Human-robot interaction with depth-based gesture recognition

Gabriele Pozzato, Stefano Michieletto
and Emanuele Menegatti
IAS-Lab
University of Padova
Padova, Italy
Email: pozzato.g@gmail.com,
{michieletto, emg}@dei.unipd.it

Fabio Dominio, Giulio Marin, Ludovico Minto
Simone Milani and Pietro Zanuttigh
LTTM-Lab
University of Padova
Padova, Italy
Email: {dominiof,maringiu,mintolud,
simone.milani,zanuttigh}@dei.unipd.it

***Abstract*—The recent introduction of novel acquisition devices like the Leap Motion and the Kinect allows to obtain a very informative description of the hand pose that can be exploited for accurate gesture recognition. This paper proposes a novel hand gesture recognition scheme explicitly targeted to Leap Motion data. An ad-hoc feature set based on the positions and orientation of the fingertips is computed and fed into a multi-class SVM classifier in order to recognize the performed gestures. A set of features is also extracted from the depth computed from the Kinect and combined with the Leap Motion ones in order to improve the recognition performance. Experimental results present a comparison between the accuracy that can be obtained from the two devices on a subset of the American Manual Alphabet and show how, by combining the two features sets, it is possible to achieve a very high accuracy in real-time.**

I. INTRODUCTION

Human-Robot Interaction (HRI) is a field of investigation that is recently gaining particular interest, involving different research areas from engineering to psychology [1]. The development of robotics, artificial intelligence and human-computer interaction systems has brought to the attention of researchers new relevant issues, such as understanding how people perceive the interaction with robots [2], how robots should look like [3] and in what manners they could be useful [4].

In this work we focused on the communication interface between humans and robots. The aim is to apply novel approaches for gesture recognition based on the analysis of depth information in order to allow users to naturally interact with robots [5]. This is a significant and sometimes undervalued perspective for human-robot interaction.

Hand gesture recognition using vision-based approaches is an intriguing topic that has attracted a large interest in the last years [6]. It can be applied in many different fields including robotics and in particular human-robot interaction. A human-to-machine communication can be established by using a predefined set of poses or gestures recognized by a vision system. It is possible to associate a precise meaning (or a command the robot will perform) to each sign. This permits making the interplay with robots more intuitive and enhancing the human-robot interaction up to the level of a human-human one. In this case also the robot plays a different role. During social interactions it must have an active part in order to really engage the human user. For example, recognizing body language, a robot can understand the mood of the human user adapting its behavior to it [7].

Until a few years ago, most approaches were based on videos acquired by standard cameras [6], [8]; however, relying on a simple 2D representation is a great limitation since that representation is not always sufficient to capture the complex movements and inter-occlusions of the hand shape.

The recent introduction of low cost consumer depth cameras, e.g., Time-Of-Flight (ToF) cameras and MS Kinect™ [9], has opened the way to a new family of algorithms that exploit the depth information for hand gesture recognition. The improved accuracy in modelling three-dimensional body poses has permitted removing the need for physical devices in human-robot interaction such as gloves, keyboards or other controllers.

HRI system based on depth cameras apply machine-learning techniques to a set of relevant features extracted from the depth data. The approach in [10] computes a descriptor from silhouettes and cell occupancy features; this is then fed to a classifier based on action graphs. Both the approaches in [11] and [12] use volumetric shape descriptors processed by a classifier based on Support Vector Machines (SVM). Another possibility is to exploit the histograms of the distance of hand edge points from the hand center in order to recognize the gestures [13], [14], [15]. Different types of features can also be combined together [16].

The paper is organized in the following way. Sections II presents the proposed hand gesture recognition strategy, while Section III describes the body-pose detection algorithm. These tools are integrated in a human-robot interaction scheme, which is described in Section IV. A case study referring to a simple rock-paper-scissors game is presented in Section V. Finally Section VI draws the

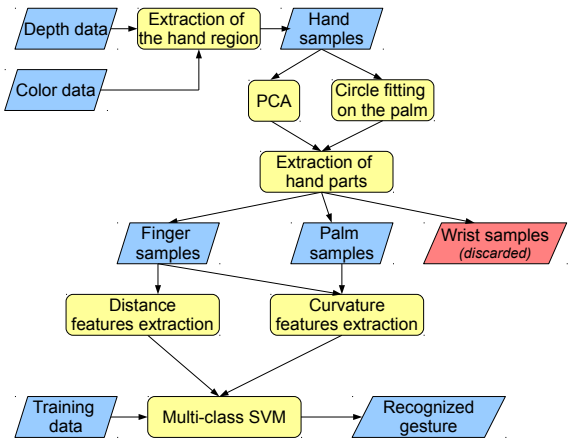


Fig. 1: Pipeline of the proposed approach.

conclusions.

II. HAND GESTURE RECOGNITION FROM DEPTH DATA

Fig. 1 shows a general overview of the employed hand gesture recognition approach. The approach is based on the method proposed in [16]. In the first step, the hand is extracted from the depth acquired by the Kinect. Then, a set of relevant features is extracted from the hand shape. Finally a multi-class Support Vector Machine classifier is applied to the extracted features in order to recognize the performed gesture.

A. Hand Recognition

In the first step the hand is extracted from the color and depth data provided by the Kinect. Calibration information is used to obtain a set of 3D points $\mathbf{X_i}$ from the depth data. The analysis starts by extracting the point in the depth map closest to the user ($\mathbf{X_c}$). The points that have a depth value close to the one of $\mathbf{X_c}$ and a 3D distance from it within a threshold of about $20[cm]$ are extracted and used as a first estimate of the candidate hand region. A further check on hand color and size is then performed to avoid to recognize possible objects positioned at the same distance as the hand. At this point the highest density region is found by applying a Gaussian filter with a large standard deviation to the mask representing the extracted points. Finally, starting from the highest density point, a circle is fitted on the hand mask such that its area roughly approximates the area corresponding to the palm (see Fig. 2). Principal Component Analysis (PCA) is then applied to the detected points in order to obtain an estimate of the hand orientation. Finally the hand points are divided into the palm region (set $\mathcal{P}$, i.e., the points inside the circle), the fingers (set $\mathcal{F}$, i.e., the points outside the circle in the direction of the axis pointing to the fingertips found by the PCA), and the wrist region (set $\mathcal{W}$, i.e., the points outside the circle in the opposite direction w.r.t. the fingertips).

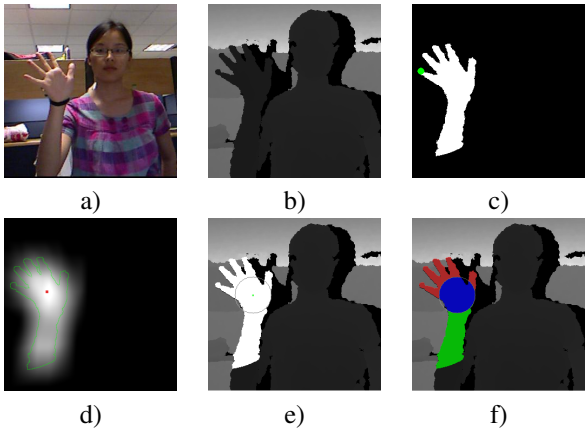


Fig. 2: Extraction of the hand: a) Acquired color image; b) Acquired depth map; c) Extracted hand samples (the closest sample is depicted in green); d) Output of the Gaussian filter applied on the mask corresponding to $\mathcal{H}$ with the maximum highlighted in red; e) Circle fitted on the hand; f) Palm (blue), finger (red) and wrist (green) regions subdivision.

B. Feature Extraction

Two different types of features are computed from data extracted in the previous step. The first set of features consists of a histogram of the distances of the hand points from the hand center. Basically, the center of the circle is used as reference point and an histogram is built, as described in [11], by considering for each angular direction the maximum of the point distances from the center, i.e.:

$$L(\theta_q) = \max_{\mathbf{X}_i \in \mathcal{I}(\theta_q)} d_{\mathbf{X}_i} \tag{1}$$

where $\mathcal{I}(\theta_q)$ is the angular sector of the hand corresponding to the direction θ_q and $d_{\mathbf{X}_i}$ is the distance between point X_i and the hand center. The computed histogram is compared with a set of reference histograms $L^r_g(\theta)$, one for each gesture g. The maximum of the correlation between the current histogram $L(\theta_q)$ and a shifted version of the reference histogram $L^r_g(\theta)$ is used in order to compute the precise hand orientation and refine the results of the PCA. A set of angular regions $I(\theta_{g,j})$ associated to each finger $j = 1, ..., 5$ in the gesture g is defined on the histogram, and the maximum is selected as feature value inside each region, normalized by the middle finger's length:

$$f^d_{g,j} = \max_{I(\theta_{g,j})} \frac{L_g(\theta)}{L_{max}} \tag{2}$$

where $g = 1, ..., G$ for an alphabet of G gestures. Note that the computation is performed for each of the candidate gesture, thus obtaining a different feature value for each of the candidate gestures.

The second feature set is based on the curvature of the hand contour. This descriptor is based on a multi-scale integral operator [17], [18] and is computed as described in

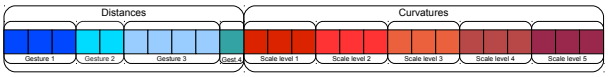


Fig. 3: Feature vectors extracted from the two devices.

[16] and [15]. The algorithm takes as input the edges of the palm and fingers regions and the binary mask representing the hand region on the depth map. A set of circular masks with increasing radius centered on each edge sample is built (we used $S = 25$ masks with radius varying from 0.5 cm to 5 cm, notice that the radius corresponds to the scale level at which the computation is performed).

The ratio between the number of samples falling inside the hand shape and the size of the mask is computed for each circular mask. The values of the ratios represent the local curvature of the edge and range from 0 for a convex section of the edge to 1 for a concave one, with 0.5 corresponding to a straight edge. The $[0, 1]$ interval is then quantized into N bins and the feature values $f^c_{b,s}$ are computed by counting the number of edge samples having a curvature value inside bin b at scale level s.

The curvature values are finally normalized by the number of edge samples and the feature vector $\mathbf{F}^c$ with $B \times S$ entries is built. The multi-scale descriptor is made of $B \times S$ entries $C_i, i = 1, ..., B \times S$, where B is the number of bins and S is the number of employed scale levels.

C. Hand Gesture Classification

The feature extraction approach provides two feature vectors, each one describing relevant properties of the hand samples. In order to recognize the performed gestures, the two vectors are concatenated and sent to a multi-class Support Vector Machine classifier. The target is to assign the feature vectors to G classes (with $G = 3$ for the sample rock-scissor-paper game of the experimental results) corresponding to the various gestures of the considered database. A multi-class SVM classifier based on the *one-against-one* approach has been used, i.e., a set of $G(G-1)/2$ binary SVM classifiers are used to test each class against each other and each output is chosen as a *vote* for a certain gesture. The gesture with the maximum number of votes is selected as the output of the classifier. In particular we used the SVM implementation in the LIBSVM package [19], together with a non-linear Gaussian Radial Basis Function (RBF) kernel tuned by means of grid search and cross-validation on a sample training set.

III. BODY GESTURE RECOGNITION FROM SKELETAL TRACKING

In this work skeletal tracking capabilities come from a software development kit released by OpenNI. In particular, the skeletal tracking algorithm is implemented in a freeware middleware called NITE built on top of the OpenNI SDK.

NITE uses the information provided by a RGB-D sensor to estimate the position of several joints of the human body at a frame rate of 30 fps that is a good trade-off between speed and precision. Differently from approaches based only on RGB images, these kind of sensors based also on depth data, allow the tracker to be more robust with respect to illumination changes.

The skeleton information provided by the NITE middleware consists of $N = 15$ joints from *head* to *foot*. Each joint is described by its position (a point in 3D space) and orientation (using quaternions). On these data, we perform two kinds of normalization: firstly we scale the joints positions in order to normalize the skeleton size, thus achieving invariance among different people, then we normalize every feature to zero mean and unit variance. Starting from the normalized data, we extract three kinds of descriptors: a first skeleton descriptor ($\mathbf{d}_P$) is made of the set of joints positions concatenated one to each other, the second one ($\mathbf{d}_O$) contains the normalized joints orientations, finally, we considered also the concatenation of both positions and orientations ($\mathbf{d}_{TOT}$) of each joint:

$$\mathbf{d}_P = [x_1 \ y_1 \ z_1 \ \dots \ x_N \ y_N \ z_N], \tag{3}$$

$$\mathbf{d}_O = \left[q_1^1 \ q_1^2 \ q_1^3 \ q_1^4 \ \dots \ q_N^1 \ q_N^2 \ q_N^3 \ q_N^4\right], \tag{4}$$

$$\mathbf{d}_{TOT} = \left[\mathbf{d}_P^1 \ \mathbf{d}_O^1 \ \dots \ \mathbf{d}_P^N \ \mathbf{d}_O^N\right]. \tag{5}$$

These descriptors have been provided as features for the classification process. A Gaussian Mixture Model (GMM)-based classifier has been used in order to compare the set of predefined body gestures to the ones performed by the user. The described technique could be extended by considering a sequence of descriptors equally spaced in time in order to apply the resulting features to human activity recognition [20], [21] and broaden the interaction capabilities of our framework.

IV. ROBOT CONTROL AND SYSTEM INTEGRATION

The information provided by both hand and skeletal gesture recognition systems are analyzed in order to obtain a proper robot motion. Robot behavior really depends on the application, but the complete framework can be separated in different layers in order to easily extend the basic system to new tasks. We created three independent layers: *gesture recognition*, *decision making*, *robot controller*. These layers are connected one to each other by means of a very diffuse robotic framework: Robot Operating System.

Robot Operating System (ROS) [22] is an open-source, meta-operating system that provides, together with standard services peculiar to an operating system (e.g., hardware abstraction, low-level device control, message-passing between processes, package management), tools and libraries useful for typical robotics applications (e.g., navigation, motion planning, image and 3D data processing). The primary goal of ROS is to support the reuse of code in robotics research and, therefore, it presents a simplified but lightweight design with clean functional interfaces.

During a human-robot interaction task, the three layers play different roles. The *gesture recognition* layer deals

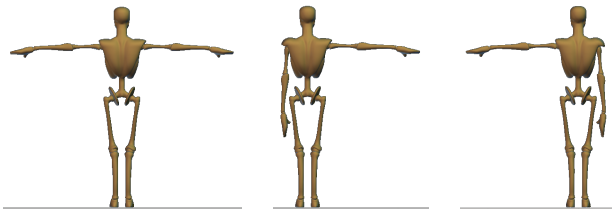
with the acquisition of information from a camera in order to recognize gestures performed by users. The *decision making* layer is on a higher level of abstraction. Information is processed by ROS and classified by an AI algorithm in order to make the robot take its decision. Finally, the *robot controller* layer is responsible of motion. This layer physically modifies the behavior of the robot sending commands to the servomotors. Even though they are strictly independent also in what concerns implementation, the three levels can interact sending or receiving messages through ROS. This means that we can use different robots, AI algorithms or vision systems by maintaining the same interface.

V. CASE STUDY: A SIMPLE ROCK-PAPER-SCISSORS GAME

In order to test the developed framework a very simple even popular game has been used, namely rock-paper-scissors. Users are asked to play against a robotic opponent on several rounds. At each round, both the players (human and robot) have to perform one of the 3 gestures depicted in Fig. 4. The winner is chosen according to the well-known rules of the game. Game settings and moves can be selected by human users without any remote controls or keyboards: only the *gesture recognition* system is involved, so that the interaction will result more natural and user-friendly as possible.

Rock Paper Scissors

Fig. 4: Sample depth images for the 3 gestures.

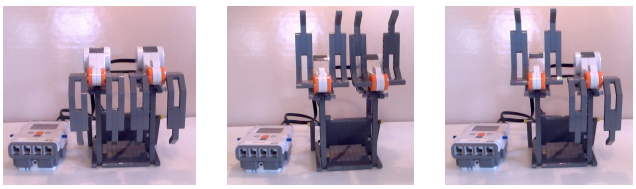


a) b) c)

Fig. 5: Poses: a) corresponds to 3 games, b) to 2 games and c) to 1 game. Each game is composed by three rounds.

Users are able to select the number of games to play by assuming one of the pose shown in Fig. 5 once they are recognized by the system. To each position corresponds a number varying from 1 to 3. Each game is composed by three rounds. Once a round is started, the hand gesture recognition algorithm looks at the move played by the human. At the same time, the *decision making* system chooses the robot gesture by using an Artificial Intelligence (AI) algorithm we developed [23]. The algorithm is based on the use of Gaussian Mixture Model (GMM). The idea under this approach is using a history of three previous rounds in order to forecast the next human move. This is a very important aspect in the natural interaction between human and robot, in fact, people are stimulated in confronting with a skilled opponent. Finally, the *robot controller* translates high level commands to motor movements in order to make a LEGO Mindstorms NXT robotic platform perform the proper gesture (Fig. 6).

Notice that the approach of Section II, that has been developed for more complex settings with a larger number of gestures, has very good performances in this sample application. In order to evaluate its effectiveness we asked to 14 different people to perform the 3 gestures 10 times each for a total of $14 \times 10 \times 3 = 420$ different executions of the gestures. Then we performed a set of tests, each time training the system on 13 people and leaving out one person for the testing. In this way (similar to the *cross-validation* approach) the performances are evaluated for each person without any training from gestures performed by the same tested user, as it will happen in a real setting where a new user starts interacting with the robot. The proposed approach has achieved an average accuracy of 99.28% with this testing protocol.



a) b) c)

Fig. 6: LEGO Mindstorms NXT performing the three gestures: a) rock, b) paper and c) scissors.

VI. CONCLUSIONS

In this paper a framework for gesture-based human-robot interaction has been proposed. Two different approaches for the exploitation of depth data from low-cost cameras have been proposed for both the recognition of hand and full-body gestures. Depth-based gesture recognition schemes allow a more natural interaction with the robot and a sample application based on the simple rock-scissor-paper game has been presented. Further research will be devoted to the application of the proposed framework in more complex scenarios.

ACKNOWLEDGEMENTS

Thanks to the Seed project *Robotic 3D video (R3D)* of the Department of Information Engineering and to the University project *3D Touch-less Interaction* of the University of Padova for supporting this work.

REFERENCES

[1] C. D. Kidd, "Human-robot interaction: Recent experiments and future work," in *Invited paper and talk at the University of Pennsylvania Department of Communications Digital Media Conference*, 2003, pp. 165–172.

[2] C. D. Kidd and C. Breazeal, "Effect of a robot on user perceptions," in *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 4. IEEE, 2004, pp. 3559–3564.

[3] C. Breazeal and B. Scassellati, "How to build robots that make friends and influence people," in *Intelligent Robots and Systems, 1999. IROS'99. Proceedings. 1999 IEEE/RSJ International Conference on*, vol. 2. IEEE, 1999, pp. 858–863.

[4] R. Gockley, A. Bruce, J. Forlizzi, M. Michalowski, A. Mundell, S. Rosenthal, B. Sellner, R. Simmons, K. Snipes, A. C. Schultz *et al.*, "Designing robots for long-term social interaction," in *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*. IEEE, 2005, pp. 1338–1343.

[5] J. P. Wachs, M. Kölsch, H. Stern, and Y. Edan, "Vision-based hand-gesture applications," *Communications of the ACM*, vol. 54, no. 2, pp. 60–71, 2011.

[6] J. P. Wachs, M. Kölsch, H. Stern, and Y. Edan, "Vision-based hand-gesture applications," *Commun. ACM*, vol. 54, no. 2, pp. 60–71, Feb. 2011.

[7] D. McColl and G. Nejat, "Affect detection from body language during social hri," in *RO-MAN, 2012 IEEE*. IEEE, 2012, pp. 1013–1018.

[8] D. Kosmopoulos, A. Doulamis, and N. Doulamis, "Gesture-based video summarization," in *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, vol. 3, 2005, pp. III–1220–3.

[9] C. Dal Mutto, P. Zanuttigh, and G. M. Cortelazzo, *Time-of-Flight Cameras and Microsoft Kinect*, ser. SpringerBriefs in Electrical and Computer Engineering. Springer, 2012.

[10] A. Kurakin, Z. Zhang, and Z. Liu, "A real-time system for dynamic hand gesture recognition with a depth sensor," in *Proc. of EUSIPCO*, 2012.

[11] P. Suryanarayan, A. Subramanian, and D. Mandalapu, "Dynamic hand pose recognition using depth data," in *Proc. of ICPR*, aug. 2010, pp. 3105 –3108.

[12] J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu, "Robust 3d action recognition with random occupancy patterns," in *Proc. of ECCV*, 2012.

[13] Z. Ren, J. Yuan, and Z. Zhang, "Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera," in *Proc. of ACM Conference on Multimedia*. ACM, 2011, pp. 1093–1096.

[14] Z. Ren, J. Meng, and J. Yuan, "Depth camera based hand gesture recognition and its applications in human-computer-interaction," in *Proc. of ICICS*, 2011, pp. 1 –5.

[15] F. Dominio, M. Donadeo, G. Marin, P. Zanuttigh, and G. M. Cortelazzo, "Hand gesture recognition with depth data," in *Proceedings of the 4th ACM/IEEE international workshop on Analysis and retrieval of tracked events and motion in imagery stream*. ACM, 2013, pp. 9–16.

[16] F. Dominio, M. Donadeo, and P. Zanuttigh, "Combining multiple depth-based descriptors for hand gesture recognition," *Pattern Recognition Letters*, 2013.

[17] S. Manay, D. Cremers, B.-W. Hong, A. Yezzi, and S. Soatto, "Integral invariants for shape matching," *IEEE Trans. on PAMI*, vol. 28, no. 10, pp. 1602 –1618, 2006.

[18] N. Kumar, P. N. Belhumeur, A. Biswas, D. W. Jacobs, W. J. Kress, I. Lopez, and J. V. B. Soares, "Leafsnap: A computer vision system for automatic plant species identification," in *Proc. of ECCV*, October 2012.

[19] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.

[20] M. Munaro, G. Ballin, S. Michieletto, and E. Menegatti, "3d flow estimation for human action recognition from colored point clouds," *Biologically Inspired Cognitive Architectures*, 2013.

[21] M. Munaro, S. Michieletto, and E. Menegatti, "An evaluation of 3d motion flow and 3d pose estimation for human action recognition," in *RSS Workshops: RGB-D: Advanced Reasoning with Depth Cameras.*, 2013.

[22] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, 2009.

[23] G. Pozzato, S. Michieletto, and E. Menegatti, "Towards smart robots: rock-paper-scissors gaming versus human players," in *Proceedings of the Workshop Popularize Artificial Intelligence (PAI2013)*, 2013, pp. 89–95.