MULTI-TOUCH SURFACE BASED ON RGBD CAMERA

Klemen Istenič, Luka Čehovin, Danijel Skočaj
University of Ljubljana, Faculty of Computer and Information Science
klemen.istenic@gmail.com, luka.cehovin@fri.uni-lj.si, danijel.skocaj@fri.uni-lj.si

ABSTRACT

The popularity of interactive surfaces is increasing because of their natural and intuitive usage. Adding 3D multi-point interaction capabilities to an arbitrary surface creates numerous additional possibilities in fields ranging from marketing to medicine. Interactive tables are nowadays present in numerous museums, schools and companies. With the advent of low-cost RGBD cameras, thee-dimensional surfaces are slowly emerging as well, attracting even more attention. This paper presents an affordable system for 3D human-computer interaction using a RGBD camera that is capable of detecting and tracking user's fingertips in 3D space. The system is evaluated in terms of accuracy, response time, CPU usage, and user experience. The results of the evaluation show that such low-cost systems are already a viable alternative to other multi-touch technologies and also present interesting new ways of interaction with a surface-based interfaces.

1 INTRODUCTION

With the reduction in size and by increasing the computational power that we have witnessed in the past decades, computers have become indispensable and ubiquitous in everyday life. Regardless of all the progress, the methods of human computer interaction most widely used have remained almost unchanged since the 1980s, when a computer mouse became a crucial part of every desktop computer. Despite the technological advancements, the ways of using a computer mouse remained the same, together with all of its shortcomings.

Only in the last decade, new technologies that enable multitouch interaction and eliminate several limitations of a mouse have become available. Decreasing the production costs of multi-touch screens greatly contributed to their inclusion in practically all new mobile devices and even in the majority of the new laptops. On the other hand, high cost of larger screens limits the technology to smaller portable devices. Bigger multi-touch surfaces have been developed using IR cameras and emitters combined with a projector and utilizing advanced computer vision algorithms. Well known examples are commercial multi-touch table *Samsung SUR40*, with *Microsoft PixelSense* technology [1] and an open-source software package *Community Core Vision (CCV)* [2]. High cost of the first and a complex construction of the second are the main reasons why they remained limited to large institutions and a handful of HCI enthusiasts. Even though these solutions provide multi-touch interaction, the interaction remains limited to a 2D plane. With the introduction of low-cost depth cameras, such as *Microsoft Kinect* and *Asus Xtion Pro*, HCI researchers have gained a cheap and efficient way of obtaining information that would have otherwise require special controllers or multi-camera systems with complex and extremely sensitive calibration.

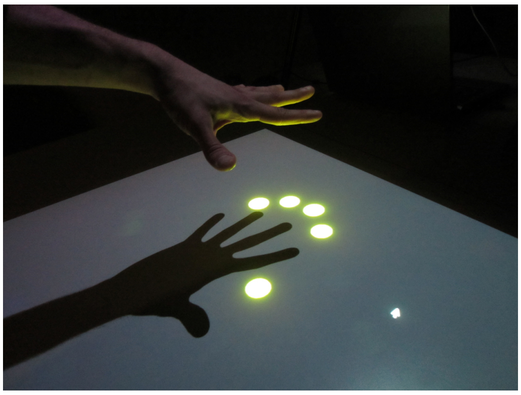

Figure 1: 3D finger interaction

The ideas for development of 2D multi-touch surfaces, by observing a small area above the surface were introduced in [14]. Enlarging the observed area above the surface, to enclose the area of the palms enabled [12] to eliminate the majority of false touch detections as well as extract additional information (hand type, finger-hand association, etc.) Portable depth camera and projector were used in [3] to allow the detection on a changing surface in real time. The research has also focused on interaction in 3D space with [5] acquired 3D model of the scene to which touch capabilities were added without restriction of the shape of the scene. [1, 6, 4] went one step forward, providing the user the ability to capture a real object and manipulate with it in virtual world. Latter also studied the 3D interaction by detecting and tracking users fingertips using a specific surface material. Researchers in [9] have developed systems that enable users multi-touch interaction on an arbitrary surface and basic 3D interaction through finger and hand gestures.

The main focus of our work is on increasing robustness and

[1] Microsoft PixelSense: http://www.microsoft.com/en-us/pixelsense/

[2] Community Core Vision: http://ccv.nuigroup.com/

generality of depth-camera based multi-touch systems using systematic evaluation the limits of technology in terms of accuracy, speed, and usability. We present a system that adds full 3D finger interaction capabilities to an arbitrary surface (shown in Figure 1) using depth camera, a projector, and a middle-ware software module that performs finger detection and has been sufficiently optimized that it can be run on a conventional desktop computer (without hardware acceleration). Besides the description of the system and the finger tracking method, a major contribution of this paper is a detailed empirical evaluation in which we highlight the capabilities and limitations of the system. In Section 2 we present the system components. In section 3 we describe the detection and tracking algorithms. In Section 4 we present the evaluation of the system and we conclude the paper with a short summary and ideas for future work in Section 5.

2 SYSTEM OVERVIEW

Our system that is capable of adding 3D multi-touch functionalities to an arbitrary surface using a low-cost depth camera, a projector and an ordinary computer, is shown in Figure 2.

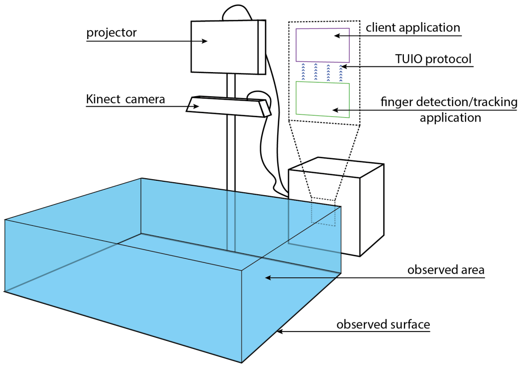

Figure 2: Components of the system

Depth camera acquires 3D information about the scene. In our prototype we use *Microsoft Kinect* camera, due to its low cost and decent support in research community, however, other cameras could be used as well. Microsoft Kinect contains an IR projector and IR camera as well as a RGB camera.

Projector is used to project the target application to the observed surface. In our setup the projector is positioned above the surface and under an oblique angle. We have used a wide lens projector that produces an image of similar size than the area captured by Kinect positioned at a similar distance to the surface.

Surface, to which we intend to add the 3D multi-touch capabilities, can be any planar surface, at any orientation. The only limitation is its material, as Kinect camera does not correctly work with reflective and transparent materials. With additional implementation of an appropriate mapping function, the planar shape limitation could also be waived.

Software of the system can be divided into finger detection and tracking middle-ware that is described in Section 3 and the client target. Fingers detected by the middle-ware are transmitted to the client application using the TUIO protocol [8].

3 FINGER DETECTION AND TRACKING

The process of a precise detection and robust tracking of users fingers is divided into the initialization, that is performed only once, and the detection stage executing constantly.

3.1 Initialization

Initialization of the system consists of building a surface model and calibration of depth camera with the projector. The surface model contains the depth model of the background, a mathematical equation of the observed surface and the information about the borders of the observed area. The depth model is the reference model used to classify pixels as foreground-background in the first step of detection. Each pixel in the depth image is modeled with an independent Gaussian model. By continuously updating the model only with pixel values classified as the background [10], we ensure that the foreground objects (such as users hand) will not be fused with the background, even if they persist at the same location for a longer period of time.

The geometry of the observed planar surface is modeled using mathematical equation of a plane in 3D space robustly estimated using RANSAC [2] method on a point cloud constructed from the depth image. The calibration of the camera and the projector is vital for the correct mapping of any detected finger to the reference frame of the target application. A transformation between both coordinate systems is obtained using barycentric coordinates. To provide the reference points, the observed surface is divided into triangle grid. In the interactive calibration process the user provides the information about the location of reference points in both coordinate systems. After the calibration, every point can be easily transformed to the other coordinate system, by finding the triangle in which it lies and computing the barycentric weights.

3.2 Finger detection and tracking

Every captured depth image is processed with a series of steps to determine the positions of the fingers. First, the background is removed using the background model. Every pixel located below the surface is instantly discarded as are the pixels that fit the depth model of the surface. The remaining regions are split and individually analyzed using the k-curvature algorithm [13] that ensures quick and robust detection of finger candidates. Each point of interaction is computed in 3D space, as the mass center of the elements in an area enclosed by the fingertip contour. Surface touch events are detected by thresholding the distance of the fingers to the surface. These events can then be used to mimic the click action, e.g. simulate a click of a computer mouse. Fingers have to be associated over frames to enable the user interaction using temporal gestures. Tracking of individual fingers is performed using Kalman filter [7] with a nearly-constant-velocity motion model. At every time step the algorithm attempts to associate detected fingertips with the detections from the previous time step. Fingertips which are not associated are considered to be new fingers. The association is done using suboptimal nearest neighbor (SNN) [11] with local optimization of the distances. The locations of the detected fingers are given in the camera coordinate space. They are then transformed to the observed surface space by computing a perpendicular projection to the surface plane. Then the location in the projector space is obtained using the barycentric coordinates provided during the calibration phase.

4 EVALUATION AND DISCUSSION

We evaluated our system in terms of accuracy, response time, and user experience. A desktop computer running Ubuntu 12.04 with Intel Core i5 and 8GB of RAM was used in the evaluation. Kinect and projector were positioned at a height of 0.91 m and 1.23 m and inclination of 15° and 10° respectively. The size of the resulting observed volume was $65 \times 49 \times 31$ cm.

Accuracy: As our main objective is to provide the user with an accurate and responsive system, we can mark a correct detection of a finger only, if the detected point of interest lies on the finger itself. Considering the average width of a finger being between 1.5 and 2 cm, the acceptable detection error is up to 0.5 cm. First we performed a calibration step using 6×5 grid of points. Two evaluation scenarios were then performed. In the first scenario, the error was measured at the center of gravity in each of the triangles, as they represent the average errors. In the second set, the accuracy was measured at 4 randomly selected points in each of the triangles.

Results of the evaluation are summarized in Table 1. Figure 3 shows the location of the evaluated points together with the detected locations in the first as well as the second test set. Dotted lines mark the borders of the calibration triangles, with the calibration points located at their vertices. The distribution of the errors combined for both sets is shown on Figure 4. This experiment shows that the system is sufficiently accurate. The detection error was less than 7 mm in 98% of the points, which still enables a satisfactory interaction with the system.

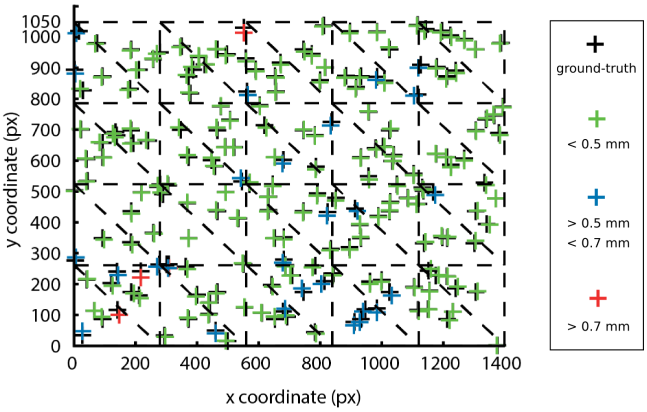

Figure 3: Accuracy evaluation

Number of points	**200**
Points with error $< 5mm$	**169 (84.5%)**
Points with error $< 7mm$	**196 (98.0%)**
Avg. error	**3.1mm**
Avg. error on axis x / y	**1.5mm / 2.4mm**
Avg. error in pixels	**6.4px**
Size of pixel	**0.46mm × 0.49mm**

Table 1: Accuracy evaluation summary

Responsiveness: The responsiveness of the system is calculated as the elapsed time between the users action and the display of the consequence. The overall delay was estimated empirically, using a camera capable of capturing 60 fps, while the processing time of each frame was computed within the software as the time elapsed between receiving a depth image and sending the positions of fingers to the client application.

The overall response time of the system was on average 120 ms for a single and 125 ms for ten fingers. Processing of a single frame took on average 8 ms, while the displaying time is 1 ms. It is evident that the main source of the delay is the depth image acquisition, which could be shortened by using a faster depth camera. Even though the total response time is relatively big, it was not noticeable in the majority of the applications. The delay only affects the user experience in applications that require quick responses, e.g. real-time games.

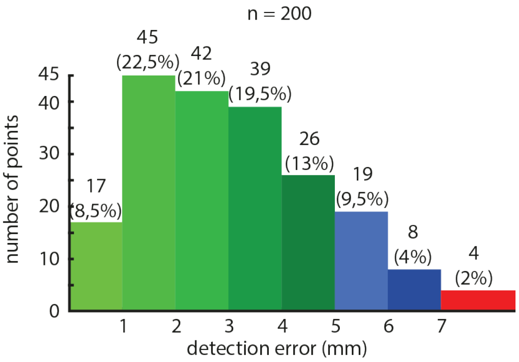

Figure 4: Distribution of errors

Computational performance: To confirm or disprove the hypothesis of an ordinary computer being sufficiently powerful to run our system, we have monitored the CPU usage, while a user was interacting with the system using both hands (10 fingers) for 30 seconds. The average usage of each of the 4 cores was 20%, leaving enough processor power to simultaneously run the client application.

User experience: To the best of our knowledge there does not exist and application that would use 3D information obtained over TUIO protocol, so we have decided to evaluate user experience using applications developed for 2D interactive tables (*Microsoft Touch Pack for Windows 7*) as well as standard TUIO-compatible applications. To observe the interaction in 3D we have designed a simple application that displays circles at the positions of the detected fingers, with the color and size depended on the fingers distance to the observed surface as shown in Figure 1. Although the application is simple it gave us the ability to observe users problems and estimate the robustness of the 3D detection.

Perceived user experience closely resembled the results obtained in the evaluation with accurate detections and barely noticeable delay. Only for applications, where quick responses are necessary (Microsoft Rebound), the delay prevented normal usage. In 3D interaction users faced some problems mainly due to the occlusions occurring among fingers. Occluded fingers, once made visible again, were marked as new fingers instead of being associated with the previous detection a few frames before.

System limitations: Our system can be used under the majority of conditions, with a few exceptions mainly due to the limitations imposed by the hardware components. Near-IR light is used by the Kinect camera, therefore the system is limited to non-reflective and opaque materials and should not be used in the direct sunlight. Maximum dimensions of the observed area are limited to 80×69 cm as the depth sensor is only able to accurately measure distances in the range between 40cm and 90cm. Static placement of the Kinect and projector is also essential for accurate operation of the system.

5 CONCLUSION

We have presented a system for adding 3D multi-touch functionalities to an arbitrary surface using a depth sensing camera using commodity hardware components, such as Kinect camera, a projector, and an ordinary desktop computer. In the presented evaluation we have shown that the system is very accurate. The average measured response time allows normal everyday usage of majority of applications. We have to emphasize that the vast majority of the processing time is the consequence of current hardware limitations.

Our future works includes development of applications exploiting the 3D information. Change of the fingers distance to the surface could be used to manipulate the level of details of information presented, enlarge/shrink an area on the map or in conjunction with graphical applications, set the size or thickness of the tools used. System could also be used in scenarios, where tracking of fingers in 3D could help understanding users motivation better or in sterile environments where frequent disinfection of the tactile surface is required.

References

[1] H. Benko, R. Jota, and A. Wilson. Miragetable: freehand interaction on a projected augmented reality tabletop. In *SIGCHI 2012*, pages 199–208. ACM, 2012.

[2] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[3] C. Harrison, H. Benko, and A. D. Wilson. Omnitouch: wearable multitouch interaction everywhere. In *ACM UIST 2011*, pages 441–450. ACM, 2011.

[4] O. Hilliges, S. Izadi, A. D. Wilson, S. Hodges, A. Garcia-Mendoza, and A. Butz. Interactions in the air: adding further depth to interactive tabletops. In *ACM UIST 2009*, pages 139–148. ACM, 2009.

[5] H. Ishii, C. Ratti, B. Piper, Y. Wang, A. Biderman, and E. Ben-Joseph. Bringing clay and sand into digital design—continuous tangible user interfaces. *BT technology journal*, 22(4):287–299, 2004.

[6] B. R. Jones, R. Sodhi, R. H. Campbell, G. Garnett, and B. P. Bailey. Build your world and play in it: Interacting with surface particles on complex objects. In *ISMAR 2010*, pages 165–174. IEEE, 2010.

[7] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[8] M. Kaltenbrunner, T. Bovermann, R. Bencina, and E. Costanza. Tuio: A protocol for table-top tangible user interfaces. In *Workshop on Gesture in Human-Computer Interaction and Simulation*, 2005.

[9] F. Klompmaker, K. Nebe, and A. Fast. dsensingni: a framework for advanced tangible interaction using a depth camera. In *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction*, pages 217–224. ACM, 2012.

[10] D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao, and S. Russell. Towards robust automatic traffic scene analysis in real-time. In *ICPR 1994*, volume 1, pages 126–131. IEEE, 1994.

[11] P. Konstantinova, A. Udvarev, and T. Semerdjiev. A study of a target tracking algorithm using global nearest neighbor approach. In *Proceedings of the International Conference on Computer Systems and Technologies*, 2003.

[12] S. Murugappan, N. Elmqvist, K. Ramani, et al. Extended multitouch: recovering touch posture and differentiating users using a depth camera. In *ACM UIST 2012*, pages 487–496. ACM, 2012.

[13] C.-H. Teh and R. T. Chin. On the detection of dominant points on digital curves. *TPAMI*, 11(8):859–872, 1989.

[14] A. D. Wilson. Using a depth camera as a touch sensor. In *ACM ICITS 2010*, pages 69–72. ACM, 2010.