Real-time Hand Pose Tracking and Classification for Natural Human-Robot Control

Bruno Lima[1], Givanildo L. N. Júnior[1], Lucas Amaral[1], Thales Vieira[2], Bruno Ferreira[1] and Tiago Vieira[1]
[1]*Institute of Computing, Federal University of Alagoas, Maceió, Brazil*
[2]*Institute of Mathematics, Federal University of Alagoas, Maceió, Brazil*

Keywords: Human-robot Interaction, Deep Learning, Convolutional Neural Networks, Skeleton Tracking.

Abstract: We present a human-robot natural interaction approach based on teleoperation through body gestures. More specifically, we propose an interface where the user can use his hand to intuitively control the position and status (open/closed) of a robotic arm gripper. In this work, we employ a 6-DOF (six degrees-of-freedom) industrial manipulator which mimics user movements in real-time, positioning the end effector as if the individual was looking into a mirror, entailing a natural and intuitive interface. The controlling hand of the user is tracked using body skeletons acquired from a Microsoft Kinect sensor, while a Convolutional Neural Network recognizes whether the hand is opened or closed using depth data. The network was trained on hand images collected from several individuals, in different orientations, resulting in a robust classifier that performs well regardless of user location or orientation. There is no need for wearable devices, such as gloves or wristbands. We present results of experiments that reveal high performance of the proposed approach to recognize both the user hand position and its status (open/closed); and experiments to demonstrate the robustness and applicability of the proposed approach to industrial tasks.

1 INTRODUCTION

Recent advances in artificial intelligence have allowed the scientific community to research and develop innovative natural human-machine interfaces. Emerging deep learning methods have allowed people to interact with virtual assistants, robotic pets and Natural User Interfaces (NUI) through voice commands and gestures (Schmidhuber, 2015).

Natural User Interfaces (NUI) are required to be easy (highly learnable), ergonomic and non-intrusive (LaViola Jr et al., 2017). In particular, clothing requirements and counter-intuitive commands are undesired, since it should feel comfortable for a good experience. These requirements are even more critical for applications in robotics control, where accuracy is crucial and latency is not desirable.

Human body gestures have been adopted in the last years for NUI as they potentially satisfy such requirements (Miranda et al., 2012; LaViola Jr et al., 2017). However, the correct identification and representation of body parts and movements in real-time is still a topic of great interest from different areas (Cheng et al., 2016).

We propose a user-friendly and intuitive Natural User Interface (NUI), where a 6-Degrees-of-Freedom (6-DOF) robot arm end-effector is controlled by mirroring the hand of the user. The user hand (position and image) is extracted from a body skeleton model obtained from a Microsoft Kinect. Such data is used to control both: (1) the robot gripper position, after an appropriate invariant encoding/decoding, and (2) its state (open or closed) according to the respective user hand state.

To classify hand poses, we use Convolutional Neural Networks (CNNs) trained by small hand depth images, which are cropped from the larger full body depth image ensuring translational and scale invariance, by centering on the projection of the hand position with a depth-dependent radius. Thus, both skeleton and depth data are used by our hybrid approach, tested in real-time with an industrial-grade 6-DOF robot arm. Potential real-world applications that could affect user physical integrity, such as: (i) landslip terrains; (ii) compromised buildings; (iii) cable and gas tunnels; (iv) areas with explosives.

In summary, this paper has the following contributions:

- We propose the use of an invariant representation to mimic human moves to a robot arm in an intuitive fashion;

Lima, B., N. Júnior, G., Amaral, L., Vieira, T., Ferreira, B. and Vieira, T.
Real-time Hand Pose Tracking and Classification for Natural Human-Robot Control.
DOI: 10.5220/0007384608320839
In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2019)*, pages 832-839
ISBN: 978-989-758-354-4

- We collected a balanced dataset (publicly available) composed by $160,000$ depth images of open and closed hands from 20 different individuals;
- We present image processing algorithms to extract and pre-filter human hand depth images from large depth images. Such algorithms are applied to provide scale and background invariant images to the CNN;
- We show that CNNs are robust to recognize human hand poses from small depth images, achieving very high accuracy.
- We propose a filter based on temporal coherence to improve per-frame classification performance;
- We show results of experiments performed to validate the method in real-world tasks.

2 RELATED WORK

Authors in (Xiao et al., 2014) used information from the upper-body such as head, arm position and hand posture to implement a Human-Robot Interaction (HRI) system. Differently from our approach, which does not require any wearable device, they used an immersion glove (CyberGlove II[1]).

Kruse *et al.*(Kruse et al., 2015) used a Kinect sensor to track and map movements of both human arms to a robotic apparatus comprising a torso and two 7-DOF (Degrees-of-Freedom) manipulators. The purpose was to evaluate the system stability whilst supporting big objects using a closed-loop force control.

Kalman filters were employed by Du *et al.* (Du et al., 2014) on data from an Inertial Measurement Unit (IMU) device, along with a Kinect sensor, to determine translation and orientation of the human hand. They propose a method to tackle the cumulative error inherent to the inertial measuring unit. Our system, in contrast, maps the absolute position of the hand relative to the user's torso, therefore invariant to body translation/rotation.

Recently, deep learning models have shown outstanding results for image classification, and some works have been proposed to recognize hand poses from images (Tompson et al., 2014; Yuan et al., 2017; Sharp et al., 2015). Fine classification of subtle hand poses is possible by constraining the acquisition of hand images: the hand is near the sensor; and the space of allowed body poses is very limited. In contrast, we face the challenge of recognizing hand poses from varying body poses, hand orientations and distances to the sensor. It is worth noting that hand pose images acquired from variable distances contain hands at different scales and levels of detail, which we successfully handle in our approach.

[1]http://www.cyberglovesystems.com/

3 METHODOLOGY

Our solution addresses two main challenges: locating, tracking and mirroring the hand movements on the robot arm; and recognizing hand poses to send binary commands to the robot gripper. Both in real-time. As depicted in Fig. 1, we adopt a Kinect v1 (Microsoft, 2018) depth sensor to locate and capture depth images of a user at 30fps. These images are used in both training and recognition phases. Although more recent depth sensors can provide even better features for hand pose recognition, Kinect v1 was sufficient to achieve high accuracy in this work.

To perform hand location and tracking, we propose the use of a parametric approach based on user skeleton models (Shotton et al., 2011; OpenNI, 2018). To provide an intuitive interface to the user, we employ a representation of the hand position that is invariant to global translations and rotations of the his body. In addition, this representation can be employed to mirror movements in a robot arm.

The hand is located and cropped from the full body depth image using the skeleton's projection in 2d image space. The cropped image is then filtered before being stored into the dataset or used for inference. The hand pose recognition is carried out by a supervised learning approach: after collecting a large training set composed of hand images (open or closed), we train a Convolutional Neural Network (CNN) to act as a binary classifier. In what follows, we describe each step of the method in detail.

3.1 Depth Image Preprocessing

Depth sensors are capable of estimating their distance (or depth) to visible objects in a scene. A depth image is defined by a function $f : \mathcal{U} \subset \mathbb{R}^2 \to \mathbb{R}$, where $z = f(x,y)$ represents the distance from the sensor to the object that is projected at pixel (x,y). Using the official Kinect SDK (Microsoft, 2018), we extract, in real-time, a skeleton composed of relevant body joint positions, given in 3d Cartesian coordinates, including the controlling hand.

Let $p(t) \in \mathbb{R}^3$ be the controlling hand position given by the sensor at frame t, and $\tilde{p}(t) = (x(t), y(t))$ the projection of $p(t)$ in the image space. To extract the hand region $\mathcal{R} \subset \mathcal{U}$ from the depth image domain, we set the center pixel of $\mathcal{R}$ to $\tilde{p}$ and its dimensions

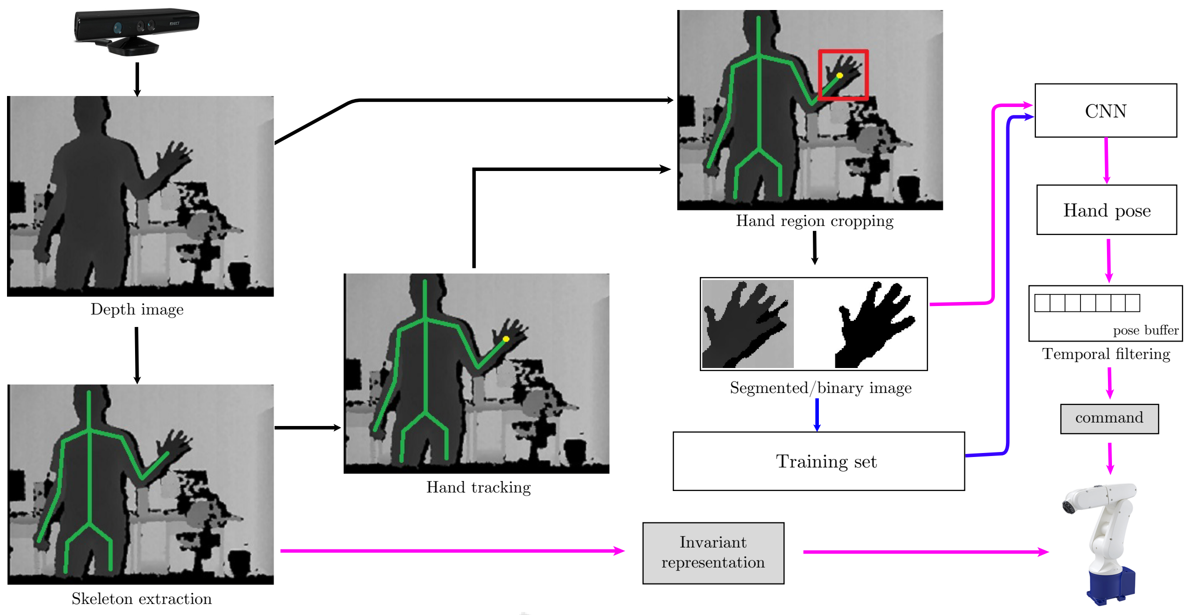


Figure 1: Method overview: a depth image is captured in real-time using a depth sensor (top-left); the skeleton is obtained (bottom-left) and used to locate the hand region (top-right); the hand is then cropped using the position $p(t)$ given by the skeleton; segmented or binarized images are computed and stored in a hand pose training set, which is later used to train a CNN (blue), or evaluated by a previously trained CNN that is capable of recognizing if the hand is opened or closed (pink). CNN classifications are sequentially stored in a pose buffer for temporal coherence filtering. Commands sent to the robot are highlighted in gray.

to $2L \times 2L$, where L is given in pixels. Since the distance from the user (and the controlling hand) to the sensor may vary over time, the hand size (or scale) in the image may also vary. Considering that depth images given by the Kinect sensor are generated by perspective projection, the hand size is inversely proportional to its distance to the sensor. Thus, we let L vary over time as a function of the hand distance, as

$$L(t) = \frac{K}{d(t)},$$

where K is a proportionality constant and $d(t)$ is the distance from $p(t)$ to the sensor in milimeters.

We empirically tuned K by analyzing several depth images from distinct users performing various poses. We searched for the smaller value of K whose generated images covered the whole hand projected area. By applying these criteria, we guarantee images that are small but still representative for the following phase of learning and recognition of hand poses. In this experiment, we found an optimum value of $K = 62$.

Let $f' : \mathcal{R} \subset \mathbb{R}^2 \to \mathbb{R}$ be the cropped hand image. This image may include, in addition to the hand, other background objects or even parts of the user body. To remove such unwanted data, we experimented two filtered representations extracted from f' for training and recognition of hand poses: segmented and binarized images. To extract the hand from the background, we consider that the hand is the closest object to the sensor and apply a depth thresholding filter, resulting in the segmented image given by

$$s(x,y) = \begin{cases} f'(x,y), & f'(x,y) \le D_{\min} + T \\ 0, & f'(x,y) > D_{\min} + T, \end{cases}$$

where $D_{\min} = \min_{\mathcal{R}}(f'(x,y))$, *i.e.* the depth of the closest hand pixel to the sensor; and T is a depth threshold used to extract the hand.

Alternatively, we compute binarized images from segmented images by

$$b(x,y) = \begin{cases} 1, & f'(x,y) \le D_{\min} + T \\ 0, & f'(x,y) > D_{\min} + T. \end{cases}$$

Both representations are experimented and compared.

3.2 Invariant Human-Robot Mimicking

In this section we focus on mapping the user body movement to the robot movement by using a local coordinate system that is invariant under rotations and translations of the body skeleton. We are interested in the relative position of the hand (which we consider to be the right hand) relative to the body. We use the following joints: right shoulder (j_{rs}); left shoulder (j_{ls}); shoulders center (j_{sc}); spine (j_{sp}); and right hand (j_{rh})

(*cf.* Fig. 2). We compute the coordinates of j_{rh} w.r.t a local coordinate system centered at joint j_{rs}.

The basis of the coordinate system must vary with body movement. We consider that the torso plane is the better reference for body movement, and compute a vertical unitary vector ($\hat{v}_t$) and a horizontal unitary vector, ($\hat{h}_t$) given by

$$\hat{v}_t = \frac{j_{sc} - j_{sp}}{\|j_{sc} - j_{sp}\|}, \quad \hat{h}_t = \frac{j_{ls} - j_{rs}}{\|j_{ls} - j_{rs}\|}.$$

The third vector of our basis is the normal vector of the torso plane ($\hat{n}_t$), given by the normalized cross product

$$\hat{n}_t = \frac{\hat{h}_t \times \hat{v}_t}{\|\hat{h}_t \times \hat{v}_t\|}.$$

The position of the right hand is initially given by $u = \overrightarrow{j_{rs}j_{rh}}$. Since users may have different arm lengths, we normalize u using the sum of the lengths of the arm and forearm vectors:

$$\tilde{u} = \frac{j_{rh} - j_{rs}}{\|j_{re} - j_{rs}\| + \|j_{rh} - j_{re}\|}.$$

Then, we change the basis of vector $\tilde{u}$ to the basis $\{\hat{n}_t, \hat{h}_t, \hat{v}_t\}$, obtaining the invariant coordinates (u_1, u_2, u_3). Finally, the goal position of the robot gripper is given by

$$P_G = \begin{bmatrix} \eta\ \omega_X\ u_1 \hat{X}_0 \\ \eta\ \omega_Y\ u_2 \hat{Y}_0 \\ \delta_z + \omega_Z\ u_3 \hat{Z}_0 \end{bmatrix}, \qquad (1)$$

where;

- $\{\hat{X}_0, \hat{Y}_0, \hat{Z}_0\}$ is the orthonormal basis of the robot coordinate system;
- ω_X, ω_Y and ω_Z are robot-dependent scale factors;
- δ_Z is an offset in the vertical direction to regulate the operation height;
- $\eta \in \{-1, 1\}$ represents an operation mode, which is $+1$ for direct operation, or -1 for mirror operation, in which case the axes on the horizontal plane of the robot are inverted.

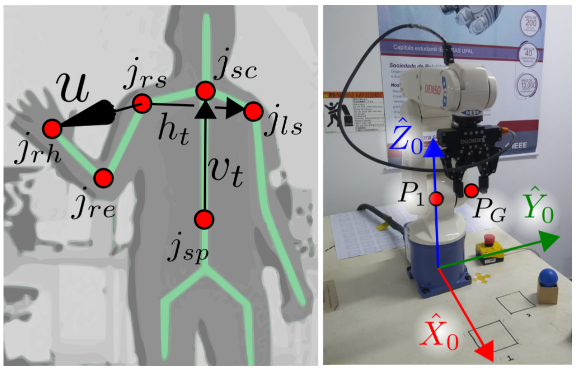


Figure 2: Example of mapping between body skeletons (left) and robot (right) coordinate systems: the right shoulder joint j_{rs} is mapped to the fixed point P_1 lying on the vertical axis $\hat{Z}_0$ of the robot coordinate system. The right hand joint is then mapped to the goal point P_G located at the tip of the gripper. Note that when the user moves or rotates w.r.t the scene keeping the same relative hand position to the body, the robot remains stationary.

3.3 Convolutional Neural Networks for Hand Pose Classification

To train our classifier, we collected a dataset comprising labeled hand images from both segmented and binarized representations. Then, we assessed which one performed better for classification.

Training and Classification. At the training stage, several examples of each hand class were feed as training data into a Convolutional Neural Network, generating a binary classification model. Then, in the classification stage, a user performs any of the trained hand poses, his hand image representation (segmented or binarized) is calculated and given as input to the trained CNN. Finally, the classifier predicts, in real-time, which class the image belongs to.

Architecture. The class of Convolutional Neural Networks for image classification includes models that: receive as input any kind of data structured as *n*-dimensional images; perform convolutions on such data in a few convolutional layers, possibly using max-pooling and dropout; flatten the resulting feature maps into an uni-dimensional array before feeding a few dense layers; and finally reach an output layer with a *softmax* activation function. Several hyper-parameters determine a unique network structure, such as the number of convolutional and dense layers, the convolutional filter sizes, and the number of units of each layer. To find the most appropriate combination of hyper-parameters for hand pose classification, we performed a grid search, evaluating several combinations of values. Fig. 3 shows the best network structure found in the experiments described in Section 4.2.

All experimented networks receive as input a single-channel image with fixed dimensions of 50 by 50 pixels. The input data is filtered by 1 or 2 convolutional layers (C), where each layer may apply 3×3 or 5×5 filters (F). For each convolutional layer, we use 8, 16 or 32 filters per layer (S). After each convolutional layer, we include a max-pooling layer with 2×2 receptive fields to downsample the feature maps.

The resulting features are then flattened into a unidimensional array that is given as input to dense lay-

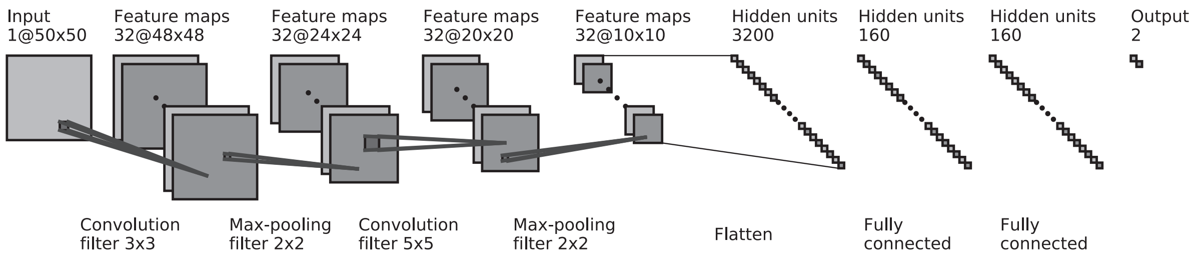


Figure 3: Architecture of the best CNN found in the experiments of Section 4.

ers. We experimented the use of 1 or 2 dense layers (U), with 80, 120 or 160 hidden units (N) each. In all convolutional and dense layers, we use the rectified linear (*ReLU*) as activation function. Finally, a *softmax* dense layer with 2 neurons outputs the final probability distribution. Note that, to generalize these networks for the classification of n poses, one can simply change the number of neurons of *softmax* layer to n. In the next Section, we present the results of the experiments, showing the best hyperparameters combinations found.

3.4 Temporal Filtering

CNN robustness may be decreased due to external factors such as: depth sensor noise; inaccurate skeleton tracking; incompatible user poses, such as keeping the hand occluded or near other objects in the scene. However, a robot may drop and break an object if, during a pick-and-place task, a closed hand is incorrectly classified as open.

To improve robustness in such situations, we propose to take advantage of temporal coherence. We consider that, in order to control a robot, the user must keep the same command along a time interval of reasonable duration. This contrasts with inaccurate hand pose classifications, which rarely occurs, considering that the CNN is robust. Thus, we employ a temporal filter that only changes command if the CNN changes its hand pose classification for at least Q consecutive frames. We set $Q = 15$, which results in half a second delay. However, this parameter should be tuned according to the nature of the application.

4 EXPERIMENTS

In this section we describe the collected dataset of hand poses images, made publicly available[2]. Then, we describe experiments to evaluate CNNs: we compare the performance using binarized and segmented image representations; we experiment several combinations of hyper-parameters to identify robust CNN configurations; and we evaluate the best CNN configuration with temporal filtering in a regular robot control task. Finally, to demonstrate the robustness and feasibility of our proposed interface in real-time robot control applications, we present results of experiments with inexperienced users.

4.1 Open-closed Depth Hand Dataset

We collected a large and diversified hand pose image dataset composed by 160,000 samples from 20 individuals (15 men and 5 women). Individuals performed the predefined hand poses: open and closed hand. From each subject, we collected depth images of the whole body, with 640×480 resolution. To improve diversity, we asked each individual to vary its global body pose as much as possible, and captured in environments with different backgrounds. From the depth images, we extracted and stored the cropped hand. It is worth mentioning that, as the distance from the hand to the sensor is variable, the cropped image resolution varies greatly. Consequently, we resampled all cropped images to the input resolution of the neural network before the training stage (the same occurring for real-time classification).

4.2 CNN Hyper-parameters and Image Representations Experiments

In this experiment, we simultaneously evaluate both the proposed image representations and the hyper-parameters combinations. We performed an exhaustive grid search through the subset of the hyper-parameter space described in Section 3.3, for each of the proposed image representations individually. To evaluate the robustness of each configuration, we applied cross-validation by considering 90% of the dataset for training and 10% for testing.

Table 1 shows the 10 best CNNs found in this experiment. The first relevant conclusion is that binarized images achieved better results than segmented

[2]http://www.im.ufal.br/professor/thales/ocdh/

Table 1: Configuration and accuracy of the best-performing CNN architectures.

image	*C*	*F*	*S*	*U*	*N*	accuracy	weights
bin	2	(3, 5)	32	2	(160, 160)	97,64%	564,194
bin	2	(5, 5)	32	2	(120, 80)	97,58%	347,466
bin	2	(5, 5)	32	2	(160, 160)	97,49%	467,426
bin	2	(3, 3)	32	2	(160, 80)	96,78%	642,290
bin	2	(3, 5)	16	2	(160, 80)	96,75%	275,778
bin	2	(5, 5)	32	2	(160, 80)	96,66%	454,386
bin	2	(3, 5)	32	2	(120, 80)	96,51%	419,914
bin	2	(5, 3)	32	2	(80, 80)	96,01%	272,802
bin	2	(5, 3)	32	1	(160, 0)	95,80%	522,562
bin	2	(3, 5)	32	2	(160, 80)	95,39%	551,154

images, which do not appear among the best results. This is more clear when we observe the histograms shown in Fig. 4, that represent the accuracy distribution of the experimented configurations for each image representation separately. For segmented images, accuracies range from 89% to 94%, while binarized images accuracies range from 96% to 99%. More importantly, we conclude that the best CNN architectures are very accurate to detect hand poses in all experimented body poses. This is a relevant result, as the user is expected to perform binary commands while simultaneously controlling the robot arm position in many distinct configurations.

By analyzing the hyper-parameters of the best architectures, we note that there is a prevalence of 2 convolutional layers with 32 filters per layer, and a diversity of values for the other evaluated hyperparameters. The best architecture is depicted in Fig. 3. However, it is worth observing that all configurations in Table 1 achieved a similar accuracy (always above 98%). Thus, we suggest that a smaller number of trainable weights may be a relevant criteria to choose an architecture among the best ones. However, hyperparameter calibration was still relevant to achieve significantly better accuracies, as revealed in Fig. 4.

4.3 CNN Evaluation for Robot Interaction

To further evaluate the best CNN architecture, we performed experiments simulating the usual pick-and-place task. We collected 10 performances of an individual picking an object and placing it in another location. It is expected that the user first places its controlling hand (open) over the object; closes his hand to grasp the object; moves his (closed) hand to another location; and finally opens his hand to release the object. Despite the simplicity of the task movements, some poses are challenging for hand pose classification, in particular when the user hand is pointing towards the Kinect sensor. In such situations, images from the hand poses are difficult to discriminate.

We evaluate and compare the robustness of the CNN in each sequence when each frame is individually classified, and when temporal filtering is applied (Section 3.4). We ignore frames from transitions between hand poses, since even a ground truth label is ambiguous. We take into account a delay of 15 frames to compare ground truth labels with labels classified after temporal filtering, as this is expected. The evaluated sequences have an average of 204 frames.

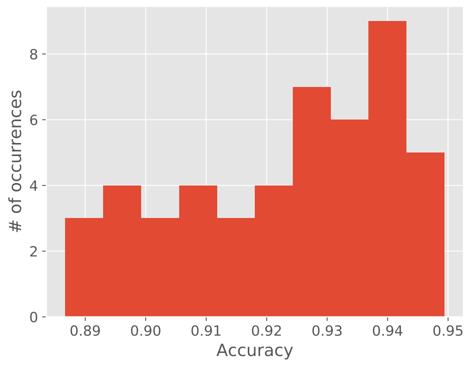
The results shown in Table 2 reveal that temporal filtering is indeed relevant to improve accuracy, achieving an excellent mean of 98.65%, against 94.73% when classification is performed in a per-frame basis. In particular, per-frame classification only outperformed the temporal filtering approach in sequence 5. In that case, a single misclassified frame in the 7th frame following a pose transition forced the command transition to be delayed by 7 frames, as it requires 15 consecutive similar classifications to change the command. Consequently, this lead to 7 incorrect results. However, we claim that this is not a critical issue, as the consequence would be just a delay in grasping or releasing an object. Is is also worth noting the worst result (sequence 7). In that case, the user performed challenging poses, as the ones shown in Fig. 5, in which case the hand was closed, but with the thumb straight. We believe that the CNN classification could be improved in these cases by providing additional training examples. Finally, we conclude that the proposed CNN approach is reliable for real-world applications.

4.4 User Experiments with a Robot Arm

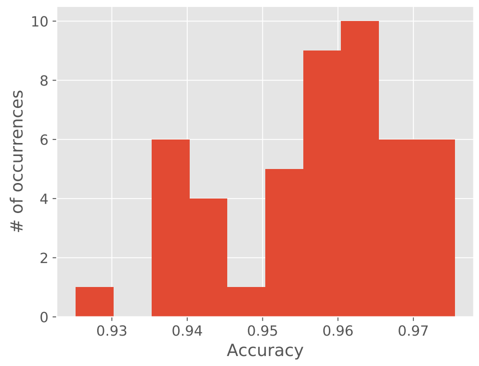
We validate our method using a 6-DOF industrial-grade robotic arm of model VP-6242, from Denso

Table 2: Comparison between straightforward per-frame classification and temporal filtering based classification: the best trained CNN is employed to classify all frames of 10 sequences simulating a pick-and-place task. Accuracy is defined as the ratio between correctly classified frames and the total number of frames of each sequence.

Sequence	Accuracy	
	Per-frame	Temporal filter
1	95.97%	100%
2	97.34%	100%
3	92.63%	100%
4	98.09%	99.04%
5	99.56%	96.92%
6	99.09%	100%
7	81.89%	91.35%
8	95.93%	100%
9	92.39%	99.21%
10	94.43%	100%
Mean ± Std. Dev.	94.73%±5.14%	98.65%±2.73%



(a) Histogram of accuracies of segmented images.



(b) Histogram of accuracies of binarized images.

Figure 4: Histograms of accuracies of the experimented architectures. Horizontal axes represent accuracy intervals, and vertical axes represent the number of architectures achieving the corresponding accuracies.

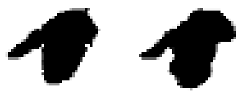
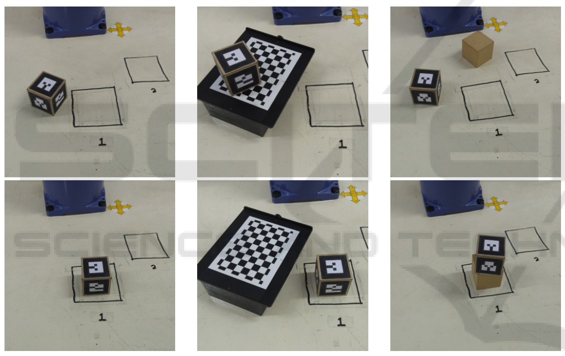


Figure 5: Challenging instances of hand poses: the user closed the hand with the thumb straight.



(a) Task 1. (b) Task 2. (c) Task 3.

Figure 6: Pick-and-place experiments with the robotic arm. Tasks 1, 2 and 3 are depicted in (a), (b) and (c), respectively. Top-rows represent initial states whereas bottom-rows illustrate the final configuration to be achieved.

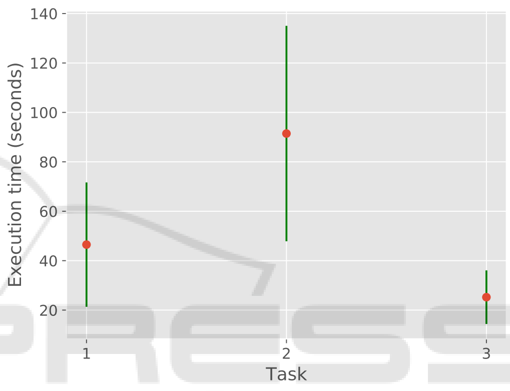


Figure 7: Execution times for the three pick-and-place experiments with the robotic arm.

Robotics[3]. It's end-effector is a gripper from Robotiq[4] (2F-85). The goal of the Human-Robot Interface is teleoperation, allowing indirect manipulation of objects by positioning the gripper according to the user hand position. Additionally, the gripper mimics the user hand state (open or closed).

We define three simple experiments (*cf.* Fig. 6) to evaluate the effectiveness of the interface: moving a box from one specific position to another (task 1); moving box 1 from a higher platform to a lower position at the table, then moving box 2 from the table to the higher platform (task 2); and pilling 2 boxes (on top of each other) on a desired position (task 3).

To validate the method, we invited 10 inexperienced users (9 men and 1 woman in their mid-twenties) to perform typical robot control tasks, always in the same order of execution of the tasks, *i.e.* firstly task 1, then task 2, then task 3. None of them had prior experience in controlling a robot using body movements. We measured the execution time that each subject needed to perform each of the three tasks, as revealed in Table 3. Average and standard deviation are shown in the last row and in Fig. 7. With a few exceptions, most tasks were successfully accomplished by the individuals. Results also suggest that users improve their performance for more complicated tasks after executing some initial experiments. More specifically, Fig. 7 reveals that, on average, task 3 was executed quicker than task 1, although task 3 is clearly harder than task 1.

[3]http://densorobotics.com/products/vp-g-series
[4]https://robotiq.com/products/adaptive-grippers

Table 3: Experimental results with inexperienced individuals. When a box is dropped, we count the execution as a failure.

Subject	Execution time (seconds)		
	Task 1	Task 2	Task 3
1	28	93	16
2	20	51	23
3	81	81	11
4	56	**fail**	22
5	82	180	40
6	44	67	20
7	18	64	**fail**
8	81	124	28
9	30	131	47
10	25	32	20
Mean ± Std. Dev.	47±27	91±46	25±11

5 CONCLUSION AND FUTURE WORK

We presented a real-time human-robot natural interaction approach suitable for the teleoperation of robotic arms. We collected a dataset of hand images (binary and segmented) and assessed different model hyper-parameters, searching for a classifier that best suited the problem. Then, we proposed a model for position control, using only hand movement, which was invariant to the user position and orientation. We performed proof-of-concept experiments with individuals with no previous training and measured elapsed times during the execution of pick-and-place tasks. The best hand pose classifier presented excellent recognition rates, which were improved by our proposed temporal filter, and allowed an intuitive user experience.

As future work, we plan to expand the number of hand state classes. Different control modes can also be analyzed, depending on the task context, such as using two hands and their relative positions. We also aim to investigate recurrent neural networks, since the input data is inherently sequential. Alternative human-robot movement mappings may also be topic of research, including user-customizable interfaces.

REFERENCES

Cheng, H., Yang, L., and Liu, Z. (2016). Survey on 3d hand gesture recognition. *IEEE Trans. Circuits Syst. Video Techn.*, 26(9):1659–1673.

Du, G., Zhang, P., and Li, D. (2014). Human–Manipulator Interface Based on Multisensory Process via Kalman Filters. *IEEE Transactions on Industrial Electronics*, 61(10):5411–5418.

Kruse, D., Wen, J. T., and Radke, R. J. (2015). A sensor-based dual-arm tele-robotic system. *IEEE Trans. Autom. Sci. Eng.*, 12(1):4–18.

LaViola Jr, J. J., Kruijff, E., McMahan, R. P., Bowman, D., and Poupyrev, I. P. (2017). *3D user interfaces: theory and practice*. Addison-Wesley Professional.

Microsoft (2018). Kinect - windows app development. https://developer.microsoft.com/en-us/windows/kinect.

Miranda, L., Vieira, T., Martínez, D., Lewiner, T., Vieira, A. W., and Campos, M. F. M. (2012). Real-time gesture recognition from depth data through key poses learning and decision forests. In *Sibgrapi*, pages 268–275.

OpenNI (2018). Openni - open-source sdk for 3d sensors. http://openni.ru.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks*, 61:85–117.

Sharp, T., Keskin, C., Robertson, D., Taylor, J., Shotton, J., Kim, D., Rhemann, C., Leichter, I., Vinnikov, A., Wei, Y., Freedman, D., Kohli, P., Krupka, E., Fitzgibbon, A., and Izadi, S. (2015). Accurate, robust, and flexible real-time hand tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 3633–3642, New York, NY, USA. ACM.

Shotton, J., Fitzgibbon, A. W., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., and Blake, A. (2011). Real-time human pose recognition in parts from single depth images. In *CVPR*, pages 1297–1304.

Tompson, J., Stein, M., Lecun, Y., and Perlin, K. (2014). Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks. *ACM Trans. Graph.*, 33(5):169:1—-169:10.

Xiao, Y., Zhang, Z., Beck, A., Yuan, J., and Thalmann, D. (2014). Human robot interaction by understanding upper body gestures. *Presence*, 23(2):133–154.

Yuan, S., Ye, Q., Stenger, B., Jain, S., and Kim, T. (2017). BigHand2.2M Benchmark: Hand Pose Dataset and State of the Art Analysis. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2605–2613.