Using Gaussian mixture models for gesture recognition during haptically guided telemanipulation

Pérez-Del-Pulgar, Carlos J.; Smisek, Jan; Rivas-Blanco, Irene; Schiele, Andre; Muñoz, Victor F.

DOI
10.3390/electronics8070772

Publication date
2019

Document Version
Final published version

Published in
Electronics (Switzerland)

electronics

MDPI

Article

Using Gaussian Mixture Models for Gesture Recognition During Haptically Guided Telemanipulation

Carlos J. Pérez-del-Pulgar [1,*], Jan Smisek [2,3], Irene Rivas-Blanco [1], Andre Schiele [2,3] and Victor F. Muñoz [1]

[1] Department of Systems Engineering and Automation, Universidad de Málaga, Andalucía Tech, 29071 Málaga, Spain
[2] Delft University of Technology, 2628 Delft, The Netherlands
[3] Telerobotics and Haptics lab, European Space Agency, 2201 Noordwijk, The Netherlands
* Correspondence: carlosperez@uma.es; Tel.: +34-951-952-324

Received: 14 June 2019; Accepted: 4 July 2019; Published: 10 July 2019

Abstract: Haptic guidance is a promising method for assisting an operator in solving robotic remote operation tasks. It can be implemented through different methods, such as virtual fixtures, where a predefined trajectory is used to generate guidance forces, or interactive guidance, where sensor measurements are used to assist the operator in real-time. During the last years, the use of learning from demonstration (LfD) has been proposed to perform interactive guidance based on simple tasks that are usually composed of a single stage. However, it would be desirable to improve this approach to solve complex tasks composed of several stages or gestures. This paper extends the LfD approach for object telemanipulation where the task to be solved is divided into a set of gestures that need to be detected. Thus, each gesture is previously trained and encoded within a Gaussian mixture model using LfD, and stored in a gesture library. During telemanipulation, depending on the sensory information, the gesture that is being carried out is recognized using the same LfD trained model for haptic guidance. The method was experimentally verified in a teleoperated peg-in-hole insertion task. A KUKA LWR4+ lightweight robot was remotely controlled with a Sigma.7 haptic device with LfD-based shared control. Finally, a comparison was carried out to evaluate the performance of Gaussian mixture models with a well-established gesture recognition method, continuous hidden Markov models, for the same task. Results show that the Gaussian mixture models (GMM)-based method slightly improves the success rate, with lower training and recognition processing times.

Keywords: robotics; telemanipulation; haptics; machine learning; gesture recognition

1. Introduction

In telemanipulation, a human operator performs a task in a distant environment by remotely controlling a robot. To allow efficient operation, the operator needs to receive sensory information from the remote site. Depending on the received information, telemanipulation can be classified as "direct" or "uni-lateral" [1], where there is no feedback to the operator, or "bilateral" [2], which enables dual interaction between the haptic and the operator. Although telemanipulation allows real-time human remote control, it is still considered to entail a rather high workload [3], at least compared with more supervisory or autonomous modes of operation. However, only telemanipulation allows reacting to unknown and unforeseen situations with spontaneous feedback. Therefore, enriching telemanipulation with additional automatic assistance would allow humans to perform complex tasks more efficiently.

In this sense, haptically guided telemanipulation [4] is shown to be a promising method that reduces the operator workload and can improve his or her performance. Haptic guidance is

usually implemented by adding virtual channels into the feedback path (e.g., the force output of a virtual spring) that generates appropriate forces to constrain the operator input along pre-described reference trajectories. This method referred to in the literature as virtual fixtures [5] or active constraints [6]. Increased precision and safety, as well as a reduction in task completion time, is the promise of this control method. It has been applied to many different fields, such as remote assembly [7], telesurgery [8], vehicle control [9], and even space-to-ground telemanipulation with long time-delay [10]. To provide effective guidance feedback, reliable and accurate task position information is required, along with the trajectories to guide the operator. This is often obtained a priori from images or markers [5]. However, this approach entails many problems in real environments, where reference positions or trajectories are often affected by measurement errors [11] or even entirely unknown during complex manipulation. For instance, during insertion operations, virtual fixtures can hardly help if the guidance system does not know exactly the insertion point (its position and orientation), which is difficult to be obtained with any vision system due to occlusions and point-of-view limitations. To mitigate this problem, van Oosterhout et al. [12] suggested combining force feedback (robot interaction data) with guidance forces. However, how to derive meaningful, accurate and sufficiently well-computed guidance trajectories for real-time manipulation such that they augment natural human manipulation is still to be resolved. This problem was addressed in our previous contribution [13], which proposed to use a learning from demonstration (LfD) approach to provide real-time haptic guidance based on the use of interaction forces and torques. This method was successfully tested with the peg-in-hole insertion task, which is a de facto benchmark test for robotics assembly [14].

The main limitation of the previous approach is that the haptic assistance should be made dependent on the kind of movements the operator is performing at a given moment. Thus, the guidance trajectories need to be generated on the fly. For example, a dashboard panel could contain different switches and connectors. Depending on the task that is being carried out by the operator, e.g., operating an on/off switch or inserting a connector, different guidance references should be applied to solve the task. In this sense, Havoutis et al. [15] proposed to create a library of previously trained models that were used to complete each defined task autonomously. However, this contribution did not take into consideration any task recognition method.

It can be assumed that a simple task is equivalent to a gesture, and a complex task consists of several gestures. In this sense, gesture recognition has been widely studied with different methods and applications [16]. For years, gesture recognition has usually been addressed using a continuous or discrete hidden Markov model (HMM) [17,18]. An HMM can encode previously trained gestures as a sequence of states with probabilistic relationships between states and measurements. It has been commonly used to detect gestures once they have finished using the forward-backward algorithm [19], i.e., for a peg insertion task, the gesture would only be detected once the operator has already inserted the peg.

In this sense, learning from demonstration (LfD) is an approach that has been widely used to generate temporally continuous trajectories by teaching, based on the robot position or interaction. Indeed, LfD uses Gaussian mixture models (GMM) or continuous hidden Markov model (CHMM) to encode training trajectories and generate the most likely trajectory through Gaussian mixture regression (GMR). This approach allows robots to perform previously trained simple human tasks such as pouring a glass of water using a bimanual robot [20], hit a table tennis ball or feed a robotic doll [21], all of them using position references. Kronander et al. [22] proposed the use of the robot pose and the exerted forces to perform automated insertion tasks based on LfD. Moreover, recent contributions used LfD for different purposes such as learning robot-collaboration skills [23], performing automated tasks of underwater remotely operated vehicles [15] or doing housework autonomously [24]. In the field of haptically guided telemanipulation, LfD has been recently used to address different tasks related to surgical robotics. Chowriappa et al. [8] used LfD to optimize the trocar placement in minimally invasive surgery (MIS). They collected a set of forces, torques, and trajectories from multiple demonstrations of the task and encoded them through the LfD approach. Then, a generalization of

this set of trajectories with its associated parameters was generated using Gaussian mixture regression (GMR). The trajectory was used to perform haptic guidance through virtual fixtures. This approach was experimentally tested in laparoscopic surgery, where the excessive load on the environment has to be avoided during the trocar insertion. Power et al. [17] proposed a LfD based framework for the position-based haptic guidance in surgical telemanipulation using gesture recognition. In this case, they used a CHMM to encode previously trained gestures (called primitive movements in the paper). This was used to solve different surgical tasks, such as needle-passing or peg-transfer. The model enabled recognition of the gesture that was performed and provided a suitable haptic guidance to the operator through virtual fixtures. However, this contribution only took the absolute instrument tip position as the measurement to detect the gesture, without taking into considerations any interaction measurement as forces or torques.

Summarizing, haptically guided telemanipulation based on interaction forces and torques solves the limitations of the methods that rely on predefined position-based trajectories. This approach has been proposed in our previous work [13] using an LfD based method. However, the best method to recognize the gesture that is being carried out, using the same model to generate the haptic guidance, remains to be investigated, e.g., some authors used GMM or CHMM to perform LfD without taking into consideration its performance for gesture recognition and vice versa. Therefore, this work is focused on a gesture recognition method based on the defined LfD approach, i.e., the use of GMM and/or CHMM. Thus, a complex task is divided into a set of simple gestures. Then, during the training stage, a GMM is encoded for each gesture and stored in a library. Hence, the system would be able to detect the gesture that is being carried out and provide a customized haptic assistance depending on the task the user is performing. Force-based gesture recognition has the additional advantage that it can be used for insertion manipulations, where position changes are hardly perceivable if some parts of a robot and an environment are in contact. Thus, we hypothesize that, for insertion and object assembly type of manipulations, a desirable guidance system should not encourage following a fixed time or position based trajectory. Furthermore, a criterion to evaluate how well each gesture has been trained to be recognized is proposed. A comparison between GMM and CHMM was carried out regarding CPU processing time and recognition accuracy. The feasibility was demonstrated in an end-to-end telemanipulation experiment in which several gestures related to the peg-in-hole insertion task were trained and recognized.

Briefly, the main contributions of this paper are, on the one hand, the use of the LfD approach to perform gesture detection, comparing the use of GMM instead of CHMM, and, on the other hand, a criterion, called GMM gesture detection score (GGDS), that can be used to choose the best number of Gaussians in a GMM, and analyze the difference between the trained gestures.

The paper is structured as follows. The proposed LfD method and the gesture recognition criteria are described in Section 2. Section 3 describes the reference task and shows the obtained experimental results. A comparison between CHMM and GMM is carried out in Section 4. Section 5 discusses the obtained results. Finally, conclusions and future works are reported in Section 6.

2. LfD for Gesture Recognition

Any complex "principle" task (such as inserting a peg in a hole) can be divided into a set of gestures $\Omega = \rho_1, ..., \rho_p$ (e.g., approach, make contact, adjust peg rotation based on force constraints, move along linear constraint, move up to rigid contact, etc.). Depending on the actual gesture, the required haptic guidance reference may be different (e.g., to align a peg, torques are predominantly used, whereas, to linearly guide during insertion, linear forces would be required). To allow such gesture-based feedback, there is a training stage that encodes the demonstrations of the operator into a library containing the set of gestures models, and a reproduction stage that recognizes the current task and provides the corresponding haptic guidance to the operator. Figure 1 shows the training stage, which is performed offline in a previous phase. During it, each of the gestures ρ_i is demonstrated u times using a training platform, e.g., a manipulator with kinesthetic movements, a teleoperated

device, or a sensorized tool. The training device provides the interaction measurements, usually forces $\overrightarrow{f} = (f_x, f_y, f_z)$ and/or torques $\overrightarrow{\tau} = (\tau_x, \tau_y, \tau_z)$. These measurements are encoded as a Gaussian Mixture Model (GMM-GGDS), the fitness of which is evaluated before storing it within the Library Ω. Then, this library is used in the reproduction stage, shown in Figure 2, to provide haptic guidance during the teleoperation of a slave robot using a master haptic device. Interaction of the slave robot with the environment generates forces $\overrightarrow{f}$ and torques $\overrightarrow{\tau}$ that are obtained from the robot sensors. They are used by the gesture recognizer to output the gesture ρ_i that is being carried, and it is used to provide the haptic guidance reference to the operator using the method described in the previous contribution [18]. Hereafter, details of the gesture recognizer system are described. Table 1 summarizes the notation used to describe the proposed framework.

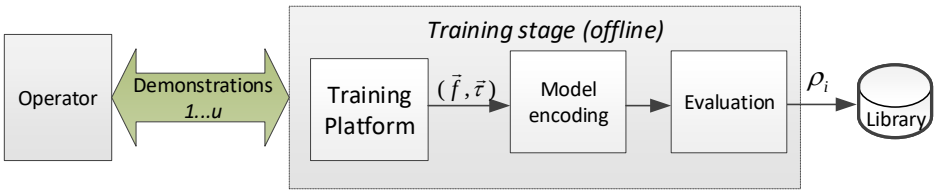


Figure 1. Training stage for gesture-based haptic guidance assistance. The operator performs a set of demonstrations for each gesture using a training platform. Then, each gesture is encoded as a Gaussian mixture models (GMM), evaluated and stored within the library Ω.

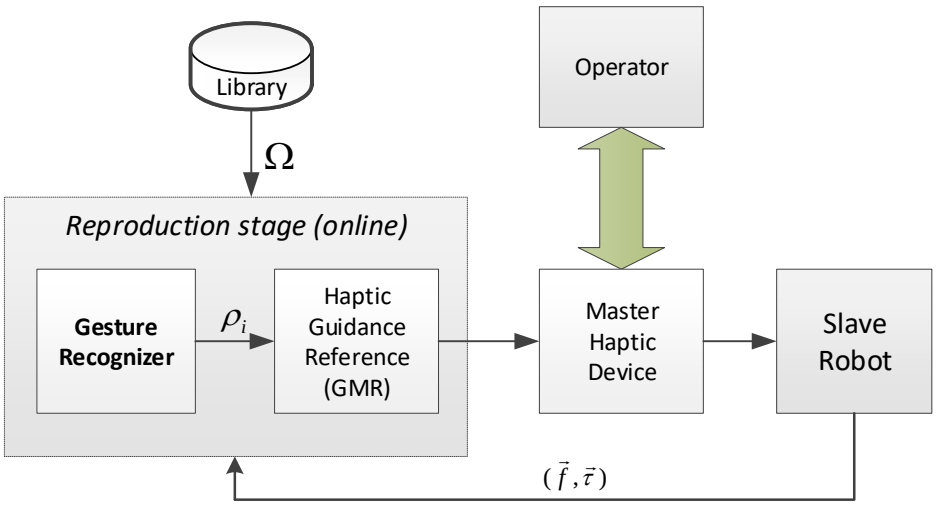


Figure 2. Method for gesture recognition and haptic guidance based on learning from demonstration. A unique library is used to detect the current gesture and provide the haptic guidance to the operator.

Table 1. Notation.

Symbol	Description
ρ_i	Encoded i gesture
Ω	Set of encoded gestures
$\overrightarrow{f}$	Exerted forces
$\overrightarrow{\tau}$	Exerted torques
ξ	Training tuple
$\widehat{\xi}(k)$	Training sequence of k tuples

2.1. Training (Offline)

First, the operator demonstrates each gesture several times using a training device. During the u demonstration of the gesture i, a training sequence of k tuples is generated as:

$$\widehat{\xi}_{iu}(k) = \xi(1), ..., \xi(k); 1 \leq i \leq p, 1 \leq u \leq U \tag{1}$$

where the training tuple is composed of the measured force and torque as:

$$\xi = (\overrightarrow{f}, \overrightarrow{\tau}) \tag{2}$$

Once the training sequences are obtained for each applicable gesture, they are used to encode a GMM ρ that will be used to recognize the gesture that is being carried out in real-time. A GMM is a probabilistic model that assumes the training sequences $\widehat{\xi}_{i1...U}$ can be included in a set of N Gaussians distributions, whereas each distribution covers a part of the training sequences. Thus, a GMM can be defined as:

$$\rho = \{\pi_n, \mu_n, \Sigma_n\}_{n=1}^{N}, \tag{3}$$

with the following parameters:

- The number of Gaussians N is one of the most important parameters since this number affects the fitness and performance of the GMM. The gesture detection score, described in Section 2.2, is used to obtain this parameter.
- The prior probabilities π_n represent the weight of each Gaussian on the demonstrations, i.e., if a nth Gaussian covers more elements of the training sequences compared with another one, its prior probability will be higher.
- The means μ_n represent the centroid of each Gaussian of the GMM.
- The covariance matrices Σ_n define the amplitude of each Gaussian n.

The encoding of a GMM consists of adjusting the parameters of the GMM such that they fit with the training sequences. In this work, it is solved using the expectation-maximization (EM) algorithm [25]. The EM is an iterative method that can approximate the Gaussians to the training sequences, maximizing the likelihood of the training sequences belonging to the encoded GMM. It has been chosen for this purpose because it provides good results, in terms of accuracy and processing time, with low dimensions in the data. It is implemented as a function EM in Equation (4), whose input parameters are the training sequences for the gesture i: $\widehat{\xi}_{i1...u}$ with u the number of demonstrations, and N the number of Gaussians.

$$\rho_i = \mathrm{EM}(\widehat{\xi}_{i1...u}, N) \tag{4}$$

The algorithm initializes each Gaussian with random parameters and they are adjusted to the training sequence iteratively as follows:

Expectation step:

$$\mathcal{P}(n|\xi_{i1...u}(j))^r = \frac{\pi_n^r \mathcal{N}(\xi_{i1...u}(j); \mu_n^r, \Sigma_n^r)}{\sum_{i=1}^{N} \pi_i^r \mathcal{N}(\xi_{i1...u}(j); \mu_i^r, \Sigma_i^r)} \tag{5}$$

$$\mathcal{E}_n^r = \sum_{j=1}^{k} \mathcal{P}(n|\xi_{i1...u}(j)^r) \tag{6}$$

Maximization step:

$$\pi_n^{r+1} = \frac{\mathcal{E}_n^r}{k} \tag{7}$$

$$\mu_n^{r+1} = \frac{\sum_{j=1}^{k} \mathcal{P}(n|\xi_{i1...u}(j))^r \xi_{i1...u}(j)}{\mathcal{E}_n^r} \tag{8}$$

$$\Sigma_n^{r+1} = \frac{\sum_{j=1}^{k} \mathcal{P}(n|\xi_{i1\ldots u}(j))^r (\xi_{i1\ldots u}(j) - \mu_n^{r+1})(\xi_{i1\ldots u}(j) - \mu_n^{r+1})^T}{\mathcal{E}_n^r} \tag{9}$$

In these equations, $\xi_{i1\ldots u}(j)$ is the tuple that corresponding to the position j on the training sequences for the gesture with k tuples, and r is the number of iterations. The iteration ends when the difference in log-likelihoods between the iterations is less than a predefined threshold C:

$$\frac{\ell(\widehat{\xi}_{i1\ldots U}|\rho^{r+1})}{\ell(\widehat{\xi}_{i1\ldots U}|\rho^r)} < C \tag{10}$$

2.2. Gesture Detection Score

As explained above, the number of Gaussians is an important parameter to be taken into account. Therefore, it may be selected based on the real-time constraint and evaluating how well every GMM fits its demonstrations. To perform this evaluation, the Bayesian information criterion (BIC) was previously used [13]. This method provides a score for different estimated GMMs, and it allows performing a comparison between them, taking the model fitness and dimension into account. However, in this paper, the trained GMMs is also used to recognize the gesture that is being carried out, and the BIC only provides information about the performance of different models for the same "gesture". It can be assumed separate gestures would be trained correctly using the BIC criteria. However, BIC does not provide information about how similar the gestures are. Thus, a GMM Gesture detection score (GGDS) has been defined in Equation (11) to overcome this issue. The GGDS provides a score for each gesture ρ_i that is calculated from the minimum difference between the log-likelihood of detecting other gestures ρ_j and the log-likelihood of detecting the gesture that is being evaluated ρ_i, using the training sequences obtained from the gesture i. The score provides information about how well the gesture will be correctly and incorrectly detected, compared with the rest of encoded gestures, with a lower score signifying a better model fitness.

$$GGDS_{\rho_i} = \min_{\rho_j \in \Omega, j \neq i} \left\{ \ell(\widehat{\xi}_{i1\ldots U}|\rho_j) - \ell(\widehat{\xi}_{i1\ldots U}|\rho_i) \right\} \tag{11}$$

The term $\ell(\widehat{\xi}_{i1\ldots U}|\rho)$ represents the log-likelihood that the training sequences belong to ρ and it is calculated as:

$$\ell(\widehat{\xi}_{i1\ldots U}|\rho) = \sum_{j=1}^{k} \log(\mathcal{P}(\xi(j)|\rho)) \tag{12}$$

with $\mathcal{P}(\xi(j)|\rho)$, the probability that the tuple $\xi(j)$ belongs to ρ, which can be calculated as:

$$\mathcal{P}(\xi|\rho) = \sum_{n=1}^{N} \pi_n \times \mathcal{N}(\xi; \mu_n, \Sigma_n) \tag{13}$$

where $\mathcal{N}(\xi; \mu_n, \Sigma_n)$ is the probability density function of ξ.

The GGDS provides a score that is useful to decide between different parametric GMMs taking into account they will be used to detect the gesture that is being carried out. Taking into account that the parameter to be optimized is the number of Gaussians N, several sets of gestures should be encoded using a different number of Gaussians, i.e., with $N = 1$ to $N = N_{MAX}$. For this purpose, Algorithm 1 performs training of p gestures for different number of Gaussians as follows. Firstly, every gesture is trained using the number of Gaussian (ranging from 1 to N_{MAX}) and the maximum GGDS score, comparing the gesture ρ_i with the rest of gestures, is stored in $SCORE_n$. Thus, the final number of Gaussians N is obtained from the minimum $SCORE_n$ for all the number of Gaussians. Finally, the trained gestures with the defined N are stored in the gesture library Ω.

Algorithm 1: Algorithm that obtains the best number of Gaussians N for the training sequences of several gestures using the GGDS criteria.

Data: N_{MAX}, p, $\widehat{\xi}_{iu}(k)$
Result: N Optimal number of Gaussians to train every gesture
for $n = 1$ *to* N_{MAX} **do**
  for $i = 1$ *to* p **do**
    $\rho_i = EM(\widehat{\xi}_{i1\ldots u}, n)$
  end
  $SCORE_n = \max_{\rho_i \in \Omega} \left[GGDS_{\rho_i}\right]$
end
$N = \underset{1 \leq n \leq N_{MAX}}{\operatorname{argmin}} \left[SCORE_n\right]$
for $i = 1$ *to* p **do**
  $\rho_i = \mathrm{EM}(\widehat{\xi}_{i1\ldots u}, N)$
  store ρ_i in Ω
end

2.3. Gesture Recognition (Online)

As stated before, each gesture was encoded into a GMM ρ in Equation (3) using training sequences that are composed of tuples ξ in Equation (2) that represent the interaction measurements. During the reproduction stage, the interaction measurements ξ are obtained each instant time from the robot sensors. Thus, the log-likelihood that the interaction measurements ξ belong to a GMM ρ can be expressed as: $log(\mathcal{P}(\xi|\rho))$, which can be calculated from Equation (13). Thus, the most likely gesture i that is being carried out can be obtained from the gesture library as:

$$i = \underset{1 \leq i \leq p}{\operatorname{argmax}} \left[log(\mathcal{P}(\xi|\rho_i))\right] \tag{14}$$

Analyzing these equations, it should be noted that the computational complexity of this method is $O(pN)$, which increases linearly with the number of Gaussians N and the number of gestures p. If N is too high, two cases can occur: (1) the processing time becomes too long for the real-time requirements during the haptic guidance owing to the large number of Gaussians in the GMM function; and/or (2) the improvement in the fitness of the GMM with respect to the training sequences is too low.

Once the gesture ρ_i in Equation (3) that is being carried out has been recognized, the corresponding model can be used to provide haptic guidance [13].

2.4. Reference Task: Peg-in-Hole Insertion with Tight Tolerance

As a de facto standard benchmark test for robotics assembly, the peg-in-hole insertion task was chosen to perform the experimental evaluation [14].

The peg-in-hole insertion task, despite being rather trivial when performed manually, has proven to be relatively challenging by the use of a robot (either teleoperated or performed autonomously) [26], in particular for long insertion dimension and tight, sub-millimeter tolerances.

As illustrated in Figure 3, we divided the peg-in-hole task into two gestures: surface contact (ρ_1) and lever effect (ρ_2), which depend on the actual interactions between the peg and hole during the execution of the task. At the beginning, during the first gesture, the operator attempts to position the peg at the entrance of the hole. This gesture is completed with establishing surface contact, still with negligence on correct orientation. Only in a second step, the operator will adjust alignment and guide the peg into the free direction for insertion.

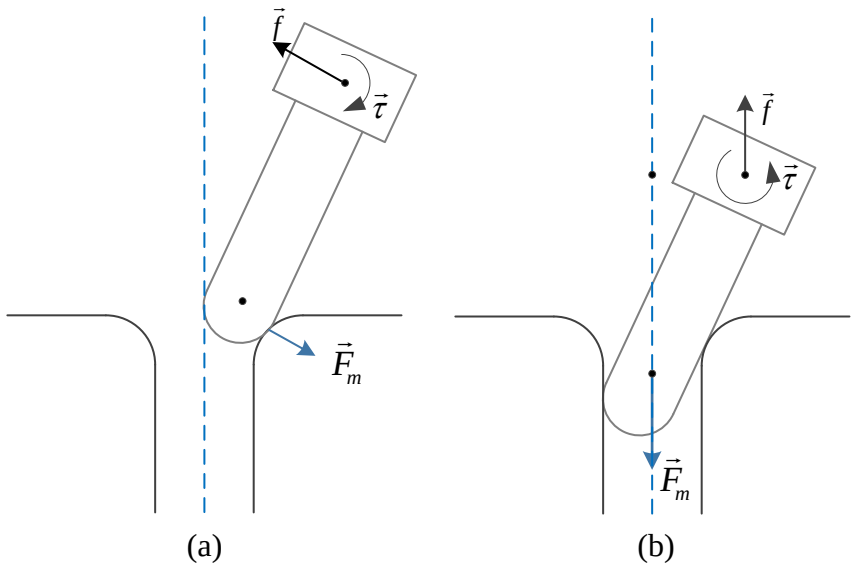


Figure 3. Peg-in-hole identified gestures: (**a**) surface contact gesture, the peg is touching the surface of the hole with lateral forces; and (**b**) lever effect gesture, the peg is already inserted in the hole without a correct orientation. Blue arrows represent the exerted forces.

In Figure 3a, the *surface contact* gesture is illustrated. The surface is pushed with a lateral force ($\overrightarrow{F}_m$). Because the peg is rigid, a reaction force of the same magnitude is transmitted to the base of the peg as $\overrightarrow{f}$. Furthermore, a small torque $\overrightarrow{\tau}$ is generated on the peg. In this case, the peg tip has to be moved horizontally to coincide with the hole. On the other hand, if the peg is already at the entrance of the hole without a correct orientation (in double contact, Figure 3b), the operator has to align the peg with the hole. In this situation, if the peg were "pushed" down with a force $\overrightarrow{F}_m$, vertical and opposite lateral forces $\overrightarrow{f}$ would arise because the peg is locked in the hole, and torques $\overrightarrow{\tau}$ occur in the opposite direction because of the lever effect. Thus, the peg may be rotated to align it with the hole. This is the *lever effect* gesture. To summarize, $\overrightarrow{f}$ and $\overrightarrow{\tau}$ represent the interaction measurements in which different magnitudes and directions are expected depending on the gesture that is being carried out. Once the interaction measurements for both gestures were obtained, the training tuple was defined as:

$$\xi = (f_x, f_y, \tau_x, \tau_y) \tag{15}$$

where f_x, f_y, τ_x and τ_y were obtained from a F/T sensor placed on the peg base. The parameters f_z and τ_z were removed from the tuple because they did not provide any relevant information to perform the guidance (only around peg symmetry axis).

3. Experimental Results

The peg-in-hole insertion task was experimentally validated by means of a telemanipulation platform located in the Telerobotics and Haptics Laboratory at the European Space Agency research center ESTEC.

The experimental validation was divided in two stages. The first one was related to the evaluation of the training stage, where two gesture were trained. The second one demonstrated the proposed gesture recognition approach and its validity for application in haptic guidance, where it should help operators to solve a task.

3.1. Experimental Setup

The experiments were carried out using a robotic telemanipulation workcell that is composed of a KUKA Lightweight Robot (LWR 4+) slave robot [27] and a Sigma.7 haptic master device [28] (Figure 4). Both devices were connected to the main computer that executed the proposed method, via the FRI Interface and UDP respectively. Figure 5 provides an experiment setup architectural overview. The taskboard, on the right-hand side of the figure, was used to research on various generic manipulation tasks. It contains different holes, pegs, and doors that can be operated on and used to validate various methods of telemanipulation and supervised autonomy. Here, the interaction with the taskboard was carried out by the manipulator Kuka LWR. During the telemanipulation, the manipulator was configured in impedance control mode to avoid any damage because of the interaction between the peg and the taskboard. During the training, the manipulator was configured in the gravity-compensated mode to allow an operator to move the robot by hand and position it freely in space to perform the training stage (hence, teach) certain task elements. This feature was used during the training stage to perform the peg-in-hole insertions by human-guided kinesthetic movements in a more direct way (as opposed to using a master device remotely for this stage).

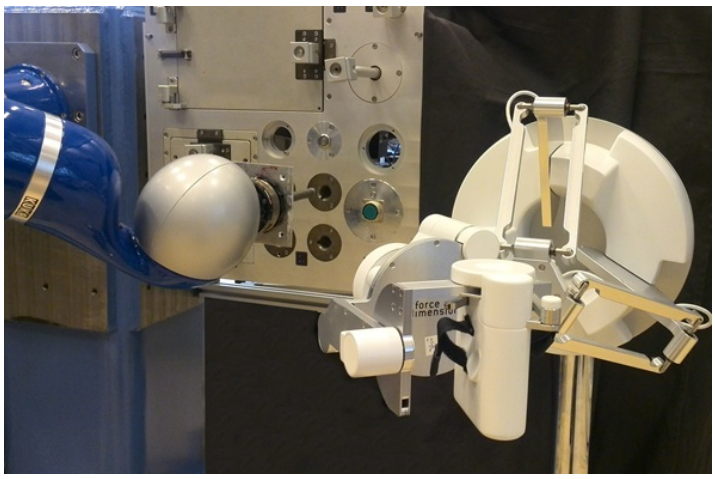


Figure 4. Overview of the Experimental setup used for learning from demonstration (LfD)-based haptic guidance, with the master device (Sigma.7 haptic master) and the KUKA lightweight slave robot manipulating a metal peg into a task-board receptacle.

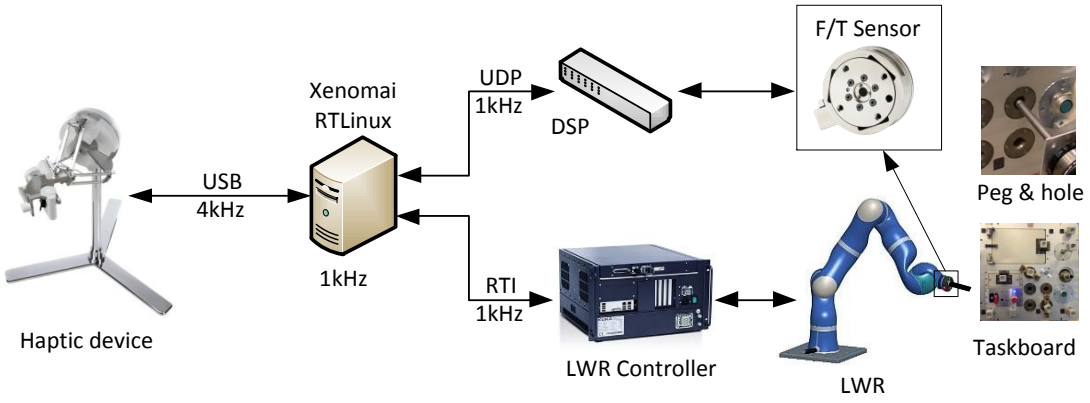


Figure 5. Software and Hardware architecture of the experimental telemanipulation setup as available at the Telerobotics Laboratory of the European Space Agency.

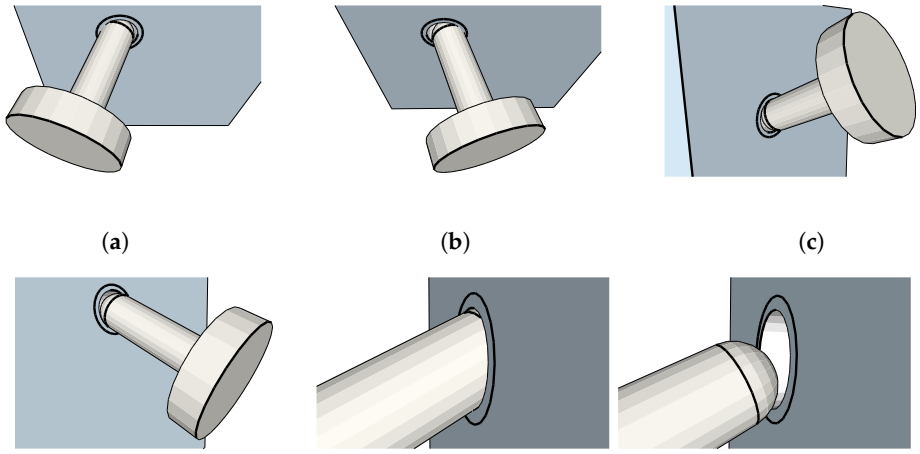
A 155 mm long titanium peg (approximate mechanical circumferential tolerance of 150 micrometers to the diameter of the metal receptacle in task-board) was rigidly mounted on a multiaxial Force/Torque (F/T) sensor (ATI Net F/T Gamma SI 65-5) attached to the robot's end-effector. The sensor measures interaction forces $\overrightarrow{f} = (f_x, f_y, f_z)$ and torques $\overrightarrow{\tau} = (\tau_x, \tau_y, \tau_z)$ on the peg base, providing the necessary information to the method described above. During task-reproduction, the manipulator was teleoperated by a force dimension Sigma.7 haptic device. It was used to receive position references from the operator, but also provided force feedback to the operator to guide him to perform the detected gesture. Forces were scaled to resist another operator movement during the assisted teleoperation. All of these devices, except the haptic one, which was connected directly through USB, were communicating with the main computer using the RTI Data Distribution Service middleware over a dedicated gigabyte LAN network. The main computer was a PC with a real-time Linux-based operating system (Xenomai). The computer algorithms were implemented with 1 ms sampling time.

3.2. Training

To obtain the training sequences for both gestures, the LWR was manually guided with its peg into the hole from eight different and 90° offset starting positions, which varied for the two types of gestures. Figure 6a–d presents the defined initial positions that cover a circle. Figure 6e,f presents the starting positions of the peg for the lever effect and the surface contact gesture, respectively. Three movements were performed for each gesture and each initial position, resulting in 12 insertion movements. They were used to train a model for each gesture. On the other hand, another 12 insertion movements were carried out using the complete telemanipulation workcell, which were used to evaluate the proposed approach.

The training insertions were used to encode both GMMs with a different number of Gaussians N, and they were evaluated through BIC and GGDS using Equation (11). Figure 7 depicts a comparison of the obtained scores between the different number of Gaussians for both gestures and scores. It demonstrates how the GGDS score was improved exponentially as the number of Gaussians is increased (larger negative score). However, the score obtained using the BIC increased almost linearly, because it does not provide information about how well the gesture would be recognized with respect the rest. Moreover, GGDS indicates both gestures would be recognized similarly as both of them received a similar score for an increased number of Gaussians. As stated before, the reproduction stage processing time increases linearly with the number of Gaussians. A test was performed to determine the maximum number of Gaussians that the main computer was able to execute within the real-time constraints of the system used in telemanipulation (characterized by a 1 ms sampling time). For this purpose, each gesture was trained with the number of Gaussians ranging from 1 to 20. Later on, the recognition algorithm was executed in the computer for each trained model to evaluate if the computer was able to execute it within the 1 ms constraint. Finally, $N = 14$ was chosen to carry out the experiments on the computer platform.

The EM algorithm in Equation (4) was used to train a GMM for each gesture. Figure 8a,b illustrates the training sequences and the Gaussians for both gestures. The training sequences $\widehat{\xi}_{i1\ldots U}$ are represented by colored dots, one color for each training sequence. Each dot represents interaction measurement tuples ξ^m obtained at every time instant. For simplicity, only forces f_x and f_y are shown for the surface contact gesture and torques τ_x and τ_y for the lever effect gesture. The reason for choosing these measurements is because they are the most relevant information for each gesture, as stated above. Finally, the encoded Gaussians are represented by crosses (means) and ellipses (covariances). Once the best-encoded GMMs were selected, they were stored in the gesture library.



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 6. Initial training positions: (**a**–**d**) four different insertion positions to cover a circle over the hole; (**e**) initial position of the surface contact gesture; and (**f**) initial position of the lever effect gesture. (**a**) Left; (**b**) right; (**c**) up; (**d**) down; (**e**) lever effect; and (**f**) surface contact are shown.

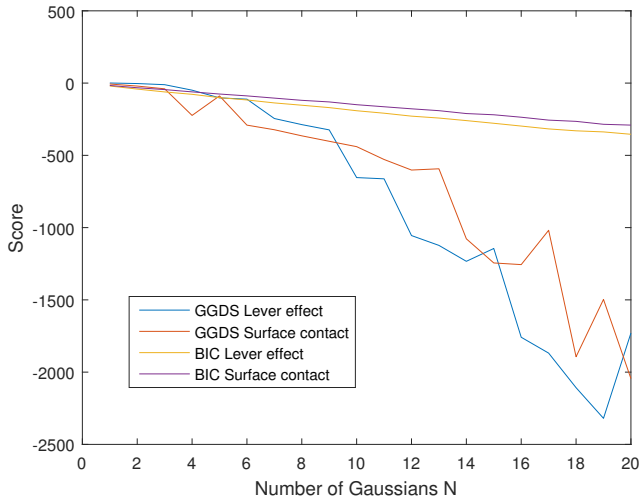
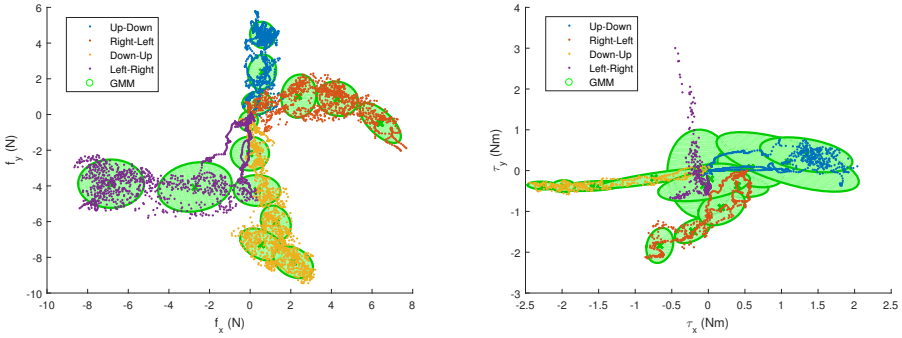


Figure 7. GMM gesture detection score (GGDS) for both gestures using different number of Gaussians. The figure shows the GGDS score decreased exponentially as the number of Gaussians was increased.



(**a**) (**b**)

Figure 8. GMM encoding for the surface contact (**a**) and lever effect (**b**) gestures showing the fit of the training sequences with the generated Gaussians. The colored points represent the training measurements for each movement.

3.3. Reproduction

Once populated and evaluated, the gesture library was used during the reproduction stage to recognize the gesture that was being carried out online, i.e., at the very instant in time. Moreover, modified models stored in the gesture library could be used to generate the appropriate haptic guidance reference to assist an operator during the telemanipulation execution [13].

As stated above, the interaction measurements $\overrightarrow{f}$ and $\overrightarrow{\tau}$ were used to calculate the most likely gesture that was being carried out through Equation (14). All evaluation movements were used to validate the proposed method. Figure 9 illustrates how the surface contact gesture is detected. Here, four evaluation movements were used, each one starting from different initial positions (see Figure 6a–d). In Figure 9a–d, the interaction measurements, obtained at each instant time, are represented by colored lines, i.e., one color for each movement from the various starting points. The trained Gaussian means are represented by crosses and the covariances by the ellipses. These figures show how the selected motion sequences enabled the algorithms to recognize the surface contact gestures (Figure 9a,b) and disabled their ability to recognize the lever effect gesture (Figure 9c,d). Figure 9e represents the log-likelihood of each evaluation sequence that belongs to the surface contact gesture GMM and Figure 9f represents the log-likelihood for the lever effect gesture. As shown, the log-likelihood of the evaluation movements for the surface contact motions was mostly higher for the surface contact gesture (-9 to 6 in the figure) than for the lever effect one (-90 to -15), which indicates the gesture was correctly recognized.

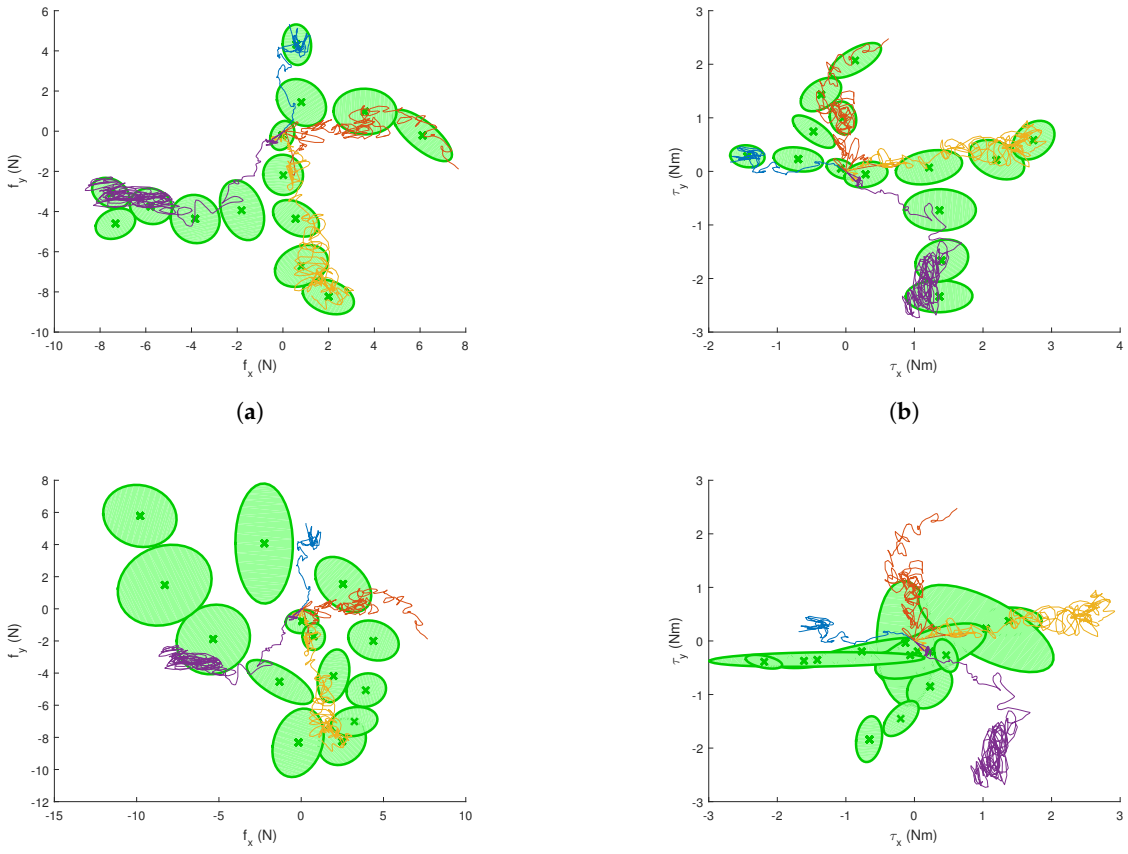
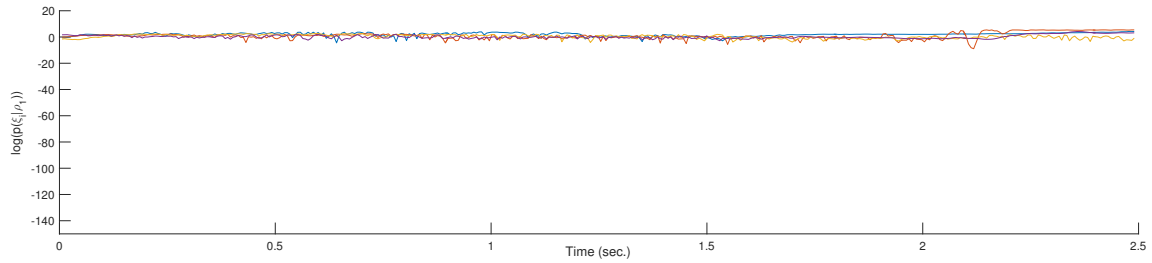
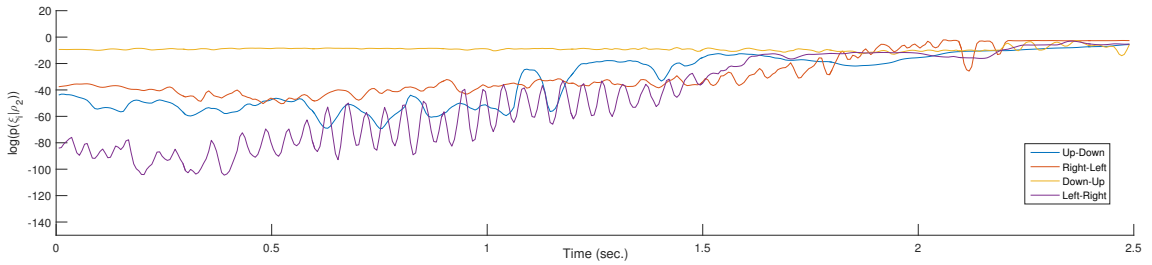


Figure 9. *Cont.*



(**e**)



(**f**)

Figure 9. Trained surface contact (**a**) and lever effect (**b**) GMMs, and their fitnesses with the surface contact gesture interaction measurements from the evaluation movements. Log-likelihood of the surface contact gesture interaction measurements for both gestures (**c**,**d**). (**a**) Interaction forces vs. surface contact gesture GMM; (**b**) interaction torques vs. surface contact gesture GMM; (**c**) interaction forces vs. lever effect gesture GMM; (**d**) interaction torques vs. lever effect gesture GMM; (**e**) log-likelihood that the interaction measurement tuple ξ_i, obtained each i instant time, belongs to the surface contact GMM ρ_1; and (**f**) log-likelihood that the interaction measurement tuple ξ_i, obtained each i instant time, belongs to the lever effect GMM ρ_2.

4. Comparison between GMM and CHMM

Taking into consideration that CHMM has been widely used for gesture recognition, GMM arises as an alternative for haptically guided telemanipulation tasks. Indeed, the main advantages of this method are the lower computer complexity for training and reproduction. The CHMM computer complexity for gesture recognition is $O(PN^2T)$ where P is the number of gestures, N is the number of states, and T is the number of measurements. Contrarily, the computer complexity of the GMM based gesture recognition method is $O(PN)$, i.e., the processing time increases linearly. In the case of the training stage, CHMM uses the k-means algorithm to define each continuous hidden state, whose computer complexity is: $O(NTK)$, where K is the number of iterations. Afterwards, the Baum–Welch algorithm is used to train the HMM itself according to the previously defined hidden states. The computer complexity of this algorithm is $O(NT)$, resulting in $O(NTK) + O(NT)$ for the CHMM. However, GMM only uses the k-means algorithm to encode a gesture model.

To evaluate the performance of GMM versus CHMM experimentally, previous measurements for training and evaluation were used to encode and evaluate a CHMM with the same parameters as the previously encoded GMM. The results obtained for the CPU processing time are shown in Figure 10, where it can be seen that GMM achieved slightly better results than CHMM. As regards the recognition rate, each model was used to recognize the performed gesture that was being carried out in real-time, i.e., every sampling period (1 ms) the gesture was recognized. As shown in Figure 11, for the lever effect gesture, CHMM was slightly better than GMM, but, in the case of the surface contact one, GMM was better than CHMM.

On the other hand, the detection of the gesture can be seen as a multi-objective optimization problem, where there are a cost function and an objective one, i.e., the CPU processing time and the log-likelihood of the detected gesture. Therefore, the objective is to minimize the CPU processing time and maximize the log-likelihood. Figure 12 represents the cost and objective for both methods and

gestures. Every evaluation movement was trained using a different number of Gaussians or states, obtaining different values as it was increased. The log-likelihood represented in this figure has been calculated according to Equation (12). Results show that GMM provides better results using the same CPU processing time than CHMM, which required approximately double the processing time to obtain equal log-likelihood for the same movement.

Finally, an evaluation of training the gestures using a different number of Gaussians or states was carried out. Figure 13 represents the CPU processing time to train both gestures using GMM and CHMM. As shown, the needed processing time for GMM was much lower than CHMM as the number of Gaussians was increased.

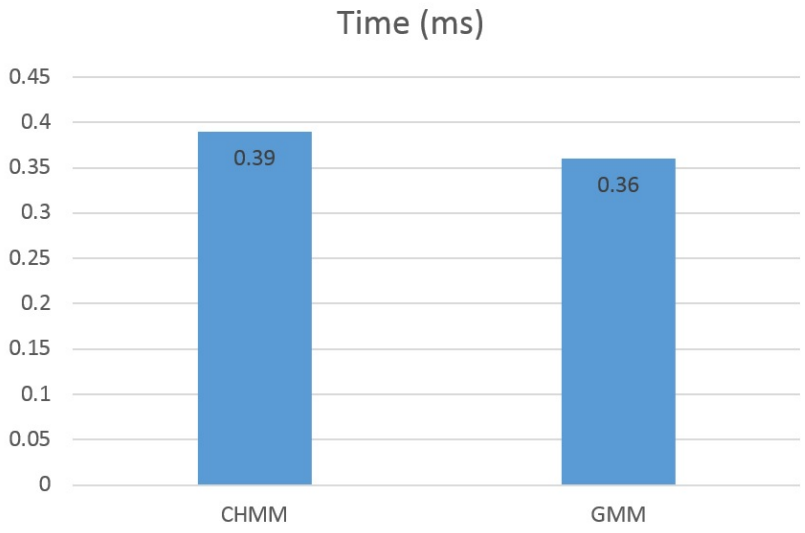


Figure 10. Average CPU processing time during the recognition of each evaluation movement using both continuous hidden Markov model (CHMM) and GMM methods.

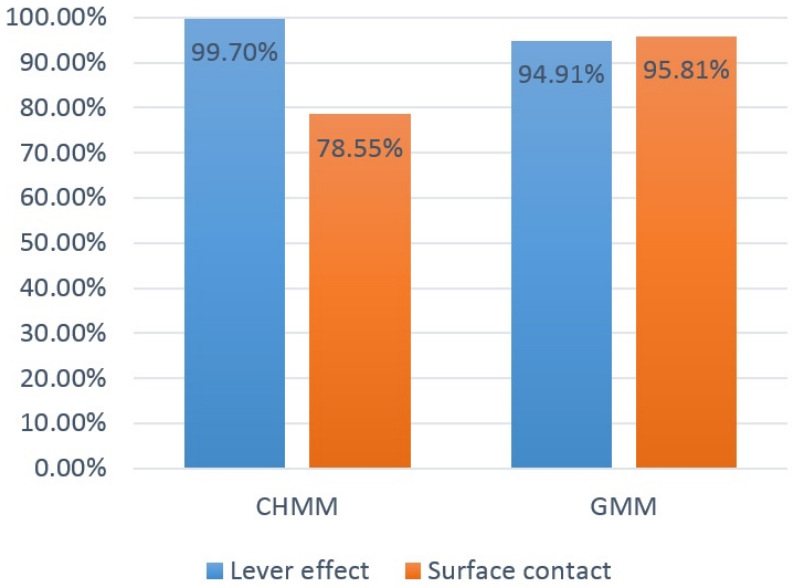
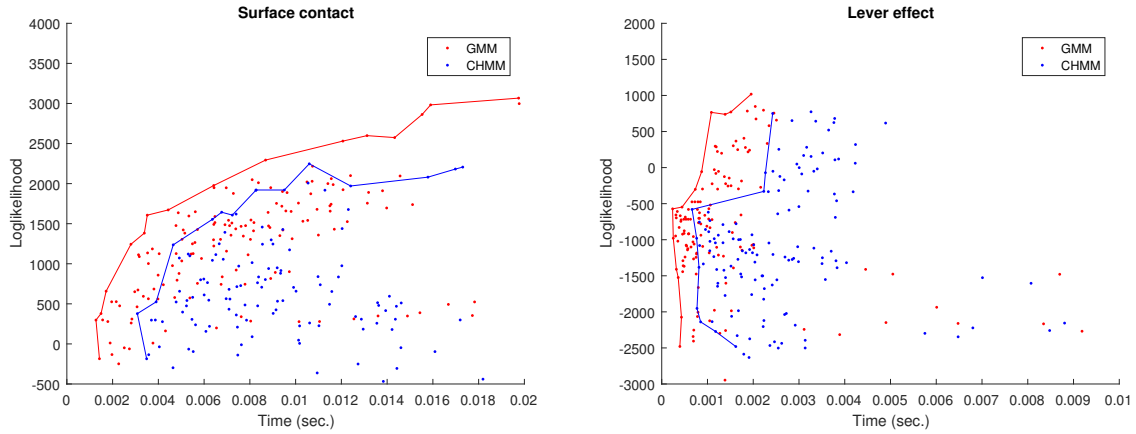


Figure 11. Percentage of real-time recognition using the evaluation movement of both gestures, and both CHMM and GMM methods.



(**a**) (**b**)

Figure 12. Pareto frontier for GMM and CHMM using the log-likelihood and processing time of each evaluation movement for both gestures: (**a**) surface contact gesture; and (**b**) lever effect gesture.

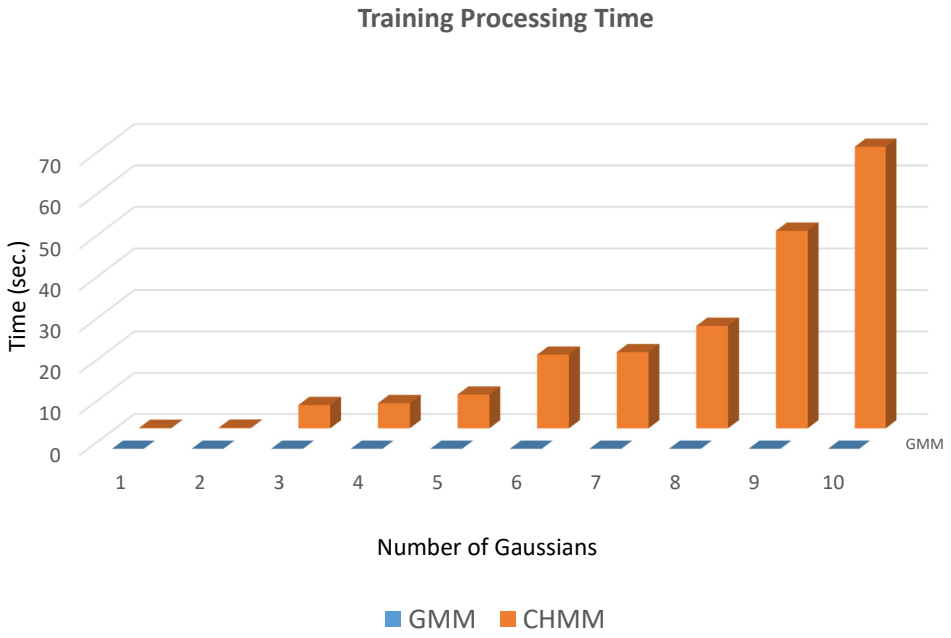


Figure 13. Training time using GMM and CHMM for different number of Gaussians/states.

5. Discussion

The above results show that it is feasible to recognize, in real-time, the gesture that is being carried out with a high success rate. The presented method was validated with the peg-in-hole insertion task under sub-millimeter tolerances. During the training stage, two gestures were defined through 24 demonstrations that generated two GMMs, composed of 14 Gaussians each. Although the presented work was focused on two manually predefined gestures (i.e., surface contact and lever effect) the proposed method could be used with recently proposed segmentation of movements techniques [29].

The defined GMM gesture detection score (GGDS) was used to analyze the GMM performance taking into account the number of Gaussians, which demonstrates that the performance gets better when the number of Gaussians is increased for the peg-in-hole insertion task. Therefore, the main

limitation to choose the maximum number of Gaussians was the real-time constraint of 1 ms. However, this method has a limitation: as it has been defined, the same number of Gaussians is used to train every gesture, which would not optimize the right number of Gaussians if there are simple and complex gestures. It is worth mentioning the GGDS is also useful to detect how different the gestures to be trained are, which would provide information about two or more similar gestures. In this case, the GGDS would help to decide the best number of Gaussians.

During the gesture recognition, the GMM-based method achieved a success rate of 94.91% and 95.81% for each gesture. The CHMM-based method provided a success rate of 99.7% and 78.6% for the same training and evaluation sequences. Although the lever effect gesture was better recognized by CHMM, the recognition of the surface contact was significantly worse. The authors consider that CHMM is intended to detect the gesture once it has ended, and it is assumed that this is the reason because GMMs provide better results than CHMM during real-time gesture recognition. Moreover, another drawback of CHMM is that the method is able to detect a gesture by a motion sequence, due to the use of hidden states. However, GMMs detect the gesture for the motion data every instant time, i.e., an incorrect detection at the beginning of the motion would not affect the entire recognition of the gesture.

Safety of the method, in terms of gesture recognition accuracy, should be taken into consideration to avoid a wrong guidance. Although this paper does not provide any solution, a good approach would be to take into consideration a log-likelihood threshold to enable the haptic guidance for the detected gesture. Depending on it, the system would be more or less conservative providing haptic guidance.

On the other hand, a Pareto analysis was carried out, which shows that GMM provides better performance than CHMM during gesture detection, by providing comparable result in half the processing time. During the training stage, both models were trained with the number of Gaussians between 1 and 10. The required processing time for GMM ranged between 0.00257 s and 0.020997 s and between 0.141 s and 67.980 s for CHMM. Such a large difference in the training stage can be especially advantageous in certain applications, for example when used in future reinforcement learning methods.

6. Conclusions

This work shows the use of a method based on LfD for real-time gesture recognition during telemanipulation. The developed method performs sufficiently well to allow gesture recognition in real-time with a one-millisecond sampling-time constraint. Moreover, it is shown that a 14 Gaussians GMM approach is sufficient to adequately recognize the gestures of a peg-in-hole insertion task (that is here characterized by two individual gestures: “establishing of surface contact” and “levering the peg into the hole”). The new “interaction-force” based method proposed in this paper provides a viable way to encode and reconstruct contact tasks in robotic assembly to guide human operators whenever either visual information is limited or when positional or time-based indexed guidance data are not available or difficult to achieve. The LfD-based framework proposed by us earlier [18] for haptic shared control is extended by an LfD-based gesture recognition module, which allows application in real-time, due to its linear increase of computational complexity with the number of Gaussian Models used to encode any gesture. We consider the proposed approach to be extendable to other complex manipulation tasks involving contact operations, such as operation on knobs, doors, hinges, and levers, and, therefore, we assume that the method can significantly improve remotely operated robotic manipulation in a variety of contexts. Moreover, a further comparison among GMM, CHMM and other recognition methods would be useful to analyze suitability in specific situations. These issues are proposed to be addressed as future work.

Author Contributions: Conceptualization, C.J.P.-d.-P., J.S. and A.S.; Methodology, C.J.P.-d.-P., A.S. and V.F.M.; Software, C.J.P.-d.-P. and J.S.; Validation, C.J.P.-d.-P., J.S. and I.R.-B.; Writing—original draft preparation, C.J.P.-d.-P., I.R.-B.; Writing—review and editing, A.S., J.S., I.R.-B. and V.F.M.; and Supervision, A.S. and V.F.M.

Funding: This research was partially funded by Ministry of Science and Innovaction (Spain), under the project code DPI2016-80391-C3-1-R.

Acknowledgments: Authors would like to thank the Telerobotics and Haptic Laboratory at the European Space Agency for supporting this work.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

LfD	Learning from Demonstration
HMM	Hidden Markov Model
GMM	Gaussian Mixture Models
GMR	Gaussian Mixture Regression
CHMM	Continuous Hidden Markov Model
MIS	Minimally Invasive Surgery
EM	Expectation-Maximization
BIC	Bayesian Information Criterion
GGDS	Gesture Detection Score

References

1. Codourey, A.; Rodriguez, M.; Pappas, I. A task-oriented teleoperation system for assembly in the microworld. In Proceedings of the 1997 8th International Conference on Advanced Robotics, ICAR'97, Monterey, CA, USA, 7–9 July 1997; IEEE: Piscataway, NJ, USA, 1997; pp. 235–240.
2. Hokayem, P.F.; Spong, M.W. Bilateral teleoperation: An historical survey. *Automatica* **2006**, *42*, 2035–2057. [CrossRef]
3. Chen, J.Y.; Haas, E.C.; Barnes, M.J. Human performance issues and user interface design for teleoperated robots. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **2007**, *37*, 1231–1245. [CrossRef]
4. Bolopion, A.; Xie, H.; Haliyo, D.S.; Régnier, S. Haptic teleoperation for 3-D microassembly of spherical objects. *IEEE/ASME Trans. Mechatron.* **2012**, *17*, 116–127. [CrossRef]
5. Abbott, J.J.; Marayong, P.; Okamura, A.M. Haptic virtual fixtures for robot-assisted manipulation. In *Robotics Research*; Springer: Berlin, Germany, 2007; pp. 49–64.
6. Nuno, E.; Rodríguez, A.; Basañez, L. Force reflecting teleoperation via ipv6 protocol with geometric constraints haptic guidance. In *Advances in Telerobotics*; Springer: Berlin, Germany, 2007; pp. 445–458.
7. Boessenkool, H.; Abbink, D.A.; Heemskerk, C.J.; van der Helm, F.C.; Wildenbeest, J.G. A task-specific analysis of the benefit of haptic shared control during telemanipulation. *Haptics IEEE Trans.* **2013**, *6*, 2–12. [CrossRef] [PubMed]
8. Chowriappa, A.; Wirz, R.; Ashammagari, A.R.; Seo, Y.W. Prediction from expert demonstrations for safe tele-surgery. *Int. J. Autom. Comput.* **2013**, *10*, 487–497. [CrossRef]
9. Griffiths, P.G.; Gillespie, R.B. Sharing control between humans and automation using haptic interface: Primary and secondary task performance benefits. *Hum. Factors J. Hum. Factors Ergon. Soc.* **2005**, *47*, 574–590. [CrossRef] [PubMed]
10. Schiele, A. Towards the Interact Space Experiment: Controlling an Outdoor Robot on Earth's Surface from Space. In Proceedings of the 13th Symposium on Advanced Space Technologies for Robotics and Automation (ASTRA), Noordwijk, The Netherlands, 11–13 May 2015; European Space Agency: Paris, France, 2016.
11. Smisek, J.; van Paassen, M.; Schiele, A. Haptic guidance in bilateral teleoperation: Effects of guidance inaccuracy. In Proceedings of the 2015 IEEE World Haptics Conference (WHC), Evanston, IL, USA, 22–26 June 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 500–505.
12. Van Oosterhout, J.; Wildenbeest, J.G.; Boessenkool, H.; Heemskerk, C.J.; de Baar, M.R.; van der Helm, F.C.; Abbink, D.A. Haptic Shared Control in Tele-Manipulation: Effects of Inaccuracies in Guidance on Task Execution. *IEEE Trans. Haptics* **2015**, *8*, 164–175. [CrossRef] [PubMed]
13. Pérez-del Pulgar, C.J.; Smisek, J.; Muñoz, V.F.; Schiele, A. Using learning from demonstration to generate real-time guidance for haptic shared control. In Proceedings of the 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Budapest, Hungary, 9–12 October 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 003205–003210.

14. Unger, B.J.; Nicolaidis, A.; Berkelman, P.J.; Thompson, A.; Klatzky, R.L.; Hollis, R.L. Comparison of 3-D haptic peg-in-hole tasks in real and virtual environments. In Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180), Maui, HI, USA, 29 October–3 November 2001; IEEE: Piscataway, NJ, USA, 2001; Volume 3, pp. 1751–1756.
15. Havoutis, I.; Calinon, S. Supervisory teleoperation with online learning and optimal control. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017.
16. Mitra, S.; Acharya, T. Gesture recognition: A survey. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **2007**, *37*, 311–324. [CrossRef]
17. Power, M.; Rafii-Tari, H.; Bergeles, C.; Vitiello, V.; Yang, G.Z. A cooperative control framework for haptic guidance of bimanual surgical tasks based on Learning From Demonstration. In Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30 May 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 5330–5337.
18. Del Pulgar, C.J.P.; Garcia-Morales, I.; Blanco, I.R.; Munoz, V.F. Navigation Method for Teleoperated Single-Port Access Surgery With Soft Tissue Interaction Detection. *IEEE Syst. J.* **2016**, in press. [CrossRef]
19. Rabiner, L.R. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* **1989**, *77*, 257–286. [CrossRef]
20. Jäkel, R.; Schmidt-Rohr, S.R.; Lösch, M.; Dillmann, R. Representation and constrained planning of manipulation strategies in the context of programming by demonstration. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3–7 May 2010; IEEE: Piscataway, NJ, USA, 2010; pp. 162–169.
21. Calinon, S.; D'halluin, F.; Sauser, E.L.; Caldwell, D.G.; Billard, A.G. Learning and reproduction of gestures by imitation. *IEEE Robot. Autom. Mag.* **2010**, *17*, 44–54. [CrossRef]
22. Kronander, K.; Burdet, E.; Billard, A. Task transfer via collaborative manipulation for insertion assembly. In *Workshop on Human-Robot Interaction for Industrial Manufacturing, Robotics, Science and Systems*; Citeseer: Centre County, PA, USA, 2014.
23. Rozo, L.; Calinon, S.; Caldwell, D.G.; Jimenez, P.; Torras, C. Learning physical collaborative robot behaviors from human demonstrations. *IEEE Trans. Robot.* **2016**, *32*, 513–527. [CrossRef]
24. Kormushev, P.; Calinon, S.; Caldwell, D.G. Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input. *Adv. Robot.* **2011**, *25*, 581–603. [CrossRef]
25. Bilmes, J.A. A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. *Int. Comput. Sci. Inst.* **1998**, *4*, 126.
26. Chhatpar, S.R.; Branicky, M.S. Search strategies for peg-in-hole assemblies with position uncertainty. In Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180), Maui, HI, USA, 29 October–3 November 2001; IEEE: Piscataway, NJ, USA, 2001; Volume 3, pp. 1465–1470.
27. Bischoff, R.; Kurth, J.; Schreiber, G.; Koeppe, R.; Albu-Schäffer, A.; Beyer, A.; Eiberger, O.; Haddadin, S.; Stemmer, A.; Grunwald, G.; et al. The KUKA-DLR Lightweight Robot arm-a new reference platform for robotics research and manufacturing. In Proceedings of the ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics), VDE, Munich, Germany, 7–9 June 2010; pp. 1–8.
28. Tobergte, A.; Helmer, P.; Hagn, U.; Rouiller, P.; Thielmann, S.; Grange, S.; Albu-Schäffer, A.; Conti, F.; Hirzinger, G. The sigma. 7 haptic interface for MiroSurge: A new bi-manual surgical console. In Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Francisco, CA, USA, 25–30 September 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 3023–3030.
29. Figueroa, N.; Ureche, A.L.P.; Billard, A. Learning complex sequential tasks from demonstration: A pizza dough rolling case study. In Proceedings of the 2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI), Christchurch, New Zealand, 7–10 March 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 611–612.