# Human-Computer Interaction through Time-of-Flight and RGB Cameras

Piercarlo Dondi, Luca Lombardi, and Marco Porta

Department of Computer Engineering and Systems Science, University of Pavia,
Via Ferrata 1, 27100 Pavia, Italy
{piercarlo.dondi,luca.lombardi,marco.porta}@unipv.it

**Abstract.** The number of systems exploiting Time-of-Flight (ToF) cameras for gesture recognition has greatly increased in the last years, confirming a very positive trend of this technology within the field of Human-Computer Interaction. In this work we present a new kind of application for the interaction with a virtual keyboard which is based on the use of an ordinary RGB webcam and a ToF camera. Our approach can be subdivided into two steps: firstly a segmentation of the entire body of the user is achieved exploiting only the ToF data; then the extraction of hands and head is obtained applying color information on the retrieved clusters. The final tracking step, based on the Kalman filter, is able to recognize the chosen hand also in presence of a second hand or the head. Tests, carried out with users of different ages, showed interesting results and a quick learning curve.

**Keywords:** Time-of-Flight camera, human-computer interaction, hand recognition.

## 1 Introduction

Time-of-Flight (ToF) cameras are able to measure depth in real-time using a single compact sensor, unlike previous multi-camera systems, such as stereo cams. In the last years, research has shown a large interest in such devices in many fields related to computer vision and computer graphics, like 3D modeling, scene reconstruction, user interaction or segmentation and tracking of moving people [1]. In some cases, the ToF contribution is combined with color informations supplied by a traditional RGB camera to achieve more complex or precise results: for instance, in [2] depth and color data are used to create a 3D ambient for mixed reality; in [3] depth information is exploited to select the best input area for a color-based segmentation algorithm (SIOX); while in [4] a fusion of colors and depth data is employed in a new segmentation and tracking method for compensating the respective weaknesses of the two different kinds of sensors.

In this paper we focus on the combined use of a RGB and a ToF camera for Human-Computer Interaction (HCI), presenting a new application which allows the user to control virtual on-screen keyboards (in particular a QWERTY keyboard and a numeric pad). The ToF stream is used for the initial search of the

G. Maino and G.L. Foresti (Eds.): ICIAP 2011, Part II, LNCS 6979, pp. 89–98, 2011.

entire body of the user, using an approach described in a previous work [5]; this solution achieves a real-time foreground segmentation which is robust to sunlight noise. Color information is then applied to the retrieved clusters to extract head and hands. The subsequent tracking step, implemented using a Kalman filter, is able to follow the hand of interest also when the other hand is detected or in presence of overlaps with the head. Experimental tests, carried out with users of both sexes aged between 20 and 50, showed interesting results: in particular a quick learning curve and a fast decrease of errors.

The paper is organized as follows: section 2 provides an overview of the main characteristics of ToF cameras; section 3 presents the state of the art related to HCI applications based on these devices; section 4 describes the proposed method; section 5 shows the experimental results; section 6, at last, draws some conclusions.

## 2 Time-of-Flight Cameras

Cameras based on the Time-of-Flight principle work in the near infrared band exploiting laser light to assess distances of elements in the scene, thus providing depth information. Practically, two major approaches exist to implement these devices, namely pulsed and modulated light. In the first case the target is hit by a coherent wavefront and the depth is measured analyzing the variations of the reflected wave; in the second case an incoherent modulated light is used and the time of flight is determined by means of phase delay detection.

Compared to stereo cameras or laser scanners, ToF cameras are characterized by some benefits: they do not employ moving mechanical components, work reliably in real-time, are not affected by shadows and can calculate 3D distances within any scenario. A drawback of these systems is however their sensitivity to sun light, which introduces considerable noise (on the contrary, in general, artificial illumination does not interact with the sensor). Although a ToF camera has a nominal working range of about 10 m, noise caused by scattering, multi-paths and environment light may decrease this value, and the actual range is therefore between two and five meters [6].

To implement the system described in this paper we exploited the SR3000 ToF camera by MESA Imaging [7], a modulated-light device which we used at 20MHz and whose active sources work in the near infrared, at about 850nm. Its frame rate is about 18-20 fps. The camera produces two maps per frame − each with a resolution of 176x144 pixels − one containing distance information and the other reporting the intensity of light reflected by objects in the scene. Since the sensor is not affected by visible light, values of intensity depend on energy in the near infrared range only. For this reason, nearer objects look clearer, since they reflect more light, while more distant elements appear darker.

## 3 Previous Works

The number of systems exploiting the ToF approach for gesture recognition has greatly increased in the last years, confirming a very positive trend of this

technology within the field of Human-Computer Interaction. In almost all implementations, the first step is the removal of background objects by means of a threshold applied to the assessed distance between hand/arm and the camera. Then, the next stages depend on the specific task.

The system described in [8], for example, carries out gesture recognition by fusing 2D and 3D images. The segmentation technique employed is based on the combination of two unsupervised clustering approaches,K-Means and Expectation Maximization, which both try to locate the centers of natural clusters in the combined data. Another gesture recognition system based on hand movement detection is presented in [9] After rejecting elements falling outside a predefined depth range, the Principle Component Analysis (PCA) technique is exploited to get a first basic estimation of the position and orientation of the hand. Afterward, a more complex (3D skeleton-based) hand model is matched to the previously acquired data. In [10], 12 static gestures are classified according to X and Y projections of the image and depth information. The projections of the hand are used as features for the classification, while the arm area is removed. Depth features are taken into account to distinguish gestures which have the same projections but different alignments. The technique described in [11] combines a pre-trained skin color model (a Gaussian mixture approach) with a histogram-based adaptive model which is updated dynamically with color information extracted from the face. The system has been tested with sample images containing six different hand postures and can identify gestures and movements of both hands.

Slideshow control is a typical application of gesture recognition. As an example in [12], the "thumbs-up" gestures towards left or right are used to switch to the previous or next slide, while pointing to the screen is interpreted as a "virtual laser pointer". The pointing direction is calculated in 3D, at first through a segmentation of the person in front of the camera with respect to the background, and then by detecting the 3D coordinates of head and hand. An analogous use for the interaction with a beamer projector is described in [7].

A further context for effective gesture recognition is represented by medical applications, where it is sometimes necessary a touchless interaction: an example is proposed in [13], where a system based on ToF camera allows the exploration and navigation through 3-D medical image data.

Usage scenarios for the ToF approach are however not limited to those quoted above, but can be extended to several settings. A very comprehensive survey of developments of ToF technology and related applications can be found in [1].

## 4 Feature Detection and Interaction

### 4.1 Foreground Segmentation

Our segmentation algorithm [5] is designed so as not to need any preprocessing operations or a priori knowledge of the environment or of the shapes of objects. This section summarizes its main steps comprehensive of noise compensations; section 4.2 describes the proposed color based extension for hand recognition.

Two steps make up our approach: a first thresholding of the distance map based on the corresponding values in the intensity image, followed by a region growing stage that starts from seeds planted on peaks of the intensity map. Considering the characteristic of the ToF camera described in section 2, we use intensity map as a guide to restrict the area of investigation in the range map and to find good candidates to become seeds.

For every frame we dinamically estimate a proper intensity threshold ($\lambda_{seed}$) applying the Otsu's method. This parameter is used to define the set of seeds $S$ (1).

$$S = \{P_x : I_x > \lambda_{seed}, \|P_x - P_s\| > \gamma, \gamma > 1\} \tag{1}$$

$P_x$ is a point of the distance map, $I_x$ is its corresponding intensity value and $P_s$ is the last seed found. The presence of a control of the distance between seeds guarantees their better distribution and reduces significantly their number in order to decrease the time needed for the following growing step.

The similarity measure $S$ between a cluster pixel $x$ and a neighboring pixel $y$ is defined in (2):

$$S(x, y) = |\mu_x - D_y| \tag{2}$$

$D_y$ is the distance value of pixel $y$ and $\mu_x$ is a local parameter related to the mean distance value around x (6). The lower $S$ is, the more similar the pixels are. When a seed is planted, $\mu_x$ is initialized to $D_x$. Considering a 4-connected neighborhood, a pixel $x$ belonging to a cluster $C$ absorbs a neighbor $y$ according to the following conditions:

$$\{x \in C, S(x, y) < \theta, I_y \in L, \theta > 512\} \rightarrow \{y \in C\} \tag{3}$$

where $L$ is the set of points of intensity generated using the equations (4) and (5) designed to threshold the data compensating the effects of noise caused by sunlight:

$$A = \{I_y : (I_y > \lambda) \vee [(I_y < \lambda) \wedge (I_{8n} > \delta * \lambda)], \delta \in [0, 1], \lambda < \lambda_{seed}\} \tag{4}$$

$$L = A \cup M \tag{5}$$

where $\lambda$ is an intensity threshold proportional to $\lambda_{seed}$, $I_{8n}$ is the intensity of all the neighbors of the pixel $y$ considering the 8-connection, and $M$ is the set A after the application of a series of morphological operations experimentally established (in order, two dilations, five erosions and a final dilation).

When a neighbor $y$ of seed $x$ is absorbed, we compute the average distance value $\mu_y$ in an incremental manner as follows:

$$\mu_y = \frac{\mu_x * \alpha + D_y}{\alpha + 1} \tag{6}$$

Parameter $\alpha$ is a learning factor of the local mean of $D$. If pixel $y$ has exactly $\alpha$ neighbors in the cluster, and if the mean of $D$ in this neighbor is exactly $\mu_x$, then $\mu_y$ becomes the mean of $D$ when $y$ is added to the cluster. Every region grows excluding the just analyzed pixels from successive steps. The process is

iterated for all seeds in order of descending intensity. Regions too small, with dimension lower than a fixed value, are discarded.

An experimental evaluation of the performances on different kinds of computers (both desktops and notebooks), made in a previous work [5], showed that the proposed approach ensures a good compromise between computational time and precision of the results: the system can reach the 44 fps with a high level computer and keeps the 18 fps of the ToF camera also with a low level one.

### 4.2 Hand Recognition

Hand detection is a complex task due to the high variability of hand shapes. An approach based only on color may be simpler but its performance is generally not good: objects in the background with color similar to skin produce inevitably false positives. This issue can be solved by limiting the region of interest: the described procedure retrieves the clusters in the foreground excluding automatically all the objects in the background. Moreover, considering the proposed interaction, in which the user must be relatively close to the camera to see what s/he is writing on the screen, we can further reduce the possible hand candidates, excluding a priori all the retrieved clusters placed too far from the camera (generally over 2 m).

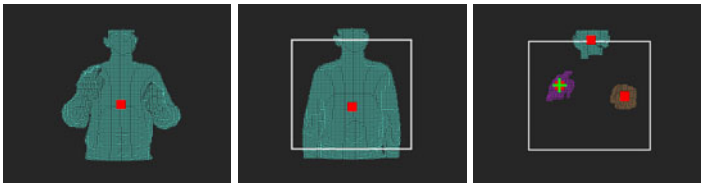
So, after these preliminary phases we obtain a cluster of a half-body user (Fig. 1(a)) from which hand detection can start using the color information supplied by a standard webcam (in our experiments a Logitech HD Pro Webcam C910 with a resolution of 640x480 pixels). The calibration is achieved using a method similar to that described in [14].

Firstly we convert the image from the RGB to the HSV color model; then we eliminate all the points of the cluster that are outside the set $W$:

$$W = \{y : 0° < H_y < 10°, 350° < H_y < 360°, S_y > TH_S, V_y > TH_V\} \qquad (7)$$

where $H_y$, $S_y$ and $V_y$ are, respectively, the hue, saturation and value of the pixel $y$. The first two constraints define the color area with a hue in the skin range; the threshold on the saturation eliminates all the white points; finally, the search for points with high values of lightness excludes clothes with skin-like colors. For our purpose we do not need a precise segmentation of the hand but only an approximation, since the pointer on the keyboard is not controlled by the shape of the hand but by the position of its centroid (section 4.3). This simplification gives two advantages: we can use very strict conditions on color thresholding for finding the hand (we can afford to lose some details in order to remove certainly wrong areas) and the user can position his or her hand in the way s/he finds more comfortable. Small inaccuracies, like holes, are in any case fixed applying a morphological dilation on the retrieved sub-clusters.

For a better performance we apply this sub-segmentation procedure not after the entire foreground segmentation (section 4.1) but at the end of the thresholding phase (equations (4) and (5)): this way we can execute a single region growing procedure on a reduced set of points.



(a) (b) (c)

**Fig. 1.** Results of the segmentation steps, visualized as cloud of points: (a) initial segmentation of the entire body; (b) search for the active area (white rectangle ); (c) hands and head extraction − the cross points the active hand

The last issue to solve is the choice of the active cluster and the exclusion of the others. We designed a training phase in which the user must stay in front of the camera and raise the hand that s/he has chosen to use. The system easily distinguishes between the hand (the cluster closer to the camera) and the head (the cluster in the upper position). For following the hand in the next iterations we use a tracking approach based on Kalman filter. The association between measured clusters and Kalman trackers is evaluated by minimum square Euclidean distance between the centroid of each cluster and the position predicted by each Kalman. This method, described in [5], is able to track multiple subjects, also in presence of short-term occlusions, and can thus be successfully applied in this situation. The use of the tracker allows the chosen hand to be recognized also in presence of other moving clusters like a second hand (Fig. 1(c)).

### 4.3 Interaction with Keyboard/Numeric Pad

We created for our application two kinds of keyboard with keys of different sizes: a reduced QWERTY keyboard (Fig. 2(a)) and a numeric pad (Fig. 2(b)). The interaction with the two keyboards occurs the same way in both cases: moving the hand moves the cursor pointer. Once the user has chosen the key s/he wants to press, s/he needs only to move the hand towards the camera, like if virtually pushing the key (the ToF camera can measure variations in distance of the tracked hand with no computational overhead). We set an appropriate distance threshold (experimentally determined) beyond which the key is considered pressed (section 5).

To estimate the position of the hand, we use its centroid, because it is the most stable point in the cluster: errors in depth evalutation caused by motion artifacts or by sunlight mainly affect points on the edges of the objects. The system performs a mapping of the position of the hand in the camera frames to the position of the cursor on the keyboard, but not all the area framed is considered valid. In fact, if the corners of the visual field of the camera corresponded to the corners of the keyboard, it would be very uncomfortable for the user to select and to press the keys placed in those positions. After specific tests, it was found that the more comfortable ″interaction zone″ for the user is a vertical

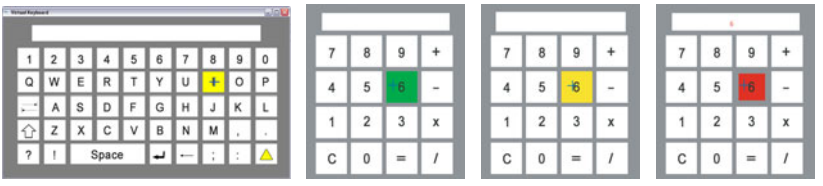



**Fig. 2.** The two kinds of keyboard used: (a) the reduced QWERTY (800x450 pixels on a 1680x1050 screen); (b)(c)(d) the numeric pad (550x600) with the three possible key states: (b) green - the pointer is on the key, but the key is not yet selected; (c) yellow - the key is selected, ready to be pressed; (d) red - the key is pressed

rectangle limited horizontally by the shoulders and vertically by the head and stomach (Fig. 1(b)). The selection of the active area for each user is automatically determinated during the training step.

Some visual feedbacks help the user to understand the state of the system. A key becomes green when the pointer is on it (Fig. 2(b)). A key becomes yellow (Fig. 2(c)), and is considered selected and ready to be pressed, when the pointer remains on it for a short period of time (around 1 sec). This status is useful to avoid accidental presses. Finally a key becomes red and gets smaller when it is pressed and the corresponding character appears in the box on top of the keyboard (Fig. 2(d)).

## 5 Experimental Results

A set of experiments were carried out to test the system and to obtain learning curve of the input method. 16 users participated in the tests (8 males and 8 females) aged between 20 and 50 (mean 28). All of them were placed in front of the cameras at a distance so that they were framed at half-body. Before starting the tests, each user was briefly trained in writing with the two keyboards (for about 2 minutes).

In the first test we asked the user to write the word $''ciao''$ ($''hello''$ in Italian) with the QWERTY keyboard; the purpose was to find the best press threshold in terms of time spent, mistakes made and personal preferences. The threshold value is the minimum distance (in cm) from the ToF camera below which a key is considered pressed. The results show that most of the users chose as the best threshold the nearest one (ThN = 50 cm) or the medium one (ThM = 60 cm). This is reasonable because the farthest threshold (ThF = 75 cm) is too much sensitive and it is easier to push a key unintentionally. Table 1 shows the writing times obtained for the three thresholds, as well as the errors made by each tester. The mean values were 16.2, 15.9 and 12.7 seconds, respectively, for ThN, ThM and ThF. A within-subjects ANOVA did not find any evident connection between threshold and times ($F = 3.03$, $p = .058$). A clear relation

**Table 1.** Test 1: writing a word. Time required and errors with different thresholds (ThN = 50cm, ThM = 60cm, ThF = 75cm). Highlighted cells indicate the threshold preferred by each user.

| | *ThN* | | *ThM* | | *ThF* | |
|---|---|---|---|---|---|---|
| | *Time (sec)* | *Errors* | *Time (sec)* | *Errors* | *Time (sec)* | *Errors* |
| Tester 1 | 18 | 0 | 20 | 0 | 13 | 2 |
| Tester 2 | 13 | 0 | 10 | 0 | 10 | 2 |
| Tester 3 | 10 | 1 | 10 | 0 | 9 | 3 |
| Tester 4 | 13 | 0 | 20 | 0 | 9 | 2 |
| Tester 5 | 19 | 1 | 14 | 3 | 10 | 3 |
| Tester 6 | 19 | 0 | 17 | 0 | 15 | 0 |
| Tester 7 | 15 | 1 | 15 | 1 | 13 | 3 |
| Tester 8 | 13 | 0 | 10 | 0 | 10 | 2 |
| Tester 9 | 13 | 1 | 12 | 1 | 18 | 4 |
| Tester 10 | 19 | 0 | 23 | 0 | 9 | 2 |
| Tester 11 | 22 | 1 | 19 | 0 | 13 | 0 |
| Tester 12 | 21 | 1 | 13 | 0 | 14 | 2 |
| Tester 13 | 14 | 0 | 21 | 0 | 13 | 2 |
| Tester 14 | 12 | 0 | 9 | 1 | 14 | 2 |
| Tester 15 | 8 | 2 | 9 | 2 | 10 | 1 |
| Tester 16 | 16 | 0 | 17 | 0 | 15 | 0 |

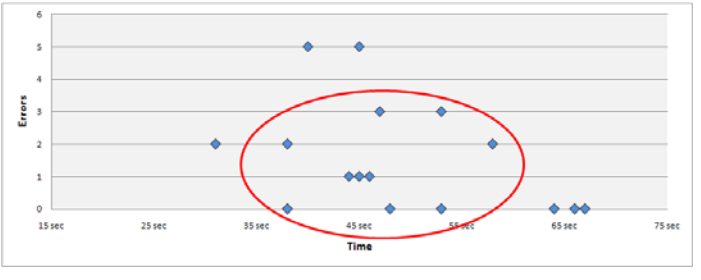



**Fig. 3.** Test2: writing a sentence with the threshold chosen in first test. Each point corresponds to a user

emerged instead between threshold and errors ($F = 9.79$, $p < .001$), with many more mistakes made with ThF.

In the second test we asked the users to write the sentence *"Ciao, come stai?"* (*"Hello, how are you?"* in Italian) using the thresholds chosen in the first test. The results obtained were interesting because they provided an indication about the user learning curve after a short period of system use. The plot in figure 3 shows a good balance between time and errors: for most users (see the area surrounded by the red circle) the task took between 35 and 60 seconds to complete, with a number of errors quite similar to that of the first part of the experiment (single word writing).

**Table 2.** Test 3: making a calculation

| | *ThN* | | *ThM* | | *ThF* | |
|---|---|---|---|---|---|---|
| | *Time (sec)* | *Errors* | *Time (sec)* | *Errors* | *Time (sec)* | *Errors* |
| Tester 1 | 10 | 0 | 9 | 0 | 12 | 2 |
| Tester 2 | 18 | 0 | 15 | 1 | 12 | 1 |
| Tester 3 | 10 | 0 | 11 | 0 | 15 | 3 |
| Tester 4 | 16 | 1 | 11 | 0 | 12 | 2 |
| Tester 5 | 12 | 0 | 12 | 0 | 7 | 1 |
| Tester 6 | 22 | 0 | 19 | 0 | 18 | 0 |
| Tester 7 | 21 | 1 | 15 | 0 | 20 | 0 |
| Tester 8 | 33 | 0 | 18 | 0 | 13 | 0 |
| Tester 9 | 22 | 0 | 13 | 1 | 13 | 2 |
| Tester 10 | 13 | 0 | 10 | 0 | 12 | 0 |
| Tester 11 | 20 | 0 | 16 | 0 | 10 | 0 |
| Tester 12 | 19 | 0 | 11 | 0 | 10 | 0 |
| Tester 13 | 14 | 1 | 12 | 0 | 13 | 2 |
| Tester 14 | 14 | 1 | 16 | 0 | 11 | 1 |
| Tester 15 | 15 | 1 | 13 | 1 | 20 | 2 |
| Tester 16 | 16 | 0 | 17 | 0 | 15 | 0 |

Finally, we carried out an experiment with the numeric pad (the user had to make the calculation $58 + 32$). Similarly to the first experiment, the purpose was to find whether there were changes in threshold preferences with larger keys (their size was twice that of the keyboard's keys). Like in the first experiment the thresholds were tested in randomized order. The results show a predictable reduction of errors (close to zero) with ThN and ThM, while with ThF their number was as high as in the first test. It is interesting to note that, whereas in the first experiment the difference in times with the three thresholds was minimal, now ThF is significantly greater. This anomaly may be explained considering that with small keys the great part of the time is spent selecting the correct key, while with big keys the selection is faster and the time required to press the key becomes more relevant. These considerations explain the nearly unanimous choice of the ThM threshold. The mean values for times were 17.2, 13.6 and 13.3 seconds, respectively, for ThN, ThM and ThF. A within-subjects ANOVA found clear relations both between threshold and times ($F = 4.04, p < .05$), with longer times with ThN, and between threshold and errors ($F = 6.3$, $p < .005$), with many more mistakes with ThF.

## 6 Conclusions

In this paper we have presented a new kind of gestural interaction with virtual keyboards that exploits the potentials of the combination of an RGB and a ToF camera. The system is totally independent of the background and of the shape of the hand, and the tracking stage enables the continuous identification of the active hand also in presence of similar clusters, such as a second hand or

part of an arm. The experimental evaluation performed with 16 users showed interesting results, in particular a rapid learning curve and a sensible reduction of mistakes after few minutes. Future improvements include a more precise color sub-segmentation for excluding possible false positives in the hand detection step and the concurrent use of two hands for writing faster.

## References

1. Kolb, A., Barth, E., Koch, R., Larsen, R.: Time-of-Flight Cameras in Computer Graphics. Computer Graphics Forum 29, 141–159 (2010)
2. Bartczak, B., Schiller, I., Beder, C., Koch, R.: Integration of a Time-of-Flight camera into a mixed reality system for handling dynamic scenes, moving viewpoints and occlusions in real-time. In: Fourth International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT 2008) (2008)
3. Santrac, N., Friedland, G., Rojas, R.: High resolution segmentation with a time-of-flight 3d-camera using the example of a lecture scene. Technical report (2006), `http://www.inf.fu-berlin.de/inst/agki/eng/index.html`
4. Bleiweiss, A., Werman, M.: Real-time foreground segmentation via range and color imaging. In: Kolb, A., Koch, R. (eds.) Dyn3D 2009. LNCS, vol. 5742, pp. 58–69. Springer, Heidelberg (2009)
5. Dondi, P., Lombardi, L.: Fast Real-Time Segmentation and Tracking of Multiple Subjects by Time-of-Flight Camera. In: 6th International Conference on Computer Vision Theory and Applications (VISAPP 2011), pp. 582–587 (2011)
6. Oprisescu, S., Falie, D., Ciuc, M., Buzuloiu, V.: Measurements with ToF Cameras and Their Necessary Corrections. In: International Symposium on Signals, Circuits and Systems, ISSCS 2007 (2007)
7. Oggier, T., Büttgen, B., Lustenberger, F., Becker, G., Rüegg, B., Hodac, A.: Swissranger SR3000 and First Experiences based on Miniaturized 3D-TOF Cameras. In: Proceedings, 1st Range Imaging Research Day, September 8-9, pp. 97–108. ETH Zurich Switzerland (2005)
8. Ghobadi, S., Loepprich, O., Hartmann, K., Loffeld, O.: Hand Segmentation Using 2D/3D Images. In: Image and Vision Computing, New Zealand, pp. 64–69 (2007)
9. Breuer, P., Eckes, C., Müller, S.: Hand Gesture Recognition with a Novel IR Time-of-Flight Range Camera–A Pilot Study. In: Gagalowicz, A., Philips, W. (eds.) MIRAGE 2007. LNCS, vol. 4418, pp. 247–260. Springer, Heidelberg (2007)
10. Kollorz, E., Penne, J., Hornegger, J., Barke, A.: Gesture recognition with a Time-Of-Flight camera. Int. J. Intell. Syst. Technol. Appl. 5(3/4), 334–343 (2008)
11. Van den Bergh, M., Van Gool, L.: Combining RGB and ToF cameras for real-time 3D hand gesture interaction. In: IEEE Workshop on Applications of Computer Vision (WACV 2011), pp. 66–72 (2011)
12. Haker, M., Böhme, M., Martinetz, T., Barth, E.: Deictic Gestures with a Time-of-Flight Camera. In: Kopp, S., Wachsmuth, I. (eds.) GW 2009. LNCS, vol. 5934, pp. 110–121. Springer, Heidelberg (2010)
13. Soutschek, S., Penne, J., Hornegger, J., Kornhuber, J.: 3-D Gesture-Based Scene Navigation in Medical Imaging Applications Using Time-Of-Flight Cameras. In: IEEE Computer Vision and Pattern Recognition Workshops, pp. 1–6 (2008)
14. Reulke, R.: Combination of distance data with high resolution images. In: Image Engeeniring and Vision Metrology, IEVM 2006 (2006)