Prediction of Human Full-Body Movements with Motion Optimization and Recurrent Neural Networks

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—Human movement prediction is difficult as humans naturally exhibit complex behaviors that can change drastically from one environment to the next. In order to alleviate this issue, we propose a prediction framework that decouples short-term prediction, linked to internal body dynamics, and long-term prediction, linked to the environment and task constraints. In this work we investigate encoding short-term dynamics in a recurrent neural network, while we account for environmental constraints, such as obstacle avoidance, using gradient-based trajectory optimization. Experiments on real motion data demonstrate that our framework improves the prediction with respect to state-of-the-art motion prediction methods, as it accounts to beforehand unseen environmental structures. Moreover we demonstrate on an example, how this framework can be used to plan robot trajectories that are optimized to coordinate with a human partner.

I. INTRODUCTION

For safe and efficient human-robot interaction it is crucial to foresee human motion in order to plan around the human partner and interact with the human partner without disturbing the natural flow of the human's motion. A good interaction strategy needs to plan trajectories that minimally intervene with the human while still retaining the ability that both, the human and the robot, can achieve their goals without having to deviate widely from their optimal trajectory.

However, human motion is the result of complex biomechanical processes that are challenging to model. As a consequence, state-of-the-art work on motion prediction focuses on data-driven models, such as recurrent neural network models [3], [4], [5]. A drawback of these architectures is that the network is only trained on the human state and therefore not able to take scene context, such as targets for reaching motion or obstacles into account. Adapting scene context directly into the architecture would require a generalizable scene representation and huge amounts of training data to be able to generalize to unseen environments.

In prior work [6], we proposed to account for environmental constraints in a later trajectory optimization step, using a Gaussian Process (GP) to model the low-level dynamics. However, GPs do not scale to large datasets of training data as they require comparing all training points in the data set to predict the next state. In this work, we instead propose to adapt a state-of-the-art recurrent neural network model [5] to learn purely kinematic predictions of the human. In order to optimize the human motion in a later stage, we introduce a modification to the network architecture to control the
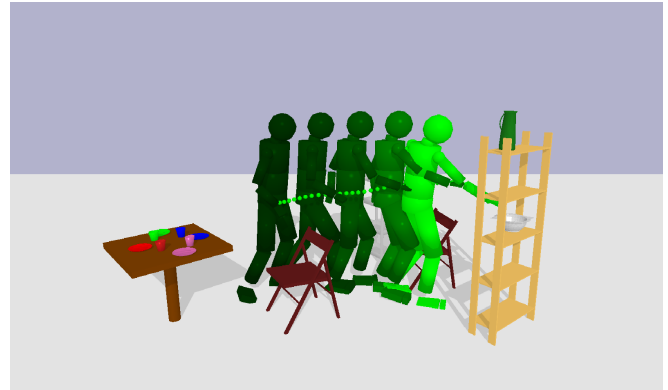

Fig. 1: Prediction of 1.5sec reaching motion towards the bowl on the big shelf by our method

velocities of the human at each prediction step. We can then differentiate the network with respect to this control input, and optimize the motion using a gradient-based optimization algorithm.

While the idea of blending motion capture data and motion planning algorithms for motion prediction is not new [7], to the best of our knowledge, this paper is the first to combine motion optimization with a recurrent neural network in order to predict human motion. This approach has the following advantages: 1) It relies on potentially infinite amount of motion capture data to train the low-level dynamics as the network does not grow with the size of the training dataset, 2) Optimization is a flexible framework for motion planning, which allows to integrate many different constraints (i.e., smoothness, obstacle avoidance, closed kinematic chain, hand orientation, goalset, human-robot interaction, etc.).

In this work, we present experiments on motion capture data recorded at the University of Stuttgart. We test our approach on segmented motions ranging between 1 and 4 seconds as shown in Figure 1 using goalset constraints, obstacle avoidance and joint human-robot coordination. Our results indicate that the additional motion optimization phase leads to higher predictive performance than state-of-the-art neural network predictors, especially in long-term prediction.

This paper is organized as follows: In Section II we discuss relevant prior work. Section III introduce our framework theoretically and explain how the implementation is done. In Section IV we evaluate our prediction framework on real motion data. Conclusions are drawn in Section V.

II. RELATED WORK

A. Human Motion Prediction in Robotics

Prior work on motion prediction in robotics has made use of graphical models. For example, Kulić et al. encoded full-body motion primitives using Hidden Markov Models and applied the model to motion imitation [8]. Koppula and Saxena focused on movement prediction using conditional random fields [9]. While these approaches are sound they generally do not scale to large databases of motion capture.

Another approach commonly used for predicting human motion is Inverse Optimal Control (IOC), which aims to find a cost function underlying the observed behavior. For example, Berret et al. investigated cost functions for arm movement planning [10]. The authors reported that such movements are closely related to mechanical energy expenditure and joint-level smoothness. Mainprice et al. investigated prediction of human reaching motions in shared workspaces [11]. Their method accounts for obstacles and a moving collaborator using iterative replanning. In IOC, bio-kinematic processes are typically represented by simplified models, which are not able to completely capture the complex bio-mechanical behaviors of human full-body motion and to accurately forecast an observed motion.

B. Neural Network Human Motion Prediction

Recent work on human motion prediction for short-term motion has focused on recurrent neural network architectures (RNN). Fragkiadaki et al. proposed a RNN based model that incorporates nonlinear encoder and decoder networks before and after recurrent layers [1]. Their model is able to handle training across multiple subjects and activity domains. Jain et al. introduced a method to incorporate structural elements into a RNN architecture [12]. Autoencoders also can be used for denoising the prediction [2]. Martinez et al. introduced a gated recurrent unit (GRU) based approach with a residual connection in the loop function and showed that this outperforms prior RNN based methods [3]. Pavllo et al. further improved the RNN-based prediction by changing the joint angle representation to quaternions [4], [13]. However, this comes at the cost of additional normalization layers and normalization penalty. Recently Wang and Feng introduced a position-velocity recurrent encoder-decoder model (VRED) [5]. Their model adds an additional velocity connection as an input to the GRU cell in the recurrent structure.

Motion prediction based on recurrent neural networks promises good results for predicting short-term motion. However, the models are trained on human data only. Handling environmental constraints is not possible yet and would require large amounts of training data. Human environments are typically cluttered with objects and obstacles. Thus, in a human-robot collaboration scenario, adapting to such environmental constraints is crucial for the prediction.

For our human prediction model we adapt the VRED architecture by Wang and Feng and modify it as described in Subsection III-D.

C. Motion Optimization

Gradient-based optimization algorithms are widely used in the field of robotics and optimal control [14], [15], [16], [17], [18], [19] for optimizing trajectories. These techniques have been shown to successfully generate motions with a variety of kinematic and dynamic objective and constraints. In 2009 [15], Ratliff et al. have used insights from differential geometry making it possible to use gradient-based optimization to solve motion planning with non-convex obstacles. Their main finding was that obstacle costs should be integrated with respect to arc-length in Cartesian space, leading to the notion of workspace geodesics [20], [21], which we use here.

Mordatch et al. use motion optimization techniques to synthesize complex behaviors [22], [23]. The authors show that motion optimization approaches for synthesizing motion and animating characters can generate realistic motions even without the use of motion capture data. While the focus of their work does not lie on forecasting an observed motion, concepts, such as constraining foot contacts, have great potential to be incorporated in our motion prediction framework.

III. METHOD

Next we formalize our approach, by presenting first our method for predicting human motion and then for planning a coordinated human-robot behavior.

A. Human Motion Optimization

When predicting human motion, the aim is to find a trajectory $s_{t+1:T}^*$ that maximizes the likelihood of the next states given our demonstration data $\mathcal{D}$ and fulfills the constraints. The constraints are the human dynamics $s_{t+1} = d(s_t)$ and further constraints h and g that arise from the environment.

This can be written as the following optimization problem:

$$\min_{s_{t+1:T}} \sum_{t+1}^{T} -\log p(s_{t+1}|s_{0:t}, \mathcal{D}) \tag{1}$$

$$\text{subject to } s_{t+1} = d(s_t)$$
$$h(s_t) = 0$$
$$g(s_t) \leq 0$$

We approximate the maximum likelihood given $\mathcal{D}$ as a regression problem and train a recurrent neural network model to find the prediction states $s_{t+1:T} = f(s_{0:t}, \delta_{t+1:T})$ as discussed in Subsection III-D. We introduce parameters $\delta_{t+1:T}$ to the model in order to be able to vary the predictions and thus account for the constraints h and g using numerical optimization. We will describe this in Subsection III-E. The dynamic constraint $s_{t+1} = d(s_t)$ is learned from the data and thus also approximated by the recurrent neural network f.

B. Coordinated Motion Optimization

In the case of joint motion optimization of the human and the robot, we not only aim to find a human trajectory $s_{t+1:T}^*$
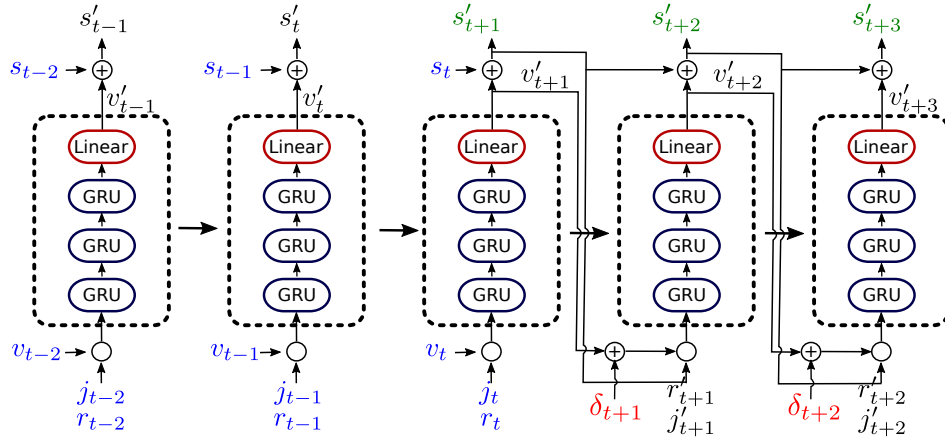
1793

Fig. 2: Architecture of the Recurrent Neural Network $s_{t+1:T} = f(s_{0:t})$. In blue the past joint angles and velocities, in green the future predicted states and in red the control input δ, which allows motion optimization.

but also a robot trajectory $x^*_{t+1:T}$. The optimization problem changes to:

$$\min_{s_{t+1:T}, x_{t+1:T}} \sum_{t+1}^{T} -\log p(s_{t+1}|s_{0:t}, \mathcal{D}) + c_R(x_{t+1:T}) \quad (2)$$

$$\text{subject to } s_{t+1}, x_{t+1} = d(s_t, x_t)$$
$$h(s_t, x_t) = 0$$
$$g(s_t, x_t) \leq 0$$

with C_R being costs on the robot trajectory $x_{t+1:T}$. The additional constraints h and g can now be function of both, s_t and x_t, and arise from joint interaction objectives, such as collision constraints.

C. Summary

Our approach works in two phases: 1) offline we learn a predictive model of the human $s_{t+1:T} = f(s_{0:t})$ where $s_{0:t}$ is the observed trajectory of human states. The aim of f is to predict the kinematic states of the human in the next time steps up to a prediction horizon T, based on a sequence of previous states. This is achieved by supervised training of a position-velocity recurrent encoder-decoder neural network model on human motion capture data, 2) online we use the learned model to predict future states. The prediction is optimized to fulfill constraints by varying the velocity inputs in the decoder's loop function at every time step.

D. Human Model

For prediction of the human we base our model on the position-velocity recurrent encoder-decoder neural network (VRED) [5]. We model the kinematic state of a human as a vector consisting of base position, base rotation and joint angles: $s = (p, r, j)$ which in our case is a 66 dimensional vector for full-body data. The joints are toes, ankles, knees, hips, pelvis, torso, neck, head, inner shoulders, shoulders, elbows and wrists.

As proposed in [5], we represent angles in the exponential map representation and convert them to quaternions for the loss computation. The velocity inputs v are computed using finite differences. At each time step the positions and

velocities are fed into the network, which then only predicts velocities. The new states s' are obtained through a residual connection by adding the velocity to the previous state.

The full architecture can be seen in Figure 2. Inputs variables are depicted blue, the new predictions are depicted green. In contrast to the original model, we stack 3 GRU cells with 1000 hidden units, which improves the prediction performance in our experiments Additionally we do not feed the position of the base (i.e., pelvis) into the recurrent unit. This avoids conditioning the model on world positions and leads to better generalization. We found that the same approach does not hold for the base rotation which encodes the direction of motion. Hence we instead offset the rotation randomly during training time to avoid overfitting.

For training, the data is sliced into same sized trajectories of 2sec. One second is fed as input to the network and 1sec is predicted by the network. The loss is computed on the full trajectory. We use a squared error loss for the base position and a quaternion loss for the base rotation and joint angles:

$$L = \sum_{s_{0:T} \in \mathcal{B}} \underbrace{||p' - p||^2}_{\text{position}} + \sum_{t,q} \underbrace{\min(||q'_t - q_t||, ||q'_t + q_t||)}_{\text{quaternion}} \quad (3)$$

with q_t being the quaternion representations. The learning rate is set to 0.0001 and the batch size is 8.

At test time we make use of numerical optimization to handle additional constraints. In order to change the predictions we add an additional connection δ to the velocity inputs (see Figure 2 depicted red). The network thus has additional input parameter $s'_{t+1:T} = f(s_{0:t}, \delta_{t+1:T})$. These parameters can be used to change the velocities that are predicted by the neural network and fed into the next time step which can be viewed as adding an additional acceleration term into the model.

During training δ is set to zero. Note that the δ_t is fed into the network at one time step but changes the predictions for all the following time steps as well. In the following, the neural network $f(\delta_{t+1:T})$ will only be parameterized by δ because $s_{0:t}$ is already observed at test time and is not able to change during prediction.

E. Trajectory Optimization

In this paper we consider optimizing predictions with the ability to account for the following additional constraints:

- *low-level:* human prediction should be close to the original network prediction.
- *goalset:* human prediction should end up with a hand close to a specified point.
- *collision:* human prediction should not collide with obstacles.
- *human-robot:* human prediction and a robotics agent should not collide.
- *robot smoothness:* robot trajectory should be smooth.
- *robot goal:* robot trajectory should end with an endeffector close to a specified point.
- *robot collision:* robot trajectory should not collide with obstacles.

Due the fact that δ is only used as input to the neural network and does not directly change the trajectory, it is ensured that the predicted states still ground on the network prediction. However, we add an additional loss term to fulfill the *low-level* constraint:

$$c_c(\delta) = \|\delta\|^2 \tag{4}$$

This term ensures that δ is not becoming too big and thus does not push the network into states that are too far from the training data to make reliable predictions.

The *goal* is given as the squared distance between the human endeffector and a reference point p^*:

$$c_g(\delta) = \|\phi_{FK}(f(\delta)_T) - p^*\|^2 \tag{5}$$

where $\phi_{FK} : s \mapsto p$ is the forward kinematics of the human, for example, calculating the hand position $p \in \mathbb{R}^3$. Thus $p_t = \phi_{FK}(f(\delta_t)))$ computes the hand position at time $t = T$, where T is the last predicted time step by the network.

We represent objects using a signed distance field (SDF). To account for the *collision* a potential function, similar as used in CHOMP [15], is used. Because moving more quickly through regions with high cost should not be penalized less, cost elements are integrated with respect to an arc-length parameterization in the workspace. Thus in our case, we have the following obstacle potential:

$$c_o(\delta) = \sum_{t=1}^T \exp\left\{-\alpha \; \text{SDF}(p_t)\right\}\Delta_H \tag{6}$$

where $\Delta_H = \|p_{t+1} - p_t\|$, with $p_t = \phi_{FK}(f(\delta_t)))$.

Similarly we define the *human-robot objective*, as a function of the human variables δ and robot states x as:

$$c_j(\delta, x) = \sum_{t=1}^T \exp\left\{-\alpha\|p_t - x_t\|\right\}\Delta_H \Delta_R \tag{7}$$

where $\Delta_R = \|p_{t+1}^R - p_t^R\|$, with $p_t^R = \phi_{FK}(x_t)$.

The *robot goal* c_{Rg} and *robot obstacle* c_{Ro} are similar to the human goal and obstacle constraints with the difference that we optimize the trajectory directly and do not pass it through the neural network.

Fig. 3: Motion Capture Setup

The *robot smoothness term* is similar, as defined in CHOMP, to a sum of squared derivatives:

$$c_{\text{smooth}}(x) = x^\top K_d x \tag{8}$$

where K is a finite differences matrix:

$$K = \begin{bmatrix} 6 & -4 & 1 & \dots & 0 & 0 & 0 \\ -4 & 6 & -4 & \dots & 0 & 0 & 0 \\ 1 & -4 & 6 & \dots & 0 & 0 & 0 \\ \vdots & & \ddots & & & \vdots \\ 0 & 0 & 0 & \dots & 6 & -4 & 1 \\ 0 & 0 & 0 & \dots & -4 & 6 & -4 \\ 0 & 0 & 0 & \dots & 1 & -4 & 1 \end{bmatrix}. \tag{9}$$

The full unconstrained proxy objective is given as the sum of the cost functions for human, robot and joint cost:

$$c(\delta, x) = c_H(\delta) + c_R(x) + c_J(\delta, x) \tag{10}$$

$$c_H(\delta) = \lambda_1 c_c(\delta) + \lambda_2 c_g(\delta) + \lambda_3 c_o(\delta) \tag{11}$$

$$c_R(x) = \lambda_4 c_{Rg}(x) + \lambda_5 c_{Ro}(x) + \lambda_6 c_{\text{smooth}}(x) \tag{12}$$

$$c_J(\delta, x) = \lambda_7 c_j(\delta, x) \tag{13}$$

with Lagrange multiplier λ. Note that implementing this framework using a principled augmented Lagrangian optimization algorithm would be straight forward and is left for future work.

We derived the respective gradients and Jacobians mainly using automatic differentiation frameworks. We implemented the recurrent neural network using the Tensorflow library and we use its automatic differentiation functionality to compute the Jacobian of the network $J_{\text{NN}} = \frac{\partial f}{\partial \delta}$. We use a limited memory version of the numerical optimization algorithm BFGS in order to optimize the trajectory [24].

IV. EXPERIMENTS

In this section we evaluate our method using real motion data. Since available motion capture datasets do not include obstacles, we gathered our own dataset, which can be made available on request.

All data was captured using an Optitrack motion capture system with 50 reflecting markers placed on the entire human body. The motion capture setup can be seen in Figure 3. Data was recorded at a rate of 120Hz and downsampled to 30Hz for use in the network. For the experiments we captured 3 datasets using one single actor. A *reaching1* dataset, which contains pick and place motions of different objects on different heights and lasts 31 minutes. A *walking*
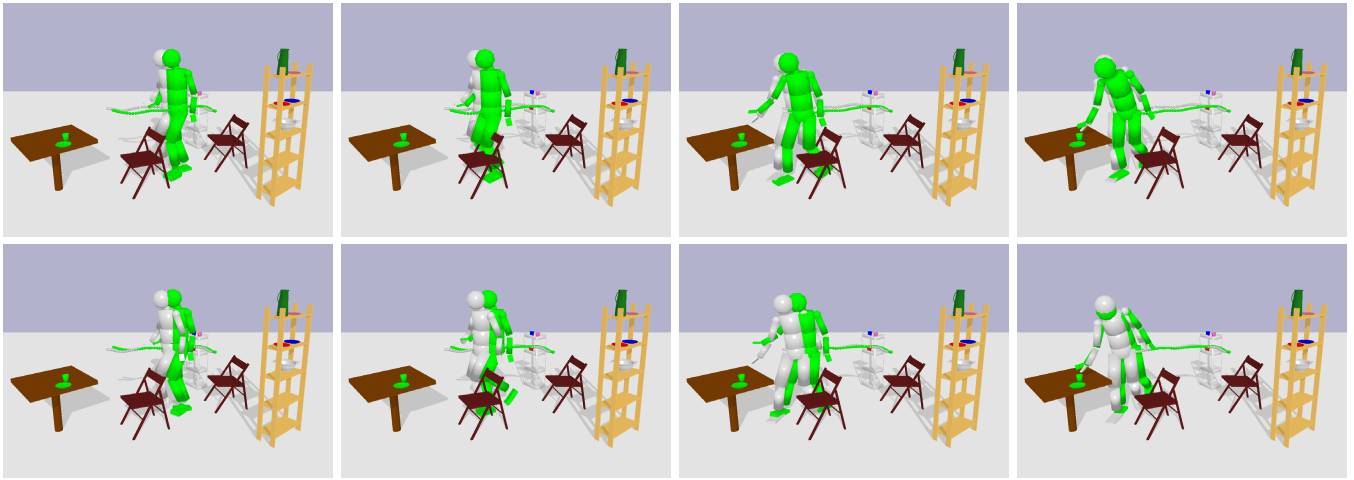
1795

Fig. 4: Human trajectory of reaching towards an object on the table for 2secs. The prediction by our method is shown in green, baseline of human motion in gray. From left to right we show the prediction after 1sec, 1.3sec, 1.6sec and 2sec. The top row shows the prediction without obstacle constraint, the bottom row with obstacle constraint. In the prediction without obstacle constraint the human collides with the chair.

dataset, which consists of 60 minutes of walking data in the motion capture area. A *reaching2* dataset, which contains 22 minutes of pick and place motions of different objects similar to *reaching1*, however, chairs have been placed in the scene so that the human has to perform the pick and place tasks while walking around the chairs. We train our model with data from all three datasets. A fraction of 10% from every dataset is held out for testing purposes.

A. Goal Set Constraints

In our first experiment we evaluate whether the prediction of the VRED network can be improved by our prediction method when we already know the target position of the reaching motion for the human hand.

From the *reaching1* test set 25 reaching trajectories with a length of 1sec have been extracted. We compare our method with three baselines. We compute the distance of key joints of the human (wrists, elbows, knees, ankles and pelvis) and compute the mean distance of the predictions to the ground truth (see Table I).

The zero velocity baseline predicts the same state for all future steps. The VRED baseline is the prediction network without the trajectory optimization part. Our method is informed with the goal position of the hand. Table I (b) shows the sum of the mean distances of the 9 key joints. The use of trajectory optimization improves the prediction among all future steps.

In Table I (w) we only compute the distance of the wrist to the ground truth. We also compute a linear interpolation baseline between the start position of the wrist and the target position. The interpolation baseline and our method are informed with the goal state and thus able to get zero error in the last time step. However, our method, where the recurrent neural network implicitly reconstructs the underlying human dynamics, outperforms the interpolation baseline on other time steps, which simply constructs a naive straight line.

The results show that our method is able to reconstruct the

ms	125	250	375	500	625	750	875	1000
Zerovel (b)	0.72	1.37	1.80	2.31	2.72	3.01	3.15	3.25
VRED (b)	**0.20**	0.36	0.45	0.57	0.68	0.78	0.86	0.94
ours (b)	**0.20**	**0.35**	**0.44**	**0.53**	**0.56**	**0.59**	**0.62**	**0.64**
Zerovel (w)	0.14	0.28	0.37	0.48	0.56	0.61	0.62	0.62
VRED (w)	**0.03**	**0.07**	0.09	0.11	0.13	0.14	0.15	0.15
ours (w)	**0.03**	**0.07**	**0.08**	**0.09**	**0.08**	**0.06**	**0.04**	0.01
Interp (w)	0.05	0.11	0.15	0.17	0.17	0.13	0.08	**0.00**

TABLE I: Error of state prediction on different time steps in the future for the whole body (b) and the right wrist only (w). Reported values are in meters. For the whole body the sum distance of 9 key joints is shown.

ms	250	500	750	1000	1250	1500	1750	2000
Zerovel	1.21	2.45	3.90	5.50	7.53	9.49	11.15	11.74
VRED	0.70	1.25	1.79	2.48	3.48	4.56	5.70	6.39
ours g	**0.68**	**1.25**	1.79	2.27	2.57	2.52	2.24	2.18
ours g+o	0.69	**1.25**	**1.74**	**2.18**	**2.45**	**2.41**	**2.15**	**2.10**

TABLE II: Error of state prediction on different time steps in the future for the whole body. Our method with goal objective (g) and goal and obstacle objectives (g+o) is shown.

full-body trajectory of the human given a target state which is useful for scenarios where we already know possible target states of the human obtained by a higher level prediction mechanism or through scene understanding, for example using affordances [9].

A limitation of the approach with goal set constraint is that we have to predefine the duration of the trajectory which is a known problem in the trajectory optimization literature [15]. Simple approaches to solve this issues include optimizing trajectories of different time horizon or reoptimizing the trajectory with a longer horizon if the target position is not reached. A more general approach would dynamically add and remove samples during optimization.

B. Obstacle Constraints

In this experiment we evaluate whether the obstacle potential helps to further improve the prediction. Therefore we extracted 22 reaching trajectories from the *reaching2* test
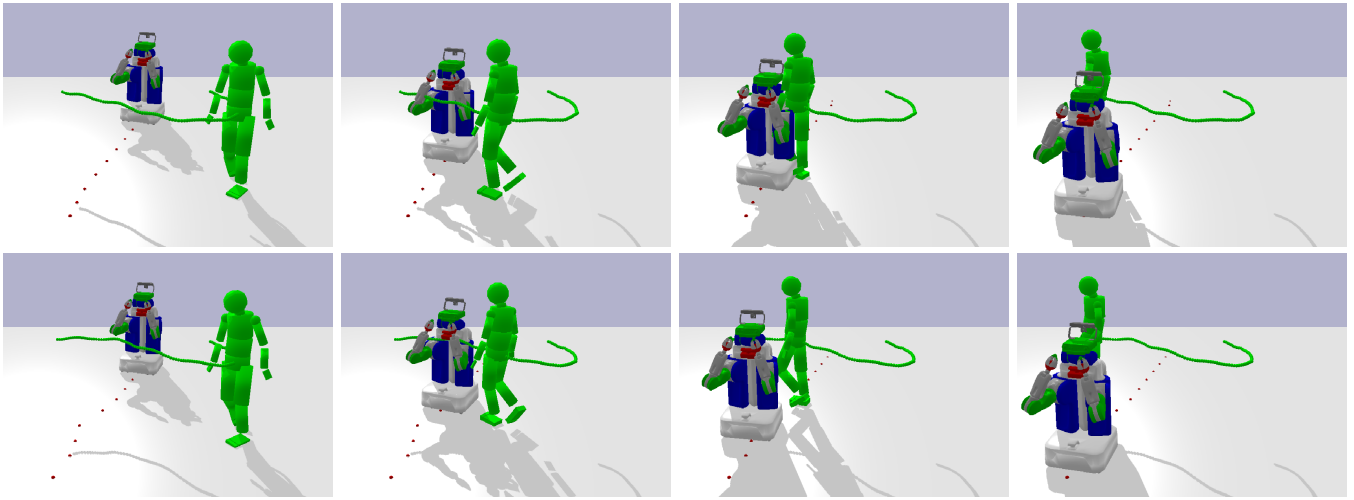
Fig. 5: Joint trajectory optimization of human and robot. From left to right the trajectories after 1sec, 2.3sec, 3sec and 3.6sec. The top row shows the trajectory without the joint optimization potential, the bottom trajectory is with joint optimization.

set with a length of 2sec. Some of the trajectories contain motion trajectories that are close to an obstacle (chair). We approximate the chair sign distance field as a sphere and add an obstacle objective in addition to the goal objective.

Figure 4 shows an example reaching trajectory with the ground truth (gray) and our prediction (green). The top row shows the prediction without obstacle objective. There the human prediction collides with the chair because the RNN has no information about obstacles in the scene. Activating the obstacle objective in the trajectory optimization step penalizes collisions and forces the prediction around the object. In the second row the improved prediction with obstacle avoidance is shown. The prediction no longer collides with the chair.

We performed a quantitative comparison averaged over the 22 extracted reaching motions (see Table II). The table shows the mean full body error of the key joints compared to the ground truth. Note that only a few of the trajectories contain motion trajectories that are close to the obstacle. The method with additional obstacle objective outperforms the other prediction methods.

C. Joint Human Robot Optimization

The last experiment is about joint optimization of human and robot for collaborative planning. We initialize a human walking trajectory from the test data of the *walking* dataset. Additionally, we add a robot model. We set a goal objective for the human base and the robot base so that the trajectories of human and robot intersect. We initialize the robot trajectory as a straight line from start to target position and the human trajectory with setting the δ to zero. The trajectories are predicted for a duration of 4sec. Using the goal objectives without the human-robot objective will lead to a collision of robot and human (see Figure 5 top). Only planning for the robot and assuming that the human stays on the predicted trajectory would lead to a route where the robot completely avoids the human and has to go for a longer path around. In reality this is not necessary because

the human would react to the robot moving and adapt his or her walking path to the robot moving. Similar, if we keep the robot trajectory fixed, the human would either need to wait till the robot passes or walk a longer way around the robot, which results in reaching the goal state later. Our aim is to find a collision free plan for human and robot so that both, the human and the robot have to leave the optimal route only a bit. Therefore we jointly optimize the robot states x and the human prediction variables δ including the human-robot objective. The resulting trajectory can be seen in Figure 5 bottom. The robot speeds up a bit and keeps farther away from the human and the human prediction walks a bit further to the right.

The experiment shows that our method can be used to plan a robot trajectory while simultaneously adapting the human prediction to it. This can be used to plan a trajectory that minimally changes the human path. How much the robot or the human has to deviate from the optimal path can be modified by adapting the optimization parameters λ.

V. CONCLUSIONS

In this paper we presented a novel prediction framework for human motion in the presence of environmental objectives. We showed that a state-of-the-art recurrent neural network model can be adapted to use it within a trajectory optimization framework. This improves the predictions and accounts to unseen environmental structures.

Furthermore, we showed an initial experiment on how the method could be used for shared human-robot planning.

For future work we plan to conduct real world experiments on real human-robot interaction tasks. We plan to optimize the trajectory optimization parameters in a way to adapt for human comfort during more complex interaction scenarios

ACKNOWLEDGMENT

REFERENCES

[1] 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, pp. 4346–4354.

[2] P. Ghosh, J. Song, E. Aksan, and O. Hilliges, "Learning human motion models for long-term predictions," in *2017 International Conference on 3D Vision (3DV)*. IEEE, 2017, pp. 458–466.

[3] 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.

[4] D. Pavllo, D. Grangier, and M. Auli, "Quaternet: A quaternion-based recurrent model for human motion," in *British Machine Vision Conference (BMVC)*, 2018.

[5] H. Wang and J. Feng, "Vred: A position-velocity recurrent encoder-decoder for human motion prediction," *arXiv preprint arXiv:1906.06514*, 2019.

[6] P. Kratzer, M. Toussaint, and J. Mainprice, "Towards combining motion optimization and data driven dynamical models for human motion prediction," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2018, pp. 202–208.

[7] J. Pettré, J.-P. Laumond, and T. Siméon, "A 2-stages locomotion planner for digital actors," in *Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2003, pp. 258–264.

[8] D. Kulić, C. Ott, D. Lee, J. Ishikawa, and Y. Nakamura, "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.

[9] 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.

[10] 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.

[11] 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.

[12] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-rnn: Deep learning on spatio-temporal graphs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5308–5317.

[13] D. Pavllo, C. Feichtenhofer, M. Auli, and D. Grangier, "Modeling human motion with quaternion-based neural networks," *International Journal of Computer Vision*, pp. 1–18, 2019.

[14] E. Todorov and W. Li, "A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems," in *Proceedings of American Control Conference, 2005*. IEEE, 2005, pp. 300–306.

[15] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 489–494.

[16] J. Schulman, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization." in *Proceedings of Robotics: Science and systems(RSS),*, vol. 9, no. 1, 2013, pp. 1–10.

[17] M. Toussaint, "Newton methods for k-order markov constrained motion problems," *arXiv preprint arXiv:1407.0414*, 2014.

[18] Z. Marinho, A. Dragan, A. Byravan, B. Boots, S. Srinivasa, and G. Gordon, "Functional gradient motion planning in reproducing kernel hilbert spaces," *arXiv preprint arXiv:1601.03648*, 2016.

[19] 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.

[20] N. Ratliff, M. Toussaint, and S. Schaal, "Understanding the geometry of workspace obstacles in motion optimization," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 4202–4209.

[21] 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.

[22] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, pp. 1–8, 2012.

[23] I. Mordatch, J. M. Wang, E. Todorov, and V. Koltun, "Animating human lower limbs using contact-invariant optimization," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 6, pp. 1–8, 2013.

[24] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, vol. 16, no. 5, pp. 1190–1208, 1995.