Distinguishing Sliding from Slipping during Object Pushing

Martin Meier and Guillaume Walck and Robert Haschke and Helge J. Ritter

{mmeier,gwalck,rhaschke,helge}@techfak.uni-bielefeld.de

Neuroinformatics Group, Center of Excellence Cognitive Interaction Technology (CITEC),
Bielefeld University, Germany

Abstract— The advent of advanced tactile sensing technology triggered the development of methods to employ them for grasp evaluation, online slip detection, and tactile servoing. In contrast to recent approaches to slip detection, distinguishing slip from non-slip conditions, we consider the more difficult task of distinguishing different types of slippage. Particularly we consider an object pushing task, where forces can only be applied from the top. In that case, the robot needs to notice when the object successfully moves vs. when the object gets stuck while the finger slips over its surface. As an example, consider the task of pushing around a piece of paper.

We propose and evaluate three different convolutional network architectures and proof the applicability of the method for online classification in a robot pushing task.

I. INTRODUCTION

In robotics, object manipulation is a key capability – especially in service robotics when interaction with complex environments is necessary. Applying a motion to an object doesn't need to involve grasping. In this case the term *non-prehensile manipulation* or more concretely *push-manipulation* is used. *Push-manipulation* has been well analyzed to model the mechanics of such motion [1], [2]. In most of the work it is supposed that the friction properties between the *pusher* and the *slider* are known or at least controlled. This allows for feed-forward execution of planned motion trajectories. For example, [3] studied pre-grasp manipulation involving object pushing. Particularly, they considered getting in contact with the object from top and using friction to move the object. However, the contact point was assumed to have enough friction for proper power transmission.

As a consequence, there is little work considering reactive controllers monitoring and adjusting the contact state during motion. In [4], a visual tracking system was employed to maintain a suitable angle of contact in order to ensure proper friction. However, online slippage monitoring and active force control call for tactile sensors within the feedback loop. The increasing availability of tactile sensors triggered a lot of research to employ them for grasp evaluation, see [5] for a review of current technologies. Some methods were also proposed for slip detection – typically for object grasping. Corresponding methods can be distinguished into two major approaches: 6D contact wrench estimation and detection of micro-vibrations at the contact.

Bicchi [6] has shown, that it is possible to determine the local contact position and the local contact wrench given a wrench measurement at the fingertip base – provided that the fingertip shape is a known quadratic surface, e.g. spherical, ellipsoid, or planar. Knowing the friction coefficient, one can then estimate slippage from the ratio of observed tangential and normal forces at the contact point [7]. In [4], the friction coefficient to be considered during pushing is estimated in advance by sliding over the fixated object and estimating the contact force from measured torques at the hand joints.

In contrast to these approaches that suppose an exact friction model to be available, vibration-based approaches attempt to predict incipient slippage from high-frequency oscillations ($\approx$ 200-400 Hz) in the tactile sensor signals, much like Pacinian corpuscles do in human skin [8], [9]. To this end, machine learning techniques are employed to directly predict slippage from data (typically only the normal force or the overall pressure value is available).

In [10], [11], slippage-induced micro-vibrations are measured with tactile sensors that allow for high-speed acquisition rates and thus enabled the detection of slippage before visual sensors or inertial measurement units (IMU) can notice. Several machine learning approaches were studied to detect slippage, (i) including SVM and random forests [12] achieving an F_{score} of 0.75, and (ii) multilayer perceptrons (MLP) achieving an accuracy of 80% [13]. The authors of the latter work also pointed out, that it is particularly important to distinguish between translational and rotational slippage, because counteracting rotational slip typically requires higher forces to be exerted onto the object for stabilization. However, to the best of our knowledge, this task wasn't solved yet.

Having successfully applied convolutional neural networks to this task of distinguishing translational, rotational and non-slip conditions recently [14], in the present paper we transfer that work from the previously used 16×16 flat sensor matrix to a 12-taxel, 3D-shaped fingertip. As a consequence of the curved fingertip shape, the contact area is typically much smaller, only activating 4-6 taxels at a time, rendering the classification task even harder. Furthermore, we consider the important task of distinguishing object sliding from object slippage in non-prehensile manipulation. When pushing thin objects, e.g. a piece of paper, one can only apply forces from the top, not from the side. In some cases, the object will then not move on the ground, but the finger(s) will slip on the object. For successful execution of such tasks, it is
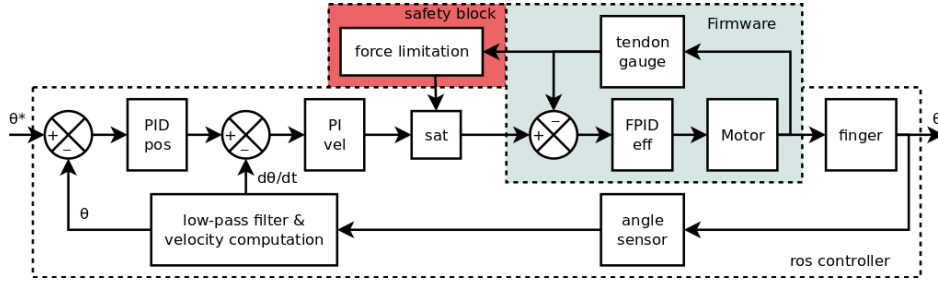
Fig. 1. Internal mixed position velocity controller with safety force limitation

essential for a robot to distinguish these conditions.

We consider different architectures of convolutional neural networks to perform this task. Additionally, to compare to existing literature, we also study MLPs. In the following sections, we will first shortly describe the tactile controller used in experiments, before covering data acquisition and preprocessing. Subsequently, the considered neural network architectures are described, evaluated and discussed.

II. ACTIVE TACTILE CONTROL

Sliding an object on a supporting surface involves two contact situations: the finger-object contact and the object-surface contact. In general, both contact situations have different friction properties and successful pushing depends on the ratio of involved friction coefficients as well as the weight of the to-be-pushed object. The weight defines a baseline gravity force $F_g = m \cdot g$ acting between the object and its support. Coulomb's friction law then defines the condition for successful pushing in a quasi-static situation:

$$\mu_{os} \cdot (F_n^{fo} + F_g) < \|F_t^{os}\| = \|F_t^{fo}\| \le \mu_{fo} F_n^{fo}, \quad (1)$$

where μ_{os} and μ_{fo} are the friction coefficients of object-surface and finger-object contacts, F_t^* are the tangential forces which are identical in both contact situations (assuming horizontal contact surfaces in both situations), and F_n^{fo} is the normal force applied onto the object by the finger. To start the pushing motion, the friction coefficients of static friction have to be considered. As soon as the object is sliding, the smaller dynamic friction coefficient can be used for μ_{os}, thus facilitating the pushing motion.

Obviously, this inequality condition can only be fulfilled (and thus allow for pushing), if the friction coefficient μ_{os} is significantly smaller than the friction coefficient μ_{fo} in order to overcome the extra contribution (F_g) on the left-hand side. Increasing F_n^{fo} will reduce the influence of F_g.

As the robot needs to apply a carefully chosen contact normal force F_n^{fo} during the pushing task, the following sub sections will detail the control scheme implemented to achieve this. This control scheme was used both for recording training data and for the application of indirect contact detection.

A. Position control with force limitation

We augmented the position controller provided by the hardware manufacturer to achieve combined position and tactile force control. In order to understand the integration, we first describe the existing control scheme and subsequently our extensions. The Shadow Robot hand comes with a mixed position-velocity controller for each joint of the hand. They run at the host computer in a pseudo-realtime loop within the ros_control framework and provide a joint torque that is commanded to the hand firmware in a 1 kHz control loop. The joint torque itself is controlled onboard by the hand's firmware at a rate of 5 kHz.

The overall controller structure is illustrated in Fig. 1, which distinguishes the firmware and host part of the controller as well as an additional safety block. The latter was added to protect the hand from damage due to long-lasting, strong tendon forces. Indirectly, this block also limits the maximum force applied on objects. It saturates the forces sent to the firmware by a dynamically determined factor that depends on an "accumulated pain level". For more details on the force limitation, we refer to [15]. For the present work, it is important to know, that this safety block introduces implicit compliance in the position control loop whenever tendon forces exceed a specified threshold. This means the finger will give back slowly due to force limitation, if the tendon force has been too high for a certain time.

However, the relation between the tendon force and the fingertip force is highly non-linear due to internal friction and complex tendon routing, such that the contact force at the tip cannot be regulated with the existing control loop. Hence, an external feedback loop was added to ensure a more precise contact force control.

B. Tactile Contact Force Control

To perform tactile contact force control, it suffices to adjust only a few joints when considering non-prehensile push manipulation. Thus, an external feedback loop was added to a selection of joint controllers, particularly proximal finger joints and the wrist flexion.

The external feedback loop shown in Fig. 2 recalls the mixed position-velocity controller with force limitation presented in Fig. 1, and augments it with a tactile contact force controller employing an inverse spring-law with gain K. The raw tactile sensor signals are (i) smoothed using an exponentially weighted moving average, (ii) normalized and (iii) averaged over all 12 tactile cells of the fingertip sensor to provide a single overall force value as feedback. All these processing steps are summarized in the filter block in Fig. 2. The experimentally-tuned gain controls the sensitivity of the
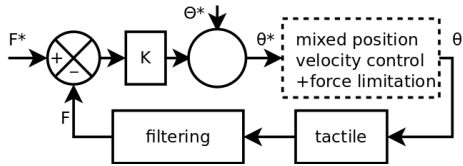
Fig. 2. External tactile contact force regulation loop

regulation loop and permits to maintain contact with a given contact force. The regulation is always active, simulating a spring around the target position.

III. DATA ACQUISITION & PREPROCESSING

In the following we will outline our data acquisition approach and the data pre-processing pipeline to compute features suitable for slip classification. However, first we introduce the tactile sensor employed in the experiments.

A. Tactile Sensor

Our dexterous Shadow Robot hands are equipped with high-speed, piezo-resistive tactile sensors developed in previous work [16]. The sensor electrode, built as a 3D-shaped molded interconnection device (MID), provides 12 tactile sensing cells per fingertip, measuring normal forces. The electrode is covered by a soft, 1 mm thick, conductive foam that changes resistance due to the applied pressure. The sensor has a high first touch sensitivity of 0.03 N/cm^2.

The design of the acquisition electronics that is directly integrated into the sensor, permits to reach a 750 Hz sampling rate, opening the possibility to detect micro vibrations and therefore also slippage, much like we have shown in previous work [10]. The sensor is directly connected to the Shadow Hand's SPI bus, thus benefiting from the 1 kHz acquisition loop. All available tactile signals are transmitted, along with other sensor data, to the host computer and made accessible, with timestamp, on a topic in the ROS environment.

B. Data Acquisition

To acquire auto-labeled tactile data for classifier training, we recorded raw tactile sensor information in the two conditions to be distinguished: successful sliding of an object on a support and slipping with a finger over an object. Both conditions are illustrated in Fig. 3. In the first case, also referred as *sliding with* the object, the finger is static w.r.t. the object and the object moves on the surface. In the latter case, referred as *slipping over*, the object is static w.r.t. the support surface and the finger slips over the object (cf. Fig. 3).

In both cases we programmed the robot to perform a linear motion in Cartesian space maintaining contact to the
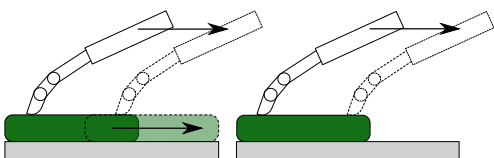

Fig. 3. Sliding motion: left *sliding with*, right *slipping over*

Fig. 4. Left: the material used in experiments. Middle: the fingertip MID tactile sensor. Right: a Mitsubishi PA-10 robot arm and the Shadow dexterous hand equipped with tactile sensors.

object employing the feedback controller described in the previous section. The only difference was that the object was fixated to the support for the *slipping over* condition, while it could freely move in the *sliding with* condition. Each motion trial was composed of several phases, namely reaching, approaching downwards, contacting, sliding, and releasing – controlled by a state machine. The linear sliding movement extended over 15 cm for a duration of 10 s. This central part of linear sliding was considered for training data in the corresponding condition. The onset of this part could be determined from specific joint angle configurations, thus automatizing the labeling process.

We recorded training data for three different objects, in three different thicknesses, and for three different support materials. This ensures a large variety of friction properties and object weights (cf. Eq. 1). Particularly, the objects considered are rectangular pieces of plywood, PVC, and aluminum. The chosen thicknesses can be categorized as thin, medium, and thick, respectively below 3 mm, around 5 mm, and around 10 mm. They are depicted in Fig. 4. The support materials were chosen to enable sliding for a large range of objects. They are made of Delrin®[1], a fabric sheet on top of soft foam, and Kömacel®[2]. Recording was repeated five times for all material combinations theoretically resulting in a total of $3^3 \cdot 5 = 135$ recordings.

However, some of the material combinations did not lead to successful *sliding with* the object, for which reason we did not considered them for training. Particularly, the plywood did not slide on the fabric, because the corresponding friction coefficient μ_{os} is too large compared to μ_{fo}. Even a human could not push the object. We also had to omit the thick versions of every object sample, because the stronger influence of gravity force F_g prevented proper *sliding with*. Consequently, $(3 \times 2 \times 3 - 2) \cdot 5 = 80$ recordings remain for the *sliding with* condition. In the *slipping over* condition, obviously the support material doesn't play a role. Hence,

[1]acetal homopolymer resin
[2]an integral skin-foam sheet made of rigid PVC

we have here $3 \times 3 \times 5 = 45$ recordings. In order to balance the amount of training data for both conditions, we considered longer sequences of training data for the *slipping over* condition.

C. Preprocessing

The raw force measurements from all 12 taxels were linearly scaled to a range of $[0..1]$ for normalization. Since high frequency micro vibrations play an important role in human slip detection [8], we decided to use similar features in the present classification task. Hence, from the raw, normalized sensor data we calculated discrete Fourier transformations individually for each of the 12 sensor cells to obtain a frequency representation of the signal. After preliminary evaluations of different window sizes and window shifts, we decided on a window size of 64 samples with a shift of 8 samples. This eight-sample buffer can absorb the rare bulk data transfer due to pseudo-realtime ROS publishing, but still permits FFT computation due to acquisition timestamps being available.

To reduce the influence of low frequency vibrations induced by the robot and slow variations introduced by the active tactile control loop maintaining contact, we employ a high-pass filter of ≈ 50 Hz, effectively cutting the four lowest frequency components. From the remaining 28 components we are using only the real part, such that we get an $4 \times 3 \times 28 = 336$ dimensional input vector per processing frame for evaluation by the neural network.

IV. LEARNING

Before we go into the introduction of the convolutional neural network model employed for learning, we will motivate our decision for that approach from a visual data analysis. To gain some initial insight into the class distribution within our dataset, we visualized the data using t-distributed stochastic neighbor embedding (t-SNE) [17]. t-SNE is a dimensionality reduction method projecting data onto a low-dimensional space (here 2D) in a non-linear fashion, trying to preserve the probability distribution of data points.
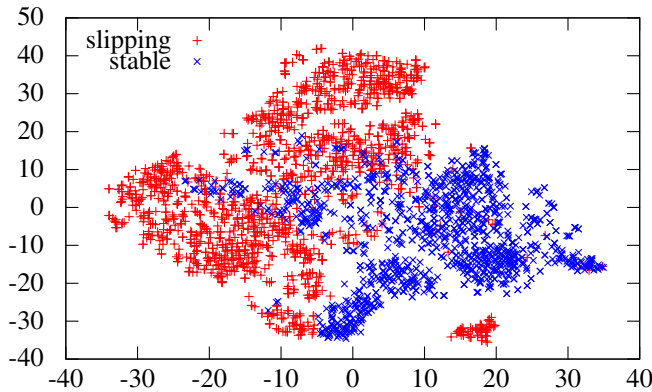
A t-SNE visualization of a baseline dataset that simply distinguishes between slip and stick conditions is shown in Fig. 5. This dataset is similar to the data recorded in other recent approaches which successfully learned to discriminate these conditions [10]–[13]. The elements of the dataset that represent slipping have stronger amplitudes in the higher frequency bands due to vibrations created through micro slip [9]. In contrast, the elements representing stable states have only small amplitudes in these bands, since no micro vibrations occur. Because of this difference, a two-dimensional t-SNE can nicely separate both classes.

On the other hand, Fig. 6 shows the t-SNE visualization of the *sliding with* vs. *slipping over* conditions described in sec. III-B. Compared to Fig. 5, the data distribution here is far more overlapping, underlining the complexity of the problem. For this reason, we decided not to investigate kernel methods like support vector machines (SVM) in detail. Preliminary experiments with SVMs using RBF-kernels showed only classification accuracies around 60%. Instead, we decided to go for artificial neural networks like the multilayer perceptron (MLP).

A neural network processes inputs by simple and identical processing units (neurons), typically performing weighted summation and a non-linear thresholding operation. Neurons are arranged in a layered architecture. The different layers are interconnected by adaptable weights **w**, trained via the backpropagation algorithm [18]. The power of neural network arises from their high flexibility (due to different architecture/layer layouts).

The authors in [13] and [10] also employed MLPs in their slip detection tasks with hidden layer sizes ranging from 50 to 1024 neurons. To assess the complexity of the task at hand, we employed MLPs as a baseline test for our classification task, using 256 and 512 neurons in the hidden layer, respectively.

Motivated by recent achievements of deep network architectures in various classification tasks, we also investigated the performance of this type of network for our purposes. Besides creating state-of-the-art results in image classifica-

Fig. 5.   t-SNE visualization of the stable and slipping classes acquired during the data recording. The two classes look reasonable separated.
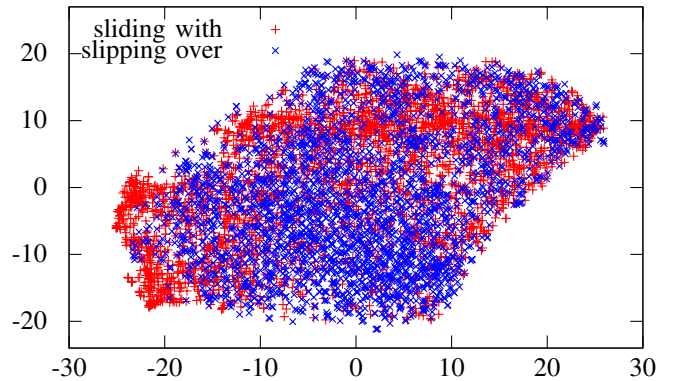

Fig. 6.   Visualization of the data we strive to classify with t-SNE. Although there are some small local clusters, both classes have a strong overlap.
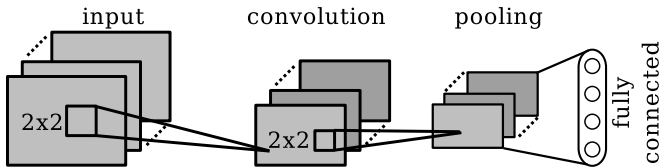
Fig. 7. Exemplary display of a convolutional neural network. The input is fed into convolution layers, which in turn are connected with a max pooling layer. The output of the pooling layer is then connected to a fully connected layer for final classification.

#	architecture
CNN1	conv 2×2×224 → fc (512)
CNN2	conv 2×2×224 → pool 2×2 → fc (512)
CNN3	conv 2×2×224 → conv 2×2×224 → fc (512)

TABLE I

NETWORK ARCHITECTURES WHICH HAVE BEEN EVALUATED.

net	test accuracy	net	test accuracy
MLP 256	71.86%	CNN1	78.66%
MLP 512	73.28%	CNN2	79.72%
		CNN3	81.34%

TABLE II

TEST ACCURACY.

tion tasks [19], convolutional neural networks (CNN) have been successfully used to enhance classification results on time series data, for example in speech recognition [20].

In contrast to MLPs that simply employ a number of fully-connected layers, CNNs exploit the underlying structure in the input data (often 2D images) to compute convolutions with a set of learnable filters (or kernels). The neurons in the convolutional layer have a small receptive field (indicated in the input layer of Fig. 7 with a 2×2 square), which are often overlapping. This structure is motivated by the organization of biological neurons that respond to small, overlapping regions of the visual field [21]. A max-pooling layer, following the convolutional layer, non-linearly down-samples the filter responses by outputting the maximum filter response within a small receptive field. Usually, CNNs for image classification consist of multiple cascades of convolution and pooling layers in sequence, computing an increasing number of more and more complex filters, always maintaining the original two-dimensional grid structure of the input image. Finally, the filter responses are fed into a set of fully-connected layers to compute the final classification result using the soft-max loss function.

In case of the tactile fingertip depicted in Fig. 4, taxels can also be arranged in a two-dimensional grid structure mimicking the physical layout of the sensor cells and thus allowing the application of spatial convolution filters. Hence, the 336-dimensional input vector can be considered as a 4×3 "image" with 28 channels representing the Fourier amplitudes.

On these image-like inputs, it is possible to use established techniques to learn convolutional filters, which then represent combinations of frequency responses along the temporal and spatial dimension. We evaluated different architectures of convolutional networks, an overview is given in table I. The width and height of the receptive fields (convolutional kernels) was fixed at 2×2, since usually not more than two neighboring taxels will be activated due to the curved 3D shape of the fingertip. The abbreviations in the table are as follows: *conv* $2 \times 2 \times N$ is a convolutional layer computing a set of N filters of size 2×2; *pool* 2×2 is a corresponding max pooling layer reducing the spatial resolution by a factor of 2; and *fc (512)* is a fully connected layer with 512 neurons. We decided to learn 8 convolution filters explicitly per channel, since the Fourier amplitudes encoded in the channels of the input data hold the important information about micro-vibrations in the different frequency bands and should be

represented accordingly across the filters. Thus, a total of 224 filters are learned per convolution layer.

V. EVALUATION

All networks have been trained with RMSProp, using an initial learning rate of 0.001 and a decay of 0.95. With the data we recorded for the detection of sliding vs. slipping movements, we did a 5-fold cross validation for each of the employed convolutional networks, meaning that we shuffled the dataset randomly before splitting it into five parts containing an equal number of samples. Four of these parts were used for training the networks and the remaining part was used for testing, which was done for every possible permutation.

Additionally, we used the presented classifier as online input for a manipulation task, where the robot pushes objects on different surfaces until they collide with an obstacle, which changes the contact condition from *sliding with* to *slipping over*.

A. Offline Classification Results

The test accuracy of each network described in table I is shown in table II. Compared to the plain MLP networks, the incorporation of convolutional layers in the network increases the classification performance by nearly 10%. Given the moderate size compared to image processing CNNs, the networks are still fast enough to be processed online within the ≈ 10 ms data acquisition cycle (given the window shift of 8 samples) on consumer grade hardware (Theano-based implementation on a Nvidia GTX 980 GPU).

Training the network architecture CNN3 (cf. table I) on the baseline dataset simply distinguishing slip vs. stick conditions, we achieved a test accuracy of 96.4%, which clearly outperforms a competing, MLP-based approach that reached only 80% [13]. This emphasizes the superior performance of the CNN-based approach.

B. Online Push Control

To evaluate the performance of our approach in a more realistic scenario, we realized a force-controlled pushing-manipulation task, where the robot should use one finger to push an object slowly towards a wall until contact occurs. As soon as the object collides with the wall, the robot should

proceed with the next object. Indirect contacts during such a manipulation do not transfer significant external forces to the finger to be detected with conventional force sensors.

The network CNN3 from table I permits to check when the object is sliding on the support and notice when the object stopped moving due to an external contact. Therefore, the feedback of the sliding vs. slipping classifier provides an input to a hierarchical state machine (HSM), which serves as a high level control instance in our setup. In the slipping over case, an event is sent to the HSM, which reacts accordingly by stopping the current manipulation and executing the next task.

For practical reasons, given a classification accuracy close to 82%, which is provided at a pace of 94 Hz, we added an additional consolidation step: The softmax probabilities generated by the classifier were averaged over 25 time steps, which yields a nice smoothing of the results over time. Furthermore, we introduced a threshold of 0.8 before the triggering event was send to the controlling HSM.

We performed 80 pushing trials with the subset of the materials mentioned in sec. III-B. As for training, we omitted the combination of plywood on fabric. Three types of behavior were distinguished: (i) *successful* collision detection, when the system would notice the contact rapidly, (ii) *failed* contact detection, and (iii) *premature* contact detection when the robot switched to early to the next object. The overall performance, among all the objects is 76.25% of success, whereas 8.75% of the motions did not detect the obstacle, and premature detection occurred 15% of the time.

We also performed additional trials with pieces of paper that were not included in the training data at all. In these trials, the collision with the obstacle would not trigger a transition to *slipping*, but simply bulge the paper. Hence, we enforced the *slipping over* condition manually fixating the paper on the surface. The distinction of slipping versus sliding was successful up to 81% of the time in this situation. Most of the errors occurred when the paper was pushed on the fabric support due to the softness of the underlying foam.

VI. DISCUSSION

We showed that learning convolutional filters on tactile time series data can achieve high classification results in different slip detection tasks required for non-prehensile object manipulation. On a baseline dataset, distinguishing slip vs. stick conditions only, the CNN achieved a test accuracy that clearly outperforms traditional approaches, thus offering new opportunities for adaptive grasp control. In the main classification task considered in this work, namely distinguishing *sliding with* vs. *slipping over* an object, which is a much more difficult task due to a high degree of similarity and overlap between classes, CNNs achieved good classification results of up to 81% enabling real-world applications. We evaluated our approach in a realistic scenario where our system could reliably track the contact state during a complex push-manipulation task.

REFERENCES

[1] M. T. Mason, *Mechanics of Robotic Manipulation*. Cambridge, MA, USA: MIT Press, 2001.

[2] K. M. Lynch, "The mechanics of fine manipulation by pushing," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2269–2276 vol.3, May 1992.

[3] D. Kappler, L. Chang, M. Przybylski, N. Pollard, T. Asfour, and R. Dillmann, "Representation of pre-grasp strategies for object manipulation," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2010.

[4] F. Ruiz-Ugalde, G. Cheng, and M. Beetz, "Fast adaptation for effect-aware pushing," in *11th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pp. 614–621, IEEE, 2011.

[5] Z. Kappassov, J.-A. Corrales, and V. Perdereau, "Tactile sensing in dexterous robot hands¿review," *Robotics and Autonomous Systems*, vol. 74, pp. 195–220, 2015.

[6] A. Bicchi, J. K. Salisbury, and D. L. Brock, "Contact sensing from force measurements," *The International Journal of Robotics Research*, vol. 12, no. 3, pp. 249–262, 1993.

[7] X. Song, H. Liu, J. Bimbo, K. Althoefer, and L. D. Seneviratne, "A novel dynamic slip prediction and compensation approach based on haptic surface exploration," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4511–4516, IEEE, 2012.

[8] R. Johansson, U. Landstrom, and R. Lundstrom, "Responses of mechanoreceptive afferent units in the glabrous skin of the human hand to sinusoidal skin displacements," *Brain Research*, vol. 244, no. 1, pp. 17–25, 1982.

[9] R. D. Howe and M. R. Cutkosky, "Sensing skin acceleration for slip and texture perception," in *IEEE International Conference on Robotics and Automation*, pp. 145–150, IEEE, 1989.

[10] M. Schöpfer, C. Schürmann, M. Pardowitz, and H. Ritter, "Using a piezo-resistive tactile sensor for detection of incipient slippage," in *41st International Symposium on Robotics (ISR), and 6th German Conference on Robotics (ROBOTIK)*, pp. 1–7, VDE, 2010.

[11] J. A. Fishel, V. J. Santos, and G. E. Loeb, "A robust micro-vibration sensor for biomimetic fingertips," *IEEE/RAS-EMBS International Conference Biomedical Robotics and Biomechatronics (BioRob)*, vol. 6106, pp. 659–663, 2008.

[12] F. Veiga, H. van Hoof, J. Peters, and T. Hermans, "Stabilizing novel objects by learning to predict tactile slip," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5065–5072, IEEE, 2015.

[13] Z. Su, K. Hausman, Y. Chebotar, A. Molchanov, G. E. Loeb, G. S. Sukhatme, and S. Schaal, "Force estimation and slip detection/classification for grip control using a biomimetic tactile sensor," in *IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 297–303, IEEE, 2015.

[14] M. Meier, F. Patzelt, R. Haschke, and H. J. Ritter, "Tactile convolutional networks for online slip and rotation detection," in *International Conference on Artificial Neural Networks*, 2016.

[15] G. Buscher, M. Meier, G. Walck, R. Haschke, and H. J. Ritter, "Augmenting curved robot surfaces with soft tactile skin," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1514–1519, IEEE, 2015.

[16] R. Koiva, M. Zenker, C. Schurmann, R. Haschke, and H. J. Ritter, "A highly sensitive 3d-shaped tactile sensor," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 1084–1089, IEEE, 2013.

[17] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 2579-2605, p. 85, 2008.

[18] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *International Joint Conference on Neural Networks*, 1989.

[19] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015.

[20] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 10, pp. 1533–1545, 2014.

[21] D. Hubel and T. Wiesel, "Receptive fields and functional architecture of monkey striate cortex," *Journal of Physiology*, vol. 195, p. 215–243, 1968.