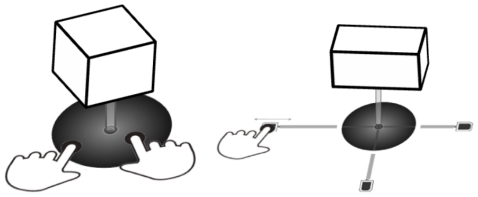
The known gestures from the mobile phone (figure 11) are implemented in an RST widget (figure 13a). For X and Y-axis manipulations, virtual rods are displayed. Rotations can be performed by dragging the finger across the rod (figure 13c). Moving the endpoint of the virtual rod outwards or inwards either enlarges or shrinks the object in the respective dimension (figure 13b). For manipulations in Z direction the user can touch the center of the widget whereupon a diagonal rod is displayed (figure 13d). Using the slider, the object



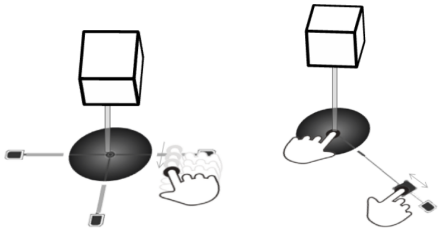
(a) RST widget (b) Virtual ray

Figure 12: Transformation widget

can be translated along the Z-axis. Scaling works in the same way as seen previously.



(a) RST widget (b) non-uniform scaling



(c) rotation (d) z-axis translation / scaling

Figure 13: Transformation widget operations

### Evaluation

A user study was perfomed with 16 participants. They were asked to perform a 3D docking task including translation, rotation and scaling. The most important findings are listed below.

- In general, the participants liked the system and were able to perform the required task quite well.
- Most difficult were Z-axis manipulations. The authors already incorporated the feedback and improved that specific part of the interface.
- Almost none of the participants experienced sickness or fatique symptoms.

Many 3D displays can cause headaches because depth cues are not consistent with what people know from the real world. The monocular depth cues[2] are mostly correct. However, when studying binocular depth cues[3], stereoscopic displays

[2]please refer to `http://en.wikipedia.org/wiki/Depth_perception#Monocular_cues` for a list

[3]details at `http://en.wikipedia.org/wiki/Depth_perception#Binocular_cues`

only provide a correct vergence distance and not focal distance. According to *Vergence-accomodation conflicts hinder visual performance and cause visual fatique* [8], the difference in these two distances can cause visual fatique.

In Toucheo, the focal distance for the floating 3D objects is almost correct, which explains why users don't experience this symptom. This is due to the fact that the focal distance is really above the touchscreen because of the mirror. For flat objects floating in the plane of the mirrored display, focal distance is completely correct. Of course, increasing the depth of the displayed virtual objects also increases the focal error. However, it is in any case considerably less than if the objects were displayed at that height from the (stereoscopic) touchscreen.

### Conclusion

Toucheo uses a clever setup to make the commonly difficult 3D docking task solvable. This proves that also other tasks that involve 3D object manipulations could benefit from Toucheo.

The developed widget with the virtual rods is an additional achievement. In theory it could also be used in other settings (even without a stereoscopic screen) to perform 3D operations on virtual objects.

## COMPARISON TOUCH TECHNOLOGIES

In this section we will juxtapose Toucheo with the two skin papers, OmniTouch and Toucheo.

**Touchscreen** Both Toucheo and the skin papers use touchscreens in the broadest sense. While Toucheo is using a traditional touchscreen to interact with virtual 3D objects, the skin papers transform real 3D objects into touchscreens.

**Display** Analogously, Toucheo also uses traditional displays, both for the 3D objects as well as the 2D widgets as opposed to the skin papers. They are creating displays in an on-demand way by projecting images onto real objects.

**Input** All papers require a user to interact with the systems using touch events. The skin papers focus on correctly recognizing where and when a touch event is triggered. This information is already available in Toucheo. The main contribution is what can be achieved using touch events in conjunction with a clever setup and user interface. For that purpose, a special user interface was designed. The skin papers use common graphical user interfaces instead.

## CONCLUSION

In this report we have seen how the third dimension can be used to create entirely new touch applications. One of the motivations for doing this has been the fact that todays mobile phones do not provide a sufficient interaction area. There are however other approaches to solving this problem. Researchers, as well as Samsung and Sony are working on developing new types of screens that can be folded, bended or rolled[4]. Using such a screen the interaction area could be improved without taking away mobility.

[4]For more details and sources visit `http://en.wikipedia.org/wiki/Flexible_display`

## REFERENCES


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

2. Chris Harrison, Desney Tan, and Dan Morris. Skinput: appropriating the body as an input surface. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 453–462. ACM, 2010.

3. T Scott Saponas, Desney S Tan, Dan Morris, Ravin Balakrishnan, Jim Turner, and James A Landay. Enabling always-available input with muscle-computer interfaces. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pages 167–176. ACM, 2009.

4. Andrew Wilson, Hrvoje Benko, Shahram Izadi, and Otmar Hilliges. Steerable augmented reality with the beamatron. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, pages 413–422. ACM, 2012.

5. Ramesh Raskar, Jeroen van Baar, Paul Beardsley, Thomas Willwacher, Srinivas Rao, and Clifton Forlines. ilamps: geometrically aware and self-configuring projectors. In *ACM SIGGRAPH 2006 Courses*, page 7. ACM, 2006.

6. Martin Hachet, Benoit Bossavit, Aurélie Cohé, and Jean-Baptiste de la Riviére. Toucheo: multitouch and stereo combined in a seamless workspace. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 587–592. ACM, 2011.

7. Oliver Bimber, L Miguel Encarnaçáo, and Dieter Schmalstieg. Augmented reality with back-projection systems using transflective surfaces. In *Computer Graphics Forum*, volume 19, pages 161–168. Wiley Online Library, 2000.

8. David M Hoffman, Ahna R Girshick, Kurt Akeley, and Martin S Banks. Vergence–accommodation conflicts hinder visual performance and cause visual fatigue. *Journal of vision*, 8(3), 2008.