2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)
Beijing, China, November 6-9, 2018

# Towards Combining Motion Optimization and Data Driven Dynamical Models for Human Motion Prediction

Philipp Kratzer[1], Marc Toussaint[1] and Jim Mainprice[1,2]
firstname.lastname@ipvs.uni-stuttgart.de
[1]Machine Learning and Robotics Lab, University of Stuttgart, Germany
[2]Max Planck Institute for Intelligent Systems ; IS-MPI ; Tübingen, Germany

***Abstract*— Predicting human motion in unstructured and dynamic environments is challenging. Human behavior arises from complex sensory-motor couplings processes that can change drastically depending on environments or tasks. In order to alleviate this issue, we propose to encode the lower level aspects of human motion separately from the higher level geometrical aspects using data driven dynamical models. In order to perform longer term behavior predictions that account for variation in tasks and environments, we propose to make use of gradient based constraint motion optimization. The present method is the first to our knowledge to combine motion optimization and data driven dynamical models for human motion prediction. We present results on synthetic and motion capture data of upper body reaching movements (see Figure 1) that demonstrate the efficacy of the approach with respect to simple baselines often mentioned in prior work.**

## I. INTRODUCTION

As robots become more capable they will inevitably share the workspace with humans. In this context predictive models of human behavior will become key for high human-robot synergy and safety. Human behavior prediction and understanding is an object of study of different fields including computer graphics, bio-mechanics and robotics, hence there are large differences in approaches developed in prior works (see Section II).

In this work we propose to decouple low-level and high-level movement prediction using data driven dynamical models and motion optimization. We believe this approach will help to generalize over environments and tasks. Motion optimization [1], [2], [3] methods are optimal control algorithms used for motion planning of complex tasks. These techniques approximate trajectory functional gradient descent by discretizing in time, allowing to produce locally optimal movements over a given time horizon. The system dynamics are usually assumed to be known.

We propose a technique that integrates non-linear system dynamics such as learned from human movement data with motion optimization. This framework allows to account for external constraints during movement that may arise form the context (environment or task), such as obstacles or orientation of held object, here we simply treat goal set constraints. Note that integrating other constraints would be straightforward, the reader may refer to [1] for examples of other such constraints.

To demonstrate the efficacy of our approach, we have gathered and segmented 250 reaching upper body movements, see Figure 1. We use this data to learn a very short term dynamical system behavior model, $\approx 0.01$ sec, $s_{t+1} = f(s_t, s_{t-1} \ldots, s_{t-T})$, where $s \in \mathcal{S}$ is a purely kinematic space. We model $f$ using a Gaussian Process (GP) [4] that abstracts all phenomena linked to complex bio-mechanical processes. In order to account for the task context (i.e. reaching goal position) and produce a longer horizon prediction, $\approx 1$ sec, we optimize the mean and variance of the GP together with the goal set constraint.

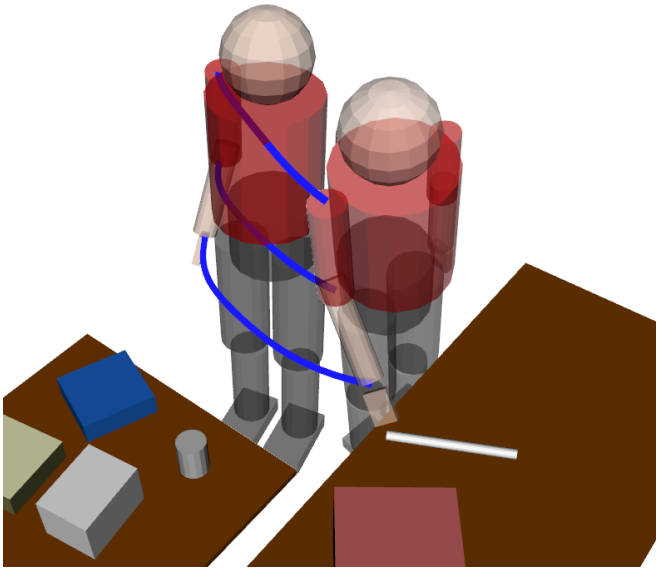


Fig. 1: Human reaching towards the long stick. The initial position and the predicted end state by our method are shown. Blue lines show the trajectories of the right shoulder, right elbow and right wrist as predicted by our method.

To our knowledge, the method presented in this paper is the first to combine motion optimization and data driven dynamical models to predict human motion. This technique has several advantages, 1) decoupling learning of the dynamics holds the promise to generalize better than learning all level of abstractions in one policy, 2) the implementation is simpler than incorporating Newtonian dynamics, 3) modularity of the model (dynamics/kinematics) makes retargeting behavior straightforward.

After reviewing the related work in Section II, we outline our method in Section III and propose a performance analysis by comparing it with some baselines in Section IV. Finally we propose conclusions in Section V.

978-1-5386-7282-2/18/$31.00 ©2018 IEEE

## II. Related Work

### A. Human Motion Prediction

Prior work has made use of graphical models, such as Hidden Markov Models (HMMs) or Conditional Random Fields (CRFs), to predict human motion. Kulić et al. used HMMs to encode full-body motion primitives from observations of human motion and used the model for motion recognition and imitation [5]. Lehrmann et al. used HMMs to retain a dynamic model of human motion and reported good results for motion completion tasks [6]. In [7], Koppula and Saxena predicted trajectories of the human hand using CRFs. Their approach samples possible trajectories by taking object affordances into account. However, while graphical approaches capture relationships between objects well they do not allow for additional constraints that may come from the environment, an issue that we address in this paper.

Recent work on human motion prediction has focused on Recurrent Neural Networks (RNN). Fragkiadaki et al. proposed a RNN based model that incorporates nonlinear encoder and decoder networks before and after recurrent layers [8]. Their model is able to handle training across multiple subjects and activity domains. With a similar approach Martinez et al. [9] reported on using a sequence-to-sequence Long Short-Term Memory (LSTM) architecture that outperformed prior RNN based methods. Although neural network based approaches for motion prediction handle the high dimensionality of the motion data very well, they require a lot of training data and propagation of the uncertainty is still an open research topic [10].

A third approach for predicting human motion is Inverse Optimal Control (IOC) which aims to find a cost function underlying the observed behavior. In [11], Berret et al. investigated cost functions for arm movement planning and report that such movements are closely linked to the combination of two costs related to mechanical energy expenditure and joint-level smoothness. In [12], Mainprice et al. investigated prediction of human reaching motions in shared workspaces. Using goal-set IOC and iterative replanning, the proposed method accounts for the presence of a moving collaborator and obstacles in the environment using a stochastic trajectory optimizer. Generally, IOC learns cost functions which allows to encode behaviors disentangled from the dynamics model and thus can transfer better to others agents [13].

Our work differs in several aspects from prior work in human motion prediction. First, our approach remains low in complexity by not relying on a bio-mechanical model, instead encoding the short term behavior in a data driven dynamical system. Second, we account for additional constraints by optimizing the predicted trajectory with respect to a cost function. This makes it possible to handle environmental constraints, such as the distance to target states. Finally, our model is able to query the predictive uncertainty of the GP that, for example, can be used to weight trajectories or predict workspace occupancy [14].

### B. Gaussian Processes

Our approach for encoding low-level aspects of human motion is based on Gaussian processes (GPs) which are kernel based Bayesian machine learning models. A Gaussian process is completely specified by a mean function $\mu(x)$ and a covariance function $c(x,\ x')$. A random function $f(x)$ can be drawn from the GP $f(x) \sim GP(\mu(x),\ c(x, x'))$. A comprehensive overview of Gaussian processes is available in Rasmussen and Williams [4].

GPs have been used to model human motion in prior works. In [15], Shon et al. learned robotic imitation of human motion using GPs, which they achieved by transforming motion-capture data of the human to a low-dimensional latent space and afterwards transforming it to a high-dimensional robotic state space. Similarly in [16], Wang et al. introduced GP dynamical models for human motion where they made use of a low-dimensional latent space and associated dynamics to represent high-dimensional human motion. In contrast to these prior works we predict the next full human state to enable multistep rollouts and optimization over a time horizon.

GPs have also been used in control to learn nonlinear models of dynamical systems. In [17], Murray-Smith and Sbarbaro developed a nonlinear adaptive control model using a GP that takes the uncertainty prediction into account. In [18], Berkenkamp and Schoellig developed a model to learn control where a GP is used to infer a linear model of the unknown dynamics around some linearization point. The authors found that their framework is a powerful tool to combine nonlinear learning methods with control algorithms.

The problem of human motion prediction is strongly related to time series forecasting. Recent work by Al-Shedivat et al. [10] combined GPs with neural networks. The authors made use of a recurrent neural network to learn kernels with an LSTM structure and found that the method outperforms state-of-the-art results on a number of datasets. We plan to combine our approach with these GP-LSTMs in future work.

## III. Method

Our approach works in three phases: 1) offline we learn a predictive dynamics model of the human $s_{t+1} = f(\xi_t)$ where $\xi_t$ is the observed trajectory and $f(\xi) \sim GP(\mu(\xi),\ c(\xi, \xi'))$, see Figure 2. The aim of the $f$ is to predict the kinematic

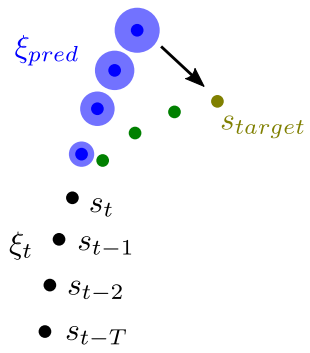



Fig. 2: Trajectory prediction with goal state optimization.

state of the human in the next time step based on a short sequence of previous states along with uncertainty. This is achieved by supervised training of a GP model on human motion capture data, 2) online we use the learned model to unroll a trajectory of future states starting with the state at the current time step $s_t$ and a short sequence of previous states $s_{t-1}, \ldots, s_{t-T}$, finally, 3) the trajectory of future states $\xi_{pred}$ is optimized further by simultaneously minimizing the distances to the mean predictions, the variances of the predictions and additional constraints, for instance goal set constraints.

### A. Problem Statement

We define a trajectory $\xi = s_{0:T} = (s_0, s_1, ..., s_T)$ as a vector of states at discrete time points $0, 1, ..., T$ with $T$ being the length of the trajectory and $s_i \in \mathcal{S}$. The state space $\mathcal{S}$ parametrizes the human posture. In our experiments we use two representations, i.e. positions of joints centers in pelvis coordinates or joint angles.

The goal of our method is to predict a future trajectory $\xi_{pred} = s_{T:T+D}$ given the currently observed trajectory $\xi_t$ that is close to the trajectory a human would perform.

### B. Predicting the Next State

To specify the GP distribution $f$, we make use of the following Radial Basis Function (RBF) kernel defined between trajectories:

$$c(\xi,\ \xi') = \alpha_1 e^{-\frac{1}{2}\alpha_2 ||\xi - \xi'||^2_\Theta} \tag{1}$$

with $\alpha_{1,2}$ being the base hyperparameters of the RBF and $\Theta$ being a $T \times d$ matrix of hyperparameters weighting the entries of the trajectory for time steps $T$ and state dimension $d$. RBF are commonly used with GPs, they have the advantage to be smooth and infinitely differentiable, thus easy to optimize in the trajectory optimization step.

We use the mean function of GP regression for predicting the next state $s_{t+1}$ when observing an unseen trajectory $\xi_t$

$$\bar{f}_{*t+1} = m(s_{t+1} \mid \xi_t, X) = k^\top C^{-1} Y \tag{2}$$

where $X$ is the training data consisting of $N$ pairs $(\xi'_n, s'_n)$ of demonstrated trajectory $\xi'_n$ and the following states $s'_n$.

$C$ is the $N \times N$ covariance matrix with elements $C_{nm} = c(\xi'_n, \xi'_m) + \beta^{-1}\delta_{nm}$, with the trajectories $\xi'_n$ and $\xi'_m$ from the training data $X$ and $\beta$ being a white noise constant. $k$ is a vector with elements $k_n = c(\xi'_n,\ \xi_t)$ and $Y$ is a matrix of states from the training data with rows $y_n = s'_n$.

The corresponding variance function of the GP is

$$\mathbb{V}[f_*]_{t+1} = \sigma^2(s_{t+1} \mid \xi_t, X) = c(\xi_t, \xi_t) - k^\top C^{-1} k \tag{3}$$

which gives the uncertainty about the prediction of the next state given training data $X$ and current trajectory $\xi_t$.

### C. Multistep Prediction

Multistep prediction consists of computing the trajectory $\xi_{t+1} = s_{t-T+1:t+1}$ by appending a predicted state to $\xi_t = s_{t-T:t}$ and iteratively repeating this step. Thus, a fixed time window of size $T$ is used.

By using the mean and the variance of the GP (Equations 2, 3) the trajectory prediction function $g(\xi_t) = \xi_{t+1}$ is

$$g(\xi_t, X) = (s_{t-T+1}, s_{t-T+2}, \ldots, s_t, s_{t+1}) \tag{4}$$

where $s_{t+1} \sim \mathcal{N}(\bar{f}_{*t+1}, \mathbb{V}[f_*]_{t+1})$. For a naive prediction $s_{t+1} = \bar{f}_*$ can be used for all $t$. Multiple trajectories can be generated by sampling multiple $s_{t+1}$ from the GP distribution $\mathcal{N}(\bar{f}_{*t+1}, \mathbb{V}[f_*]_{t+1})$.

### D. Optimizing the Trajectory

In order to improve the predicted trajectory $\xi_{pred} = s_{t+1:t+D}$, with $D$ being the number of predictions, we want to minimize the distance to the mean prediction $\bar{f}_*$ for each future time step as well as the variance $\mathbb{V}[f_*]$ at each time step. Additional cost objectives can, for example, promote close distances to possible target states or penalize close distances to obstacles.

We optimize the cost function $\mathrm{V}(\xi_{pred})$ that is a sum of mean distance costs and variance costs over the states of the predicted trajectory $\xi_{pred}$ to obtain an optimized trajectory $\xi^*_{pred}$:

$$\begin{aligned} \xi^*_{pred} &= \arg\min_{\xi_{pred}} V(\xi_{pred}) \\ V(\xi_{pred}) &= \sum_{d=t}^{t+D} \underbrace{\|s_{d+1} - \bar{f}_{*d+1}\|^2}_{\text{mean cost}} + \underbrace{\gamma \mathbb{V}[f_*]_{d+1}}_{\text{variance cost}} \\ &\qquad \text{subject to } h(\xi_{pred}) = 0 \end{aligned} \tag{5}$$

where $\bar{f}_{*d+1} = m(s_{d+1}|\xi_d, X)$ is the mean and $\mathbb{V}[f_*]_{d+1} = \sigma^2(s_{d+1}, \xi_d, X)$ is the variance at step $d+1$. $\gamma$ is a weight describing the importance of the variance minimization, $h$ being an additional goal constraint and the trajectories $\xi_d = (s_{d-T:d})$ describing the trajectory $\xi_t$ shifted by predicted states. Because of the RBF kernel structure the derivation of the gradient $\nabla_\xi V$ is straightforward. We minimize the cost function using sequential least square programming (SLSQP), an optimization algorithm designed for constrained non-linear optimization problems [19].

Figure 2 shows an example of a two-dimensional trajectory that should be optimized to end at some target state $s_{\text{target}}$. In blue the multistep prediction is shown, green shows the expected trajectory after optimization.

### E. Tuning the Hyperparameters of the Gaussian Process

The covariance function $c(\xi, \xi')$ of the Gaussian Process relies on the scalar hyperparameters $\alpha_1$ and $\alpha_2$ and the matrix hyperparameters $\Theta$.

Because integrals over the parameters in a GP are analytically tractable it is possible to compute the marginal likelihood. The log marginal likelihood for a column vector $y \in Y$ is given by:

$$\ln\, p(y|\alpha_{1,2}, \Theta) = -\frac{1}{2}\ln|C| - \frac{1}{2} y^\top C^{-1} y - \frac{N}{2}\ln(2\pi) \tag{6}$$

Details about how the marginal log likelihood is obtained are available in Rasmussen and Williams [4]. The hyperparameters of the covariance function can be adapted to the

data by minimizing the Negative Log Marginal Likelihood (NLML) of the GP with respect to the hyperparameters (see Equation 6). We can obtain the NLML for each column in $Y$ corresponding to each dimension of the state space and then minimize the sum of the NLML using a Conjugate Gradients (CG) algorithm.

### F. Algorithm

**Algorithm 1** GP Trajectory Optimization

*Offline:*

1: $\alpha_{1,2},\ \Theta \leftarrow$ initialize
2: **while** not converged **do**
3: compute $C, C^{-1}$ from $X$
4: $\alpha_{1,2}, \Theta \leftarrow \text{update}_{\text{CG}}(C, C^{-1}, Y)$

*Online:*

5: Input: $\xi_t$
6: **for** $d = 1$ to $D$ **do**
7: $\xi_{t+d} \leftarrow g(\xi_{t+d-1}, X)$ (see Equation 4)
8: $s_{t+d} \leftarrow$ predicted state from $\xi_{t+d}$
9: $\xi_{pred} \leftarrow (s_{t+1}, s_{t+2}, ..., s_{t+D})$
10: **while** desired accuracy not reached **do**
11: $\xi_{pred} \leftarrow \text{update}_{\text{SLSQP}}(\xi_{pred}, X)$
12: Output: $\xi_{pred}$

Our complete method can be seen in Algorithm 1. Lines 1 to 4 show the offline phase of the algorithm that is used to adapt the hyperparameters of the GP minimizing the NLML loss of the GP. In lines 6 to 9 an initial prediction for the future states $\xi_{pred}$ is computed based on a rollout of the naive multistep prediction. Afterwards, the trajectory is optimized in line 11 using the SLSQP algorithm to minimize the cost function $V(\xi_{pred})$, which we defined in Equation 5, and to fulfill the additional constraints. After the algorithm converges or the desired accuracy is reached it outputs the optimized predicted trajectory $\xi_{pred}$.

## IV. Experiments

To test our method we run different versions of it as well as baselines on one-dimensional synthetic data and on real motion data recorded with a motion capture system.

### A. Datasets

*1) Synthetic Data:* We created a discrete 1D dataset based on cubic splines. To generate the synthetic dataset we randomly sampled values from the set $\{0,\ 0.3,\ 0.8,\ 1\}$ and performed a cubic interpolation between these values. The data was discretized by sampling 50 additional points between the values from the interpolated curve. Starting at random time steps, shorter trajectories from this dataset for training and testing the algorithms were sampled.

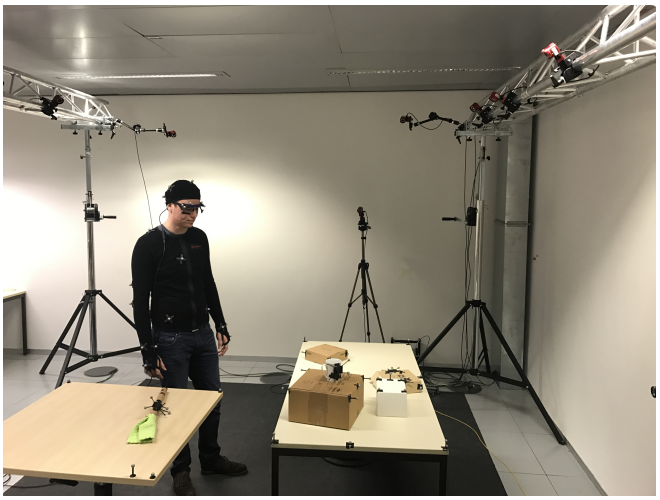


Fig. 3: Motion capture system used in the experiments.

*2) Motion Capture Data:* The human motion dataset was captured using an Optitrack motion capture system. The subject wore a motion capture suit with 25 markers placed on the upper body of the human. The subject was instructed to perform tasks with objects placed on two different tables in the motion capture area (see Figure 3). Possible tasks were placing, drinking, pouring, opening, closing and scrubbing. Each task was preceded by a reaching motion to pick up the objects involved. Marker position data was recorded at a rate of 120 Hz. In total we recorded 132 minutes of motion capture data with two different actors.

Reaching motions naturally yield goal set constraints for the hand, which can be inferred from the object location on the table. In order to evaluate the efficacy of our model to handle such constraints, 250 reaching trajectories of the right hand have been segmented from all tasks. We used a training set size of 200 trajectories and a test set size of 50 trajectories. Three types of state representation are compared with different state representation: 1) **wrist dataset**: only wrist positions, 2) **arm dataset**: joints center positions of the wrist, elbow and shoulder, and 3) **joint angle dataset**: 12 joint angles and 5 translations of the human's upper body and right arm.

### B. Prediction Methods

Test and training set trajectories are sampled of length $T = 10$ corresponding to $0.083$ sec on the real data starting from a random index. We want to predict $D = 30$ points corresponding to $0.25$ sec on the real data in the future. In our experiments we use the following prediction methods:

- **vel_pred:** Predict a trajectory based on the velocity of the current state. The velocity is assumed to be constant.
- **lin_pred:** Linearly interpolate between the current state and the target state.
- **lstm:** Predict a trajectory based on a one-layer recurrent neural network with long short-term memory (LSTM) blocks.

- **gp_multistep:** Use the GP multistep prediction without trajectory optimization.
- **gp_trajopt:** Optimize the gp_multistep prediction regarding mean distance and variance with goal state constraint.
- **ja_multistep:** Predict a trajectory in joint angle space based on the GP multistep prediction method in joint angle space.
- **ja_trajopt:** Optimize the ja_multistep prediction regarding mean distance and variance with goal state constraint. To calculate the distance of the wrist to its goal position, the position of the wrist in real world coordinates is calculated by forward kinematics. We then use the squared distance between the two points. For optimization we additionally use the Jacobian of the wrist.

Note that lin_pred, gp_trajopt and ja_trajopt need additional information about the target state. For the motion capture datasets we consider that only the target state of the wrist is given, but not the target state for the other joints. For example, this situation is given when one wants to predict a reaching motion and knows which objects can be gripped by the human because the wrist will end up close to the object the human wants to pick up.

### C. Hyperparameter Tuning

Training of the GPs is done by hyperparameter tuning as described in Section III-E. We use the same hyperparameters for gp_multistep and gp_trajopt and the same hyperparameters for ja_multistep and ja_trajopt.

Figure 4 shows the results of the hyperparameter tuning in joint angle space. The hyperparameter tuning automatically performs weighting and scaling of the individual dimension of the joints for every time step. Note that certain joints move more than others which makes comparing between rows difficult. For instance, *rArmTrans* corresponding to the translation of the arm has a high value for all time steps, however this translation only changes by less than a millimeter throughout the dataset, hence the influence on the kernel remains very marginal.

Within a row it can be observed that the most recent time step ($t = 9$) has high values that most of the time are decreasing when going further back in the past. This result is expected because the latest time step has most information about how the motion will be continued.

### D. Training set size for synthetic and wrist data

Figure 5 shows a comparison over training sizes for different prediction methods. The prediction is applied at the last 30th time steps of a trajectory. The test set is the same for all training sizes. Because the constant velocity prediction and the linear prediction methods work on the test trajectory without using training data, their loss does therefore not change when increasing the training size.

The first row in Figure 5 shows the performance on the synthetic dataset while the second row shows the performance on the wrist dataset.

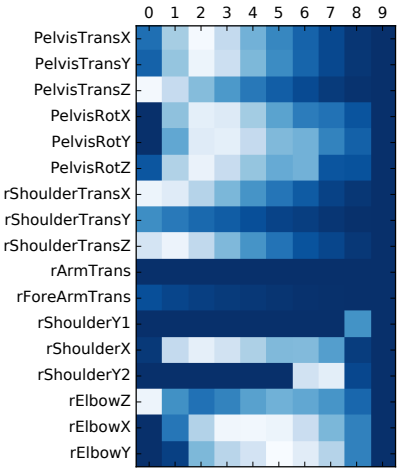



Fig. 4: Results for optimizing the hyperparameters $\Theta$ of the GP regarding the distance between trajectories in joint angle space. The y-axis shows the name of the joint angle, the x-axis shows the previous time steps with time step 9 being the latest step. Darker colors correspond to higher values.

The prediction of the gp_multistep method performs better with increasing training sizes for both datasets (first column). The loss for the gp_trajopt method also slightly decreases with training size. However, the improvements are not as high as for the gp_multistep method because the goal constraint already fixes the target state and only the states in between are improved.

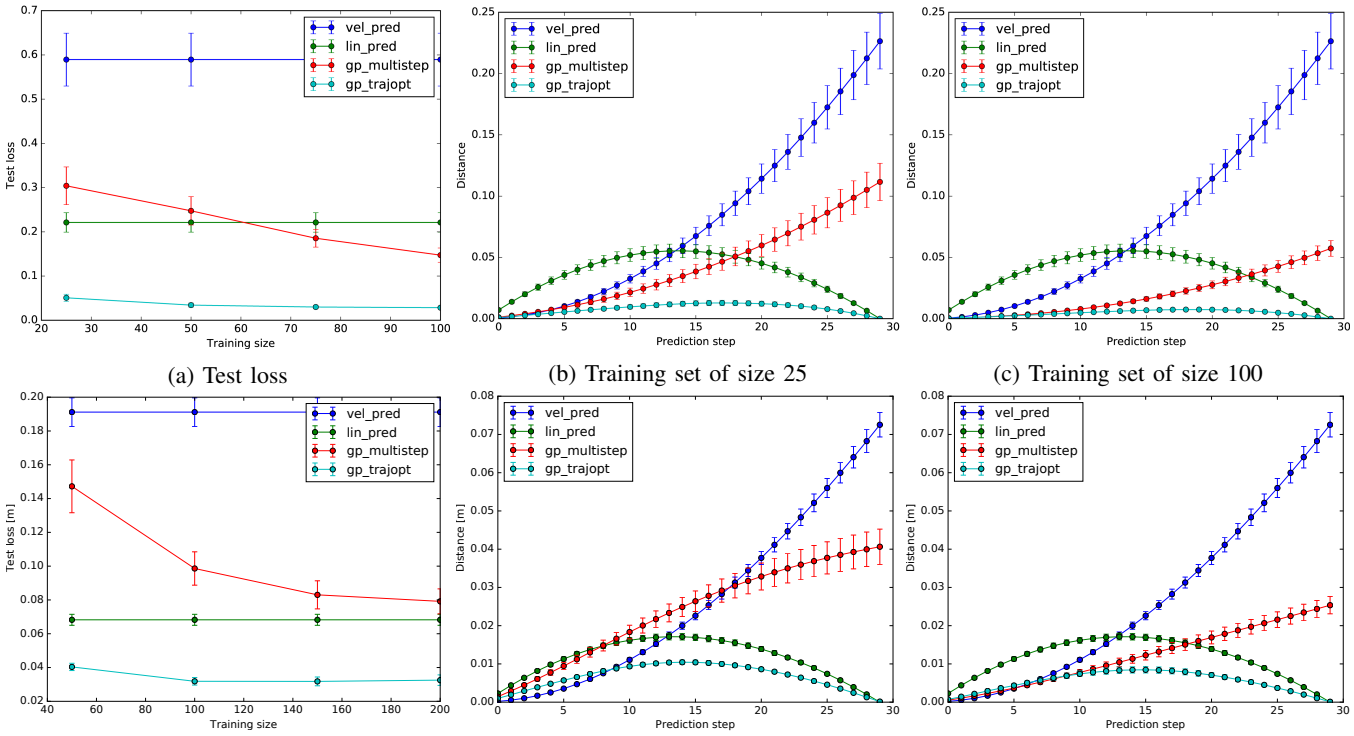
The plots in the second and third row show the prediction step on the x-axis and the mean Euclidean distance to the ground truth of the corresponding prediction step on the y-axis. The second row shows plots for a small training size and the third row for a larger training size. The precision of the predictions of the GP based methods increases with larger training size. The gp_multistep method outperforms the vel_pred method and the gp_trajopt method outperforms the lin_pred method on both datasets, which indicate the efficacy of combining trajectory optimization and GP prediction.

Note that because gp_trajopt and lin_pred are informed with the target position they both end at the correct position, leading to first increasing loss and afterwards decreasing when approaching the target position.

### E. Joint positions accuracy with goal optimization

Figure 6 shows a comparison of the methods in position space (lstm, gp_multistep and gp_trajopt) and the methods in joint angle space (ja_multistep and ja_trajopt). The methods in position space use the arm dataset based on the positions of shoulder, elbow and wrist. To compare the methods in joint angle space to the methods in position space we perform forward kinematics for each configuration in the trajectory to obtain the positions of shoulder, elbow and wrist.

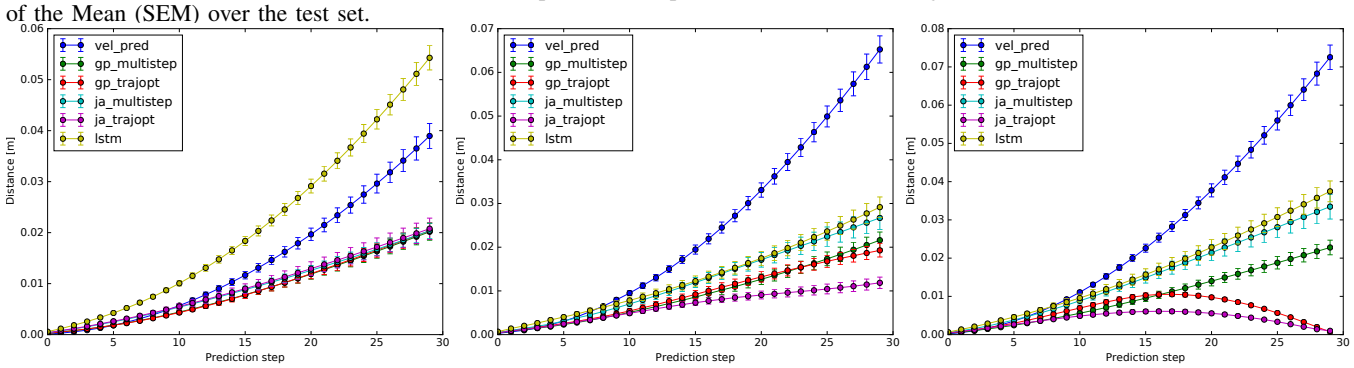
Note that we do not display the LSTM prediction in joint angle space as it performed worse than the other methods. We assume that this is due to the one-layer LSTM not being able to capture the high-dimensional state space correctly and that more complex recurrent neural networks, such as Encoder-




(a) Test loss (b) Training set of size 25 (c) Training set of size 100

(d) Test loss (e) Training set of size 50 (f) Training set of size 200

Fig. 5: Comparison of prediction methods on the synthetic dataset (first row) and the wrist data (second row). The first plot in each row shows the mean Euclidean norm between predicted and correct trajectory over training set size. The other two plots show the mean Euclidean distance between states as a function of the prediction steps for two different training sizes. Error bars show the Standard Error of the Mean (SEM) over the test set.




(a) Shoulder joint (b) Elbow Joint (c) Wrist Joint

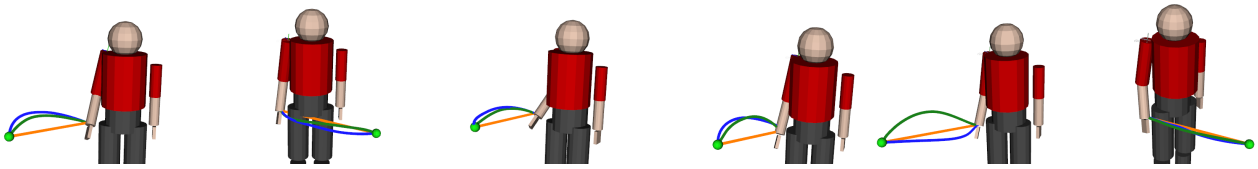
Fig. 6: Comparison of prediction methods on the arm dataset with goal optimization for the wrist joint. The y-axis shows the Euclidean distance to the correct position, the x-axis shows the prediction time step, error bars show the SEM.

Recurrent-Decoder models [8], could be used to overcome this issue.

The results show that all our models outperform the constant velocity prediction on the test set. The ja_multistep performs slightly worse than the gp_multistep, especially for the elbow joint and the wrist joint. This result is expected because the positions of the joints are calculated through the kinematic chain and prediction errors of single joint angles sum up through the chain. However, it can be seen that in joint angle space the prediction improves more when setting the target constraint to the wrist than in position space. This is also expected because the constraint influences the whole kinematic chain which is not the case with the joint center position representation. Thus, although the goal constraint is only used for the wrist joint, the trajectory optimization in joint angle space with goal constraint also improves the prediction for the elbow joint significantly.

### *F. Longer trajectories*

While we focused on short-term trajectories of 0.25 sec in the previous experiments, in this section we propose a study of longer reaching trajectories lasting 1 sec, which are more challenging. We computed 15 test trajectories on the reaching dataset in joint angle space. The mean distance between the predictions for the wrist position by our method and the ground truth is 0.078 meters ($SD = 0.061$) which is



(a) (b) (c) (d) (e) (f)

Fig. 7: Ground truth (green), prediction by our method (blue) and linear prediction baseline (orange). All trajectory durations are 1 sec.

less than the linear prediction with a mean distance of $0.087$ meters ($SD = 0.045$). While our prediction method outputs a trajectory for the whole human the linear prediction is only a baseline for the wrist position.

Figure 7 shows some examples of the predicted trajectories along with the ground truth and the linear prediction baseline. Some of the predicted trajectories are very similar to the ground truth trajectory, however, some trajectories remain further away from the ground truth in the specific case, for example, Figure 7 (e) and (h). We assume that this could be further improved with a GP method able to scale to larger datasets such as [10], as well as optimization methods that combine global and local optimization. We leave these for future work.

## V. Conclusions and Future Work

In this paper we proposed an approach for prediction of human motion that models the dynamic behavior of humans using a Gaussian process and combines it with trajectory optimization to account for additional constraints.

Our experiments on synthetic and motion capture data demonstrate the efficacy of the approach. The experiments show that the prediction using an iterative multistep GP can be improved by optimizing for an additional goal constraint. Moreover, we found that optimizing for a goal constraint for the wrist in joint angle space significantly improves the prediction for the elbow joint as well. Finally, we demonstrated that the prediction method also works well for longer reaching motions.

In future work we investigate the scalability of the approach to handle additional constraints, such as obstacle constraints or increase smoothness by minimizing jerk. We also wish to investigate combining goal constraints for other activities, such as placing or drinking, which could also be obtained by sampling object affordances as proposed by Koppula et al. [7].

## Acknowledgment

This work is funded by the research alliance “System Mensch” of Baden-Württemberg, Germany. All authors are together with the International Max Planck Research School for Intelligent Systems (IMPRS-IS).

## References

[1] J. Mainprice, N. Ratliff, and S. Schaal, “Warping the workspace geometry with electric potentials for motion optimization of manipulation tasks,” in *IEEE/RSJ Int. Conf. on Intel. Rob. And Sys. (IROS)*, 2016.

[2] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, “Chomp: Covariant hamiltonian optimization for motion planning,” *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[3] M. Toussaint, “A tutorial on newton methods for constrained trajectory optimization and relations to slam, gaussian process smoothing, optimal control, and probabilistic inference,” in *Geometric and numerical foundations of movements*. Springer, 2017, pp. 361–392.

[4] C. E. Rasmussen and C. K. Williams, *Gaussian process for machine learning*. MIT press, 2006.

[5] D. Kulić *et al.*, “Incremental learning of full body motion primitives and their sequencing through human motion observation,” *International Journal Of Robotic Research*, vol. 31, no. 3, pp. 330–345, 2012.

[6] A. M. Lehrmann, P. V. Gehler, and S. Nowozin, “Efficient nonlinear markov models for human motion,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[7] H. S. Koppula and A. Saxena, “Anticipating human activities using object affordances for reactive robotic response,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 1, pp. 14–29, 2016.

[8] K. Fragkiadaki, S. Levine, P. Felsen, and J. Malik, “Recurrent network models for human dynamics,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[9] J. Martinez, M. J. Black, and J. Romero, “On human motion prediction using recurrent neural networks,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017.

[10] M. Al-Shedivat *et al.*, “Learning scalable deep kernels with recurrent structure,” *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 2850–2886, 2017.

[11] B. Berret, E. Chiovetto, F. Nori, and T. Pozzo, “Evidence for composite cost functions in arm movement planning: an inverse optimal control approach,” *PLoS computational biology*, vol. 7, no. 10, 2011.

[12] J. Mainprice, R. Hayne, and D. Berenson, “Goal set inverse optimal control and iterative replanning for predicting human reaching motions in shared workspaces,” *IEEE Trans. Robotics*, vol. 32, no. 4, pp. 897–908, 2016.

[13] J. Fu, K. Luo, and S. Levine, “Learning Robust Rewards with Adversarial Inverse Reinforcement Learning.” *arXiv preprint arXiv:1710.11248*, 2017.

[14] J. Mainprice and D. Berenson, “Human-robot collaborative manipulation planning using early prediction of human motion,” in *IEEE/RSJ Int. Conf. on Intel. Rob. And Sys. (IROS)*, 2013.

[15] A. P. Shon, K. Grochow, and R. P. Rao, “Robotic imitation from human motion capture using gaussian processes,” in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. IEEE, 2005, pp. 129–134.

[16] J. M. Wang, D. J. Fleet, and A. Hertzmann, “Gaussian process dynamical models for human motion,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 283–298, 2008.

[17] R. Murray-Smith and D. Sbarbaro, “Nonlinear adaptive control using nonparametric gaussian process prior models,” *IFAC Proceedings Volumes*, vol. 35, no. 1, pp. 325–330, 2002.

[18] F. Berkenkamp and A. P. Schoellig, “Safe and robust learning control with gaussian processes,” in *European Control Conference (ECC)*. IEEE, 2015, pp. 2496–2501.

[19] D. Kraft, “Algorithm 733: Tomp–fortran modules for optimal control calculations,” *ACM Transactions on Mathematical Software (TOMS)*, vol. 20, no. 3, pp. 262–281, 1994.