Lightweight Palm and Finger Tracking for Real-Time 3D Gesture Control

Georg Hackenberg* Fraunhofer FIT

Rod McCall‡ Fraunhofer FIT

Wolfgang Broll△ Fraunhofer FIT

ABSTRACT

We present a novel technique implementing barehanded interaction with virtual 3D content by employing a time-of-flight camera. The system improves on existing 3D multi-touch systems by working regardless of lighting conditions and supplying a working volume large enough for multiple users. Previous systems were limited either by environmental requirements, working volume, or computational resources necessary for real-time operation. By employing a time-of-flight camera, the system is capable of reliably recognizing gestures at the finger level in real-time at more than 50 fps with commodity computer hardware using our newly developed precision hand and finger-tracking algorithm. Building on this algorithm, the system performs gesture recognition with simple constraint modeling over statistical aggregations of the hand appearances in a working volume of more than 8 cubic meters. Two iterations of user tests were performed on a prototype system, demonstrating the feasibility and usability of the approach as well as providing first insights regarding the acceptance of true barehanded touch-based 3D interaction.

KEYWORDS: 3D user interfaces, hand tracking, computer vision, multi-touch, gesture recognition.

INDEX TERMS: H5.2 [Information interfaces and presentation]: User Interfaces. - Input devices and strategies; Graphical user interfaces, I3.6 [Computer Graphics]: Methodologies and Techniques. – Interaction Techniques, I4.6 [Image Processing and Computer Vision]: Segmentation. – Edge and Feature Detection.

1 INTRODUCTION

Multi-touch interaction techniques have become widely available recently, being used for instance in table top systems such as Microsoft's Surface [30] projection-based systems such as CityWall [24], desktop systems such as HP's TouchSmart series as well as in several mobile devices, in particular smartphones such as the Google Nexus and, of course, the iPhone. The introduction of multi-touch interaction techniques has probably been the most important change to user input since the introduction of the mouse.

To date, multi-touch interaction typically is surface based. Selection of objects is required before actually manipulating them by touching the corresponding surface with the hands.

Freehand multi-touch has been explored within various approaches, e.g. Oblong's g-speak[1], after initially having been introduced in the popular Hollywood movie "Minority Report". They usually depended heavily on hand-worn gloves, markers, wrists, or other input devices, and typically did not achieve the intuitiveness, simplicity and efficiency of surface (2D) based multi-touch techniques[2]. In contrast to those approaches, the goal of our approach was to use barehanded interaction as a replacement for surface based interaction.

Only a vision-based approach will allow for freehand and barehanded 3D multi-touch interaction. The system must also provide sufficient solutions for the following four steps: detection of hand position without prior knowledge of existence; for each appearance determine pose from image cues; track appearances over time; recognize gestures based on their trajectory and pose information. Various approaches exist to solve all four problems, each featuring different advantages and disadvantages.

Barehanded 3D interaction has recently been presented by Mygestyk[3] and provides the basis for Microsoft's Kinect interface for Xbox 360[4]. However, these approaches are limited either to a single pair of hands, or to a rather coarse detection of the hand (in contrast to individual fingers and finger tips).

Our approach allows for barehanded multi-touch interaction in immersive 3D settings using hand and finger based interaction. As such, our approach extends intuitive multi-touch interaction to Virtual Reality (VR) and Augmented Reality (AR) applications, allowing for full 3D interaction rather than restricting the user to a certain interaction plane. The approach is not limited regarding the number of segments it can track (making it fully multi-user capable), while providing a fine granularity for the development of specific interaction techniques.

The paper begins with a review of related work, then goes on to describe the system design before providing results from an early user study followed by a discussion and conclusion.

2 RELATED WORK

2.1 Hand Detection

Several image cues such as color, texture, and shape have been used to approach the problem of hand detection. Donoser and Bischof [5] used statistical models for skin color to give clues about possible hand segments. Matas et al. [21] extended this work, and extracted hand segments using the Maximally Stable Extremal Region operator.

To overcome the problem of susceptibility towards changing lighting conditions in predefined color models, Rosales et al. [25] used a neural-network based face detector to build and maintain active color models. Using another approach, Barhate et al. [3] integrated a second motion cue into the detection framework, where color and motion are used to initialize an EigenTracker.

* e-mail: ghackenberg@gmail.com
‡ e-mail: roderick.mccall@uni.lu
△ e-mail: wolfgang.broll@fit.fraunhofer.de

[1] See http://www.oblong.com/, last visited March 2010.

[2] See http://www.billbuxton.com/multitouchOverview.html, last visited March 2010 for an overview and history.

[3] See http://www.mgestyk.com/, last visited March 2010.

[4] See http://www.xbox.com/de-de/kinect/, last visited Sept. 2010.

Another approach developed by Delamarre et al. [4] uses dense stereo reconstructions to achieve robustness against background clutter and other environmental noise, utilizing depth information for hand region segmentation. To overcome the disadvantage of expensive to compute dense disparity maps, Liu and Jia [18] employ specialized hardware such as the FingerMouse.

Other methods have been developed, i.e. Kolsch and Turk [13] used a learning-based method. They adapted the Viola-Jones face detector in the context of hand detection. The final detector is build from a cascade of weak classifiers, which are trained for a specific initialization posture using the AdaBoost algorithm. A similar idea is exploited by Ghobadi et al. [7], but a combination of range data taken by a time-of-flight camera and regular color information is used instead to increase robustness and efficiency.

2.2 Pose Estimation

The complexity of the pose estimation tasks depends on the hand model employed and its degrees of freedom. Limited reconstruction techniques have been used in a number of real-time systems. A basic approach is to estimate the palm position as the centroid of the hand blob [27] Abe et al. [1] used a more stable estimate given by the point with the maximum distance to all boundary edges. In contrast, fingertips can be found using circular mask convolution [16] curvature analysis [28] or palm-edge distance transform analysis [22] which is considered to be more robust with respect to segmentation noise.

Full pose reconstruction is a more complex task due to the number of parameters that have to be estimated simultaneously. An elegant approach is to train a set of Specialized Mappings [25] with functions based on arbitrary image features, such as the Hu moments. A major problem of this approach is the huge variability in articulation and appearance, requiring an extensive database of training samples.

2.3 Hand Tracking

Techniques for hand tracking can roughly be divided into appearance- and model-based approaches. One approach to appearance-based object tracking is correlation-based feature matching, i.e. evaluated by Kolsch and Turk [14] using KLT features selected from the bounding box of the hand.

Barhate et al. [3] apply the ideas of the EigenTracking framework instead. The main disadvantage of this method is its limited robustness towards background clutter, which is essential in multiuser scenarios. Other systems, i.e. those by Argyros and Lourakis [2] use prediction models and assume constant linear motion in the image plane between two consecutive video frames, while i.e. Kim et al. [12] use second order auto-regression of the limb motion instead.

Lu et al. [19] describe a prototype of model-based hand tracking techniques. They are using an iterative fitting procedure to gradually reduce the matching error between model and observation, but real-time operation could not be achieved. Thus, among others, Delamarre and Faugeras [4] proposed to simplify fitting systems by using dense stereo map 3D data directly to reduce ambiguity. They derive 3D forces using the Iterative Closest Point algorithm to update the model. In contrast, Ueda et al. [29] reconstructed the entire hand volume using an octree representation. Still, real-time operation is difficult to achieve due to the expensive 3D reconstruction step.

A problem with the above methods is that temporal occlusions of entire objects cannot be handled well. Gumpp et al. [9] address this limitation by means of hypothesis generation using Particle Filters. To improve performance, several optimizations exist [17], however a whole computing cluster is required to achieve near real-time performance. Finally, Guðmundsson et al. [10] increase the efficiency by employing real-time range data from a time-of-flight camera while reducing the dimensionality of the joint angle space for their particle filter using Principle Component Analysis. However, their system is limited to a few hand poses only.

2.4 Gesture Recognition

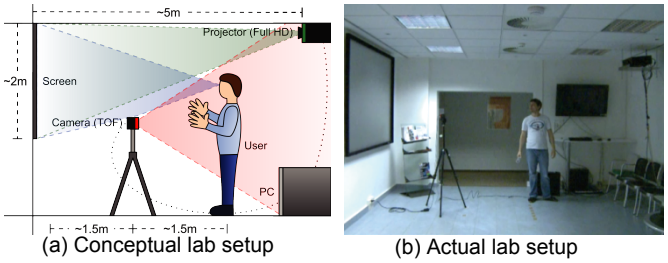
In the simplest case, gestures are derived from static hand appearances only. e.g. Wang et al. [31] use Fourier Descriptors for deriving a rotation, translation and scale-invariant shape descriptor, while Liu and Jia [18] use the Maximum Posterior estimate (MAP) to derive the most probable hand gesture from samples of a particle filter set.

To account for spatio-temporal variety in hand gestures, hidden Markov Models (HMM) haven been proposed. For example, Keskin et al. [11] use position-relative velocity vectors, restricting the variability to palm motion only. In contrast, El-Sawah et al. [6] fed a sequence of posture symbols into the HMM, with posture symbols derived from the hand appearances using Support Vector Machines.

As our work does not focus primarily on gesture recognition discussion of further techniques has been omitted.

3 Implementation

As the related work shows, the choice of the algorithms chiefly depends on the environment in which the system is required to operate. A larger setup of the system in an open environment, such as a shopping mall or outdoors [24] requires robust hardware that can cope with many users and difficult lighting conditions such as sunset. The choice of hardware to capture the users is a defining element of the reliability of the final system.



(a) Conceptual lab setup (b) Actual lab setup

Figure 1: Setup of the lab as used for usability testing

As a consequence, a time-of-flight (ToF) camera was selected, capturing depth maps directly by using an active infrared signal emission and sensing scheme [15]. We selected the SwissRanger SR4000 as it provides a resolution of 176 by 144 pixels at a frame rate of 55 Hz, which is sufficient for the algorithms to support at least two users in the detection range of the camera, creating a working volume of roughly 8 cubic meters. Alternatively, GPU-based stereo matching algorithms could be employed, which would provide similar results, but at computationally much higher costs for real-time performance.

Using the ToF camera, a commodity desktop workstation with an Intel Core 2 Duo 2.6 GHz and 3 GB RAM is more than sufficient to support our tracking system and application for full real-time operation. The final deployment of the system components is illustrated in Figure 1a. The geometric setting supports immersive environments with 3D stereo rendering where the user is interacting with content on a large screen. The ToF camera is mounted between the screen and the user, and captures the hands from the front. A projector with full high-definition resolution is mounted behind the user under the ceiling. Both

camera and projector are connected to a workstation that carries out the processing.

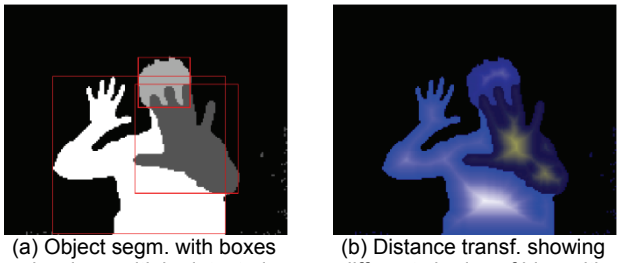
While our system is capable of operating in diverse indoor and outdoor environments with arbitrary lighting conditions, to accommodate an initial user study, a restricted laboratory setting was selected (Figure 1b).

3.1 Vision Algorithm

Our Fast Light-Independent Gesture and Hand Tracking (FLIGHT) vision algorithm, which has been developed for the 3D multi-touch prototype is divided into three stages of low- to high-level vision tasks: Data preparation, feature extraction, and hand tracking.

3.1.1 Data Preparation

The *data preparation* step aims at pre-processing the raw image data to enable robust feature extraction in the next step.

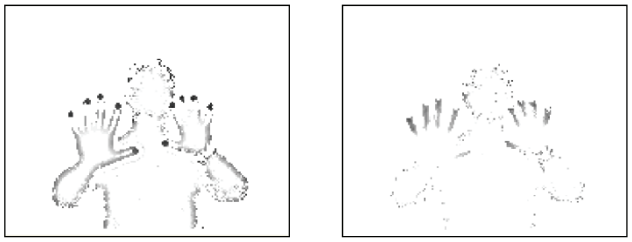


(a) Object segm. with boxes showing multiple detected objects with depth in gray

(b) Distance transf. showing different shades of blue with multiple distances detected

Figure 2: Object segmentation and distance transformation

In the first step, the depth map (a 16 bit unsigned integer array) is filtered to suppress noise effects originating from the sensing process of the ToF camera. A non-linear median filter with mask size 3×3 was chosen due to its ability to preserve object boundaries. On depth maps, object boundaries correspond to depth discontinuities providing, valuable information for later steps in the vision algorithm.



(a) Tip feature map with dots showing detected features

(b) Pipe feature map with areas showing detected features

Figure 3: Result of FAST feature mapping

In the next step, the image is segmented into a set of coherent objects. The objects define the regions of interest for subsequent processing steps. Segmentation is realized by determining connected components on the depth map using the 4-connected flood fill algorithm with constant deviation threshold $d = 1000$, corresponding to a depth resolution of approximately $15cm$. The effect of the segmentation process is illustrated in Figure 2a. Red boxes demark the bounding areas of the regions, while shades of grey visualize the masks of the individual components.

In the next step, a feature detector inspired by the FAST detector [26] is used to classify each remaining pixel $\boldsymbol{P} = (\boldsymbol{x}, \boldsymbol{y})$ according to its tip- or pipe-likeness. The algorithm uses a Bresenham circle of radius 4 to segment the pixel's periphery into arcs $\boldsymbol{A} = (\boldsymbol{\alpha}, \boldsymbol{\beta})$ of consistently lower, equal or higher depth compared to the center pixel. For comparison again the maximum deviation threshold $\boldsymbol{d}$ is used. Finally, the arcs are assigned to the respective sets $\boldsymbol{\mathcal{A}_P^-}$, $\boldsymbol{\mathcal{A}_P^=}$, $\boldsymbol{\mathcal{A}_P^+}$ to build the tip and pipe feature maps $\boldsymbol{\mathcal{M}_T}$ and $\boldsymbol{\mathcal{M}_P}$ with

$$\mathcal{M}_T(x,y) = \begin{cases} |A_\alpha - A_\beta| & if \quad \left|\mathcal{A}^=_{(x,y)}\right| = 1 \\ 2\pi & else \end{cases} \tag{1}$$

and

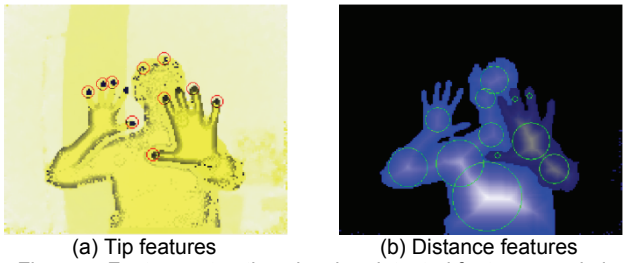
$$\mathcal{M}_P(x,y) = \begin{cases} \sum_{A \in \mathcal{A}^=_{(x,y)}} |A_\alpha - A_\beta| & if \quad \left|\mathcal{A}^=_{(x,y)}\right| = 2 \\ 2\pi & else \end{cases} \tag{2}$$

The result of the feature mapping procedure is illustrated in Figure 3. Pixels belonging to both tips and pipes consistently exhibit low intensity values in the respective representations, and hence can be found as local minima.

In the final step, during preparation a depth-discontinuity enhanced distance transform map $\mathcal{M}_D$ is calculated on top of the segmented objects. Connectivity between pixels for distance propagation is defined analogous to the flood fill algorithm for region segmentation. As a consequence, depth discontinuities are considered obstacles with the goal to incorporate intra-object boundaries for hands and arms. As a result, palms are shown as local maxima even if they considerably overlap the body region, provided that they are held far enough apart. Sample output of the method is given in Figure 4b. The blue channel demarks the connected components, a.k.a. objects, which for example cause the user's left arm in Figure 4b to appear in a darker shade of blue than the head or the body. In contrast, the red and green channels provide the distance transform results, which cause a transition to white from yellow, depending on the strength of the blue component.

3.1.2 Feature Extraction

The *feature extraction* step aims at reducing the pixel-based image data to a small set of features, from low-level tips and pipes to high-level compound constellations.



(a) Tip features

(b) Distance features

Figure 4: Feature extraction showing detected features as circles

In the first step, the sets $\mathcal{F}_T, \mathcal{F}_D$ of local turning points in the tip feature map $\mathcal{M}_T$ and the distance transform map $\mathcal{M}_D$ are extracted. To improve their localization, both images are convolved with a Gaussian filter mask first. For the distance transform map a mask of size 11×11 was chosen to suppress multiple maxima in a small region, for the tip feature map the size 3×3 proved to be sufficient due to the overall smaller feature appearance. Finally, local turning points are selected using a non-maximum suppression in a 3×3 window. The result of the procedure is illustrated in Figure 4. Tip features are annotated using red circles with constant radius 4, while distance features are annotated using cyan circles with radius equal to the depth discontinuity distance value instead.

In the next feature extraction step, an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is built by connecting those distance features whose radius is larger than a minimum threshold of 5 pixels. The threshold was

selected empirically based on the average distance of the user to the camera, with the goal to exclude features that are too small to represent a palm (e.g. the circles on the fingertips in Figure 4b). To derive the edges of the graph, a distance transform is calculated, which finds for each region pixel the closest distance feature from the set $\mathcal{V}$. Thereby, connectivity between pixels is defined equivalent to the previous distance transform.

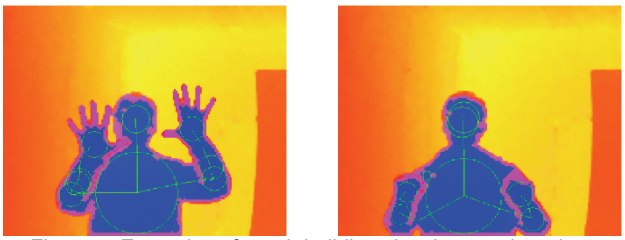


Figure 5: Examples of graph building showing graph nodes as circles and connections as lines

However, additionally a minimum distance of 2 for depth discontinuities is required to prevent the algorithm from crossing depth boundaries with pixels of depth value in between foreground and background. Finally, unique edges are introduced whenever two neighbouring pixels point to different closest distance features. The result of the procedure is illustrated in Figure 5. The graph nodes are annotated using circles, while edges are annotated using lines. Furthermore, the red channel contains the distance transform, the green channel the median-filtered depth map and the blue channel the connected object components.

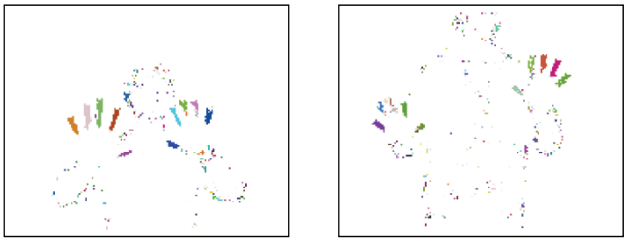


Figure 6: Examples of pipe segmentation with connected segments in random colors

In the next step, connected components are segmented from the pipe feature map $\mathcal{M}_P$. Therefore, the connectivity between pixels is defined along the four cardinal directions if the feature map values of both neighbouring locations are below the threshold $\theta = 0.7 \cdot 2\pi$, i.e. only pixels with exactly two arcs of equal depth are considered, which cover a maximum of 70% of the Bresenham circle. The result of the segmentation process is illustrated in Figure 6. Connected components are visualized with random colors. Fingers are represented clearly using this method.

In the next step, the tip and pipe features are combined to finger structures $\mathcal{F}_F$. The process relies on the assumption that each tip coincides with an image location, which only has a single arc with equal depth compared to the center pixel (due to noise other constellations are possible). To find the connected pipe feature the arc's pixels are traversed and tested for the existence of an underlying pipe segment. If multiple matches are found along the arc, the largest segment is selected. Finally, if matching was successful, a finger object is built using the location of the tip and the point on the pipe segment, which has the largest distance to the tip – also called base. The result of the procedure is shown in Figure 7. Both tip and base of the fingers are annotated using circles, which are connected by a line.

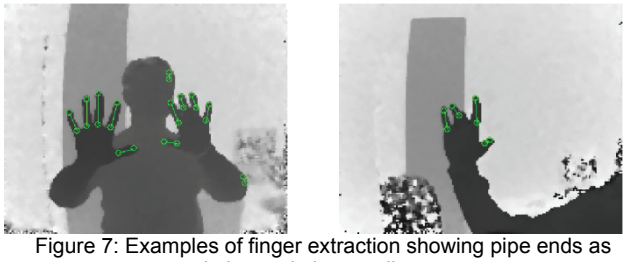
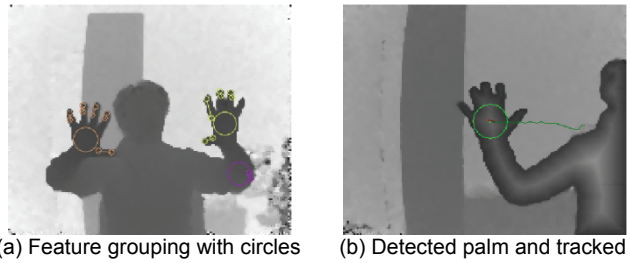


Figure 7: Examples of finger extraction showing pipe ends as circles and pipes as lines

In the final step during feature extraction, the set of palm candidates $\mathcal{F}_C$ and their connectivity to finger structures is determined. The palm candidates are defined as the topological endpoints of graph $\mathcal{G}$, i.e. those vertexes that have at maximum one connection. Further the set of candidates is limited to those distance features whose radius does not exceed a threshold of 13, which reflects the maximum palm size given the average distance of the user to the camera. This condition was added to filter the location of the head, which usually appears larger than the palm counterparts. For each remaining candidate the set of plausible finger matches $\mathcal{P}_C$ is derived using the palm center point as well as the finger base points. For successful matching, a maximum depth deviation of d between palm center and finger base is required, which reflects the constraint of equal/similar depth. Further, the distance between the two points in image space is required to lie below 20 pixels, which marks the empirical upper threshold for plausible associations given the average distance of the user to the camera. Likewise, for each finger structure the reverse mapping $\mathcal{P}_F$ is established, i.e. the set of plausible palm candidate matches.



(a) Feature grouping with circles marking detected features

(b) Detected palm and tracked movement path of palm

Figure 8: Examples of feature grouping and palm tracking

Based on the previous sets for each palm candidate C ∈ $\mathcal{F}_C$, a score s_C is calculated with

$$s_C = \sum_{F \in \mathcal{P}_C} \|C - F_B\| \tag{3}$$

Finally, using this score the optimal palm to finger assignment $\mathcal{S}_C$ is calculated as

$$\mathcal{S}_C = \left\{ F \in \mathcal{F}_F : C = argmin_{C' \in \mathcal{P}_F} \frac{s_{C'}}{|\mathcal{P}_{C'}|} \right\} \tag{4}$$

i.e. each finger is allocated to the plausible palm candidate with minimal average distance to its plausible finger connections. The result of the procedure is illustrated in Figure 8a. Both distance features and finger structures of the candidates are visualized using the previous annotation conventions.

3.1.3 Hand Tracking

The *hand tracking* step aims at identifying the true hand structures, and following their appearance through a sequence of frames.

In the first step, the set of palms $\mathcal{F}_{P,n}$ in frame n is determined from the set of palms $\mathcal{F}_{P,n-1}$ using feature-based optical flow [20]

on the squared distance transform maps for pronouncing the individual feature locations. In cases where a palm was already tracked successfully in the previous frame, the expected location in the current frame is predicted by assuming constant linear motion in the image plane to account for time-related displacements. The window size for the optical flow algorithm is set to the maximum palm radius, which was encountered in the previous frame. Otherwise the convergence performance drops due to unreliable matching of window contents. After retrieving the converged location for each palm of the previous frame, the respective closest palm candidate of the current frame is selected, and an association is established if the distance between candidate and converged location does not exceed the maximum admissible palm radius of 13 pixels. In addition to the successfully tracked palms, those palm candidates are added to the set of palms, which are connected to exactly 5 finger structures. The last step defines the strategy for hand detection that is employed in the system. The result of the procedure is shown in Figure 8b. The previously detected palm is annotated using a green circle; the trajectory is annotated using a green line. The current linear predication is annotated using a thin red line.

In the final step, for each palm of the previous frame, the temporal association of the associated fingers is determined. Similar to palm tracking, the algorithm uses feature-based optical flow on the tip feature map $\mathcal{M}_T$ in order to track the location of the fingertips. The size of the search window is set to 7 x 7, which reflects the radius of the Bresenham circle. For prediction, again linear motion is assumed in the image plane. However, the apparent displacement of the palm is used instead of a finger-specific velocity component. After retrieving the converged location for each finger of the previous frame, the closest finger structure in the current frame is selected and a connection is established if the distance between tip and converged location does not exceed the radius of the Bresenham circle.

3.2 Interaction Design

As part of our iterative development approach of the algorithm, we developed a demo application with basic 3D interaction. The interaction was designed to demonstrate interaction metaphors of existing multi-touch systems, such as selection, translation, rotation and scaling, in a three-dimensional context. To utilize familiarity effects with systems such as CityWall [24], the metaphors are applied to the problem of spatial picture arrangement. The formulation of the gestures is based on a 3D model of the picture frames $O = (C, X, Y, Z, W, H)$, where C represents the center point and X, Y, Z denote the orthonormal basis vectors for frame rotation. Finally, W, H represent the width and height of the frame in space units. In the following, the implementation of the gestures is described.

3.2.1 Create frame

Usually, a session starts out with an empty screen space. The first action a user can perform is pull up new pictures from a picture database. The design of the corresponding gesture was inspired by the idea of popping up elements out of the empty space. This design naturally translates into a closed human hand, which suddenly extends all five fingers. Technically, the gesture is recognized for palm P_n in frame n if

$$\frac{1}{3}\sum_{i=n}^{n-2}\left|\mathcal{S}_{P_i}\right| \geq 3 \wedge \frac{1}{9}\sum_{i=n-3}^{n-12}\left|\mathcal{S}_{P_i}\right| \leq 1 \qquad (5)$$

i.e. the average of the number of detected fingers is constrained by the respective thresholds. This step improves reliability of finger detections due to noise and appearance problems in the captured image.

3.2.2 Select frame

Before the user can start manipulating frames, the desired target has to be selected. Borrowing from existing systems, the respective gesture is based on the simple touch event. Touching pictures in 3D geometrically amounts to the hand intersecting the picture plane within the width and height boundaries. A pilot user study, however, revealed that this definition is too restrictive, as users frequently missed the target. As a consequence, the touch event was extended to a box volume, which is aligned with the frame coordinate system. Technically, the select gesture is recognized for palm P_n and frame O if

$$R \cdot O_X \leq \frac{O_W}{2} + 5cm \wedge R \cdot O_Y \leq \frac{O_H}{2} + 5cm \wedge R \cdot O_Z \leq 1dm \qquad (6)$$

with

$$R = [\mathcal{C}_n(P_n) - O_C] \qquad (7)$$

The mapping $\mathcal{C}_n$ is used to retrieve the space coordinates for image location P_n, i.e. the palm center.

3.2.3 Move frame

After selecting a frame with one hand, the user has the option to move it along all three coordinate axes. The respective gesture is inspired by the idea of grabbing small objects, which fit in the user's hand, such as drinking glasses, before changing their position. Technically, the grab event is recognized for palm P_n if

$$\bigwedge_{D\in\{X,Y,Z\}}\left(\sum_{i=n}^{n-4}(\mathcal{C}_n(P_n)_D - \mu_D)^2 \leq 3cm^2\right) \wedge \left|\mathcal{S}_{P_n}\right| = 0 \qquad (8)$$

with

$$\mu = \frac{1}{5}\sum_{i=n}^{n-4}\mathcal{C}_n(P_n) \qquad (9)$$

i.e. the hand position is required to be stable within a small window by means of the covariance constraint, and no fingers should be detected in the last frame. The stability constraint helps to prevent moving hands from accidentally grabbing frames due to unreliable finger detection in case of blurred motions resulting from 3D camera exposure time requirements. To release the object, a fully opened hand is required, which is defined analogously to the *create* action.

3.2.4 Rotate/scale frame

Both frame scaling and rotation around the Y and Z axes is supported as well. In accordance with other multi-touch systems, the interaction metaphor is based on two touch points, which operate on the object simultaneously. Following the previous conventions, the action is triggered if two hands are grabbing the same frame. Technically, grabbing is again defined as a covariance and zero finger constraint. Also, the release is supported on both hands to either switch back to simple movement or entirely detach from the object.

3.2.5 Delete frame

Deletion of frames is supported to clear the screen and make space for new objects. The respective gesture is inspired by the metaphor of throwing dispensable objects into the trash. However, as currently fast motion cannot be supported by the optical hand tracking due to 3D camera exposure time requirements, this action was simply translated to moving frames across a certain spatial border in either the X or Y direction. Technically, the delete event is triggered for hand P_n which is moving a frame O if

$$|\mathcal{C}_n(P_n)_X| \geq 35cm \vee |\mathcal{C}_n(P_n)_Y| \geq 30cm \qquad (10)$$

The coordinates and boundaries are expressed relative to the camera coordinate system.

4 EVALUATION

The methods that have been selected to evaluate the system were mainly oriented towards the two main design goals of the project: (1) The development of a lightweight real-time algorithm with possible adaption to mobile hardware in the near future, and (2) the exploitation of this novel non-intrusive 3D motion capture platform with respect to existing gesture-based 3D interaction techniques. To account for these points, first a number of computation statistics have been collected across a series of input sequences. Second, the usability of the system has been evaluated through a series of user studies.

4.1 Performance

To develop a picture of the computational complexity, the algorithm has been applied to four test sequences on an Intel Core 2 Duo with 2.6 GHz and 3GB RAM while recording average execution times over 10 iterations per frame (see Figure 9). Sequence 1 (12s, avg. of 0.84 hands/frame detected) contains one hand while the body of the user is hidden outside the camera view. Sequence 2 (18s, 1.23 hands/frame) captures a user that is showing and moving both hands at the same time. Sequence 3 (23s, 0.62 hands/frame) captures again one hand but including all upper body parts and larger motion distances. Sequence 4 (12s, 0.87 hands/frame) shows one-handed closed and open gesture transitions. All sequences include the user walking into and out of the view. The average hands/frame is to be seen as a measure of complexity for the sequences rather than accuracy (which is not inspected here). For performance analysis the average execution times are broken down into the portions that each individual algorithm step contributes from input filtering to finger tracking.

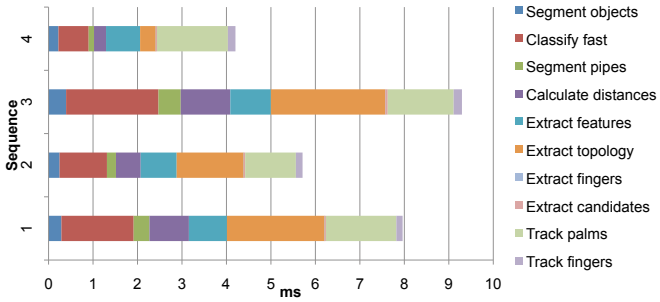


Figure 10: Average execution time per test sequence

When comparing the sequences 2 through 4, a strong similarity regarding the relative contributions of the individual algorithm steps can be recognized. In particular, across all three sequences fast classification uses 16-22% while topology extraction uses 25-28% and palm tracking 16-20%. In contrast, sequence 1 mainly stands out due to the relatively low percentage of the topology extraction step with 8%. Since topology extraction aims at describing the structure of the segmented regions this divergence is easily explained by the fact that only the arm of the user was visible while the remaining body parts were hidden.

Overall, it can be concluded that the relative execution times may vary significantly depending on a number of factors such as the diverse feature counts (segmented regions, distance maxima, tip map minima, pipe map segments, etc.). However, as proven by the four test sequences and many more test runs the absolute execution times seem to behave fairly well within the 10ms range, which is more than sufficient for real-time interaction.

4.2 Evaluation with Users

The purpose of the usability test was to evaluate the suitability of the non-intrusive hand and finger motion capture platform for standard gesture-based 3D interaction tasks. As noted by Nielsen [23], subjective satisfaction is an important part of acceptance and use of future technologies. To date no prior literature exists on user studies of ToF-based hand and finger tracking. As a result our study was primarily aimed at identifying whether such an approach was perceived as user friendly by users to the extent that they would consider using it in future systems. As part of this we were interested in testing for perceived accuracy of the system within its given tasks and gauging their subjective responses to its use in particular as to whether it was suitable for 2D and/or 3D tasks. It is important to indicate we were not at this stage focussing on absolute numerical accuracy of the system.

In common with earlier work on usability testing we adopted a quasi-iterative design perspective similar to that of Gould and Lewis [8] e.g. early involvement of users and tasks, empirical evaluation, and iterative design. This consisted of an early pilot study in which usability issues were identified and rectified. During the pilot phase the process of testing users was repeated and changes or redesigns were made until we were satisfied that problems had been rectified before the formal study was undertaken. Among the issues identified during the pilot/iterative phase were: (1) Inclusion of an explicit grab gesture, which was previously triggered by touch only. (2) Tuning the gesture recognition thresholds. And (3) extending the tracking algorithm to prevent frequent drift-off to the elbow. It should further be noted that the final study is intended to inform future design and development of such systems and is therefore also part of an iterative and on-going process.

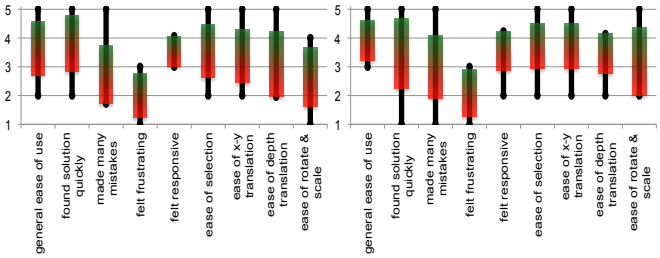
4.2.1 Method

Participants were recruited from among administrative staff, researchers and students. They were aged between 21 and 53, had vary degrees of experience with computer games, multi-touch devices, motion tracking systems, and augmented reality software. All participants were asked to complete three tasks, an introduction phase, followed by a 2D and 3D puzzle. The introduction phase was used in order to familiarize the users with the system, such as the boundaries of the interaction volume and the individual gestures. The two main tasks were administered randomly. The 2D puzzle task involved reassembling a photograph from various parts, requiring the use of selection, rotation and scaling. The 3D task involved placing objects in the correct depth within a scene. After completion of each task, the users were asked to complete a simple questionnaire, which contained 5-point Likert-scale. On completion of all tasks, the subjects were asked for their overall opinions by providing written responses. In addition, the researcher took notes during each user test. In total 11 participants (7 male, 4 female) took part in the study, with a mean age of 29 (std. dev. 3.9).

4.2.2 Results

On average, the users rated the interaction scheme neutral to positive. A summary of the Likert-scale feedback is provided in Figures 11. In particular, the general concept of the application was well received. Concerning the puzzles, the participants took on average 165 seconds to complete the 2D task, with time taken ranging from 78 to 290 seconds. In contrast, they took less time to complete the 3D task, with times ranging from 56 to 269 seconds (mean 135). Across all three tasks they rated the interaction scheme neutral to positive, with scores typically being 3 or more for user actions such as selection and translation, with

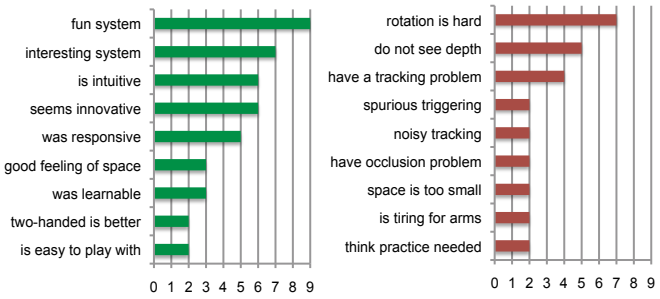
positioning in the depth plane and rotation (see Figure 13) being identified as the most difficult actions across all tasks.



(a) 2D puzzle (b) 3D puzzle

Figure 11: User rating on Likert-scale from 1 (disagree) to 5 (agree)

The written user responses were grouped and are summarised in figure 12. The results point to the system being well received and this is indicated by comments such as: *"Cool application, it's really fun!"* or *"Original experience, it's fun!"* The second most common theme (63% of users indicating this) was interesting which could be found in comments such as *"Interesting experience, which shows, that there is also a world after the touch-interfaces."* Third to fifth place in terms of positive comments was that the system was innovative (55%), intuitive (55%) and responsive (36%). The issue of haptics was also noted but in a positive way with one user noting: *"I expected that the absence of haptics is disturbing, but it was not like that!"*



(a) Positive user comments (a) Negative user comments

Figure 12: Summary count of comments made during tests

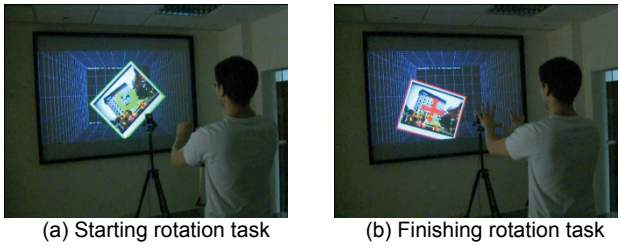
Although the system was in general well received, problems with rotation tracking were noted in 63% of tests, while general tracking problems were identified in 36% of tests. Typical comments were: *"The tracking errors are a little disturbing, but do not take the appetite to interact."* or *"The experiment was completely positive regardless of the issues with tracking."* Problems with depth perception were identified in 45% of tests. This was, however, not related to the system but rested with the display, which did not have a stereoscopic 3D setting. However had a 3D stereoscopic display been used this would have added an additional variable to the test.

5 Discussion

The performance data as well as the positive user feedback indicates that the system provides an effective environment for implementing and studying novel barehanded 3D interaction techniques. In this context the overall approach benefits notably from the simple and lightweight setup, allowing for individual usage in most environments within a couple of minutes. Also the technical requirements allow such a system to be operated on most currently available PCs.

Through a user study the FLIGHT algorithm proved to be capable of robustly detecting hands and reliably estimating the 3D poses of the palms, fingers, and fingertips for a number of users, which can be seen in the many positive comments and impressions. Moreover, the calculated hand information seems to be stable enough for supporting real-time 3D gesture input and completing a range of interactive 3D tasks. This claim is supported by the fact that every participant was able to complete the given tasks while the immediate feedback was neutral to positive for the individual interaction styles (selection, translation, rotation, scaling) as depicted in Figure 11. In particular the graph formulation $\mathcal{G}$ for connecting distance features has been recognized as significantly improving the palm selection procedure through introducing an endpoint constraint that incorporates information about feature constellations. This contribution has been noted after introducing the graph concept in response to major tracking problems during a pilot study.

Nevertheless, the user study also revealed issues with the current system, most importantly the robustness of the tracking component as indicated by the comment counts in Figure 12(b). More precisely, one has to distinguish between rotation tracking and general tracking errors. The first seem to have a fairly simple explanation: many subjects tended to cross their hands for rotating objects 180 degrees around the Z axis (see Figure 13), which caused either of the hands to be occluded from the view of the camera. As our algorithm does not support this case it is not surprising that problems were encountered.



(a) Starting rotation task (b) Finishing rotation task

Figure 13: Example of a user rotating an object

The remaining tracking issues, however, suggest that the optical flow based tracking concept has general limitations, e.g. under higher motion in the image plane. To overcome these problems more advanced tracking methods such as the more robust shape-based EigenTracker or the occlusion-sensitive Particle Filter approach may have to be adopted.

Another known issue with the system is the long exposure time of the ToF camera limiting the maximum speed of motion before introducing a significant blur. Further, the rather low camera resolution requires the fingers to be clearly separated. These two effects become even more critical in currently available cheap ToF cameras. While standard stereo cameras may currently provide an alternative solution – in particular since they are cheap, robust, and provide a better resolution – it can also be expected that, based on recent developments regarding ToF cameras, appropriate modules will become significantly cheaper, providing increased frame rates, and at higher resolutions. In summary, the user study found that while there is room for improvement that the users liked the basic concept of bare-handed 3D interaction and that it is perceived as providing sufficient accuracy thus allowing them to complete desired tasks. However, it is acknowledged that improvements can be made and that future technologies will improve the system. In particular, a more powerful pose estimation method coupled with a more robust and accurate output would greatly enhance the degree of expressiveness that could be exploited for gesture definition.

6 Conclusion

In this paper we presented our approach to lightweight palm and finger tracking, by providing a basis for 3D gesture-based interaction. Our approach allows for barehanded 6-DOF 3D multi-touch interaction is based on a fast light-independent gesture and hand tracking algorithm (FLIGHT). It first preprocesses the depth information received from a time-of-flight (ToF) camera, classifying pixels for their tip- or pipe-likeness. It then extracts higher-level features identifying palm and finger candidates. Finally, the true hand structure is identified applying an optical flow approach to the palm and the fingertips.

We use this as a basis to support a set of basic 3D multi-touch interaction techniques, naturally extending common 2D multi-touch input gestures into the 3rd dimension. The overall performance of our approach allows for 50fps, mainly restricted due to the camera refresh cycle as the overall computation time usually is in the 10ms range.

As the result of our qualitative user study, our approach proved to be suitable for 3D interaction and intuitive usage by inexperienced users. Restrictions still apply regarding fast movements, depth and detail resolution, and robustness regarding the connection of features in the graph.

In our future work, we will explore possibilities to increase the robustness and the working volume e.g. by using additional time-of-flight cameras. Working with different cameras we will further develop mathematical formulations for current "magic values" tailored to the single ToF camera available for early prototyping. This work will further enable us to conduct more elaborate evaluations of the algorithm performance. We will further examine the applicability in stereoscopic VR and AR scenarios as part of another user test to gather quantitative data. Finally, we will investigate particular gestures and possible application areas requiring the usage of individual fingers rather than the complete hand, as finger pipe and tip tracking is a unique feature of the presented approach.

Acknowledgements

We thank Thorsten Froehlich for his contributions to this paper.

References

[1] K. Abe, H. Saito and S. Ozawa. "3-D drawing system via hand motion recognition from two cameras," in *Proc. of IEEE SMC,* 2000, pp. 840-845 vol. 2.

[2] A. Argyros and M. Lourakis. "Tracking multiple colored blobs with a moving camera," in *Proc. of IEEE CVPR*, 2005, 1178 vol. 2.

[3] K. Barhate et al. "Robust shape based two hand tracker," in *Proc. of IEEE ICIP*, 2004, pp. 1017-1020.

[4] Q. Delamarre and O. Faugeras. "Finding pose of hand in video images: a stereo-based approach," in *Proc. of IEEE FG*, 1998, pp. 585-590.

[5] M. Donoser and H. Bischof. "Real time appearance based hand tracking." in *Proc. of ICPR*, 2008, pp. 1-4.

[6] A. El-Sawah, C. Joslin, N. Georganas and E. Petriu. "A Framework for 3D Hand Tracking and Gesture Recognition using Elements of Genetic Programming," in *Proc. of IEEE CRV,* 2007, pp. 495-502.

[7] S. Ghobadi et al. "Real Time Hand Based Robot Control Using 2D/3D Images," in *Proc. of ISVC*, 2008, pp. 307-316.

[8] J. Gould and C. Lewis. "Designing for usability: key principles and what designers think." *CACM*, vol. 28, no. 3, pp. 300-311, 1985.

[9] T. Gumpp, P. Azad, K. Welke, E. Oztop, R. Dillmann and G. Cheng. "Unconstrained Real-time Markerless Hand Tracking for Humanoid Interaction," in *Proc. of IEEE Humanoids*, 2006, pp. 88-93.

[10] S. Guomundsson et al. "Model-Based Hand Gesture Tracking in ToF Image Sequences," in *Proc. of AMDO*, 2010, pp. 118-127.

[11] C. Keskin, A. Erkan and L. Akarun. "Real time hand tracking and 3d gesture recognition for interactive interfaces using hmm," in *Proc. of Int. Conf. on Artificial Neural Networks*, 2003.

[12] H. Kim, K. Kwak, S. Chi and Y. Cho. "AR-KLT based Hand Tracking," in *Proc. of IEEE RO-MAN*, 2006, pp. 611-616.

[13] M. Kolsch and M. Turk. "Robust hand detection," in *Proc. of IEEE FG*, 2004, pp. 614-619.

[14] M. Kolsch and M. Turk. "Hand tracking with Flocks of Features," in *Proc. of IEEE CVPR*, 2005, p. 1187 vol. 2.

[15] R. Lange and P. Seitz. "Solid-state time-of-flight range camera." *IEEE J. of Quantum Electronics*, vol. 37, no. 3, pp. 390-397, 2001.

[16] J. Letessier and F. Bérard. "Visual tracking of bare fingers for interactive surfaces," in *Proc. of ACM UIST*, 2004, pp. 119-122.

[17] W. Liang, Y. Jia, F. Sun, B. Ning, T. Liu and X. Wu. "Visual Hand Tracking Using MDSA Method, "in *Proc. of IMACS Multiconf. on Computational Eng. in Syst. Applications*, 2006, pp. 255-259.

[18] Y. Liu and Y. Jia. "A Robust Hand Tracking and Gesture Recognition Method for Wearable Visual Interfaces and Its Applications," in *Proc. of IEEE ICIG*, 2004, pp. 472-475.

[19] S. Lu, D. Metaxas, D. Samaras and J. Oliensis. "Using multiple cues for hand tracking and model refinement in *Proc. of IEEE CVPR*, 2003, pp. 443-450 vol. 2.

[20] B. Lucas and T. Kanade. "An iterative image registration technique with an application to stereo vision," in *Proc. of IJCAI*, 1981, pp. 674-679.

[21] J. Matas, O. Chum, M. Urban and T. Pajdla. "Robust wide baseline stereo from maximally stable external regions, "in *Proc. of British Machine Vision Conference*, 2002, pp. 384-393.

[22] Z. Mo, J. Lewis and U. Neumann. "SmartCanvas: a gesture-driven intelligent drawing desk system," in *Proc. of ACM IUI*, 2005, pp. 239-243.

[23] J. Nielsen. "The usability engineering life cycle." *IEEE Computer*, vol. 25, no. 3, pp. 12-22, 1992.

[24] P. Peltonen et al. "Extending large-scale event participation with user-created mobile media on a public display," in *Proc. of ACM MUM*. 2007, pp. 131-138.

[25] R. Rosales, V. Athitsos, L. Sigal and S. Sclaroff. "3D hand pose reconstruction using specialized mappings," in *Proc. of IEEE ICCV*, 2001, pp. 378-385 vol. 1.

[26] E. Rosten, R. Porter and T. Drummond. "Faster and Better: A Machine Learning Approach to Corner Detection." *IEEE Trans. Pattern Anal. Mach. Intelligence*, vol. 32, no. 1, pp. 105-119, 2010.

[27] Y. Sato, M. Saito and H. Koike. "Real-time input of 3D pose and gestures of a user's hand and its applications for HCI, " in *Proc. of IEEE VR,* 2001, pp. 79-86.

[28] J. Segen and S. Kumar. "Gesture VR: vision-based 3D hand interface for spatial interaction," in *Proc. of ACM MULTIMEDIA*, 1998, pp. 455-464.

[29] E. Ueda, Y. Matsumoto, M. Imai and T. Ogasawara, "A hand-pose estimation for vision-based human interfaces." *IEEE Trans. Ind. Electronics*, vol. 50, no. 4, pp. 676-684, 2003.

[30] J. Wall. "Demo I Microsoft Surface and the Single View Platform," in *Proc. of Int. Symp. on Collaborative Technologies and Systems,* 2009, pp. xxxi-xxxii.

[31] X. Wang, X. Zhang and G. Dai. "Tracking of deformable human hand in real time as continuous input for gesture-based interaction," in *Proc. of ACM IUI,* 2007, pp. 235-242.