3D Hand Gesture Recognition using the Hough Transform

Serban OPRISESCU[1], Erhardt BARTH[2]
[1]*LAPI, University Politehnica from Bucuresti, Romania*
[2]*Institute for Neuro- and Bioinformatics, University of Lübeck, Germany*
soprisescu@imag.pub.ro

***Abstract*—This paper presents an automatic 3D dynamic hand gesture recognition algorithm relying on both intensity and depth information provided by a Kinect camera. Gesture classification consists of a decision tree constructed on six parameters delivered by the Hough transform of projected 3D points. The Hough transform is originally applied, for the first time, on the projected gesture trajectories to obtain a reliable decision. The experimental data obtained from 300 video sequences with different subjects validate the proposed recognition method.**

***Index Terms*—image processing, computer vision, gesture recognition, Kinect camera, Hough transform.**

I. INTRODUCTION

The field of research on human-computer interaction has known a major growth in the recent years, coming from the need of a more natural way to interact with a computer. The set of possible applications is also growing, from computer games to robot interaction, operating-room assistance etc. This growth has been promoted by the introduction of new 3D image sensors such as time-of-flight (ToF) cameras and the Kinect.

ToF cameras are "active illumination" sensors, meaning that the scene is illuminated by the camera, using some IR LEDs or laser IR diodes. The light emitted by the camera is usually sinusoidally modulated, and the camera computes on each pixel the phase shift between the emitted light signal and the received one. This range measurement principle is depicted in Fig. 1.

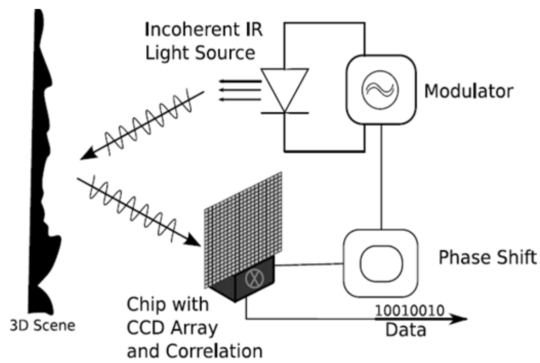

Figure 1. The ToF camera distance measurement principle

The maximum distance computed is from 5m to 10m, depending on the modulation frequency used, and it comes from the 2π maximum phase shift. Typically the image resolution is low, for instance 176x144 pixels (due to the very fast electronics needed for light phase shift computation).

In Fig. 2 one shows some examples of ToF cameras present on the market today.

Figure 2. Examples of ToF cameras

In Fig. 3 one shows the very popular Kinect sensor, mainly used for video games.

Figure 3. The Kinect sensor

Kinect is not a ToF camera, although the range image is similar to the one obtained with a ToF sensor. In order to compute the depth, the camera uses an IR laser projector which creates a pattern of dots on the objects. Comparing with the ToF camera which provides only depth and a gray-level infrared image, the Kinect camera has also a RGB camera, and the fusion between depth and RGB is done within the driver. Hence, Kinect has all the advantages of ToF cameras (works in low light conditions, does not relies on textured objects like stereo vision, can be highly miniaturized etc.) and offers also true color images, which can be directly used for face recognition, skin color detection, and other image processing algorithms.

In this paper we will focus on the recognition of hand gestures based on a video sequence acquired with a Kinect sensor. The work is motivated by different applications, for example gesture-based interfaces in a car, where only a simple set of gestures is used but the requirements on robust recognition are high.

Within the last years, the number of research activities

The work has been co-funded by the Sectoral Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labour, Family and Social Protection through the Financial Agreement POSDRU/89/1.5/S/62557.

Digital Object Identifier 10.4316/AECE.2013.03012

1582-7445 © 2013 AECE

relying on range cameras has increased dramatically. A comprehensive state of the art on this issue can be found in [1]. The authors concluded that, from 2006 to 2008, the number of publications has increased by a factor of 5, and the research focus changed from basic applications (like sensor characteristics or static scene acquisition) to human machine interaction and surveillance.

Hand-gesture recognition using 3D cameras is relatively new. Among the first authors that followed this approach were Liu and Fujimura [2]. They used the range data only for segmentation (to extract the hand and the head), and then performed 2D hand-shape comparisons (using the Chamfer Distance) and template matching for hand motion trajectories. Many different gesture-recognition techniques have been reported [3]. They involve various methods of segmentation, feature extraction and recognition strategies. For instance, in [4] the authors use a SR4000 ToF camera to recognize arm gestures with 3D motion primitives made invariant to rotation by spherical harmonic basis functions. In [5] a comparison between gesture recognition based on skin color (from RGB images only) and on depth data from a Kinect camera is performed. The authors emphasize the usefulness of the Kinect camera; gesture recognition is based on a dynamic programming method (Dynamic Time Warping (DTW)). A Kinect sensor is also used in [6] for the recognition of ten digits written in the air. The authors also use DTW to improve gesture recognition accuracy of various graphical models: Hidden Markov Model (HMM), input-output HMM (IOHMM), hidden conditional random field (HCRF) and explicit duration model (EDM). They compare the results and conclude that, after clustering, model complexities are reduced and recognition rates increase. The authors of [7] present a medical image exploration system based on the Kinect. Their system is able to recognize body postures and hand motions (pointing, zooming etc.) after a calibration step (skeleton fitting, hand area calibration etc.). Also a touch-free 3D medical image visualization and navigation system is presented in [8]. The system is based on the SR-3100 ToF camera, and a set of five gestures (translation, cursor, click, rotation and reset) are defined. The gestures consist on a hand posture and 3D hand motion interpretation. To reduce the recognition error, a majority vote among three classifiers (k-d Tree Based k-Means Clustering, Bayesian Plug-In Classifier and Nearest Neighbor) has been used.

In [9] a recognition scheme based on the Growing Neural Gas (GNG) algorithm (a variant of the self-organizing feature map) is implemented to recognize three hand gestures, captured with a Kinect camera, and to control a robot. A learning mechanism based on user feedback is used to improve recognition rates. A system of hand gesture recognition by image processing for human robot interaction is presented in [10]. The authors use a single RGB video camera to control a robotic arm having six main joints (Basis, Shoulder, Elbow, Pitch and Roll for wrist, and one for Clamp). The hand postures are recognized with a classifier that combines geometric, heuristic and structural pattern recognition techniques.

In [11] a “GestureMouse” application is proposed, that can perform different actions (move icons, select, zoom etc.) based on hand gestures captured by the Kinect. The authors use a nearest-neighbor classifier with a metric defined by the Frobenius norm of the difference between log-covariance matrices (containing statistical features from skeleton joints).

In [12] the Hough transform, and the generalized Hough transform in [13], are used to recognize static hand postures representing letters of sign language. In [14] the Hough transform is used to compute the angles of fingers within some hand postures; these measured angles are then used to control a robot.

Our method also relies on the Hough transform (for lines), which is used in an original way, to detect dynamic hand gestures. More precisely, we defined five gestures consisting of linear hand motions in the 3D space, and the main idea is that the Hough transform can be used to detect these hand trajectories in space-time because they are piece-wise approximately linear. The results obtained are promising, and they show that this approach is robust and appropriate for detecting such hand gestures. Furthermore, by using this method, we are able to discriminate, with a recognition rate of about 84%, between a defined gesture and a non-defined or random gesture; handling such random non-intended gestures is an important and difficult problem in gesture recognition.

The paper is organized as follows: Section II describes the proposed gesture recognition algorithm, Section III the results and Section IV the conclusions.

II. The Proposed Gesture Recognition Method

The input for the image processing chain consists of a movie (of around 100 frames) from the Kinect camera. In fact, one has at each instance of time two images: a color image of the scene, and a depth image.

In order to make the algorithm as simple as possible, without the need of hand segmentation or hand shape recognition and tracking (which requires hard computation and is not 100% reliable), we need to impose a constraint. The constraint is that the camera should not move, and that the object that has the predominant (or the fastest) motion within the scene must be the hand. Also, with this approach, we only detect a single hand motion.

How can we obtain the hand trajectory fast, without complex segmentation or tracking algorithms? The idea was to use a simple motion detection algorithm, namely the absolute value of the difference between two consecutive frames. We call this the DIF operator. We use this method in order to detect motion based on the color image converted to gray scale (because the noise within the color image is much lower than the noise within the distance image).

Based on this simple segmentation, the next step is to estimate the parameters of the motions. For this, we use the Hough transform. However, the classic Hough transform is in 2D (takes as input a binary image) and our motion is in 3D. To cope with this, we use projections: we will have as input for the Hough transform 3 projection images: a XT image, an YT image, and a ZT image. More precisely, from the entire movie, we will construct the above-mentioned three 2D images by projecting the 3D data.

For instance, the XT projection is obtained as follows: at each instance of time one computes the DIF image mentioned above. One thresholds this image, with a fixed threshold of 97% of the maximum value, and then one projects the resulting binary image on the X axis. In case of a constant motion, the resulting points will lie on a line. This line will be a line in the XT plane, where the T axis will count for the frame number. In this manner, if the hand describes a linear trajectory at a constant speed along the X axis, in the XT plane we will obtain a line at a certain angle in the XT plane. If the hand moves perpendicular to the X axis, in the XT plane we will have a line perpendicular to the X axis, and in the YT plane we will obtain a line at a certain angle.

In order to obtain trajectories in the ZT plane, we must use the Z axis (perpendicular to the XY plane of the image), which is a virtual axis. In our algorithm the Z axis contains values in the interval (0.4, 2) meters, which ensures that moving objects farther than 2 meters are invisible in the ZT plane. In order to construct the ZT plane we use the absolute differences between the histograms of two consecutive distance frames, the histograms being computed on the (0.4, 2) meters interval. This histogram difference, thresholded at 90% of the maximum peak, will yield a binary line in the ZT plane. This method relies on the fact that if the hand moves along the Z axis, there will be a peak in the histogram difference at that particular hand-distance value.

In Fig. 4 we show an example of a trajectory in the XT plane for an ideal right-to-left hand gesture. By applying the Hough transform, we detect the line and obtain the angle θ, which we will use in the subsequent decision tree.

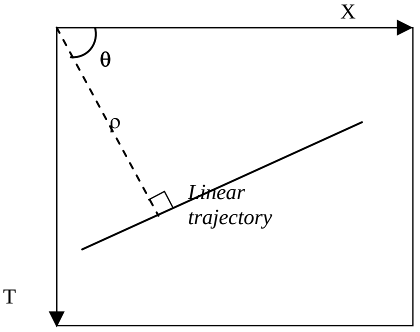

Figure 4. Hough transform example

The gesture recognition algorithm we used is the following.

Input: intensity frames A_i and distance frames D_i
i=2; {the frame number}
repeat {for each frame}
DIF = $|A_{i-1} – A_i|$;
H_{i-1}=histogram(D_{i-1}); H_i=histogram(D_i);
HD = $|H_{i-1} – H_i|$; {the histogram difference}
HD=HD > (90/100*max(HD)); {a binary line}
Project points to XT, YT (using DIF) and ZT (using HD)
until i=100
{here we have XT,YT and YT images for 100 frames}
Apply the Hough transform on XT, YT and YT.
{the gesture recognition decision tree is as follows}
if $(((|\theta_{ZT}| > 60)$ **and** $(H_Z > 1))$ **and** $(((|\theta_{YT}| > 60)$ **and** $(H_Y < 1))$ **or** $(|\theta_{YT}| < 60)$)) **then** gesture ← Stop;
elseif $(((|\theta_{XT}| > 60)$ **and** $(H_X > 1)$ **and not**$(1+\mathrm{sign}(\theta_{XT})))$ **and** $(((|\theta_{YT}| > 60)$ **and** $(H_Y < 1))$ **or** $(|\theta_{YT}| < 60)$) **and** $(((|\theta_{ZT}| > 60)$ **and** $(H_Z < 1))$ **or** $(|\theta_{ZT}| < 60)$)) **then** gesture ← Left-right;
elseif $(((|\theta_{XT}| > 60)$ **and** $(H_X > 1)$ **and not**$(1-\mathrm{sign}(\theta_{XT})))$ **and** $(((|\theta_{YT}| > 60)$ **and** $(H_Y < 1))$ **or** $(|\theta_{YT}| < 60))$ $(((|\theta_{ZT}| > 60)$ **and** $(H_Z < 1))$ **or** $(|\theta_{ZT}| < 60)$)) **then** gesture ← Right-left;
elseif $(((|\theta_{YT}| > 60)$ **and** $(H_Y > 1)$ **and not**$(1+\mathrm{sign}(\theta_{YT})))$ **and** $(((|\theta_{XT}| > 60|$ **and** $(H_X < 1))$ **or** $(|\theta_{XT}| < 60))$ $(((|\theta_{ZT}| > 60)$ **and** $(H_Z < 1))$ **or** $(|\theta_{ZT}| < 60)$)) **then** gesture ← Up-down;
elseif $(((|\theta_{YT}| > 60)$ **and** $(H_Y > 1)$ **and not**$(1-\mathrm{sign}(\theta_{YT})))$ **and** $(((|\theta_{XT}| > 60)$ **and** $(H_X < 1))$ **or** $(|\theta_{XT}| < 60))$ $(((|\theta_{ZT}| > 60)$ **and** $(H_Z < 1))$ **or** $(|\theta_{ZT}| < 60)$)) gesture ← Down-up;
else gesture ← Random;
end
Output: gesture

The meaning of the decision parameters and the choice of the thresholds will be explained in the next section.

III. Experimental Results

In order to test our algorithm we recorded 300 movies with the Kinect camera (10 movies x 6 gesture types x 5 different subjects performing the gestures). Each movie has 100 frames (about 3 seconds at 30 fps). The camera was directly connected to a portable computer, via a USB cable. Range and color images were directly acquired from the camera at 30FPS using the driver provided by the OpenKinect online project. The image processing and gesture recognition algorithms were developed in Matlab.

In Fig. 5 we show 5 single frames for each of the defined gestures, and the arrow represents the hand trajectory (the Stop gesture is towards the camera). The random gestures class contains random hand motions recorded within 3 seconds.

The five orthogonal hand gestures defined form a reasonable working set, because introducing more gestures can increase the complexity of the algorithm and hence reduce the computation speed, and perhaps reduce the recognition rate (by creating confusions). Our aim was to develop an interface which works in real time (and this demands a simple, fast and reliable algorithm) and one can associate five different commands (for database navigation, robot control etc.).

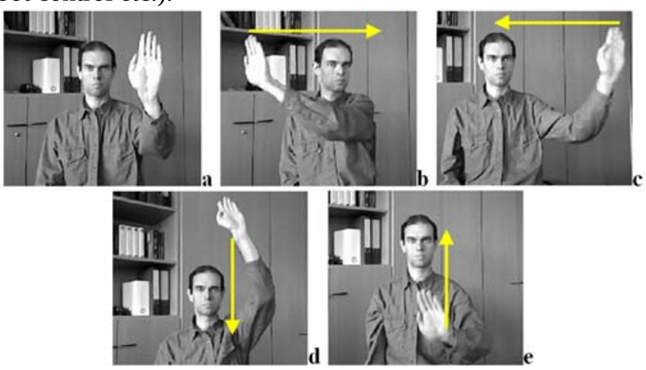

Figure 5. The 5 defined gestures: a) Stop, b) Left-right, c) Right-left, d) Up-down and e) Down-up

In Fig. 6 a) one shows an example of an intensity frame from the Kinect camera and in Fig. 6 b) the corresponding depth image (distance is showed in mm). One mentions that our method is insensitive to the clothes worn by the human.

In Figs. 7 to 11 we show examples of obtained trajectories within the projection planes for some of the 5 defined gestures.

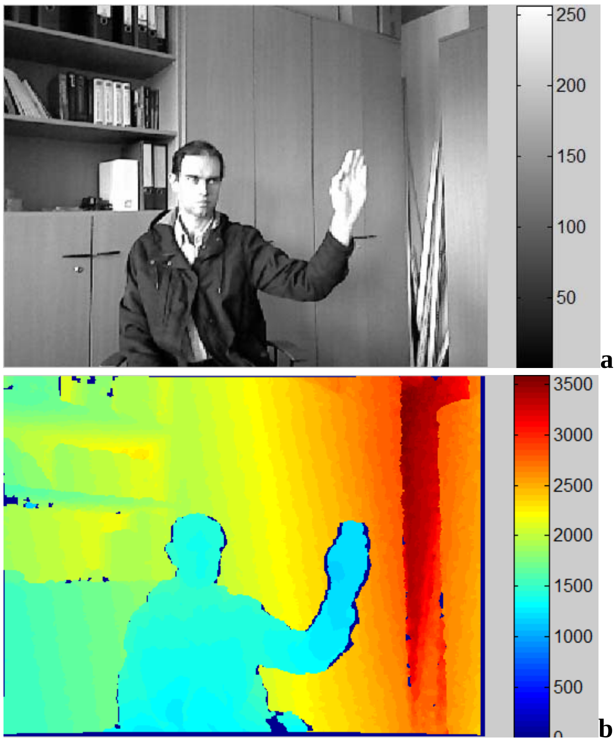

Figure 6. a) Intensity image; b) Depth image.

Notice that the remarks made in Section II concerning the trajectory shape are confirmed by the experiments. If the subjects would perform an ideal gesture (with constant speed) then we would obtain perfect lines. Since the resulting patterns are not perfect lines, the use of the Hough transform is appropriate in this case, as we will see in the following paragraphs. Figs. 12 and 13 show some

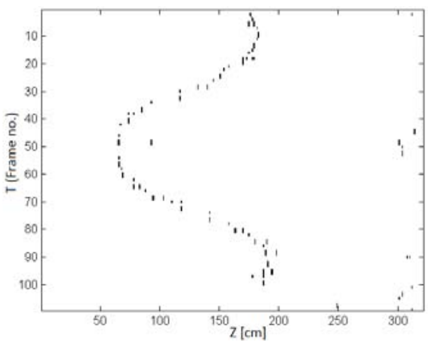

Figure 7. ZT Plane for “Stop” gesture

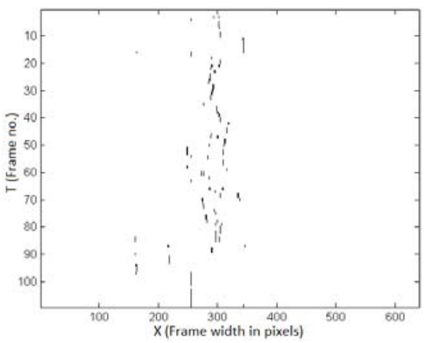

Figure 8. XT Plane for “Stop” gesture

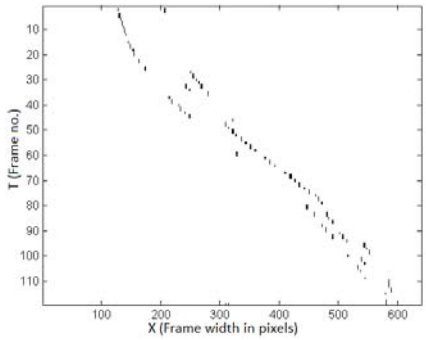

Figure 9. XT Plane for “left-to-right” gesture

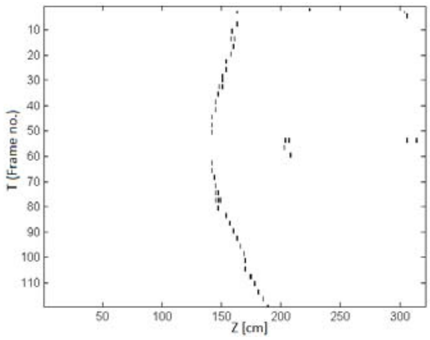

Figure 10. ZT Plane for “left-to-right” gesture

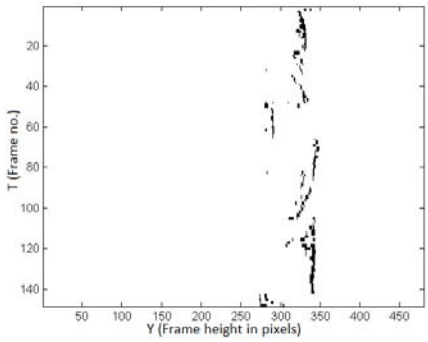

Figure 11. YT Plane for “right-to-left” gesture

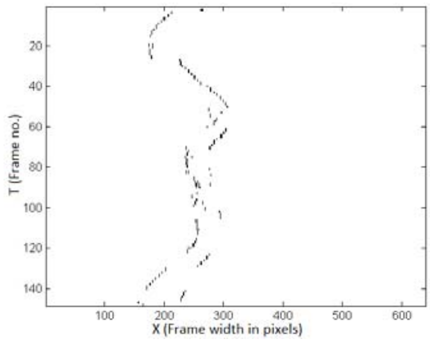

Figure 12. XT Plane for a random gesture

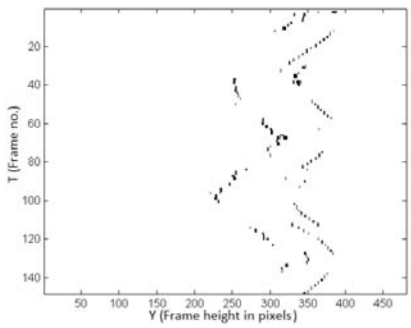

Figure 13. YT Plane for a random gesture

trajectories for random gestures, which are very fragmented.

In Table I we show, for each gesture, some numerical examples for the recognition parameters that we will use.

TABLE I. θ AND H VALUES PER GESTURE

Gesture type	θ_{XT}	θ_{YT}	θ_{ZT}	H_X	H_Y	H_Z
Stop	-2	-9	**-73**	1.28	0.68	**1.51**
Left-to-Right	**-81**	-6	-2	**1.81**	0.76	1.14
Right-to-Left	**81**	-32	32	**1.71**	1.26	0.9
Up-Down	-8	**-82**	30	1.59	**2.08**	1.69
Down-Up	6	**80**	-27	1.53	**1.95**	2.21
Random	**-77**	**75**	10	**0.86**	**0.64**	1.22

Looking at the values obtained for the angle θ in Table I, one can conclude that a decision based on the angle θ should be possible. Hence one can decide if the motion is parallel to the Z axis (like in the Stop gesture), or to the X axis (Left-to-right and Right-to-left gestures), or to the Y axis (Up-down or Down-up gestures). To take the decision, one has to check if the absolute value of θ is greater than a threshold, let's say 60 degrees (this threshold value was determined experimentally). This rule is valid if the system is used to recognize one of the 5 pre-defined gestures, because each gesture consists of a mainly linear motion along one axis (X, Y or Z), and theoretically no motion on the other axes. However, users would usually not perform gestures perfectly parallel to X, Y or Z axis. Hence, angles slightly greater than 60° may appear on axes different than the gesture's axis. To cope with this, we considered also the normalized maximum peak value from the Hough transform accumulator space (multiplied by a factor of 10). We have chosen this multiplication factor experimentally in order to work with a threshold of 1 instead of 0.1.

These are the values HX, HY and HZ shown in Table I. As expected, when a set of points is closer to a line, the peak in the accumulator space increases. We used these values in order to validate an angle θ. For instance, if $|\theta_{XT}| > 60°$ and HX > 1, then the gesture is parallel to the X axis.

In order to discriminate between the pairs of gestures parallel to the X axis or to the Y axis, we only have to look at the sign of the maximum value of the angle θ. For instance if we consider the X axis, if θ is positive, then the motion is from right to left (the origin is the upper-left corner of the image), and if θ is negative, then the motion is from left to right. The alternating sign obtained for θ when the Stop gesture is performed is due to the fact that we recorded the motion as a go and return motion, and the Hough transform simply takes the most important of the two lines. Even though the Hough transform can detect multiple lines, we decided not to use this property for the Stop gesture in order to keep the algorithm as simple as possible.

A gesture is classified as random if we cannot classify it among one of the 5 gestures, i.e. if we don't have only one pair $|\theta| > 60°$ and the corresponding $H > 1$.

When looking at the values in Table II, one can notice that, if we use the same threshold (60°), then "linear motion" will be detected on two (or perhaps all three) axes, or all angles will be below the threshold (see line 4 of the table). Hence, one can say that the system will recognize a non-defined gesture, by the fact that not only one angle θ is larger than the threshold (as before).

TABLE II. MEAN ABSOLUTE VALUES

Gesture type	θ_{XT}	θ_{YT}	θ_{ZT}	H_X	H_Y	H_Z
Stop	38	27	**69**	1.10	0.97	**1.44**
Left-to-Right	**80**	28	32	**1.84**	0.81	1.39
Right-to-Left	**81**	39	33	**1.89**	0.87	1.49
Up-Down	12	**78**	15	1.33	**1.71**	1.40
Down-Up	19	**76**	15	1.33	**1.67**	1.51
Random	**79**	**82**	29	**1.01**	**1.04**	1.21

Form the values of Table III one can say that there is no confusion between the 5 defined gestures since all 5 can only be confused with a random gesture. For instance, an up-down gesture that has been performed very slowly was classified as random because θ_{YT} was -55.

TABLE III. CONFUSION MATRIX

Gesture type	Stop	L-R	R-L	U-D	D-U	RND
Stop	**47**	0	0	0	0	3
Left-to-right	0	**48**	0	0	0	2
Right-to-left	0	0	**48**	0	0	2
Up-down	0	0	0	**49**	0	1
Down-up	0	0	0	0	**49**	1
Random	3	3	2	0	0	**42**

Table IV shows the recognition rate for each gesture type. The recognition rate of a random gesture is low because random gestures can contain hand trajectories similar to one or more of the defined gestures. However, the high recognition rates obtained for the 5 intended gestures validate the choice of the thresholds used in the decision tree.

TABLE IV. RECOGNITION RATE PER GESTURE

Gesture type	Stop	L-R	R-L	U-D	D-U	RND
Recognition rate (%)	94	96	96	98	98	84

IV. CONCLUSION

A method for the recognition of dynamic hand gestures using a 3D Kinect camera has been presented. The hands are detected by using only frame differences and hand trajectories are then estimated based on XT, YT and ZT projections by using the Hough transform, which is applied on these planes. The transform yields six parameters (θ_{XT}, θ_{YT}, θ_{ZT}, H_X, H_Y and H_Z), which are then used in a decision tree to recognize the gestures. The confusion matrix shows that there is no confusion between the 5 defined gestures (a defined gesture can only be miss-classified as random gesture if for instance it is performed to slowly). The rather low recognition rate obtained for the random gestures is due to the fact that often random gestures contain trajectory segments similar to one or more trajectories of the defined gestures.

The number of consecutive frames needed to recognize a gesture was around 100 (about 3 seconds at 30 fps). This number was determined experimentally for gestures performed at "normal speed" (not too fast or too slow).

However, the recognition rate does not decrease if, for instance, only 60 frames are used (the gesture should be performed in about 2 seconds). However, the hardware limitation of 30 fps exists, and, for fast motions, this limitation leads to a small number of points for a trajectory. Nevertheless, although simple and of low computational complexity, the proposed algorithm is robust and accurate.

As possible further work, a generalization of the approach is possible, in order to include circular or elliptical hand gestures. This extension is possible because the Hough transform can be applied also for circles and ellipses.

As for algorithm portability, the code can be very easily adapted to any ToF camera available on the market, for instance for SR4000.

REFERENCES

[1] A. Kolb, E. Barth, R. Koch, R. Larsen, “Time-of-Flight Cameras in Computer Graphics,” In Computer Graphics Forum, 29(1), pp 141-159, 2010 [Online]. Available: http://dx.doi.org/10.1111/j.1467-8659.2009.01583.x

[2] X. Liu and K. Fujimura, “Hand gesture recognition using depth data,” Proc. of the 6th IEEE international conf. on Automatic face and gesture recognition (FGR' 04), Washington, DC, USA, 529-534, 2004.

[3] S. Mitra and T. Acharya, “Gesture Recognition: A Survey,” IEEE Trans. on Syst., man, and cybernetics, Part C: Applications and Reviews, pp. 311-324, vol. 37, no. 3, may 2007 [Online]. Available: http://dx.doi.org/10.1109/TSMCC.2007.893280

[4] M. B. Holte, T. B. Moeslund, and P. Fihl, “View-invariant gesture recognition using 3D optical flow and harmonic motion context,” Comput. Vis. Image Underst. 114, 12, pp. 1353-1361, 2010. [Online]. Available: http://dx.doi.org/10.1016/j.cviu.2010.07.012

[5] P. Doliotis, A. Stefan, C. McMurrough, D. Eckhard, and V. Athitsos, “Comparing gesture recognition accuracy using color and depth information,” in Proceedings of PETRA, pp. 20:1-20:7, 2011. [Online]. Available: http://dx.doi.org/10.1145/2141622.2141647

[6] C. Keskin, A. T. Cemgil, and L. Akarun, “DTW Based Clustering to Improve Hand Gesture Recognition,” in Proceedings of HBU’11, pp. 72-81, Amsterdam, 2011. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-25446-8_8

[7] L. Gallo, A.P. Placitelli, and M. Ciampi, “Controller-free exploration of medical image data: experiencing the Kinect,” Proc. of. 24th IEEE CMBS'11, Piscataway, NJ, USA, 2011. [Online]. Available: http://dx.doi.org/10.1109/CBMS.2011.5999138

[8] S. Soutschek, J. Penne and J. Hornegger, 3D gesture-based scene navigation in medical imaging applications using time-of-flight cameras, IEEE Conf. on Computer Vision & Pattern Recogn., Workshop on ToF-Camera based Computer Vision (2008).

[9] P. Yanik et al., “Use of Kinect Depth Data and Growing Neural Gas for Gesture Based Robot Control,” in Proc. of PervaSense, pp. 283-290, 2012. [Online]. Available: http://dx.doi.org/10.4108/icst.pervasivehealth.2012.248610

[10] Prodan, R.-C., Pentiuc, S.-G., Vatavu, R.-D., “An Efficient Solution for Hand Gesture Recognition from Video Sequence,” Advances in Electrical and Computer Engineering, vol. 12, no. 3, pp. 85-88, 2012, [Online]. Available: http://dx.doi.org/10.4316/AECE.2012.03013

[11] K. Lai, J. Konrad, and P. Ishwar, “A gesture-driven computer interface using Kinect camera,” in Proc. Southwest Symposium on Image Analysis and Interpretation, Apr. 2012.

[12] Q. Munib, M. Habeeb, B. Takruri and H. A. Al-Malik, “American sign language (ASL) recognition based on Hough transform and neural networks,” Expert Systems with Applications, vol. 32, 1, pp. 24-37, 2007. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2005.11.018

[13] O. Altun, S. Albayrak, “Turkish fingerspelling recognition system using Generalized Hough Transform, interest regions, and local descriptors,” Patt. Rec. Letters, vol. 32, 13, pp. 1626-1632, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.patrec.2011.06.010

[14] N.N. Bhat, “Real time robust hand gesture recognition and visual servoing,” India Conference (INDICON), Annual IEEE, pp.1153-1157, 7-9 December 2012 [Online]. Available: http://dx.doi.org/10.1109/INDCON.2012.6420791