Chapter 1

Deep learning techniques for modelling human manipulation and its translation for autonomous robotic grasping with soft end-effectors

Visar Arapi[1], Yujie Zhang[1,2], Giuseppe Averta[1,2,3], Cosimo Della Santina[4], and Matteo Bianchi[1,2]

One of the key enablers for the extraordinary dexterity of human hands is their compliance and capability to purposefully adapt with the environment, to multiply their manipulation possibilities. This observation has also produced a significant paradigm shift for the design of robotic hands, leading to the avenue of soft end-effectors that embed elastic and deformable elements directly in their mechanical architecture. This shift has also determined a perspective change for the control and planning of the grasping phases, with respect to the classical approach used with rigid grippers. Indeed, instead of targeting an accurate analysis of the contact points on the object, an approximated estimation of the relative hand-object pose is sufficient to generate successful grasps, exploiting the intrinsic adaptability of the robotic systems to overcome local uncertainties. This chapter reports on deep learning techniques used to model human manipulation and to successfully translate these modelling outcomes for enabling soft artificial hands to autonomous grasp objects with the environment.

1.1 Introduction

Achieving stable and purposeful grasps with robotic hands is a challenging problem, especially under the framework of autonomous operations. Classical approaches for grasp planning and execution targeted the exact definition of the contact points on the object (object-centric approach) [1]. These approaches defined a set of available contact points and then identified point locations and contact forces, relying on the knowledge of object properties. These methods were proven to work well with

[1]Centro di Ricerca "Enrico Piaggio", University of Pisa, Largo L. Lazzarino 1, 56162 Pisa, Italy
[2]Department of Information Engineering, University of Pisa, via G. Caruso 16, 56122 Pisa, Italy
[3]Soft Robotics for Human Cooperation and Rehabilitation, Fondazione Istituto Italiano di Tecnologia, via Morego, 30, 16163 Genova, Italy
[4]Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA

rigid robotic hands, but can be hardly applied to a new generation of end-effectors, which can deform and interact with the environment. The latter category of robotic end-effectors, which can be continuously deformable [2] or soft-articulated [3], envisions the purposeful introduction of elastic elements in their mechanical structure to allow the adaptation around different objects and with the environment, the increase of the device robustness and the reduction of the control burden (often capitalizing upon under-actuation schemes). The motivation for the design of these soft hands is the human example: every-day human hands purposefully leverage on their softness to exploit the environmental constraints [4] for multiplying grasping opportunities and degrees of freedom, while reducing the computational complexity needed for task execution. Lifting a coin from a table, pivoting the object around one or more fingers interacting with the surface, represent some exemplary cases of this ability. Under a robotic point of view, soft manipulators have also produced a perspective shift in grasp planning problems, requiring only an approximated estimation of the relative hand-object pose is sufficient, letting the elasticity of the end-effectors and the interaction with the environment doing the rest [5]. It is hence clear that this paradigm shift also requires new mathematical tools, to model human behavior (which represents the golden standard for artificial manipulation) and to translate it in well-defined control laws for autonomous manipulators. Under this regard, machine-learning (ML) and, especially, deep-learning (DL) [6], have emerged as promising techniques to tackle the aforementioned twofold goal. Indeed, ML and DL allow to overcome the modelling challenges that arise when dealing with soft bodies and their interaction with the external environment, targeting solutions close enough to the desired ones, rather than exact, leveraging on the adaptation capabilities of the soft hands to overcome local uncertainties.

However, so far only few works in literature have applied learning methods for (i) studying human hands in every-day grasps with the environment, and (ii) controlling soft hands, by suitably translating human observation outcomes on the robotic side. Regarding (i), pioneer work on convolutional neural network (CNN) applications to hand gesture recognition dates back to 90s. Among the more recent examples it is worth mentioning the EgoHands [7], a CNN based framework trained on 15.000 segmented hands (obtained through a manually pixel-level process on 4800 egocentric video frames − first person videos of people playing four game board activities). EgoHands detects accurately one or more hands from each video frame, with very robust performance with respect to (w.r.t.) changes in environment conditions, particularly during Environmental Constraint Exploitation (ECE) [4] for grasping. Regarding (ii), in [8] authors introduced a mixed approach combining learning by demonstration with reinforcement learning to transfer grasping capabilities of known objects from a human operator to the robotic system. In [9], GRNN (generalized regression neural networks) and autoencoders were adopted to learn from human demonstration examples how to manipulate previously unseen thin objects with a soft gripper. In [5] a library of reactive strategies was collected from a subject operating a soft hand, and successfully translated for robotic grasping of new items, in a human-robot handover scenario. In [10] a 3D convolutional neural

network was trained with tens of thousands of labeled images. The network output provides the control input for the hand approaching direction.

All these papers are extremely promising, especially for the integration of deep learning with the intrinsic compliance of soft end-effectors, i.e. their embodied intelligence. However, they failed in terms of result generalization and in effectively grasping the full potentiality that the human example and the environment exploitation could represent for achieving autonomous grasps with softhands.

In this chapter, we report on two successful applications of deep learning to the study of human hands and its translation for autonomous grasping with soft grippers. We will discuss the results and finally comment on future avenues and possibilities of these approaches.

1.2 Investigation of the human example

Everyday, humans purposefully take advantage from the interaction with the environment, to accomplish successful manipulation actions, relying on the intrinsic compliance of their hands. This is one of the key enablers for human extraordinary manipulation capabilities, which have not been yet matched on the robotic side. For these reasons, the observation and modelling of the human example could provide useful insights for the control of soft robotic hands, which can take advantage from the exploitation of the environmental constraints, similarly to human hands. In [11] authors modelled human grasping behavior introducing transition probabilities for the identification of the conditions leading to the decision between one path of action with respect to another one. The outcomes of this investigation resulted in different action sequences with respect to different object shapes. Translation of these results for the control of soft robotic end-effectors could allow the robots to effectively evaluate the transition conditions, relying on suitable sensors and computational tools. Towards this goal, the first mandatory step is the recognition of human gestures, which is usually accomplished through wearable [12, 13] or remote devices. Regarding remote systems, the most commonly used strategy is represented by video recordings. In literature there have been important examples for the extraction of features describing human gestures in video sources, which include methods based on k-means classification or Hidden Markov models [12, 13].

At the same time, deep learning approaches have emerged as a promising tool for feature extraction, e.g. image classification [14], object detection [15], hand gesture recognition in video sources. Regarding the latter point, it is worth mentioning EgoHands [7], where four actions were recognized by a Convolutional Neural Network (CNN) − trained with 4800 segmented hands − in combination with windowing at fixed temporal size. CNNs can be synergistically used in combination with Recurrent Neural Networks (RNNs), which allow to efficiently and robustly manage spatio-temporal features [16, 17, 18]. RNNs usually rely on Long Short Term Memory (LSTM) cells, which can put in memory a compressed representation of medium-range temporal relationships in the input sequence [19].

However, to the authors' best knowledge, there is currently no approach based on deep learning applied to videos, for the characterization of the dynamic and time-

-dependent content underpinning human hand pose evolution, during the interaction with the environment. To bridge this gap, we proposed a framework, named DeepDynamicHand presented in [20], which targets visual features related to the hand shape only (instead of considering the whole video frame) and dynamic video annotation encompassing both the pre-grasp and grasping actions. DeepDynamicHand consists of two neural architectures: the first one is a CNN for segmenting human hand in each video frame, enabling the extraction of a compressed and rich encoding of the hand posture; the second architecture is a RNN based on LSTM recurrent units [21].

The latter architecture takes as input the sequence of encodings provided by the CNN and generates as output a sequence of action primitives – i.e. a dictionary of meaningful behavior components whose composition enables to successfully interact with the environment in grasping tasks – performed by human hands in the videos.

1.2.1 Methods

Let us consider a set of videos

$$< \mathbf{V}_1, \mathbf{V}_2, \cdots, \mathbf{V}_T > . \tag{1.1}$$

A single video $\mathbf{V}_j$ comprises a sequence of frames

$$\mathbf{V}_j = (\mathbf{I}_{j,1}, \mathbf{I}_{j,2}, \cdots, \mathbf{I}_{j,n}), \tag{1.2}$$

where $\mathbf{I}_{j,t} \in \mathbb{R}^{w \times h \times 3}$, w and h are the width and height of the video frame at time t, respectively. In general, the number of frames n composing the video varies from case to case. Our goal is to automatically convert each video into a sequence of labels

$$\mathbf{Y}_j = (\mathbf{y}_{j,1}, \mathbf{y}_{j,2}, \cdots, \mathbf{y}_{j,n}), \tag{1.3}$$

where each label $\mathbf{y}_{j,t}$ (j refers to the actual video, while t is the temporal frame) is the *action primitive* – included in a pre-defined (finite) dictionary S – executed by the participant's hand in each frame. Moreover, considering that, in each video, actions performed by the participant are executed in a dynamic temporal sequence, implies that the label $\mathbf{y}_{j,t}$ associated to the action primitive at frame t depends on features observed both in prior as well as in the actual frames $(\mathbf{I}_{j,1}, \mathbf{I}_{j,2}, \cdots, \mathbf{I}_{j,t})$.

To address the aforementioned challenge, we propose a two-stage architecture (Figure 1.1). In the first stage (Figure 1.1-(a)) we leverage on a CNN based approach to recognize the hand in each video frame $\mathbf{I}_{j,i}$ and subsequently extract a condensed and, still, informative characterization of the hand pose. To this end, we employ two units, named *window proposal* and *window classification*, respectively. The *window proposal* unit

$$H_P : \mathbb{R}^{w \times h \times 3} \longrightarrow \mathbb{R}^{w_k \times h_k \times 3}, \tag{1.4}$$

where $w_k \leq w$ and $h_k \leq h$, is realized through the pre-trained four-dimensional Gaussian Kernel Density Estimator (KDE) fitted on the EgoHands dataset [7]. The aim of such unit is to discover the most likely regions that enclose the hand within a proba-

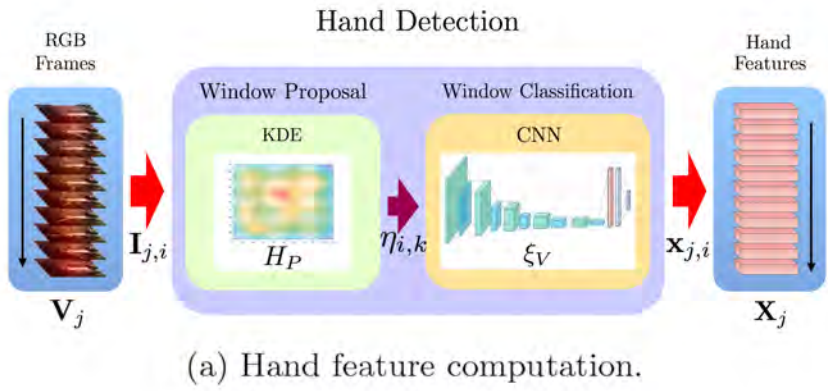


Figure 1.1 General structure of the proposed architecture. In a) we show a CNN-based approach, used to detect hands in the frames extracted from videos, encoding the associated bounding boxes into feature vectors corresponding to the activation of the penultimate fully connected layer of the CNN. In b) we show a LSTM model, trained to learn to predict the sequence of n input vectors, corresponding to the n video frames, into a sequence of n hand action primitives.

bilistic context. Accordingly, the output includes a set of candidate windows that are then passed to the *window classification* unit

$$\xi_V : \mathbb{R}^{w_k \times h_k \times 3} \longrightarrow (p, 1-p) \in \mathbb{R}^2, \tag{1.5}$$

where $p \in [0,1]$, whose target is to detect the presence of the hand in the candidate window (hereinafter, bounding box). For this purpose, we use the pre-trained CNN

developed as part of the Egohands framework [7]. The structure of the CNN is summarized in Table 1.1. Convolutional layers $-$ Conv_i $-$ extract input features through convolution operations represented by the number of filters (kernels) b with spatial size $f \times f$, applied to the input with stride s. The outcomes of convolutional layers are sequentially: saturated by ReLU_i layers (introducing non-linearity into the CNN); down-sampled by Pool_i layers (which apply a mask on $f \times f$ input regions and stride s); and, normalized by Norm_i layers. On the top of the CNN structure, there are two fully connected layers FC_6 and FC_7, respectively. These layers connect all neurons of previous layer with 4096 nodes. Finally, the *Softmax* function capitalizes on the last FC_7 layer encodings to generate a probability distribution over two classes (*hand*, *no-hand*).

Supposing $\eta_{i,k} \in \mathbb{R}^{w_k \times h_k \times 3}$ be the most likely scored bounding box enclosing the hand. We leverage on FC_6 layer encodings to characterize hand pose features. This layer comprises less task-specific information than FC_7 layer, which is closer to the Softmax layer and, thus, more dedicated to the recognition of the hand rather

Table 1.1 Description of the CNN structure. Each row describes a layer of the network, organized from input to output. (kernels b) refer to the number of kernels (each of them is a $f \times f$ matrix, whose dimensions are determined by the spatial size f) containing the convolutional parameters, stride s controls the shift step of the kernel (convolution layer) or the mask (pooling layer) around the input volume, and output size represents the output dimension (height, weight and depth in the case of a volume or the vector size).

type	**kernels (b)**	**spatial size (f)**	**stride (s)**	**output size**
Conv_1	96	11	4	55×55×96
ReLU_1	-	-	-	55×55×96
Pool_1	-	3	2	27×27×96
Norm_1	-	-	-	27×27×96
Conv_2	256	5	1	27×27×256
ReLU_2	-	-	-	27×27×256
Pool_2	-	3	2	13×13×256
Norm_2	-	-	-	13×13×256
Conv_3	384	3	1	13×13×384
Conv_4	384	3	1	13×13×384
ReLU_4	-	-	-	13×13×384
Conv_5	256	3	1	13×13×256
ReLU_5	-	-	-	13×13×256
Pool_5	-	3	2	6×6×256
FC_6	-	-	-	4096
ReLU_6	-	-	-	4096
FC_7	-	-	-	4096

than its high level information (i.e. edges, shapes, etc.). As a result, for each frame we represent the hand pose features as fixed vectorial encondings $\mathbf{x}_{j,i} \in \mathbb{R}^g$, where $g = 4100$ (the first four components refer to the bounding box coordinates, while the remaining ones include the FC_6 layer encodings). Therefore, the whole video $\mathbf{V}_j$ is now represented as sequence

$$\mathbf{X}_j = (\mathbf{x}_{j,1}, \mathbf{x}_{j,2}, \ldots, \mathbf{x}_{j,n}), \tag{1.6}$$

of such hand-pose encodings.

In the second stage (Figure 1.1-(b)), the *Sequence learning* model is trained to process the input sequence $\mathbf{X}_j$ − hand pose encodings obtained by the CNN − to predict a related output sequence of action primitives

$$\mathbf{Y}_j = (\mathbf{y}_{j,1}, \mathbf{y}_{j,2}, \ldots, \mathbf{y}_{j,n}). \tag{1.7}$$

Moreover, to model the dynamic temporal behavior, we employ a RNN − implemented using LSTM recurrent units − in the *Sequence learning* component of our architecture. Specifically, LSTM is able to provide as output the prediction for the current frame, while taking into account the history of the poses and action sequences performed by the hand in the previous instants. To this end, the action primitives (which −as aforementioned − are symbols from a discrete and finite alphabet) are converted to a numeric vector using a *one-hot* encoding. The approach encodes the k-th symbol of the action primitive alphabet as a vector of length equal to the alphabet size, where only the k-th element is set to 1, while the rest is equal to zero. More formally, the one-hot encoding of the action label for frame $\mathbf{x}_i$ is the vector $\mathbf{y_i} \in \mathbb{R}^{|S|}$ defined as

$$\mathbf{y_i}^k = \begin{cases} 1, & \text{if } k = \text{ind}(\mathbf{y_i}) \\ 0, & \text{otherwise} , \end{cases} \tag{1.8}$$

where $\text{ind}(\mathbf{y_i})$ is the index of the current label in the dictionary S.

Moreover, the details of the LSTM network are specified by model selection, using validation data outside of the training and test samples for ensuring robustness and avoiding results biased towards high precision on the test-set. Such details include, among others, the number of hidden layers and the number of LSTM units in each layer. The following experimental analysis provides details on the final configuration. More information and technical details can be found here [20].

1.2.2 Experiments

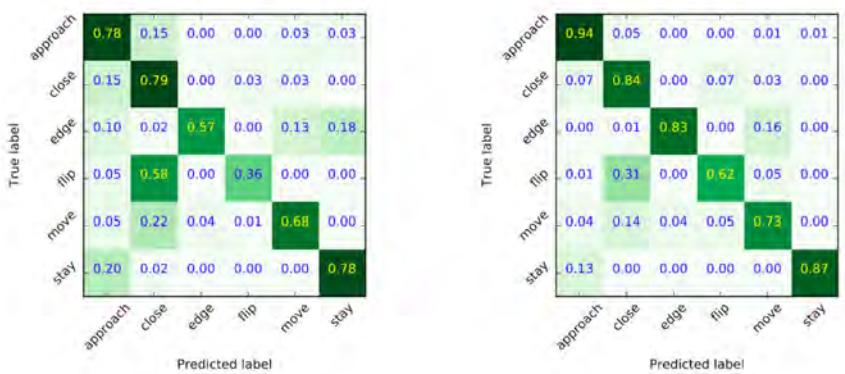
Videos were manually segmented and labeled by an experienced person using the action primitives described below. Each grasping video is represented by a combination of a subset of the following action primitives: *rest, approach, close, slide, flip, edge*.

Note that the CNN we leverage on to detect hands was trained on RGB images (hands), instead our videos are in black and white. In addition the participants wore a glove during the experiments. We thus apply a simple image segmentation filter using the *imbinarize* function in MATLAB to convert each frame into RGB coding.

In this way, we can select the pixels connected to the glove, which roughly represent the hand, and colorize them with the same color, chosen as the mean value of EgoHands ground truth [7].

1.2.2.1 Evaluation on ECE dataset

Training and testing procedures are executed on a NVIDIA Tesla M40 GPU with 12GB of onboard memory. We propose two types of cross-validation: *hold out* and *leave one out* to verify the generalization and robustness of action primitive prediction. The goal of cross-validation is to estimate the expected level of model predictive accuracy in a way that is independent from the data used to train the model.



(a) SVM benchmarking. (b) Proposed LSTM.

Figure 1.2 Performance of the proposed network, reported as confusion matrices evaluated on a benchmark SVM method and on the proposed LSTM. For each matrix, on the y-axis are reported the true classes, on the x-axis the predicted classes. Optimal performances are represented by purely diagonal matrices. Results show significantly better performances for the LSTM case, confirming the crucial relevance of temporal information for an accurate classification of action primitives.

Considering that *hold out* approach requires less computation time compared to *leave one out*, we employed it to determine both network hyper-parameters (i.e. LSTM depth and width) and learning hyper-parameters (i.e. batch size, learning rate, number of epochs and dropout). We trained 30 different network configurations which were obtained by varying respectively: number of LSTM hidden--layers in $\{1,2,3\}$, number of LSTM cells per layer in $\{64,128,256,512\}$, batch size in $\{5,10,15,20\}$, learning rate in $\{10^{-2},10^{-3},10^{-4}\}$, number of epochs in $\{10,20,30,40\}$, and dropout in $\{0.4,0.5,0.6\}$. Relying on the results of each simulation, we consider the configuration which provided both the highest *min-score* accuracy – the lowest accuracy with respect to the six classes – and *f1-score* [22] accuracy on the validation dataset – which are 73% and 91% respectively. In such

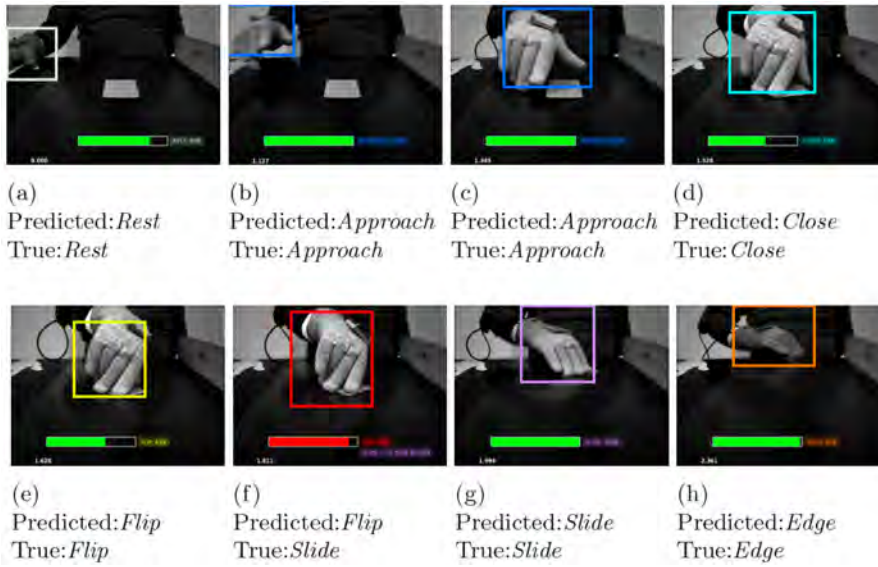


Figure 1.3 Action primitives classified by the proposed network during the reaching and grasping of a credit card. Each picture represents a time-frame of the action execution. For each frame, we show a bounding box around the hand with a color dependent on the classified action, the label associated to the predicted action, a bar indicating the level of prediction confidence (green if the classification is correct, red otherwise).

configuration, we consider three hidden layers, with respectively 256, 256, and 128 dimensions for the size of the LSTM memory. We train the network for 30 epochs using RMSprop optimizer with a fixed learning rate of 10^{-3}, batch size 20 and dropout 0.5. Furthermore, with such architecture and such learning parameters, the network is able to predict the dynamic hand strategies in the test dataset with an accuracy ranging from 75% up to 96%, depending on the action class. Normalizing scores with respect to the total number of classes, we obtain an accuracy of 85% – please refer to [20] for details.

Results of the leave one out cross-validation analyses of network performance, which necessitate a longer computation time compared to hold out approach, but guarantee more robust validation results, are reported in Figure 1.2-(b). We show the per class normalized confusion matrix index of the predictor of all six classes that were detected. What we can observe from Figure 1.2-(b) is that there is a class (*approach*) with a precision of over 94%, three classes with a precision over 83% (*edge, close, rest*), one class (*slide*) with a precision of 73% and one (*flip*) that reaches a reasonable 62%. This is a very strong result given the fact that the predictor is trained on videos with poor quality – videos are in black and white format, and subjects wore gloves. Furthermore, we provide a comparison with a benchmark Support Vector

Machine (SVM) method in Figure 1.2-(a). This approach is implemented with the purpose of noticing the accuracy discrepancy when action primitives are predicted without considering any temporal information. It is also evident from the results that LSTM outperforms SVM, confirming the crucial relevance of temporal information for an accurate classification of action primitives.

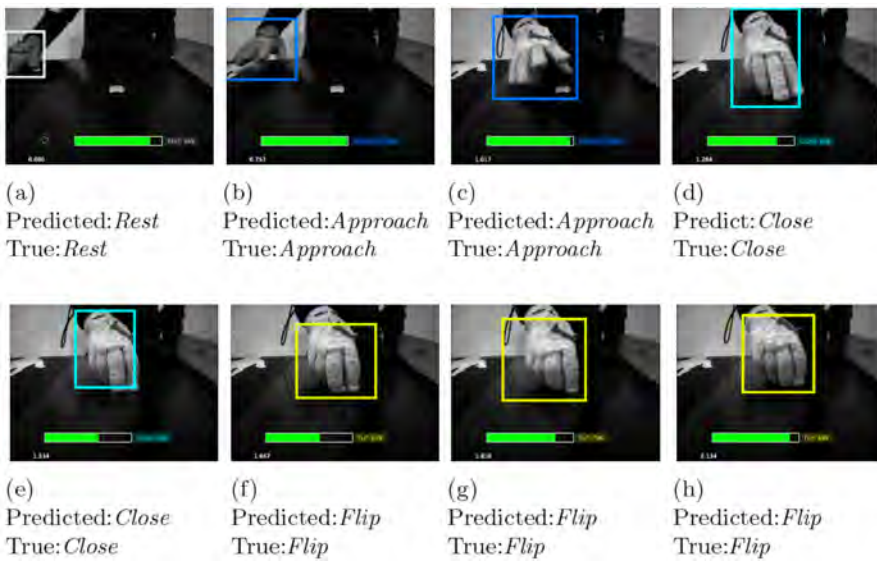


Figure 1.4 Action primitives classified by the proposed network during the reaching and grasping of a french chalk. Each picture represents a time-frame of the action execution. For each frame, we show a bounding box around the hand with a color dependent on the classified action, the label associated to the predicted action, a bar indicating the level of prediction confidence (green if the classification is correct, red otherwise).

Figures 1.3 and 1.4 show some examples of action primitives predicted by our DeepDynamicHand model on ECE dataset. In each video frame we overlaid (i) the bounding box that likely encloses the hand, color changes depending on the action primitive the hand currently performs, (ii) the predicted action label, and in case of failure, the true label, and (iii) the level of confidence represented with a filled rectangle – the color is green if the predicted action is true, red otherwise. We can observe that wrong classification usually happens when a transaction between primitives occurs. Frame 1.3-(f) refers to such example, showing that confusion may arise in situations where hand shapes are hardly distinguishable also by a human.

1.3 Autonomous Grasping With Anthropomorphic Soft Hands

In the previous section, we reported on how the human example can be investigated using DL techniques to identify common action primitives, which take into account the compliance of the hands for the exploitation of the environmental constraints during manipulation tasks. In this section, we describe the usage of DL to translate human observations on the robotic side, for autonomous grasping with soft hands [23]. More specifically, we considered a soft articulated robotic hand, the Pisa/IIT SoftHand [3], although the approach can be generalized to any soft end effector. In our solution, which was presented in [23], the intelligence is distributed on three levels of abstractions, see Figure 1.5: (i) *high-level*: a classifier for planning the right action, choosing among a set of available ones, ii) *medium level*: a set of human-inspired low level strategies implementing both the approaching phase and the sensor-triggered reaction, iii) *low level*: a soft hand whose embodied intelligence mechanically manages local uncertainties. All the three levels are human-inspired. We report in the next sub-sections the detailed description of these components.

1.3.1 High-Level: Deep Classifier

The target of the deep neural network is the association to an object, detected from the scene using a RGB camera, the primitive, intended as the temporal evolution of the hand pose, that humans would likely perform for grasping it. The object detection is implemented using the state of the art detector YOLOv2 [24], which produces as output a set of labeled bounding boxes containing all the objects in the scene from the RGB input image. This output is fed to the second classifier, which is built moving from Inception-v3 [25] trained on the Image Net dataset. The goal is not to obtain a one-to-one signature of a particular object, but, on the contrary, to achieve a semantic description that can be applied to objects with similar geometric characteristics. We modified the Inception-v3 and added two fully connected layers (2048 neurons each, with ReLU activation), which perform an adaptive non-linear combination and refinement of the features. The outcome of the last layer acts as input to the Softmax, whose output is a probability distribution on the set of motion primitives.

The deep classifier was trained using 6336 first person RGB videos (single-object, table-top scenario), which were collected from 11 right-handed subjects grasping 36 objects, see Figure 1.6 − which cover most of grasping possibilities in everyday life − from different points of view (4), see Figure 1.9. Videos were visually inspected to extract and label the main strategies, resulting in a set of ten primitives, i.e. *Top*, *Top Left*, *Top Right*, *Bottom*, *Pinch*, *Pinch Right*, *Slide*, *Flip* and *Lateral*. For more details on primitive description, please refer to [23]. The choice of these primitives was done taking inspiration from literature [4, 26], and to provide a representative yet concise description of human behavior, without any claim of exhaustiveness. Note that the selection of the action primitive is also object-configuration dependent.

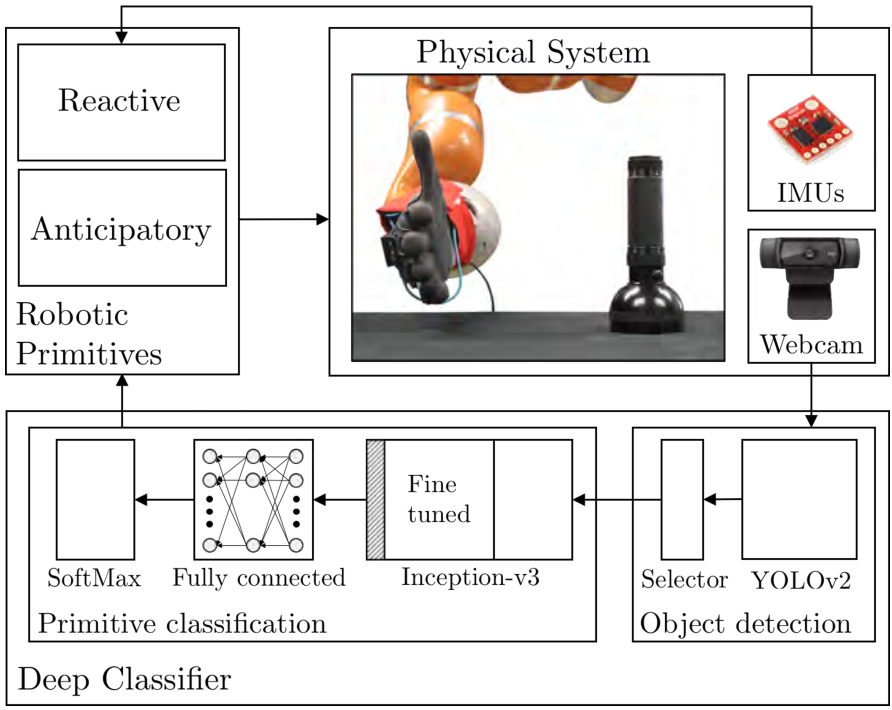


Figure 1.5 General description of the proposed methodology, combining reactive actions and anticipatory behavior. A deep classifier observes the scene and − given a specific object − predicts the approaching strategy a human would implement to grasp the object. This semantic description is used to select a corresponding motion primitive on the robot, in terms of hand posture over time. IMUs fastened on the fingers detect contact with the object and triggers a reactive grasp behavior.

1.3.1.1 Object detection

The activity of object detection is achieved through a YOLOv2 detector [24]. More specifically, given a RGB image as input, YOLOv2 is able to provide as output a series of labeled bounding boxes that contain all the objects in the scene. Among all, we first discard all the ones labeled as person. Then, assuming that the target is placed close to the center of the picture, we select the bounding box closest to the scene center. Note that this is an arbitrary selection, made only for implementation purposes and can be easily generalized. After the identification of the particular object, the picture is automatically cropped around the bounding box, and resized to 416×416 pixels. The result of this procedure is then used as input for a second block used for classification.

Figure 1.6 Set of objects (36) used during the experiments with human participants. Pictures are not in scale.

1.3.1.2 Primitive Classification

Network Architecture

The Primitive Classification is based on a transfer learning approach [27]. This approach leverages on prior knowledge – learned from one environment – to solve a new problem, typically related but different in general. With this approach, a reduced amount of samples is needed to train the model, thus resulting in shorter training time while preserving high accuracy of results. To do this, we leveraged on Inception-v3 [25], trained on the ImageNet dataset [28] to classify objects from images. We keep the early and middle layers and remove the Softmax layer. This enables a direct access to the informative, highly refined, neural features that Inception-v3 uses for the classification. It is worth noticing that the object classification is not intended as one-to-one, but rather it aims at extracting high level – semantic – descriptions that can be easily transferred to objects with similar characteristics. On the top of the

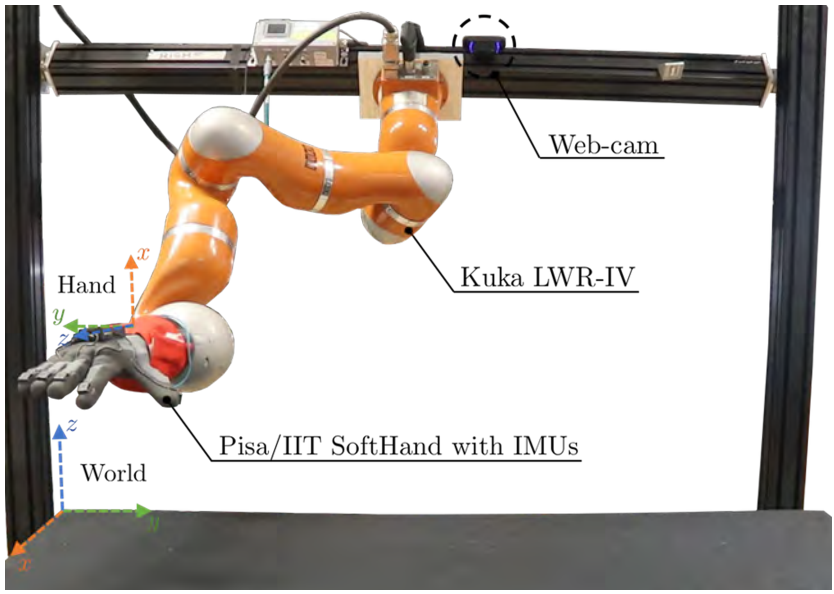


Figure 1.7 Robotic platform used for the experiments. A Kuka LWR is mounted on a rigid framework and equipped with a Pisa/IIT SoftHand as end-effector. The scene is recorded through a camera mounted at the robot base. Hand is sensorized through Inertial Measurement Units (IMUs) placed on the back of the fingertips. Local and Global reference systems are also reported.

original architecture we also included two fully connected layers containing 2048 neurons each (with ReLU activation function). These layers introduce a non-linear, adaptive, combination of the high-level features identified by the convolutional and pooling layers, further refining the information. In this way, the geometric features are implicitly linked each other to serve as the base for the classification. The output of the last fully-connected layer is thus fed into the Softmax layer, which produces as output a probability distribution over the considered set of motion primitives. The output of the overall network is then the motion primitive that shows the maximum probability.

Training and validation

The network was trained using the labeled dataset introduced above. More specifically, while the original parameters of the Inception-v3 were fine-tuned for the specific task, the parameters of the two fully-connected layers placed at the top of the architecture were trained from scratch. This was achieved using a different learning rate for the layers. Indeed, the weights of the first 172 layers – which are more devoted to universal features, like curves and edges – were maintained unchanged, while the remaining 77 layers – used to capture features more related to the specific dataset – were re-trained in this implementation. Changes on the latter were limited

by a relatively small learning rate λ_{ft}. Finally, the last two fully-connected layers were randomly initialized and trained with a higher learning rate.

We minimized the risk of over-fitting by using a dropout policy [29]. More specifically, for every new training sample presented to the network, we randomly disconnected a set of neurons by masking their activation. Each neuron has a specific disconnection probability p_{drop}. This results in a new – different – topology of the network after each training, with the ultimate result to introduce variability and minimize the arising of unwanted co-adaptation of weights. Network design and training was performed through Keras library [30] leveraging on a NVIDIA Tesla M40 GPU with 12GB of memory on-board.

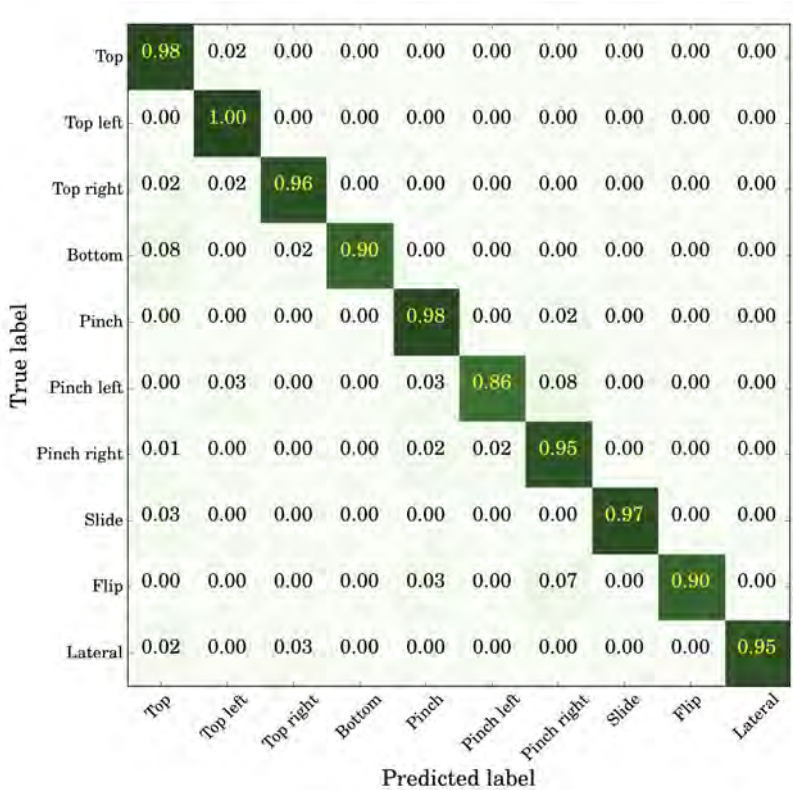


Figure 1.8 Performance of the proposed deep classifier, reported as confusion matrix evaluated on the test set. Each element of the matrix reports the percentage of cases in which the primitive identified (identified by the row label) is classified as the primitive associated with the column label. Optimal performances are represented by purely diagonal matrices. Numerical values are color-coded: white is low rate, dark green is high rate.

A hold-out validation was then used to verify the robustness and the generalization capabilities of primitive classification. To do this, we split our samples in three datasets: 70% of samples used for the training phase; 20% of samples used for the validation phase; 10% of samples used for the testing phase. The three sub-datasets were organized with a balanced representation of objects for each class we considered. We performed the training using 30 different configurations of the network, leveraging on *cross entropy* cost function to adjust the weights by calculating the error between the output of the Softmax layer and the label vector of the given sample category. Each configuration resulted from the variation of the most relevant hyper-parameters of the learning phase, i.e. probability of dropout $p_{\text{drop}} \in \{0.4, 0.5, 0.6\}$, learning rates $\lambda_{\text{ft}} \in \{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$ and $\lambda_{\text{tr}} \in \{10^{-2}, 10^{-3}, 10^{-4}\}$, number of epochs in $\{10, 20, 30, 40\}$ and batch size $\in \{10, 20, 30, 40\}$. The training time employed $-$ for each network $-$ was between 1 and 5 hours. Among all the configurations trained, we selected the one that provided the highest $f1$-score accuracy [22] on the validation set. In our case, we had a maximum value of 97% with the following hyper-parameters: $p_{\text{drop}} = 0.5$, $\lambda_{\text{ft}} = 10^{-5}$, $\lambda_{\text{tr}} = 10^{-3}$, 30 epochs and batches size 20.

The network with these parameters resulted able to correctly classify the motion primitives with an accuracy between 86% and 100%, depending on the primitive, with an average value equal to 95%. In Figure 1.8 we show the normalized accuracy of the proposed network for the 10 classes considered in this work. It is worth noticing that the occasional failure of the network is related to two main reasons. First, the formulation of the problem itself makes intrinsically not possible an accuracy of 100% , since different subjects may use different grasping strategy for the same object in the same configuration. An example is the grasp of the coin, which is typically grasped through a flip strategy, but sometimes may be grasped leveraging on a sliding primitive instead. Another reason for occasional failure is related to the experimental setup. Indeed, the usage of a single RGB camera sometimes results in a misinterpretation of the object size which, for example, could lead to the prediction of a top grasp instead of a bottom grasp for a bowl. This particular problem can be solved through the usage of a stereo-camera; this extension is currently under evaluation and is left for future developments of this work.

1.3.2 Transferring Grasping Primitives to Robots

In [31], Johansson and Edin hypothesized that the Central Nervous System (CNS)

> *monitors specific, more-or-less expected, peripheral sensory events and use these to directly apply control signals that are appropriate for the current task and its phase.*

This means that control signals descending from the CNS are more likely computed in advance (i.e. feedforward/anticipatory actions). Motivated by this observation, our translation of human behavior in a robotic framework was implemented leveraging mostly on feedforward actions. To do this, we moved from the observations discussed in the previous section of this chapter and defined a basis of human-inspired motion primitives, to be triggered by specific events. The first triggering

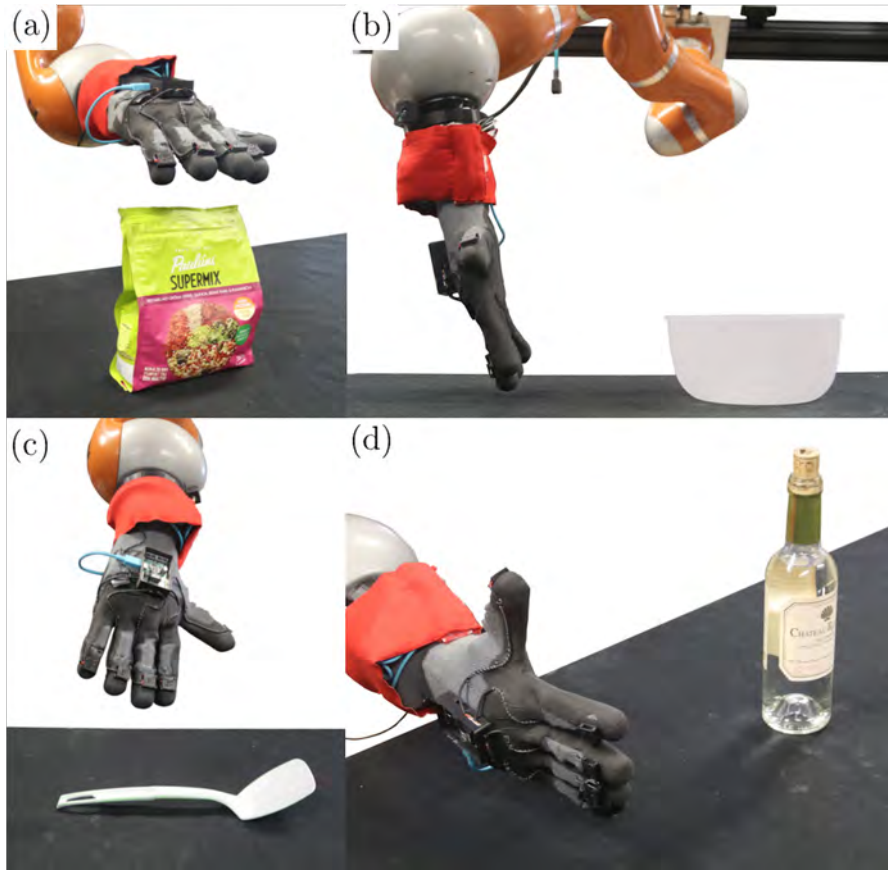


Figure 1.9 Initial configuration of the hand w.r.t. the object at the beginning of four different motion primitives: a) top grasp; b) bottom grasp; c) pinch grasp; d) lateral grasp. Moving from these initial poses, the hand translates until the contact with the object is detected by the IMUs. The contact triggers a reactive behavior.

event is associated with the object detection. To each object, the framework associates a corresponding grasping primitive. Among all the primitives included in this work, we do not consider here the flip, because this cannot be implemented by the particular end-effector used in this work. Motor primitives are divide in two main phases: i) approaching and ii) reactive adaptation. The transition between the two phases is triggered by the detection of a contact event perceived by the Inertial Measurement Units (IMUs) mounted on the back of the fingers of the softhand.

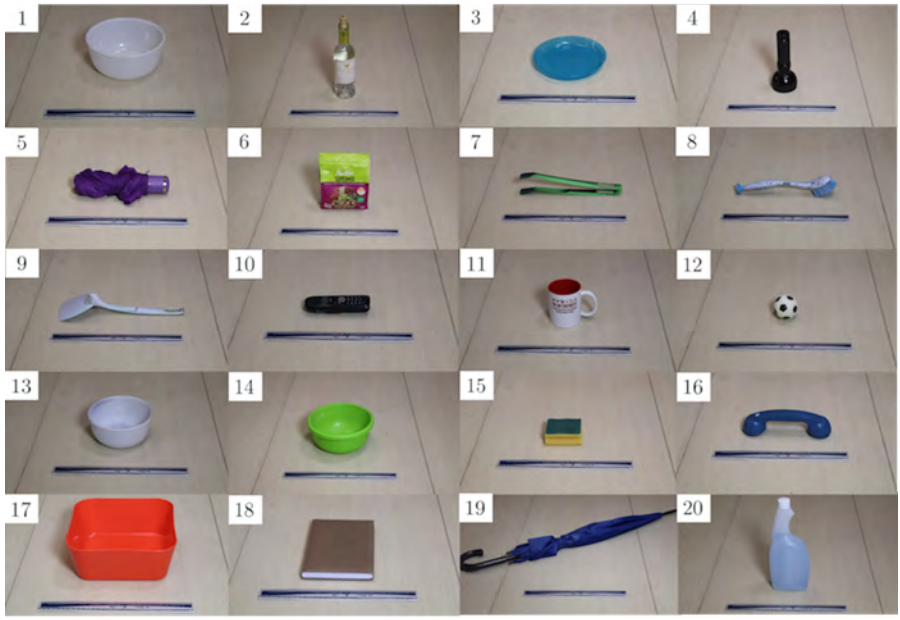


Figure 1.10 *Set of objects (20) used during the experiments with the robotic platform for the experimental validation. Note that all the selected objects are different w.r.t. the ones used during the training phase. A* 30*cm ruler is placed in all the pictures to enable a qualitatively perception of the objects size.*

1.3.3 Experimental setup

Even if the proposed approach is general and not limited to the specific robotic architecture, it is convenient to report here its description to increase clarity. The platform we used is composed by a Pisa/IIT SoftHand [32] mounted as end-effector on a KUKA LWR-IV arm. The particular end-effector used is an anthropomorphic soft robotic hand with 19 Degrees of Freedom, which are jointly (following synergistic covariation patterns) actuated by one single Degree of Actuation. It is worth noticing that the particular design and control of the end effector represent a low-level *intelligence embodied*, integral part of the control architecture itself. Video information are recorded through a RGB camera, fastened close to the robot base to generate a first-person point-of-view.

The soft hand is endowed with inertial sensors (IMUs) placed on the back of the fingertips. Measures recorded from IMUs (i.e. accelerations) are needed to detect contacts and trigger reactive grasping [5] strategies. Figure 1.7 shows the whole experimental setup, with reference frames for the global and end effector reference systems.

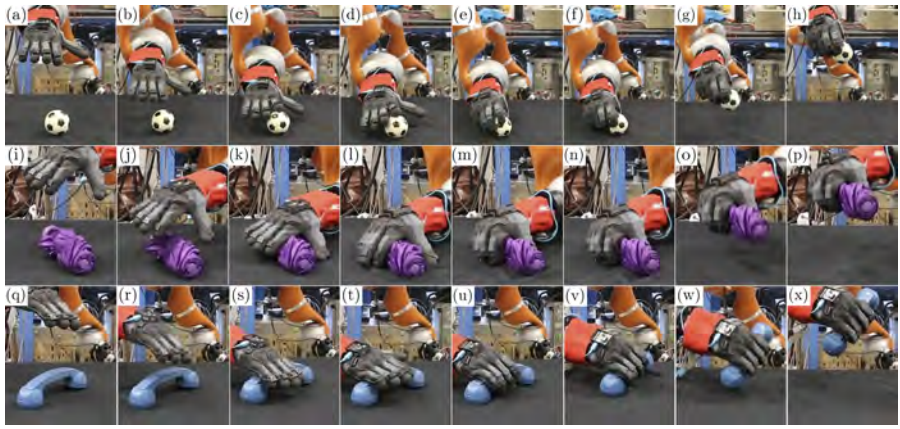


Figure 1.11 *Snapshot of power top grasps generated through the proposed architecture. Subfigs a-h show a top grasp performed on object 12; subfigs i-p show a top-left grasp performed on object 5; subfigs q-x show a top-right grasp performed on object 16. For each line, it is possible to observe an approaching phase (e.g. subfigs a-b), a contact detection (e.g. subfig c), a reactive behavior for posture adaptation (e.g. subfigs c-f) and a firm grasp and lift of the object (e.g. subfigs g-h).*

Figure 1.12 *Snapshot of a bottom grasp performed on object 14. Subfig a shows the initial configuration of the primitive. In b the contact triggers the reactive routine. Finally, in subfig f the item is firmly lifted.*

1.3.3.1 Approach phase

Given the starting pose, the approaching phase toward the object position is encoded via the following Cartesian trajectory

$$x(t) = x_0 + dtQ(t) = Q_0, \tag{1.9}$$

where $x \in \mathbb{R}^3$ represents the hand 3D Cartesian position and $Q \in \mathbb{R}^4$ is the quaternion that encodes its orientation (all w.r.t. the global reference system); $x_0 \in \mathbb{R}^3$ and $Q_0 \in \mathbb{R}^4$ refer to the initial position and orientation and $d \in \mathbb{R}^3$ is the approaching direction. All these three quantities are defined by the selected primitive, and dictated by the aim of heuristically reproducing as close as possible the human behavior observed in the videos. Figure 1.9 depicts the initial configuration of the hand for four directions of approach, while Table 1.2 collects numerical values of initial orientations and approaching directions for all the primitive implemented in this work.

Table 1.2 Initial orientation Q_0 and normalized direction of approach $\hat{d}$ for each primitive.

Strategy	Q_0^{T}	$\hat{d}^{\mathrm{T}}$
Top	[0.0 0.711 0.0 0.703]	[0 0 −1]
Top left	[0.269 0.6570 −0.2721 0.6496]	[0 0 −1]
Top right	[0.269 −0.657 −0.272 −0.649]	[0 0 −1]
Bottom	[0.145 −0.696 0.701 0.030]	[0 1 0]
Pinch	[0.084 0.816 0.17 0.458]	[0 0 −1]
Pinch left	[0.116 0.733 0.483 0.463]	[0 0 −1]
Pinch right	[0.186 0.890 −0.110 0.400]	[0 0 −1]
Slide	[0.0 0.711 0.0 0.703]	[0 0 −1]
Lateral	[0 −1 0 0]	[0 1 0]

1.3.3.2 Grasp phase

Following the approaching phase, the action is intended to proceed with the grasp of the detected object. This is the phase in which primitives differentiate more from each others. On the following, we will review the grasping strategies implemented in this work. Note that, when not differently specified, hand movements are intended in local coordinates.

Top and lateral grasps

The local wrist/hand adaptation around the object leverages on the reactive grasp framework [5], moving from a set of 13 basis adaptation movements of the end effector w.r.t. the contacted object. A detailed description of these strategies is reported in [5], on the following we will recall some concepts, useful for the understanding of this chapter. In [33], a human participant was asked to reach and grasp a tennis ball placed on a table, while actively controlling the SoftHand through an interface endowed with a lever, which is used to accomplish a grasping task. The user was instructed to continuously move the hand until a contact with the object was perceived, then adapt the hand orientation w.r.t. the object so to favour the grasp. Movements were replicated 13 times, considering different approaching direction. For each movement, the 3D pose of the hand was recorded via motion tracking (PhaseSpace) and synchronized with the recordings of fingers accelerations $\alpha_1 \dots \alpha_{13} : [0,T] \rightarrow \mathbb{R}^5$ from IMUs, and the hand motor current (to keep track of the hand closure). This resulted in the association between acceleration profiles and corresponding reactive adaptation movements of the hand w.r.t. the object. In [23], when the Pisa/IIT SoftHand touches an object, acceleration profiles $a : [0,T] \rightarrow \mathbb{R}^5$ are recorded through the IMUs. Then, one reactive strategy − defined by the local

rearrangement − is selected as the one that maximizes the covariation of measured IMUs signals with the recorded ones:

$$j = \arg\max_i \int_0^T a^{\mathrm{T}}(\tau)\alpha_i(\tau)\mathrm{d}\tau \,. \tag{1.10}$$

When this adaptation w.r.t. the object is completed, the hand closes around the object. Preliminary experiments on top grasps, discussed in [33], proved the effectiveness of this approach, which is here extended to top right, top left and lateral strategies.

Bottom
In case of large concave objects, to mimic human behavior, when the contact is triggered, the hand rotates along the x axis of an angle equal to $\pi/3$ and translate along y axis of 300mm. This roto-translation is determined to let the palm move over and enables the thumb entering into the object concave part during hand closure.

Pinches
For pinch strategies, the hand is programmed to close without any change in relative pose. Note that the end effector we used in this work is onceived for power grasps, but the interaction with the environment enables the execution of pinch grasping actions.

Slide
To account for objects difficult to be grasped via power of pinch strategies (e.g. a book), we implemented a multi-phase anticipatory strategy, triggered, as usual, by the contact with the object:

1. exert a normal force on the object (x axis) to preserve the contact during sliding (commanding a ref. position 10mm below the contact position);
2. translate toward the table edge (sliding);
3. remove the normal force (inverse of step 1);
4. translate (translation 100mm along x and 50mm along z) and rotate ($\pi/12$ radians around y) the hand to favour the grasp;
5. close the hand.

1.3.3.3 Control Strategy

A Jacobian-based Inverse kinematic algorithm is used to map the Cartesian references provided by the motion primitives to the joint level q_r. Then, a joint impedance control is used to realize the movement, with stiffness $K = 10^3 \frac{\mathrm{Nm}}{\mathrm{rad}}$ and damping $D = 0.7 \frac{\mathrm{Nms}}{\mathrm{rad}}$. The resulting control law is $\tau(t) = Ke(t) + D\dot{e}(t) + \mathbb{D}(q, \dot{q})$, where τ is the vector of joint torques, while $e = q_r - q$ and $\dot{e} = \dot{q}$ are the joint tracking error and its derivative, respecitvely. $\mathbb{D}$ is a compensation of the robot dynamics evaluated by the KUKA embedded controller. Control and strategies implementation was developed in ROS.

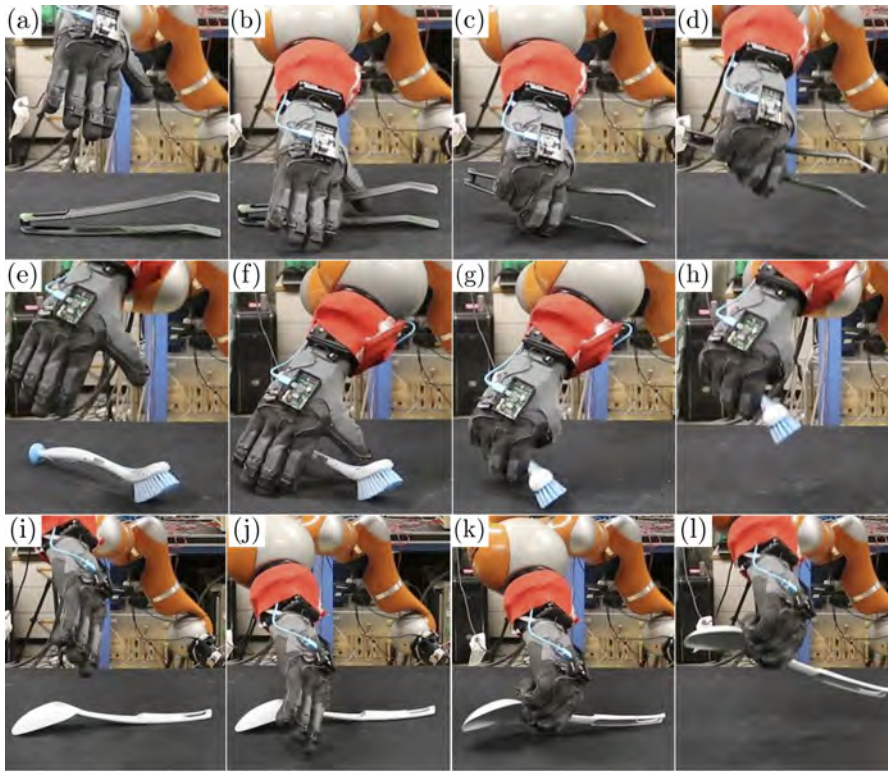


Figure 1.13 *Snapshot of pinch grasps generated through the proposed architecture. Subfigs a-d show a pinch grasp performed on object 7; subfigs e-h show a pinch-left grasp performed on object 8; subfigs i-l show a pinch-right grasp performed on object 9. For each line, it is possible to observe an approaching phase (e.g. subfigs a-b),a contact detection (e.g. subfig b), and a firm grasp and lift of the object (e.g. subfig c-d).*

1.3.4 Results

We evaluated the performances achieved by the proposed framework by performing a set of experiments in a table-top scenario. A flat surface is placed in front of the manipulator, as reported in Figure 1.7. One of the objects is placed by an operator approximatively in the center of the table. RGB data extracted from the camera are used to classify through the proposed deep neural network. The classification outcome is then associated to the corresponding motion primitive. Each object is randomly proposed three times. Position and orientation of the object can vary at every time, taking a random position inside a circle of $\sim$ 100mm centered in the table center. Tests were performed with 20 different objects (see Figure 1.10). None of these was used during the training phase. The particular selection was made so as to consider all the grasping strategies included in the study. Note that for objects 5,

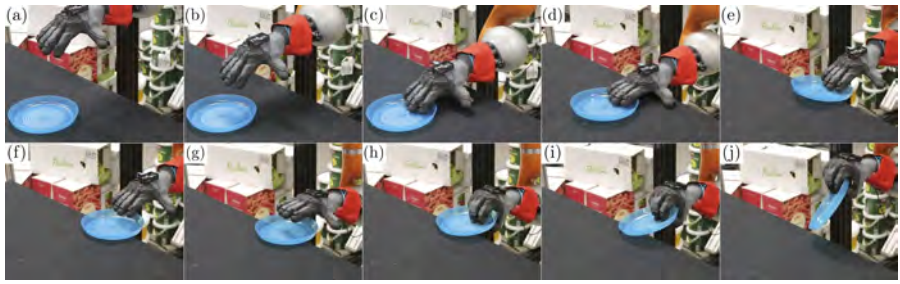


Figure 1.14 *Snapshot of slide grasp performed on object 3. Subfigs a-c show the approaching phase. Subfigs d-e show how the system exploits the environment toward the table edge. Subfigs f-g show the hand changing its relative configuration w.r.t. the object, established in Subfigs h-i. Finally, in Subfig j the item is firmly lifted.*

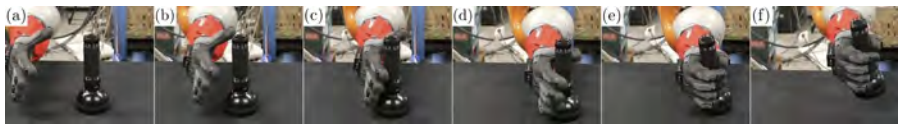


Figure 1.15 *Snapshot of lateral grasp performed on object 4. Subfigs a-c show the approaching phase. In e the grasp is achieved and in f the item is firmly lifted.*

6, 7, 8, 9, 10, 16, and 19 the classification can point to different strategy depending on the object position/orientation. The total amount of configurations tested is 111.

Table 1.3 collects the results of this study. We achieved an overall success rate equal to 81.1% (a grasp is labeled as successful if the closure is maintained for 5 seconds). It is worth noticing that objects 15 and 12 − which are rotationally symmetric − are classified as associated to top grasp primitives, regardless the specific orientation under testing. Regarding the success rate for each primitive, our results show the following: Top 85.7% (Figure 1.11 (a-h)), Top left 73.3% (Figure 1.11 (i-p)), Top right 100% (Figure 1.11 (q-x)), Bottom 100% (Figure 1.12), Pinch 55.6% (Figure 1.13 (a-d)), Pinch left 55.6% (Figure 1.13 (e-h)), Pinch right 66.7% (Figure 1.13 (i-e)), Slide 83.3% (Figure 1.14), Lateral 86.7% (Figure 1.15).

1.4 Discussion and Conclusions

One of the key enablers for the extraordinary dexterity of human hands is their compliance and capability to purposefully adapt with the environment, to multiply their manipulation possibilities. This observation has also produced a significant paradigm shift for the design of robotic hands, leading to the avenue of soft end-effectors that embed elastic and deformable elements directly in their mechanical architecture. This shift has also determined a perspective change for the control and

Table 1.3 Strategy used, successes and failures for each grasp.

Object	Strategy	Successes	Failures
1	bottom	3	0
2	lateral	2	1
3	slide	2	1
4	lateral	3	0
5	top	3	0
	top left	2	1
	top right	3	0
6	lateral	3	0
	top	3	0
	top left	2	1
	top right	3	0
7	pinch	3	0
	pinch left	2	1
	pinch right	3	0
8	pinch	2	1
	pinch left	2	1
	pinch right	2	1
9	pinch	0	3
	pinch left	1	2
	pinch right	1	2
10	top	3	0
	top left	2	1
	top right	3	0
11	lateral	3	0
12	top	2	1
13	bottom	3	0
14	bottom	3	0
15	top	2	1
16	top	2	1
	top left	3	0
	top right	3	0
17	bottom	3	0
18	slide	3	0
19	top	3	0
	top left	2	1
	top right	3	0
20	lateral	2	1
Total	-	90	21

planning of the grasping phases, with respect to the classical approach used with rigid grippers. Machine-learning (ML) and, especially, deep-learning (DL) can be promising tools to overcome the modelling challenges that arise when dealing with soft bodies and their interaction with the external environment, targeting solutions close enough to the desired ones, rather than exact, leveraging on the adaptation capabilities of the soft hands to overcome local uncertainties.

More specifically, in this chapter we report on how DL can be used to model human behavior and to translate this modelling for autonomous grasping with robotic softhands. In the first section, we propose an approach that combines CNN and RNN, and represents the first attempt to include also dynamic information for classifying different time related action primitives, which are used by humans for grasping and manipulation tasks. This idea will be further explored and tested to extract an exhaustive description of human grasping and manipulation with the environment, and to devise effective guidelines for the translation of human observations on the robotic side [20]. In the second section, we report on how the human example can be used for autonomous grasping of the softhands using DL [23]. This work represents − together with [10] − the first attempt to validate − over a large set of objects − the combination of soft robotic hands and deep learning techniques for autonomous grasping, with a success rate of 81%.

In the future we will work to integrate the results from [20] within the framework reported in [23]. The objective will be to increase the dataset of the primitives to be implemented with soft manipulators. On the other side, we aim at testing the DL human-inspired approach with other robotic softhands, e.g. continuously deformable [2], and investigate sensing strategies to predict grasping failures. We do believe, that Deep Learning can be an effective solution to devise control guidelines for soft end-effectors that autonomously perform manipulation and grasping tasks following the human example.

1.5 Acknowledgment

This work has received founding from the European Union's Horizon 2020 research and innovation program under grant agreement No.688857 (SoftPro), No. 732737 (Iliad), and by the Italian Ministry of Education and Research (MIUR) in the framework of the CrossLab project (Department of Excellence). The content of this publication is the sole responsibility of the authors. The European Commission or its services cannot be held responsible for any use that may be made of the information it contains.

References

[1] Prattichizzo D, Trinkle JC. Grasping. In: Springer handbook of robotics. Springer; 2016. p. 955–988.

[2] Deimel R, Brock O. A novel type of compliant and underactuated robotic hand for dexterous grasping. The International Journal of Robotics Research. 2016;35(1-3):161–185.

[3] Catalano MG, Grioli G, Farnioli E, et al. Adaptive synergies for the design and control of the Pisa/IIT SoftHand. The International Journal of Robotics Research. 2014;33(5):768–782.

[4] Eppner C, Deimel R, Alvarez-Ruiz J, et al. Exploitation of environmental constraints in human and robotic grasping. The International Journal of Robotics Research. 2015;34(7):1021–1038.

[5] Bianchi M, Averta G, Battaglia E, et al. Touch-Based Grasp Primitives for Soft Hands: Applications to Human-to-Robot Handover Tasks and Beyond. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE; 2018. p. 7794–7801.

[6] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521(7553):436–444.

[7] Bambach S, Lee S, Crandall DJ, et al. Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions. In: Proceedings of the IEEE International Conference on Computer Vision; 2015. p. 1949–1957.

[8] Gupta A, Eppner C, Levine S, et al. Learning dexterous manipulation for a soft robotic hand from human demonstrations. In: Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on. IEEE; 2016. p. 3786–3793.

[9] Nishimura T, Mizushima K, Suzuki Y, et al. Thin plate manipulation by an under-actuated robotic soft gripper utilizing the environment. In: Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on. IEEE; 2017. p. 1236–1243.

[10] Choi C, Schwarting W, DelPreto J, et al. Learning Object Grasping for Soft Robot Hands. IEEE Robotics and Automation Letters. 2018;.

[11] Puhlmann S, Heinemann F, Brock O, et al. A compact representation of human single-object grasping. In: Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on. IEEE; 2016. p. 1954–1959.

[12] Ong EJ, Bowden R. A boosted classifier tree for hand shape detection. In: Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on. IEEE; 2004. p. 889–894.

[13] Beh J, Han DK, Durasiwami R, et al. Hidden Markov Model on a unit hypersphere space for gesture trajectory recognition. Pattern recognition letters. 2014;36:144–153.

[14] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:14091556. 2014;.

[15] Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2014. p. 580–587.

[16] Karpathy A, Toderici G, Shetty S, et al. Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition; 2014. p. 1725–1732.

[17] Ng JYH, Hausknecht M, Vijayanarasimhan S, et al. Beyond short snippets: Deep networks for video classification. In: Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on. IEEE; 2015. p. 4694–4702.

[18] Sudhakaran S, Lanz O. Convolutional Long Short-Term Memory Networks for Recognizing First Person Interactions. In: Computer Vision Workshop (ICCVW), 2017 IEEE International Conference on. IEEE; 2017. p. 2339–2346.

[19] Nguyen A, Kanoulas D, Muratore L, et al. Translating Videos to Commands for Robotic Manipulation with Deep Recurrent Neural Networks. arXiv preprint arXiv:171000290. 2017;.

[20] Arapi V, Della Santina C, Bacciu D, et al. DeepDynamicHand: A Deep Neural Architecture for Labeling Hand Manipulation Strategies in Video Sources Exploiting Temporal Information. Frontiers in Neurorobotics. 2018;12:86. Available from: https://www.frontiersin.org/article/10.3389/fnbot.2018.00086.

[21] Hochreiter S, Schmidhuber J. Long short-term memory. Neural computation. 1997;9(8):1735–1780.

[22] Yang Y, Liu X. A re-examination of text categorization methods. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. ACM; 1999. p. 42–49.

[23] Della Santina C, Arapi V, Averta G, et al. Learning from humans how to grasp: a data-driven architecture for autonomous grasping with anthropomorphic soft hands. IEEE Robotics and Automation Letters. 2019;4(2):1533–1540.

[24] Redmon J, Farhadi A. YOLO9000: better, faster, stronger. arXiv preprint. 2017;.

[25] Szegedy C, Vanhoucke V, Ioffe S, et al. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 2818–2826.

[26] Feix T, Romero J, Schmiedmayer HB, et al. The grasp taxonomy of human grasp types. IEEE Transactions on Human-Machine Systems. 2016;46(1):66–77.

[27] Thrun S, Pratt L. Learning to learn. Springer Science & Business Media; 2012.

[28] Deng J, Dong W, Socher R, et al. Imagenet: A large-scale hierarchical image database. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. Ieee; 2009. p. 248–255.

[29] Srivastava N, Hinton G, Krizhevsky A, et al. Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research. 2014;15(1):1929–1958.

[30] Chollet F, et al.. Keras; 2015.

[31] Johansson RS, Edin BB. Predictive feed-forward sensory control during grasping and manipulation in man. BIOMEDICAL RESEARCH-TOKYO-. 1993;14:95–95.

[32] Della Santina C, Piazza C, Gasparri GM, et al. The quest for natural machine motion: An open platform to fast-prototyping articulated soft robots. IEEE Robotics & Automation Magazine. 2017;24(1):48–56.

[33] Bianchi M, Averta G, Battaglia E, et al. Tactile-Based Grasp Primitives for Soft Hands: Applications to Human-to-Robot Handover Tasks and Beyond. In: Robotics and Automation (ICRA), 2018 IEEE International Conference on. IEEE; 2019. .