Fast Target Prediction of Human Reaching Motion for Cooperative Human-Robot Manipulation Tasks using Time Series Classification

Claudia Pérez-D'Arpino[1] and Julie A. Shah[2]

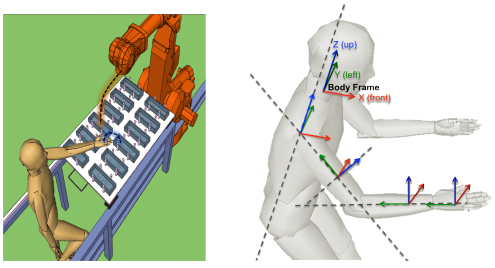
***Abstract*— Interest in human-robot coexistence, in which humans and robots share a common work volume, is increasing in manufacturing environments. Efficient work coordination requires both awareness of the human pose and a plan of action for both human and robot agents in order to compute robot motion trajectories that synchronize naturally with human motion. In this paper, we present a data-driven approach that synthesizes anticipatory knowledge of both human motions and subsequent action steps in order to predict in real-time the intended target of a human performing a reaching motion. Motion-level anticipatory models are constructed using multiple demonstrations of human reaching motions. We produce a library of motions from human demonstrations, based on a statistical representation of the degrees of freedom of the human arm, using time series analysis, wherein each time step is encoded as a multivariate Gaussian distribution. We demonstrate the benefits of this approach through offline statistical analysis of human motion data. The results indicate a considerable improvement over prior techniques in early prediction, achieving 70% or higher correct classification on average for the first third of the trajectory ($< 500msec$). We also indicate proof-of-concept through the demonstration of a human-robot cooperative manipulation task performed with a PR2 robot. Finally, we analyze the quality of task-level anticipatory knowledge required to improve prediction performance early in the human motion trajectory.**

I. Introduction

In the manufacturing setting, robots increasingly coexist with human workers on the same factory floor. Significant productivity benefits can be achieved if the human and robot can fluently share the same work volume and function in close proximity to one another. This new work paradigm is of particular interest for automotive final assembly, where much of the work is dexterous and must still be performed by humans, but where robots can reduce task completion time by concurrently positioning tools and parts for human workers. Successful execution of this coexistence requires that the robot reliably anticipate human motions and subtly adjust planned trajectories in real-time, as a human would do. In this work, we present an approach that synthesizes anticipatory knowledge of both the human's motions and the next action steps in order to predict the intended target of a human performing a reaching motion in real-time.

Our primary contribution is a method to conduct real-time online prediction of the target of a human reaching motion using time series analysis, wherein each time step of the motion is encoded as a multivariate Gaussian distribution over the degrees of freedom of the human arm. Early prediction is accomplished through Bayesian classification using a partial initial segment of the trajectory of the human arm. Our approach involves an offline training phase, during which motion-level anticipatory models are constructed using multiple demonstrations of human reaching motions. Each demonstration is represented as a reaching motion from the initial pose of a human arm to the target of the manipulation action. Quantities necessary for fast online prediction are cached for quick reference during online data processing. The online stage consists of the execution of the human-robot cooperative task. The statistical models representing a learned library of motions serve as the set of possible motion classes for the real-time classification algorithm.

This approach outperforms prior art [1][2] by achieving considerably higher prediction performance very early in the trajectory execution. We demonstrate 70% or higher correct classification on average for the first third of the trajectory ($< 500msec$), and use this anticipatory signal to adjust the robot's next action to avoid conflict between human and robot motions. We also discuss the generalization of the approach to synthesize both task-level and motion-level anticipatory information. At the task level, we use the prior term in the Bayesian model to represent the prior probability of the human's next task step, which can be derived from task procedures. We establish the performance requirements for task-level prediction in relation to the quality of the motion-level prediction, such that synthesis of the two results improved prediction performance early in the human motion trajectory. Finally, we demonstrate proof-of-concept of the approach using a tabletop manipulation task, during which the human and robot collect parts from and/or bring parts to a shared workspace, as illustrated in Fig.1a.



(a) (b)

Fig. 1: (a) Tabletop task. (b) Coordinate systems used to recover the trajectory of the kinematic chain during a demonstration.

[1]Claudia Pérez-D'Arpino is with Department of Electrical Engineering and Computer Science, Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology cdarpino@mit.edu

[2]Julie Shah with the Department of Aeronautics and Astronautics, Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology julie_a_shah@csail.mit.edu

II. Related Work

A manipulation planning framework for human-robot collaboration was recently presented in [1], wherein the authors developed a prediction algorithm at the motion level using Gaussian mixture models (GMM) and an adaptive path planning algorithm for the robotic manipulator based on the STOMP algorithm [3]. In that work, a library of human motions was built from real Kinect data (30Hz) and then tested together with the motion planning algorithm using a PR2 robot in simulation. On the motion prediction side, the algorithm achieved 92% correct classification after having processed 80% of the test trajectory on average. However, neither the prediction nor the motion planning processing time achieved sufficient real-time performance to be used online for a real application, and the average correct classification at early stages of the trajectory was still very low, with 50% correct classification achieved after processing 43% of the motion trajectory.

The learning-from-demonstration (LfD) community has also utilized GMMs [4][5][6] to learn a set of robot motions from human demonstrations. However, these works do not specifically address the online prediction problem, and the known limitations of GMM make its application for motion prediction problematic [7][2]. GMM is an exchangeable model meaning that the temporal order of the data is not taken into account, as the sequence of observations can be modified without affecting the algorithm result. This drawback is usually handled by introducing the time of the observations as another dimension of the GMM; we have found that this improves the accuracy with respect to time-myopic GMM, but still does not encode the time dependence explicitly.

We contribute an algorithm for motion prediction that is able to run online in real-time during the execution of a task, with considerably better rates of correct classification at early stages of the human motion trajectory than those exhibited in prior work. We achieve this by adapting the Bayesian approach developed in [7][2] and exploiting its capability of reporting partial results as trajectories are being processed.

In [2], the probabilistic flow tubes (PFT) algorithm was designed to learn and recognize (or classify) tasks demonstrated by a human operator who is physically moving the end-effector of a robotic manipulator. The main goal is to teach the robot how to execute manipulation tasks in variable environments. This prior work only addressed classification using the full motion trajectory offline, once the robot motion had been completely executed. The demonstration was represented as the sequence of the end-effector positions through the trajectory. Here, we pose the prediction problem as one of classification, using a partial segment of the trajectory, and instead use human motion trajectories as features comprised of the full degrees of freedom of the human arm.

There are challenges associated with using human motion instead of human-guided robot motion to build the motion library. Motion tracking may be less reliable and the algorithm must be robust with regard to noise in the data; a larger set of features results in increased algorithmic complexity and greater difficulty in extracting relevant features. The differences in learning via demonstrations from human motion vs. human-guided robot motion are described in [8].

Other works aim to utilize anticipatory information derived from task procedures for the prediction of subsequent action. These works analyze the discrete action sequence of human and robot task steps [9][10][11][12]; however, they do not take advantage of geometric or kinematic information. Approaches include the prediction of subsequent human action through inference in a Bayesian network [9], and consideration of task ambiguity and sensor uncertainty [10] during an assembly task similar to ours. Other techniques, such as simple temporal problems (STP) and nonlinear programming, have been used to produce optimal flexible schedules for human-robot collaboration [11]. One case [12] utilized a Markov decision process (MDP) model to encode information about the roles of the human and the robot during a task, and allowed for the inference of a subsequent action using entropy rate in the MDP. Another related approach sought objects in the environment to identify affordances that could potentially be the targets of human manipulation actions [13]. This method for anticipating the target incorporates a combination of knowledge about the affordances and the motion itself to build temporal conditional random fields. However, it does not take into account the temporal sequence of actions or task procedures for multi-step action prediction.

The aforementioned works approach the problem from the robot's point of view, by trying to predict human motion and adapt to it by planning the robot motion in spatio-temporal synchronization. This is also our approach; however, it is also interesting to consider the problem from the human perspective, as it is also relevant that the human understand the robot's motion. This perspective was explored in [14], where the concepts of legibility (the ability to predict the goal) and predictability (the ability to predict trajectory) of the motion were introduced. A functional gradient optimization technique for generating legible motion was proposed in [15], and this technique yielded favorable performance within a region containing the expected trajectory. We believe these two points of view should be combined in future applications, with a robot both aware of the future actions of a human and capable of taking this information into account in the generation of legible motion.

III. Learning a Library of Motions

The first step of the proposed algorithm involves learning a library of motions through the observation of human demonstrations. This library then serves as a reference data set to perform online classification of trajectories in order to predict an intended target. Stochastic modeling allows the library to store information about both the nominal trajectory of features and about a sense of variability across demonstrations. In this section, we formally describe a data representation of this library of motions.

The generatrix form of this representation begins with the data D taken from one demonstration d, which consists of

a set of vectors v_i, with each vector containing a sampled version, of length $K_{samples_{i,d}}$, of the process of each of the $N_{variables}$ sensed variables. The number of samples can be different for each variable, given different sampling rates from different devices; moreover, the number is different across different demonstrations. The discrete notation for time $[k]$ is used to refer to time step k.

$$D^d = \{v_i[k]\}_{k=1:K_{samples_{i,d}}, i=1:N_{variables}} \tag{1}$$

This collection of raw data is then used to generate a set F^d of features f_i that are more suitable to represent the motion and environment state. The features are derived from the original variables, and can also include signal processing steps, such as re-sampling and filtering, to improve the quality of the data.

$$F^d = \{f_i[k]\}_{k=1:K_{samples_{i,d}}, i=1:N_{features}} \tag{2}$$

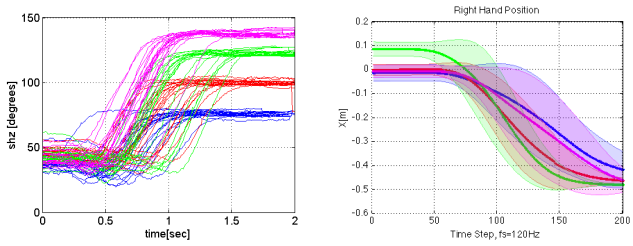
Then, a data set T^t, containing all the $N_{demonstrations_t}$ demonstrations of the same task t, is allotted to serve as the learning source (training set). The length of the features varies between demonstrations since each demonstration can have a different duration in time. For this reason, full dynamic time warping (DTW) [16] is used during this step to determine an optimal alignment among the demonstrations that preserves the waveform of each time series. Full DTW requires the complete time series to be aligned, as we do at the library stage, and runs in $O(n^2)$ time and space. Then, from T^t, we build a library at the motion level, L, with a probabilistic representation for each task L^t based on the mean, μ, and the covariance, Σ, of all the features, as opposed to the task-level probabilistic flow tubes used in [2].

$$\begin{aligned} T^t &= \left\{F^d\right\}_{d=1:N_{demonstrations_t}} \\ L^t &= \{(\mu_i[k], \Sigma_i[k])\}_{k=1:K_{samples_t}, i=1:N_{features}} \\ L &= \left\{L^t\right\}_{t=1:N_{task}} \end{aligned} \tag{3}$$

Finally, the library in Eq.(3) provides a probabilistic model of human motion. In this work, we record the following DOFs of the human arm: roll, pitch and yaw of the upper arm (shoulder rotation); yaw of the lower arm with respect to the upper arm (elbow rotation); roll and pitch of the hand (due to elbow and wrist rotation); and the position of the right hand, (x, y, z), with respect to a defined base reference system (origin). This reference system can be static or dynamic, such that the variables are measured in a global frame or according to a moving system attached to the user, respectively. For rotational joint variables, we attached the origin to the back of the user, and for position variables, such as the position of the hand, we set references in the coordinate frames of the targets, such that the overall system is independent of particular locations but is referenced to the subject and objects of interest. Alternatively, we can use a dimensionality reduction technique to avoid the curse of dimensionality problem by using a smaller but relevant set of features that can be constructed via the collected raw data.

The evolution of all of these features over time can be represented as a multivariate time series, as shown in Fig2, where multiple demonstrations for a task have been drawn, together with the mean and variance, at a given time step. Finally, the model consists of a multivariate Gaussian per time step with mean and covariance computed as shown in Eq.4.

$$\begin{aligned} \mu_i[k] &= \frac{1}{N_{dem_t}} \sum_{i=1}^{N_{dem_t}} f_i[k], \\ \Sigma_i[k] &= \frac{1}{N_{dem_t} - 1} \sum_{i=1}^{N_{dem_t}} (X_i[k] - \mu_i[k]) (X_i[k] - \mu_i[k])^T \end{aligned} \tag{4}$$



(a) (b)

Fig. 2: (a) Raw trajectories of the shoulder's yaw component during 20 demonstrations of four tasks (by color). (b) Modeling of the trajectories of the right hand (x), where the solid line represents the mean and the shaded area represents the variance per time step.

IV. Time Series Classification Algorithm

The objective of the algorithm is to receive the time series of the features corresponding to the human motion to be classified, and to make a decision about what motion class the human is currently executing. The proposed algorithm runs in real-time, iteratively receiving a new set of values corresponding to the features per time step $f_l[k]$ and processing them using the previously collected library of motions L. (We will refer to features stored in the library as f_l, and to features being generated by the human motion as f_h, and will use the bold notation $\mathbf{f}_l[k]$ to represent a vector of features at time step k, instead of the scalar notation $f_i[k], i = 1 : N_{features}$ used in Eq.2).

We compute the log posterior to determine the motion class t that best corresponds to the human trajectory encoded in $\mathbf{f}_h$. We adapt the Bayesian approach proposed in [2] to compute the probability of being in the presence of one of the motion classes t, given the partial segment of the trajectory being executed by the human $f_h[1:k]$, as expressed en Eq.5 using the Bayes rule. We further incorporate task-level knowledge through the inclusion of a hyperparameter that leverages structured information from a customized task level predictor $T(\cdot)$, which takes the set of previous actions a and computes a distribution over the possible motion classes t.

$$P(t \mid f_h[1:k], T(a)) \propto P(t \mid T(a)) \cdot P(f_h[1:k] \mid t), \tag{5}$$

where $P(t \mid T(a))$ is the prior probability of the motion class t given by a task level predictor $T(a)$, and $P(\mathbf{f}_h[1:k] \mid t)$ is the likelihood, which can be modeled as the product of the probability functions of each multivariate Gaussian per time step stored in the library of motions, as expressed in Eq.6. Note we assume that task-level prior is not influenced by the current trajectory $\mathbf{f}_h[1:k]$, and the likelihoods of the motion classes t are independent of the previous set of actions a.

$$P(f_h[1:k] \mid t) = \prod_{k=1}^{K} [\mathcal{N}(\mu_t[k], \Sigma_t[k])]^{\frac{1}{K}} \tag{6}$$

From the log of Eq.5, combined with Eq.6, the log posterior of each motion class at time step $k = K$, is obtained:

$$log(P(t \mid T(a))) + \frac{1}{K} \sum_{k=1}^{K} \left[-log\left((2\pi)^{\frac{N_f}{2}} |\Sigma_t[k]|^{\frac{1}{2}} \right) - \frac{1}{2} \left(\delta^T[k] \Sigma_t{}^{-1}[k] \delta[k] \right) \right], \tag{7}$$

where $\delta[k]$ is the difference between $\mathbf{f}_h[k]$ and $\mu_l[k]$ at a time step k defined by an online DTW algorithm [17] [18] [19].

Unlike the full DTW algorithm used to build the library that requires complete signals, online DTW computes a time alignment between the time series in the library and the partial trajectory being classified. It runs in linear time, $O(n)$, when used to match a segment of trajectory, and runs in constant time, $O(1)$, when using an incremental implementation: at each time step we find a temporal alignment only for the new samples of the partial trajectory [18].

Notice that in Eq. 7, the first term of the sum (g for short) can be computed when building the library of motions offline, as well as the inverse of the covariance matrix in the second term of the sum (h for short), reducing the computation time needed online [2]. Eq.8 reforumations Eq. 7 as a function of g and h and diagrams the task-level and motion-level contributions to the prediction computation.

In Section V we evaluate the motion-level predictor on real data and present a set of tests designed to assess the impact of the task-level prior over the overall performance of the algorithm and establish its performance requirements.

$$\overbrace{\underbrace{log(P(t \mid T(a)))}_{\text{Task Level}} + \frac{1}{K} \sum_{k=1}^{K} [g(\Sigma[k]) + h(\Sigma^{-1}[k], \delta[k])]}^{\text{Prediction}} \tag{8}$$
$$\text{Motion Level}$$

A. Complexity and Scalability

We found that the PFT algorithm in [7][2] is not scalable with respect to the number of features (the number of DOF in this application) and the number of motion classes even when using incremental online DTW. The classification algorithm runs incremental online DTW per each class per each DOF to find the best time step at which to query the log likelihood value. This drives the DTW complexity from $O(1)$ for the alignment of each new observation to $O(n_{class} \times n_{dof})$. However, we found that this result does not typically enable real-time classification for more than three classes. Our application requires up to twelve classes. Given that the likelihood computation of each class is independent, in this work we parallelize the algorithm by querying each class concurrently using a multi-threaded implementation.

V. Performance

In order to evaluate the performance of the classification algorithm, we collected human reaching motion data and evaluated performance offline using repeated random sub-sampling validation (RRSV). This process involves recording N_{Dem} demonstrations per motion class and randomly splitting the data into training and test sets in order to learn the library of motions via the former and run the classification in real-time via the latter (each random split forms a "random library"). This allows us to explore the effect of using different sizes for the training set without biasing the results, as can occur with k-fold cross-validation. We observed no relevant Monte-Carlo variations throughout experimentation.

In the following two subsections, we present the results of this offline validation using real data from a tabletop manipulation task set along one axis (four targets along a line) and two axes (a 2x2 grid of targets in the plane).

A. Manipulation Task Along One Axis

This task required the collection of parts from a table, with four possible targets located along a single axis. Fig.3 shows the 3D trajectories of the right hand of the human operator performing $N_{dem} = 20$ demonstrations of each motion class.

Each motion class is defined by an initial position and a target object to be reached. For this experiment, three different initial positions and four different target objects were used, as indicated in the plot of Fig.3, for a total of 12 possible motion classes. We fully explored this task at the motion level in order to show the internal behaviors of time series classification and prediction (with uniform prior).

In this experiment, target objects were separated by a distance ranging from 25-30 cm. The subject stood 30 cm away along an axis perpendicular to that of the objects. The different initial positions of the subject were separated from the objects by 35 cm along a parallel axis. The initial position of the hand and posture of the arm were not necessarily repeated across demonstrations.

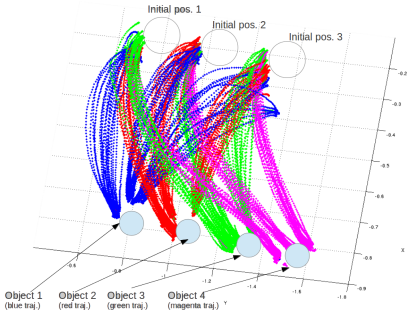
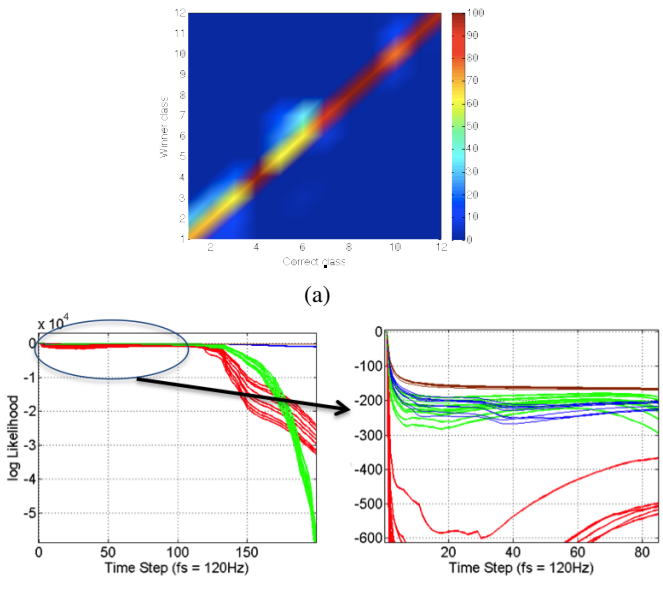


Fig. 3: 3D trajectories of the right hand in the single-axis task.

The confusion matrix is depicted in Fig.4a. Each entry is defined as the average of the percentage of time the algorithm selected a particular motion class W (vertical axis), given an actual correct class C (horizontal axis), while processing each trajectory over the entire library of motions. This result indicates that the algorithm was successful at identifying the correct class. When the classification was incorrect, the chosen class was frequently one of the adjacent classes in space (immediate neighbor object), confining the result to one of the nearest options.

A typical evolution of the log likelihood of each class while the algorithm processes a test trajectory online is depicted in Fig. 4b. The log likelihoods are initially similar, and remain stable or decay as the trajectory is processed. Motion classes that are further from the correct class in the geometric space decay rapidly compared with the correct class and the neighboring classes.

The average processing time per time step was 0.96*msec* when the classification was performed without using DTW. Processes time increased to 12.63*msec* when using incremental on-line DTW without the multi-threaded implementation. As mentioned in Sec.IV-A, this time does not satisfy the on-line requirements. In contrast, the multi-threaded version achieved an average time of 4.96*msec*. In this work, we used a Vicon motion capture system with a sampling frequency of 120 Hz, requiring the algorithm to process each new set of samples in fewer than 8.3*msec* in order to run online. The processing time per time step across the entire library was less than 60% of the time requirement. Comparative results of this and other performance metrics used in previous work are presented in Table I. All tests were performed using an Intel Core i7-3920XM (2.90 GHz, 8MB L3, 1600MHz FSB) and 16 GB of RAM.



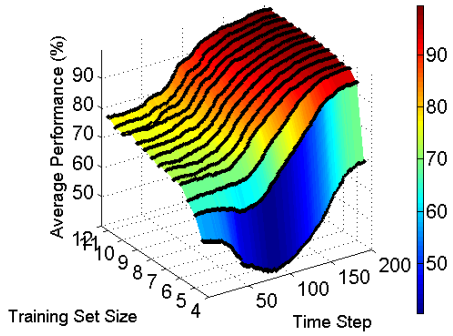
(a)

(b)

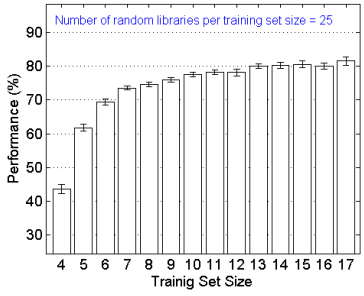
Fig. 4: (a) Confusion matrix for 1D task. (b) Behavior of the log likelihood of each motion class.

Validation was performed using 25 random libraries of motions per training set size (ranging from 4-14 training trajectories) to record the percentage of correct classification per time step across all 12 motion classes in the 25 random libraries and five random test trajectories, for a total of 125 tests per class. The results are plotted in Fig. 5a as a performance surface, indicating the evolution over time of the average correct classification per time step as a function of the number of demonstrations used. Fig.5b shows one transversal slide of the surface at one third of the trajectory processed.

The result suggests that a large number of human demonstrations is not needed for this technique, making it a viable option for real implementation. Previous work in this area [1] [2] had not defined the amount of data required for achieving a goal performance using a partial segment of the trajectory (online early prediction).



(a)



(b)

Fig. 5: Results of the motion-level classification for the single-axis task, using position (x,y,z) of the hand as features. (a) Performance surface; (b) bar graph with error bars (SEM) of the average performance after processing one third of the trajectory, as a function of the training set size.

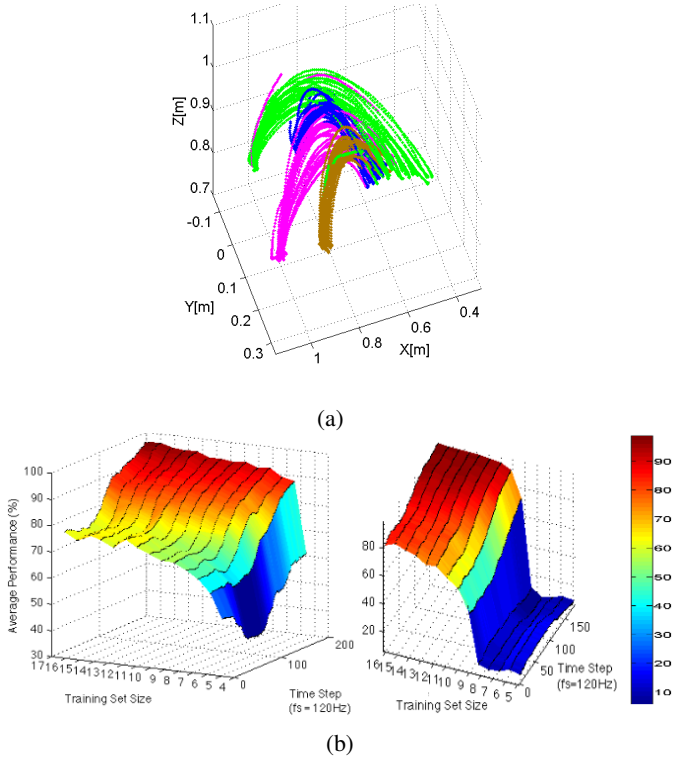
The selection of the features for each task type is informed by a preliminary PCA analysis. However, strict PCA-based feature selection is not guaranteed to work due to nonlinearity within the data [20].

B. Manipulation Task Along Two Axes

Similar performance can be achieved with a two-axes task setup consisting of target objects distributed on a plane. As a study case, we present a task with four motion classes

corresponding to a 2x2 grid. In this experiment, the four target objects were located on a 30 x 40 cm grid, and the subject stood 30 cm away from the first row of objects.

As a baseline comparison, performance was tested using the position of the right hand in three dimensions as the feature set for the classification path. This test resulted in the performance surface shown in Fig.6b(left), which achieved 70% performance with one third of the trajectory, using nine demonstrations. However, in this case, PCA analysis suggested the use of all the DOF of the arm, as opposed to hand position alone. The results of this test are presented in Fig.6b(right), which indicates that the algorithm required a bigger training set (approximately 14 demonstrations) to achieve similar performance, due to the larger feature space. Once enough data were made available, the algorithm outperformed the previous case.



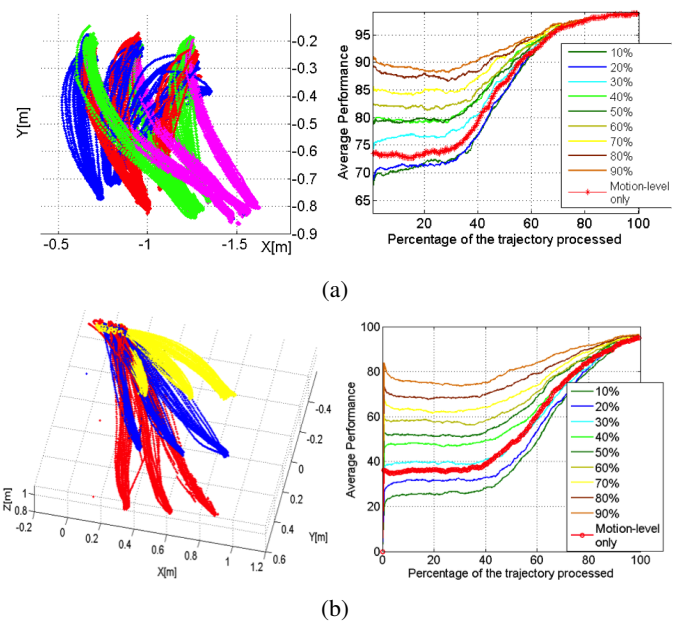
(a)

(b)

Fig. 6: (a) 3D trajectories of the right hand during 20 demonstrations for each of the four targets in the two-axes task. (b) Performance surface results from the motion-level classification for the task in (a) from 25 random libraries, using position (x, y, z) of the right hand (left), and the improved results incorporating all DOFs (right).

C. Study of performance requirements for the task-level prior algorithm

In this section, we analyze the performance requirements a task-level prediction algorithm should meet in order to offer a positive contribution when used to set the prior probability of Eq.8. The evaluation method consisted of multiple simulations of real motion data using random subsampling to build libraries of motions, where the confidence of the task-level algorithm ranged from 10% to 90%. The libraries were then tested using sequences of trajectories.



(a)

(b)

Fig. 7: Performance improvement of using both motion- and task-level information, as a function of the hit rate of the task-level component (right column), for the case of: (a) the single-axis task; and (b) the two-axes task with a nine-target grid. Trajectories are shown in the left column for reference.

The evaluation was performed using the single-axis task presented in SectionV-A, the trajectories for which are plotted in Fig.7a(left). We used a fixed training set size of 10 trajectories, which provided a balance of data size and performance, and present the evolution of performance with respect to the executed fraction of the trajectory. We compared the result of motion-level classification against adding the task-level prior in Fig.7a(right), where the red dotted line is motion-only and the set of all other curves correspond to a hit rate range of 10%-90% for task prediction. Note that in the presence of 12 possible motion classes, a predictor with a hit rate of 8.3% would be comparable to uniform random prediction. In this case, the predictors with hit rates of 10% and 20% actually resulted in a decay of overall performance, as the poor prediction caused the correct motion-level prediction to be neglected according to the probabilities of each side. By contrast, task predictors with a performance over 30% contribute positively to overall performance.

It is important to note that the performance of the dual approach is such that task-level contribution is very relevant during the initial stages of the trajectory, when it is crucial for early prediction of the motion class, before converging to motion-level performance as the trajectory reaches its end. The results also reveal that the contribution of information added by the task prior is asymptotically upper-bounded by the motion level information once the complete trajectory has been processed by the proposed algorithm.

In order to further test the contribution at the task level, we studied a two-axes case with nine possible motion classes

that frequently overlapped. The 3D trajectories of the right hand for these motion classes are shown in Fig.7b(left). As previously indicated, this type of task is not well-suited for the proposed motion classification. However, we show that the use of the task prior can increase performance to levels comparable to successful cases of motion-level prediction. In Fig. 7b(right), the results are presented in the same format as in the previous study case, and indicate similar behavior. The percent improvement *ab initio* from the use of task-level information is approximately 29% and 144% for the single-axis and two-axes cases, respectively, using task predictors with confidence of over 70%. Other cases in which the task-level contribution may be relevant, apart from the nature of the trajectories themselves, include when sensing is not reliable for the tracking of the human DOF, such as current algorithms that use 3D point clouds [21].

D. Summary of Results and Comparison With Related Work

A summary of the results described in Sections V-A and V-B is presented in Table I. We present the following performance metrics, using the ones that are also reported in previous work.

- **M1** The percentage of trajectories correctly classified at the end of the human trajectory.
- **M2** The percentage of time that the classification is correct during the analysis of a trajectory, averaged over all test trajectories.
- **M3** - Processing time per time step [msec], on average.
- **M4** - Partial performance, defined as the percentage of trajectories correctly classified at a given time step. Reported in pairs {% of correct class. / at given % of traj. processed }

Table I shows results reported in previous work (1st and 2dn rows), as well as a direct comparison with GMM under the same test conditions (task, data, number of demonstrations, host machine) in rows 3 to 6. We implemented the GMM classification algorithm as reported in the literature [1], using the expectation maximization (EM) algorithm to fit one mixture of Gaussians per motion class in the library seeded by the output of K-means clustering. We improved upon that implementation by adding a regularizer to avoid singularities, using the Bayesian information criterion (BIC) to select the number of components per model and DTW for temporal alignment [6], and using the same level of optimization as in PFT by pre-computing and storing in the library the quantities that are available in advance, such as the inverse of the covariance matrix.

The proposed method performed better than GMM when the number of Gaussian components in the GMM was limited so that the GMM likelihood computation ran in real-time comparable to PFT. The improvement was most pronounced during the initial segment of the trajectory, and decreased as more observations of the trajectory were processed. Note that early prediction requires high performance at the initial state of the trajectory. The difference in average performance between the two approaches is shown in Fig.8, where PTF achieved 15% improvement in the initial part of the trajectory.

TABLE I: Summary of performance metrics, including related work, at both the motion level (ML) and task level (TL).

Method	N_{dem} / N_{class}	M1	M2	M3	M4
GMM, as reported in [1]	24	92	–	–	50/43
	8				80/60
PFT as reported in [2]	5	97	81.93	21.43	92/80
	7				–
PFT 1-axis ML	7	97.16	92.03	4.96	**73.26/20**
	12				**89.55/43**
					95.76/60
					98.06/80
GMM 1-axis ML	7	99.44	84.58	6.10	**57.08/20**
	12				**85.83/43**
					96.94/60
					99.24/80
PFT 2-axes ML	13	97.38	94.37	5.01	80.00/16.6
	4				90.12/54.44
PFT 2-axes ML & TL(70%)	10	96.46	87.45	6.03	71.21/43.64
	9				79.09/60.00
					90.00/80.00

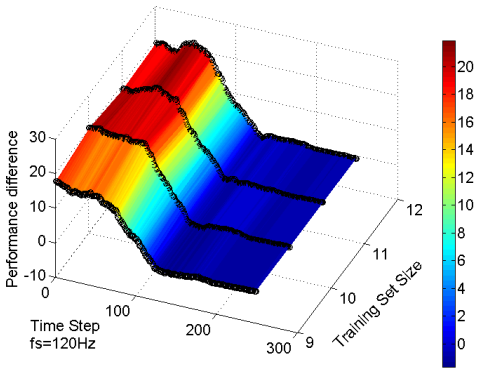


Fig. 8: Performance improvement of PFT over GMM for similar runtime.

VI. Human-Robot Cooperative Task Application

The motion-level classification algorithm was tested online within the framework of a collaborative tabletop task with a PR2 robot. As a proof of concept, we present single-axis and two-axes tasks, as previously defined, in which a human cooperated with a PR2 robot to organize different manufacturing pieces in bins on top a common table. Fig. 9 presents a picture of the task setup. Accompanying this paper, we present a video of the execution of the tasks, available online at http://goo.gl/EiZybP

The implementation consisted of a multi-threaded system that handled the streaming of data from the Vicon motion capture system (120Hz), computed the human joint angles using kinematic transformations, captured the set of features per time step and queried the library of motions to compute the log likelihood as described in Sec. IV.

The task has been designed to exhibit concurrent robot and human motions, where the robot initializes the manipulation motion as required by its task and adjusts its target bin according to the result of the human motion predictor. The human motion predictor runs from the beginning of the human reaching motion up to 416.6*msec* (50 time steps), the instant at which the classification decision is made according to the current likelihood values of each motion

Fig. 9: Pictures of the experimental setup of the single-axis task (left) and two-axes task (right).

class. The time limit for the predictor is set to the point in the performance surface for which the classification is expected to be correct more than 80% of the time for the number of demonstrations used to build the library. The sequence of events is as follows: (1) The initialization of the task in asynchronous between human and robot. (2) The robot starts the motion from the side table to an intermediate fixed position while the prediction is received. (3) The human starts the reaching motion. (4) The predictor runs for 416.6*msec* and sends the decision. (5) The robot goes to a target position at the common table that differs from the prediction; and, finally the process is repeated.

The training set size was 10 demonstrations for each experiment task. Note that our current experimental procedure consists of recording the demonstrations from the same person that executes the task. We intend to study the use of a common library for multiple humans in future work. Out of a total of 20 trials per task, the online performance achieved 80% and 70% correct classification for single-axis and two-axes tasks, respectively.

VII. Conclusion

In summary, the main contributions of this work include the development of a real-time human motion prediction method, with validated performance, that enables online use within the planning loop to inform a path-planner algorithm. This differs from previous work in that the performance of the PFT prediction capability is explored as a function of time in order to assess the feasibility of early prediction, and the scalability of the algorithm is reached through a multi-threading implementation. The approach also incorporates data from real human motion instead of human-guided robot motion to enable human-only demonstrations while using a Vicon motion capture system running at 120Hz, instead of Kinect data at 30Hz. This work also provides an analysis of the performance requirements for a task-level predictor and the behavior of this contribution with respect to the motion-level component, showing its relevance, particularly during the initial part of the trajectory. Future work will involve the testing of different task-level predictors during real task applications, and closing the loop with a path-planner algorithm that interacts with the predictor online. Other future improvement in the performance aspect will include the implementation of a GPU version of the classification queries.

References

[1] J. Mainprice and D. Berenson, "Human-robot collaborative manipulation planning using early prediction of human motion," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pp. 299–306, 2013.

[2] S. Dong and B. Williams, "Learning and recognition of hybrid manipulation motions in variable environments using probabilistic flow tubes," *International Journal of Social Robotics*, vol. 4, no. 4, pp. 357–368, 2012.

[3] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 4569–4574, 2011.

[4] S. Calinon, F. Guenter, and A. Billard, "On learning, representing, and generalizing a task in a humanoid robot," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 2, pp. 286–298, 2007.

[5] S. Calinon and A. Billard, "A probabilistic programming by demonstration framework handling constraints in joint space and task space," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pp. 367–372, IEEE, 2008.

[6] M. Muhlig, M. Gienger, S. Hellbach, J. J. Steil, and C. Goerick, "Task-level imitation learning using variance-based movement optimization," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pp. 1177–1184, IEEE, 2009.

[7] S. Dong and B. Williams, "Motion learning in variable environments using probabilistic flow tubes," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 1976–1981, 2011.

[8] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469 – 483, 2009.

[9] K. P. Hawkins, N. Vo, S. Bansal, and A. F. Bobick, "Probabilistic human action prediction and wait-sensitive planning for responsive human-robot collaboration," in *IEEE Humanoids 2013*, 2013.

[10] K. P. Hawkins, S. Bansal, and A. F. Bobick, "Anticipating human actions for collaboration in the presence of task and sensor uncertainty," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, 2014.

[11] R. Wilcox, S. Nikolaidis, and J. Shah, "Optimization of temporal dynamics for adaptive human-robot interaction in assembly manufacturing," in *Proceedings of Robotics: Science and Systems*, 2012.

[12] S. Nikolaidis and J. Shah, "Human-robot cross-training: Computational formulation, modeling and evaluation of a human team training strategy," in *Proceedings of the 8th ACM/IEEE International Conference on Human-robot Interaction*, HRI '13, pp. 33–40, 2013.

[13] H. Koppula and A. Saxena, "Anticipating human activities using object affordances for reactive robotic response," in *Proceedings of Robotics: Science and Systems*, (Berlin, Germany), June 2013.

[14] A. Dragan, K. Lee, and S. Srinivasa, "Legibility and predictability of robot motion," in *Human-Robot Interaction*, March 2013.

[15] A. Dragan and S. Srinivasa, "Generating legible motion," in *Robotics: Science and Systems*, 2013.

[16] J. Kruskall and M. Liberman, "The symmetric time warping problem from continuous to discrete.," in *Time Warps String Edits and Macromolecules The Theory and Practice of Sequence Comparison*, pp. 125 – 161, 1983.

[17] S. Dixon, "Live tracking of musical performances using on-line timewarping," in *Proc. of the 8th Int. Conference on Digital Audio Effects (DAFx 05)*, (Madrid, Spain), 2005.

[18] S. Dixon, "An on-line time warping algorithm for tracking musical performances," in *International Joint Conference on Artificial Intelligence (IJCAII)*, 2005.

[19] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," in *Intelligent Data Analysis*, pp. 561 – 580.

[20] J. Wang, D. Fleet, and A. Hertzmann, "Gaussian process dynamical models for human motion," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 2, pp. 283–298, 2008.

[21] S. Knoop, S. Vacek, and R. Dillmann, "Sensor fusion for 3d human body tracking with an articulated 3d body model," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pp. 1686–1691, 2006.