Advanced Engineering Forum *Online: 2013-06-27*
ISSN: 2234-991X, Vols. 8-9, pp 553-562
doi:10.4028/www.scientific.net/AEF.8-9.553

2D Sensor Based Design of a Dynamic Hand Gesture Interpretation System

Ciprian David[1,a], Vasile Gui[1,b]

[1]Electronics and Telecommunications Faculty,

"Politehnica" University of Timisoara,

No. 2, Vasile Parvan Blvd., 300223, Timisoara, Romania

[a] ciprian.david@etc.upt.ro (corresponding author), [b] vasile.gui@etc.upt.ro

Keywords: dynamic hand gestures, human computer interfaces, trajectory encoding.

Abstract. A complete 2D sensor based system for dynamic gesture interpretation is presented in this paper. A hand model is devised for this purpose, composed of the palm area and the fingertips. Multiple cues are integrated in a feature space. Segmentation is carried out in this space to output the hand model. The robust technique of mean shift mode estimation is used to estimate the parameters of the hand model, making it adaptive and robust. The model is validated in various experiments concerning difficult situations like occlusion, varying illumination, and camouflage. Real time requirements are also met. The gesture interpretation approach refers to dynamic hand gestures. A collection of fingertip locations is collected from the hand model. Tensor voting approach is used to smooth and reconstruct the trajectory. The final output is represented by an encoding sequence of local trajectory directions. These are obtained by mean shift mode detection on the trajectory representation on Radon space. This module was tested and proved highly accurate.

Introduction

Human computer interfaces (HCIs) gained a steadily growing interest over the last decades. A wide range of applications make use of HCIs, some examples being: remote control [1], virtual desktops [2], robot control [3], and many others. Hand gestures are a powerful means of nonverbal communication among humans and one of the most widely used means of interaction between humans and computing systems. There are two different types of hand gesture based HCIs. One is focused on interpreting static hand poses and the other one is based on dynamic hand gestures. Our proposal refers to dynamic gestures HCIs and their applications. As we will see in the following, our gesture recognition algorithm also, exploits the robustness of dynamic inputs (relative to static gesture interpretation). Recently, various proposals [4, 5, 6] use 3D sensors for hand gesture recognition. However, there are numerous applications where the use of such sensor types is not possible. Moreover a working 2D sensor based hand gesture recognition system represents a cheaper solution than the ones using 3D sensors. The proposed system is intended for indoor applications that use 2D cameras. Even if the extension of the approach to a 3D sensor use is trivial, we intend here to prove that an efficient design is possible using but a 2D sensor (for the sake of having a cheaper hardware implementation).

The first stage in a HCI system is to construct an appropriate hand model. Secondly, the gesture recognition algorithm is to be designed accordingly to the used hand model. The gesture recognition stage has the task of extracting and labeling groups of features obtained from the hand model.

There are two main directions concerning the design of a hand model: appearance based and model based. Appearance based methods search for a mapping between extracted features and different hand postures. Generally, appearance based methods [7] are best suited to detect hand poses from a limited set of vocabulary. Model based approaches [8, 9], fit a synthetic hand model to the observed hand. The main downside of this approach consists in its high computational cost. Our system proposes a simpler model based related design. We find that for dynamic based gesture

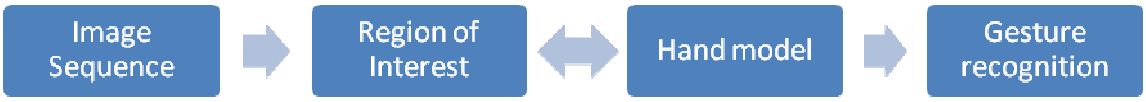
interpretation it is sufficient to have the hand modeled by the fingertips and the palm area. One can argue that detecting fingertips could suffice for such cases. We added the palm area to the model in order to increase the robustness by imposing a spatial constraint imposed the two components.
Adequate features have to be extracted for model components detection. A very common and robust approach involves the use of multiple cues. Stenger [9] for example, successfully uses color and shape as cues to hand detection in a model based framework. Our approach uses skin tone, shape, edges and motion information as cues for hand modeling.
As for the gesture interpretation part, our approach focuses on the case of trajectory segmentation. The trajectory is represented by the temporal collection of extracted fingertips. Some of the previous proposals in this direction use the condensation algorithm [10] or clustering of hidden Markov states [11] to segment the trajectory. However, these approaches are not designed specifically for dynamic hand gestures, and some of the problems encountered concern pause detection. Our model gives us the ability to define a small set of static gestures which are used to treat pauses between consecutive gestural inputs. We aim to design a suitable trajectory segmentation approach, resulting in an encoding sequence. Different actions to be performed by the computer can be assigned to such sequence. Various approaches that treat dynamic hand gestures concern the use of hidden Markov models (HMM) [12, 13], dynamic Bayesian networks [14] or the more recent motion divergence fields [15, 16]. Approaches using HMM present the drawback of compromising between training effort and a better classification rate. The proposals involving dynamic Bayesian networks lack in computational efficiency when training the model. The motion divergence field approach relies on optical flow computation that can prove not robust enough when the scene includes other moving or skin tone colored objects. Also, the method needs a predefined database for a matching stage of the processing chain. The efficiency of the approach is dependent on the database and adding a new gesture to be recognized will need a serious update of this database. We surpass these problems by designing a new model of the hand based on simple geometric features; and an online and versatile gesture recognition algorithm that allows constructing a large set of dynamic gestures. Another novelty and advantage of our proposal is that the interpretation system is not limited to a predefined set of gestures. Any new trajectory is easily labeled and can be straightforward incorporated in the human-computer communication protocol.
The rest of the paper is structured as follows. Next three sections detail our design. First an overview of the architecture is presented, then, the composing blocks are detailed: the hand model extraction block followed by the gesture recognition algorithm we have designed. We proceed by a section dedicated to experiments in order to validate the design block by block. The final section is dedicated to some concluding remarks.

System Description

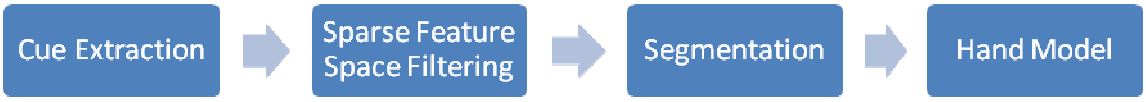
Overview. The system architecture is presented in Fig. 1. The general framework is represented in Fig. 1a. A region of interest (ROI) strategy is adopted in order to reduce the computational load. The ROI parameters are estimated in the hand model detection stage. The input for the gesture recognition algorithm consists of a sparse temporal collection of fingertip locations, also obtained from the hand model.
Fig. 1b details the hand model detection stage. First the cues are extracted. Our cues are represented by line segments extracted from binary maps in horizontal/vertical scans. We use both horizontal and vertical scans in order to achieve hand orientation invariance. Only line segments from one of the two scans are used, the one that holds the majority of valid such segments. The binary maps involved are represented by the foreground map, edge map and skin tone map, thus resulting in three types of cues. An important fact is that the same chain of Fig. 1b applies to both fingers and palm area. The only difference is in scale. The entire set of extracted line segments forms the sparse feature space. We use relaxed detectors in order to obtain all valid line segments. The outliers are filtered out in the next stage. Here we apply a mean shift mode detection [17] to estimate the length of the line segments. This estimate is used to filter the next frame associated feature space. Another filtering stage consists in imposing relative location constraints between finger and palm related line

segments. For example, if the hand is vertical, only finger and palm line segments that are directly above, are allowed. The segmentation process searches for groups of vertically adjacent line segments, in a top to bottom scan. Having segmented the palm and fingers the hand model is complete.

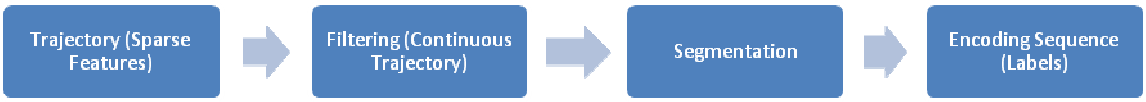
Fig.1c presents the gesture recognition algorithm. One active finger is used to input locally linear trajectories, composed of its fingertip locations. Next, the sparse set of trajectory points is filtered by a tensor voting technique [18], to obtain a continuous trajectory. The segmentation is carried out in Radon transformed space of the trajectory, by local maxima detection. The maxima in this space correspond to local orientations of the trajectory. Obtaining these points of local maximum implicitly gives the final encoding sequence. We code the trajectory by the succession of local orientations of the composing segments (e.g. E-NW-E). These encoding sequences determine only one dynamic gesture and a computer action can be attached to them (e.g. scroll, rotate, and so on).



(a) General architecture.



(b) Hand model extraction.



(c) Gesture interpretation.

Fig. 1. System schematics.

Hand Model Extraction

Cue Detection. As mentioned before, we obtain the three types of cues from various binary maps. Cue detection is represented by a simple operation of horizontal/vertical scan to identify the adequate line segments on the binary maps. By adequate we mean that valid line segments are considered only those lying in an interval dependent of the previous estimated length. In our implementation this interval is taken to be [0.5, 1.5] x *estimated length*. Handling binary maps provides an important reduction in computational time. The binary maps involved, are represented by the foreground map, skin tone map and edge map. The approaches chosen for this purpose are simple, fast and relaxed.

The foreground map is obtained with the approach proposed by Kim et al. [19]. The approach is proven to be real time, robust and capable of updating the background model. The latter is important for applications that run for long periods of time. The skin map is obtained with a threshold based approach [20]. Skin segmentation is conducted in the RGB space. The thresholds are trained on an extensive database of skin tone image samples, which meets our requirement of relaxed detector. Also, using previously trained thresholds significantly increases the computational speed. Finally, for edge map extraction we use the Canny edge detector. For a more stable detection, edge extraction is confined to skin colored foreground areas within the ROI. Furthermore, static edges are cleared out by frame differencing, to avoid confusion from background clutter.

The reunion of all the extracted cues represents the feature space *FS*:

$$FS = \bigcup_i (FGC_i \vee SKC_i \vee EGC_i) \quad (1)$$

With FGC_i the set of foreground map cues, SKC_i the set of skin map cues and EGC_i the set of edge map cues. An example of the cue sets from above is presented in Fig. 2 (gray level).

Fig. 2. Examples of cue sets in gray level, from left to right: FGC_i, SKC_i, EGC_i.

Feature Space Filtering. An important stage is represented by feature space filtering. Since the feature extraction used relaxed detectors, it is necessary to eliminate the outliers. For this purpose we estimate the segment length by employing a mean shift type robust estimator [17]. The estimated length is given by the dominant mode in the feature space. Using the Epanechnikov kernel:

$$k_E(u) = \frac{3}{4}(1-u^2)\cdot \mathbf{1}_{\{|u|\le 1\}} \tag{2}$$

Where:

$$\mathbf{1}_{\{|u|\le 1\}} = \begin{cases} 1 & |u| \le 1 \\ 0 & elsewhere \end{cases} \tag{3}$$

The estimated length is obtained by iterating to convergence the following equation:

$$\hat{t}^{(j+1)} = \mathrm{f}^{(j)}(\hat{t}^{(j)}) \tag{4}$$

With:

$$\mathrm{f}^{(j)}(\hat{t}) = \sum_t \frac{g_E(\frac{|\hat{t}-t|}{s_j})h(t)}{\sum_{t'} g_E(\frac{|\hat{t}-t'|}{s_j})h(t')}t \tag{5}$$

Where h is the histogram of lengths from the detected line segments, g_E is the derivative of the Epanechnikov kernel, and s_j is the mean shift algorithm scale (usually equal to 1). It follows that:

$$\mathrm{f}^{(j)}(\hat{t}) = \frac{\sum_{t=\hat{t}-s_j}^{t=\hat{t}+s_j} th(t)}{\sum_{t'=\hat{t}-s_j}^{t'=\hat{t}+s_j} h(t')} \tag{6}$$

ROI Construction. The estimated length corresponds to the finger/palm width. This parameter varies with the user's position and gesturing, relative to the camera. Also, the estimate will influence the ROI. In our implementation we consider a square ROI centered on the extracted palm center and its size about 3 times the palm width. Evidently, the ROI is adapted as a valid hand model is detected. When there is no user interacting with the system the ROI is the entire image.

Hand Model Segmentation. For the segmentation we consider the morphological operation of reconstruction by geodesic dilation. The first marker is the middle point of the upmost extracted line segment. By geodesic dilation we detect and group downwards adjacent neighbors, thus obtaining a segmented finger and the fingertip location. If multiple fingers are active we eliminate from the feature space previously segmented line segments and iterate the above operations until all fingers

are obtained (empty feature space). For palm area segmentation only one pass is needed and the palm center is retained. Examples of segmented fingers, palm area and the final hand model are presented in Fig. 3.

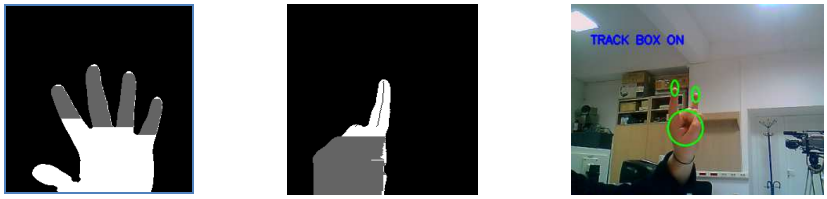


Fig. 3. (from left to right) Examples of segmented fingers, segmented palm, and final hand model.

Primitive chains are segmented in this paper by means of a morphological tool: reconstruction by geodesic dilation. The markers are first detected in the primitive list. Fingertip coordinates represent the markers, meaning the middle point of the primitive (line segment) having the bigger y coordinate. Considering the vertical orientation of the hand the uppermost fingertip is detected in a simple horizontal scan. Connected neighbors are searched then downwards by geodesic dilation. The set of connected primitives represents the segmented finger. The segmented primitive set is then removed from the list and the same procedure is iterated to find the remaining fingers. The algorithm ends when the list is empty or a small number of primitives are left. At the same location multiple primitives can be present in the list, due to multiple cue character of the design. The right primitive for the segmentation is chosen with respect to angle conservation relative to the previous one or the starting marker.

Gesture Interpretation

Using one active finger a dynamic trajectory can be collected, composed of the fingertip locations. We address the case of locally linear trajectories. The encoding sequence is represented by the local directions of the trajectory. We find that a total of 8 directions (N, S, E, W, SW, SE, NW, NE) suffice, and gives the possibility of designing a large set of meaningful gestures. The number of the composing trajectory segments is not limited; however, in our experiments we used up to a succession of 4 oriented segments to compose the trajectory.

Trajectory Filtering. First step in recognizing the gesture is to obtain a smooth continuous trajectory from the sparse collection. For this purpose we use the tensor voting technique [18]. Its advantage comes from the perceptual based design that is able to naturally reconstruct interrupted lines. Trajectory points are encoded by the following tensor, carrying information on their directional character:

$$T = \begin{bmatrix} \vec{e}_1 & \vec{e}_2 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} \vec{e}_1^{\,T} \\ \vec{e}_2^{\,T} \end{bmatrix} \tag{7}$$

Where $\lambda_i, \vec{e}_i$, $i=1,2$ represent the eigenvalues and eigenvectors obtained with a principal component analysis (PCA) on the location of the neighboring trajectory points. The first term of the tensor encodes the linear saliency of the neighborhood and the second term corresponds to the isotropic saliency.

In the next step the tensor is accumulated in the neighboring trajectory points to reinforce the local directionality. Then, it is propagated and accumulated in the missing locations of the trajectory. The accumulated tensor is given by the following equation:

$$T_{acc} = \lambda_1 \cdot \vec{e}_1 \vec{e}_1^{\,T} + \lambda_2 \cdot \vec{e}_2 \vec{e}_2^{\,T} = (\lambda_1 - \lambda_2) \cdot \vec{e}_1 \vec{e}_1^{\,T} + \lambda_2 \cdot (\vec{e}_1 \vec{e}_1^{\,T} + \vec{e}_2 \vec{e}_2^{\,T}) \tag{8}$$

Performing a PCA on every point that accumulated such tensors we obtain the smooth trajectory by displaying the linear saliency map given by the quantity $(\lambda_1 - \lambda_2)$. An example of smoothed trajectory is presented in Fig. 4(middle).

Fig. 4. Sparse trajectory (left), smoothed trajectory (middle) and corresponding Radon transform (right).

Gesture Interpretation. The final output of the system is the succession of composing segment directions. For this purpose, the trajectory is transformed and represented in Radon space:

$$\Re f(s,\theta) = \int_{-\infty}^{\infty} f(x(t), y(t))dt \tag{9}$$

With:

$$\begin{aligned} x(t) &= t\sin\theta + s\cos\theta \\ y(t) &= -t\cos\theta + s\sin\theta \end{aligned} \tag{10}$$

Because the angle of the Radon transform of Eq. 5, spans the interval $[0,\pi]$ it is not possible to extract the previously mentioned 8 directions. Instead of the transform of Eq. 5 we consider the following, with the angle spanning the interval $[0,2\pi]$:

$$\Re f(s,\theta) = \int_{0}^{\infty} f(x(t), y(t))dt \tag{11}$$

In Fig. 4(right) an example of trajectory represented in Radon space, is showed. The maxima correspond to the three directions of the trajectory segments. To obtain the final encoding sequence the angle corresponding to each maximum is extracted by mode detection. Our choice is to use again, mean shift mode detection, and the resulting output sequence for the example of Fig. 4, is: E-SW-E. The gesture started in the top left corner.

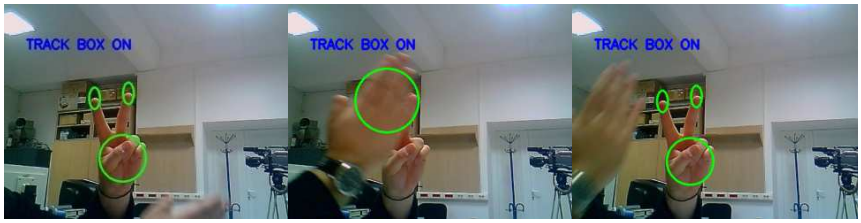
Experiments

Hand Model Validation. The proposed design is intended for HCIs. In this context a very important role has the robustness of the hand model. There are a number of situations where the system could fail to respond properly. In the following, an assessment of the system behavior in such cases is presented. Varying illumination could result in misdetections concerning any of the three cues involved. Camouflage caused by moving objects behind the user or by skin like objects present in the background can lead to an erroneous hand model extraction. Also, we need to address the problem of occlusion. In such a case, the model extraction will evidently be perturbed, but the question is how quickly can the correct model be recovered? Fig. 5 presents the results of the hand model validation experiments.

(a) Varying illumination experiment.

(b) Moving object camouflage experiment.

(c) Skin like objects in the background experiment.



(d) Occlusion experiment.

Fig. 5. Hand model validation experiments.

The examples of Fig. 5a show that the model is stable for a large interval of illuminations. Moreover, if used in an indoor environment, such extreme values for the illuminations are not very probable during a user interaction. Fig. 5b,c validate our model in the two cases of camouflage. The combination of multiple cues offers the robustness in these cases. Foreground, skin and edge cues are expected to act differently in various cases and to aid the system passing similar situations. Fig. 5d shows the test conducted in the case of occlusion. Our measurements showed that the system recovers a valid hand model in about 4-7 frames from moment the model was lost.

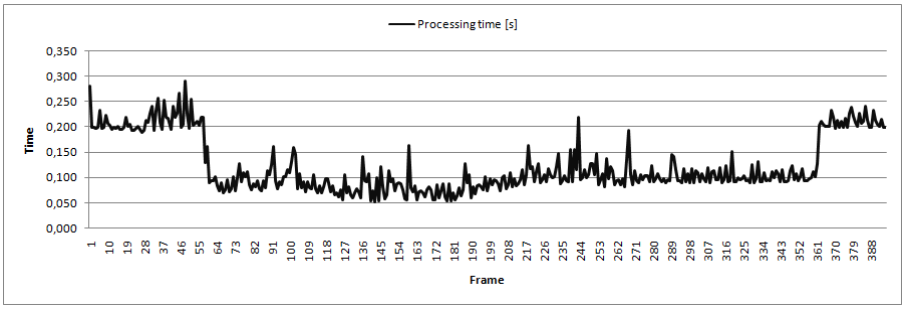


Fig. 6. Processing time per frame.

Real Time. As expected, meeting real time constraints is an important issue for a HCI. These constraints apply mainly for the hand model extraction block, since it is used mostly in the user's input phase. Also, it is here that the majority of computations are done. Fig. 6 shows the computational speed related behavior of our design. From the point of view of a user interaction, we can distinguish between two functioning modes of the system. One is the active mode, when there is an interaction and second is the search mode when the system looks for a valid hand model. The factor that mostly influences the system speed is ROI adaptation. As the hand model is extracted the system adapts and processes only a small part of the entire frame. This leads to a significant decrease in processing time. There is an average speed of 15 fps in active mode and about 5 fps in

searching mode. It is to be noted that our implementation is not optimized and the measurements were conducted using a laptop (Intel Core2 Duo CPU at 2,5GHz, 2GB RAM, NVIDIA GeForce 8600M GT and 1GB VRAM) and the incorporated web camera.

Gesture recognition accuracy. In order to have a fair assessment of the systems' accuracy we need to have an idea about the accuracy of the users input. For this purpose, human inputs of two simple lines oriented at 0° and 45° were collected from several subjects. Using our algorithm we estimated the angle, standard deviation and the error interval. The results are presented in Table 1.

Table 1. Human input accuracy assessment.

Real angle	Estimated mean angle	Standard deviation	Interval
0°	0°	0.66	$[-4^\circ...3^\circ]$
45°	41.3°	2.25	$[31^\circ...52^\circ]$

These results also motivated the use of only 8 encoding directions, as we see that human input is not that accurate. For comparison, synthetically perturbed trajectories were fed to the dynamic gesture recognition block. In Fig. 7 a plot of the mean estimation error and the standard deviation is shown. Note that the detection error is very low and that human input error is higher, proving that the gesture interpretation block is accurate enough to be used in a HCI application.
A final validation test consisted in measuring the recognition rate. Since we assessed the human input accuracy the recognition rate was computed on synthetically generated trajectories in two cases. First case considered severe perturbations of the test trajectories; a higher perturbation with respect to the human input accuracy, and the recognition rate was of 96%. In the second case, we tested the recognition on mildly perturbed trajectories; within the range of human accuracy input. In this case we obtained a 100% recognition rate. These tests show that due to its simplicity and when the human error is eliminated the system performs at its best.
Some typical examples of results of the gesture interpretations are presented in Fig. 8, all gestures started in the upper left corner.

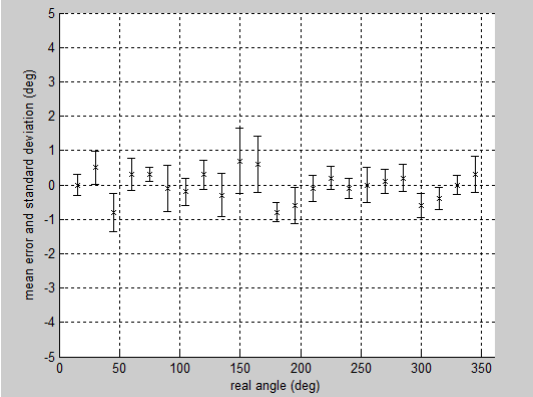


Fig. 7. Recognition accuracy experiment.

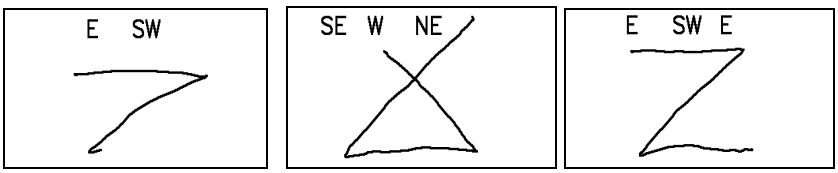


Fig. 8. Examples of dynamic gestures and system interpretation.

Conclusion and Discussion

We have presented a new complete design for gesture recognition to be used in HCI applications. Our system is intended mainly for dynamic gestures. It is composed from a hand model extraction block, followed by gesture interpretation. The hand model is obtained by detecting the fingertips and the palm area which need to be spatially correlated. Each of the two model components are by segmentation in a sparse feature space. The feature space is constructed from the reunion of multiple cues. Out approach uses foreground, skin and edge cues in the form of binary maps to reduce the computational load. The parameters are estimated by means of robust estimators, making the system very adaptive. A series of tests validate the hand model in challenging situations like occlusions, camouflage and varying illumination. Real time constraints are proven to be met by the system.

As for the gesture interpretation approach, it takes as input a collection of fingertip locations, obtained from the hand model. The output is a sequence that encodes trajectory by directions. The application refers to locally linear trajectories. First a smooth trajectory is constructed by tensor voting filtering. The succession of directions is obtained by mode detection in Radon space. The use of 8 possible directions and not imposing limitations on the trajectory length offers the possibility of devising a large dictionary of meaningful gestures.

We should mention that pauses between consecutive gestures can be marked by a static gesture. In fact, the hand model allows a limited number of such gestures, by active finger counting (e.g. 2 active fingers, see Fig. 5). We note also, that the measured response time of the gesture interpretation block is sufficiently low (for gestures up to 4-5 segments) to not encumber user's interaction with the system. Finally, one can argue about the relevance of this work, especially nowadays, when 3D sensors are more and more available. These sensors could spare much of the trouble that 2D sensors create. The answer is simply because it offers a cheaper solution, and moreover, there are numerous applications that do not need an elaborate HCI the 3D sensors could offer. Still, the framework presented here is versatile and a 3D sensor can easily be incorporated. Of course, the use of foreground detection will become obsolete and the related cue can be replaced with depth maps, for example.

Acknowledgement

This paper was supported by the project "Development and support of multidisciplinary postdoctoral programmes in major technical areas of national strategy of Research - Development - Innovation" 4D-POSTDOC, contract no. POSDRU/89/1.5/S/52603, project co-funded by the European Social Fund through Sectoral Operational Programme Human Resources Development 2007-2013.

References

[1] W.T. Freeman, C.D. Weissman, Television control by hand gestures. Intl. Workshop on Automatic Face and Gesture Recognition, Zurich, Switzerland, June, 1995.

[2] N. Stefanov, A. Galata, R. Hubbold, Real-time hand tracker using variable-length markov models of behaviour. Computer Vision and Image Understanding, 108(1-2), 98–115, 2007.

[3] C. Wang, K. Wang, Hand posture recognition using adaboost with sift for human robot interaction. Intl. Conf. on Advanced Robotics, Jeju Island, South Korea, 21-24 August, 2007.

[4] A. Kurakin, Z. Zhang, Z. Liu, Real Time System for Dynamic Hand Gesture Recognition with a Depth Sensor, 20th European Signal Processing Conf., Bucharest, Romania, August 27-31, 2012.

[5] M.Van den Berg, L. Van Gool, Combining RGB and ToF Cameras for Real-time 3D Hand Gesture Interaction, IEEE Workshop on Applications of Computer Vision, 5-7 January 2011.

[6] Z. Li, R. Jarvis, Real Time Hand Gesture Recognition Using a Range Camera. Australasian Conf. on Robotics and Automation, Sydney, Australia, December 2-4, 2009.

[7] M. Kolsch, M. Turk, Fast 2D hand tracking with flocks of features and multi-cue integration. IEEE Workshop on Real-Time Vision for Human-Computer Interaction, pp. 158–165, 2004.

[8] I. Oikonomidis, N. Kyriazis, A. Argyros, Markerless and efficient 26-dof hand pose recovery. Asian Conf. on Computer Vision, Queenstown, New Zealand, 8-12 November 2010.

[9] B. Stenger, A. Thayananthan, P.H.S. Torr, R. Cipolla, Model-based hand tracking using a hierarchical bayesian filter. IEEE Trans. Pattern Analysis and Machine Intell., 28(9), 1372–1384, 2006.

[10] M. J. Black, and A. D. Jepson, Recognizing Temporal Trajectories using the Condensation Algorithm. Third IEEE Intl. Conf. on Automatic Face and Gesture Recognition, Nara, Japan, April 1998.

[11] G. Shaogang, M. Walter, A. Psarrou, Recognition of Temporal Structures: Learning Prior and Propagating Observation Augmented Densities via Hidden Markov States. Proc. Seventh Intl. Conf. on Computer Vision, 1999.

[12] S.Rajko, G. Qian, T. Ingalls, J. James, Real-time gesture recognition with minimal training requirements and on-line learning. Intl. Conf. on Computer Vision and Pattern Recognition, 2007.

[13] S. Wang, A. Quattoni, L. P. Morency, D. Demirdjian, T. Darrell. Hidden conditional random fields for gesture recognition. Intl. Conf. on Computer Vision and Pattern Recognition, 2006.

[14] H.-I. Suk, B.-K. Sin, S.-W. Lee, Hand Gesture Recognition Based on Dynamic Bayesian Network Framework. Pattern Recognition, 2010, 43(9), 3059-3072.

[15] X. Shen, G. Hua, L. Williams, Y. Wu, Motion Divergence Fields for Dynamic Hand Recognition. Proc. of Automatic Face and Gesture Recognition, 2011, 492-499.

[16] X. Shen, G. Hua, L. Williams, Y. Wu, Dynamic Hand Gesture Recognition: An Exemplar-Based Approach from Motion Divergence Fields. Journal of Image and Vision Computing, 2012, 30(3), 227-235.

[17] Comaniciu, D.; Meer, P. Mean Shift: A Robust Approach Toward Feature Space Analysis, IEEE Trans. Pattern Recognition and Machine Intell., 24(5), 603-619, 2002.

[18] M. Nicolescu, and G. Medioni, Perceptual grouping from motion cues - a 4-D voting approach. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25(4), pp. 492–501, 2003.

[19] K. Kim, T.H. Chalidabhongse, D. Harwood, L. Davis, Real-time Foreground–Background Segmentation Using Codebook Model. Real-time Imaging, 11(3), 167-256, 2005.

[20] A. Cheddad, J. Condell, K. Curran, P. A. Mc Kevitt, Skin Tone Detection Algorithm for an Adaptive Approach to Steganography. Signal Processing, 2009, 89, 2465–2478.