Real-Time Motion Prediction for Efficient Human-Robot Collaboration

by

Aadi Kothari

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2023

Authored by: Aadi Kothari
Department of Mechanical Engineering
September 01, 2023

Certified by: Kamal Youcef-Toumi
Professor of Mechanical Engineering, Thesis Supervisor

Accepted by: Nicolas Hadjiconstantinou
Chairman, Department Committee on Graduate Theses

Real-Time Motion Prediction for Efficient Human-Robot Collaboration

by

Aadi Kothari

Submitted to the Department of Mechanical Engineering
on September 01, 2023 in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

ABSTRACT

Human motion prediction is an essential step for efficient and safe human-robot collaboration. Current methods either purely rely on representing the human joints in some form of neural network-based architecture or use regression models offline to fit hyper-parameters in the hope of capturing a model encompassing human motion. While these methods provide good initial results, they are missing out on leveraging well-studied human body kinematic models as well as body and scene constraints which can help boost the efficacy of these prediction frameworks while also explicitly avoiding implausible human joint configurations. We propose a novel human motion prediction framework that incorporates human joint constraints and scene constraints in a Gaussian Process Regression (GPR) model to predict human motion over a set time horizon. This formulation is combined with an online context-aware constraints model to leverage task-dependent motions. It is tested on a human arm kinematic model and implemented on a human-robot collaborative setup with a UR5 robot arm to demonstrate the real-time capability of our approach. Simulations were also performed on datasets like HA4M and ANDY. The simulation and experimental results demonstrate considerable improvements in a Gaussian Process framework when these constraints are explicitly considered.

Thesis supervisor: Kamal Youcef-Toumi
Title: Professor of Mechanical Engineering

Acknowledgments

I would like to extend my heartfelt appreciation to all those who have contributed to this significant milestone in my academic journey.

I am indebted to Professor Kamal Youcef-Toumi for his unwavering support and guidance throughout this research project. Your expertise, insightful feedback, and continuous encouragement have been instrumental in shaping the trajectory of my work. I am fortunate to have had the privilege of learning under your mentorship.

I owe a debt of gratitude to my parents, family, and friends for their unwavering belief in me and their constant encouragement. Your love and support have been my foundation and motivation throughout my academic pursuits.

To my lab mates, Xiaotong Zhang, Tony Tohme, and others, your camaraderie, assistance, and ongoing support have been invaluable. The collaborative environment that we shared has enriched the quality of my work and made this journey more fulfilling.

I extend my gratitude to the King Abdulaziz City of Science and Technology (KACST) for their generous sponsorship of my studies. This support has been pivotal in affording me the opportunity to pursue higher education and delve into the realm of knowledge and discovery. The resources and opportunities provided by KACST have played a vital role in shaping my academic pursuits.

I am also thankful to the Mechanical Engineering Department at MIT for awarding me a fellowship during my first year of graduate school. Your investment in my education has alleviated financial constraints and demonstrated the department's commitment to fostering academic growth. This fellowship has enabled me to engage wholeheartedly in my studies and research from the outset of my journey.

Lastly, I would like to extend my appreciation to all those whose names may not appear here but have contributed to my growth and development in various ways. Your encouragement, advice, and presence have not gone unnoticed.

In conclusion, this thesis represents the culmination of a collective effort, and I am humbled by the immense support I have received. The lessons learned, skills acquired, and experiences gained will undoubtedly shape my path ahead. I am committed to utilizing the knowledge and opportunities provided to make a meaningful impact in the future.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction and overview

Motion prediction is an essential step for efficient and safe human-robot collaboration. Often times there are tasks that necessitate human expertise and robotic precision, requiring varying degrees of collaboration between the two. Humans often tend to anticipate each other's motion to avoid collisions while efficiently achieving short-term and long-term objectives in a collaborative setting. Thus, it is crucial to predict human motion for robots to navigate safely around humans while also making sure that the planned motion is efficient in space and time.

Among the numerous approaches to human motion prediction, some still result in improbable or impractical predictions, leading to lower prediction accuracy. Methods ranging from model-based methods that exploit the inherent dynamics of physical systems [1], to approaches like Inverse Optimal Control (IOC) [2], [3], which seek to emulate cost functions that rationalize observed movements, have been attempted to make accurate prediction. In more recent developments, data-driven strategies have emerged that try to decipher not only the underlying dynamics but also the intricate causal relationships with the environment, contingent on the task at hand. Yet, for the real-world application of these techniques, it is necessary to consider sensor noise, an inherent pain point when tracking human joints, especially for markerless methods. In doing so the propagation of this uncertainty into subsequent prediction stages becomes important. Regrettably, the current techniques that are

capable of quantifying this uncertainty [4], [5] often omit the critical aspect of using the kinematics of the physical system at hand and also fail to account for the context of the workspace within which it operates. The neglect of these factors leads to an unadjusted probability distribution function (pdf) for predictions. This underlines the pivotal role of accounting for physical systems, their intrinsic constraints, and workspace limitations when handling prediction uncertainty. These elements can significantly sway predictive accuracy. Disregarding them can result in models generating improbable or impractical predictions, despite their mathematical feasibility, thus underscoring the imperative to integrate these elements for a more practical motion prediction model.

Uncertainty propagation from pose estimation to prediction is essential for accurate motion prediction of a physical system. This uncertainty can also be propagated in the downstream layers of a robotics stack [6]. Modelling these physical systems makes them more interpretable as well as helps in considering task-specific parameters. Human joints are well-studied and it is important to make use of the kinematic constraints of the body as well as the physical constraints of the operating space for making accurate predictions. For accurate human motion prediction with sufficient representability, not only do we need to track action based on past motion of human joints [7], which inherently has uncertainty associated with it, we need to propagate this uncertainty through the prediction layer while making sure that the probability distribution function of the prediction satisfies kinematic and workspace constraints.

The contributions of this thesis are as follows. A framework for human motion prediction is proposed that takes measurement uncertainty into account to predict future motion with corresponding uncertainty while respecting the kinematic and physical constraints of the human body. Further, this uncertainty is propagated into the task space and the predicted probability distribution function is modified to respect task space constraints. An overview of the framework is shown in Fig. 1.1. This formulation not only helps in getting rid of implausible predictions that violate human joint configurations but also incorporates context into the uncertainty framework. This framework is evaluated on two different human motion datasets, demonstrating the benefit of considering such constraints on the output pdf. Lastly, we implement it on a human-robot collaborative setup, where we use human

motion predictions to make informed decisions by a UR5 robot arm.

1.2 Related works

1.2.1 Uncertainty based motion prediction

Current methods do not explicitly take human body joint characteristics, and contextual physical constraints into account, especially when these constraints operate in different representation spaces. Gaussian Process Dynamical Models (GPDM) [8] have been a traditional method for modelling human dynamics in order to predict human motion while also quantifying uncertainty. State-of-the-art deep learning architectures have also been recently modified for uncertainty quantification [5], or work like [9] attempts to purely quantify regression based uncertainty. Unfortunately none of them account for constraints in their framework to avoid predicting implausible configurations.

1.2.2 Constrained Gaussian Process Regression

For human motion prediction, we can enforce several different kinds of constraints like position, velocity, etc. constraints when operating in joint space and collision constraints with physical objects when operating in the task space. As for a GPR problem formulation, the constraints can be bound constraints, monotonicity and convexity constraints, as well as differential equation constraints [10]. In the realm of GPR, warping functions [11], as well as non-gaussian likelihood functions are among approaches that enforce bound constraints. When considering non-gaussian likelihood functions to enforce constraints, the posterior is often analytically intractable, and truncated Gaussian distributions have been considered instead to enforce discrete constraints [10]. We leverage truncated normal distribution along with rejection sampling for our framework.

1.2.3 Model/Kinematics for human motion prediction

It is important to leverage joint angle position, velocity, etc. constraints to get rid of implausible human motion predictions. Kinematics and dynamics modelling has traditionally been

an important component for understanding how humans move. The human body has been well studied and the nature of kinematic chains and physics-based methods have been leveraged in various models[1][12]. Specifically, for human motion prediction, there are different representations which can be used as a starting point for the prediction model, the simplest one being the x,y,z coordinates of each joint at a given time. In recent literature [13], human joints representation like 3D position, rotation matrix, angle-axis, or quaternions have been considered. For our formulation, it can be assumed that the motion is independent in the joint angle space, i.e. movement of one joint does not inherently affect movement of another, since it helps in dimensionality reduction while leaving room for imposing constraints. The joint angle space, as part of the kinematic model, is thus a feasible representation as a starting point of our prediction framework.

1.2.4 Context based human motion prediction

Human motion prediction is context driven and it is necessary to include scene context within the prediction framework. Context-based interactions have been attempted and addressed in [14][15]. Though these methods, via incorporating different neural network-based architectures like Graph Neural Networks [16], spatio-temporal transformers [13], etc., are expected to inherently account for the physical attributes of a scene and collision avoidance in prediction, they lack elements that explicitly enforce context-based constraints in the prediction and also the corresponding uncertainty that should be modified accordingly. We take care of physical scene-based constraints using a rejection sampling-based approach in our framework.

1.3 Objectives and Contributions

The contributions of this paper are as follows. A framework for human motion prediction is proposed that takes measurement uncertainty into account to predict future motion with corresponding uncertainty while respecting the kinematic and physical constraints of the human body. Further, this uncertainty is propagated into the task space and the predicted probability distribution function is modified to respect task space constraints. An overview

of the framework is shown in Fig. 1.1. This formulation not only helps in getting rid of implausible predictions that violate human joint configurations but also incorporates context into the uncertainty framework. This framework is evaluated on two different human motion datasets, demonstrating the benefit of considering such constraints on the output pdf. Lastly, we implement it on a human-robot collaborative setup, where we use human motion predictions to make informed decisions by a UR5 robot arm.

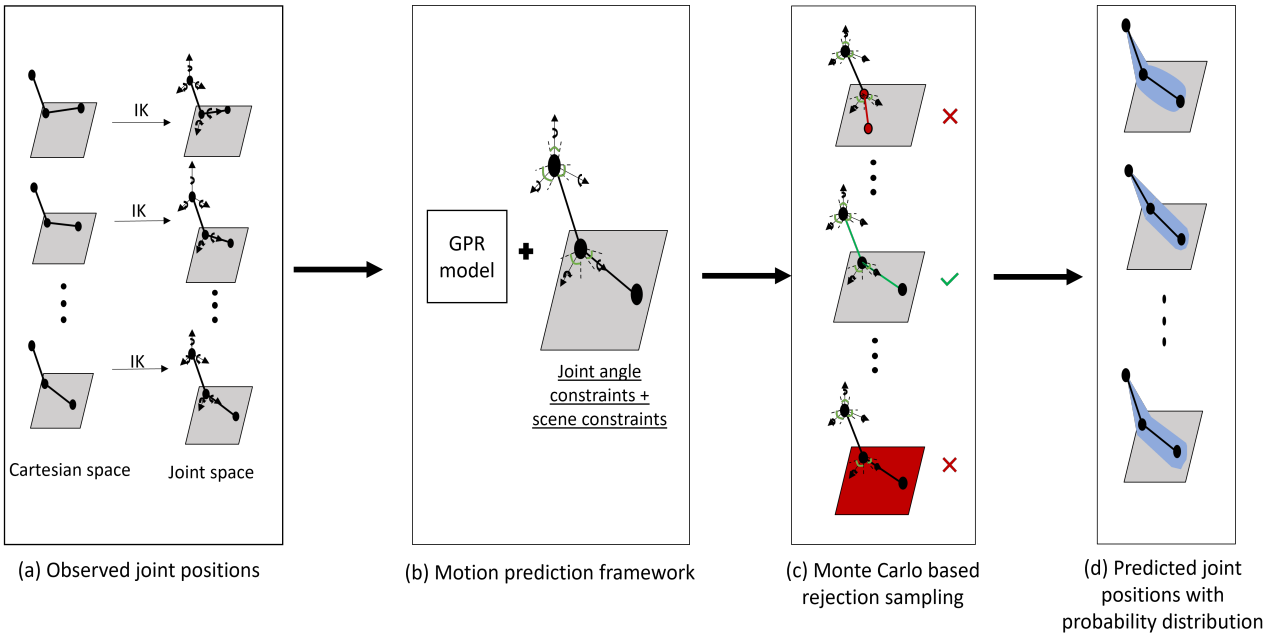

Figure 1.1: Overview of the proposed constrained probability distribution prediction for human motion. Inverse kinematics is used to get to joint space for each observed time step, passed through a GPR model with constraints which are imposed using rejection sampling. Red joints represent a violation of joint angle or velocity constraints while red plane denotes collision or intersection of the prediction with an object in the scene.

1.4 Thesis outline

The thesis is outlined as follows. In chapter 2 we describe the problem formulation of our proposed framework with details on the algorithm used. In chapter 3 we evaluate the framework on two different datasets and compare results on average errors as well as negative log-likelihood (nll). In chapter 4, we discuss the implementation of the framework in a human-robot collaborative setup. In the appendices, we dive deeper into normalizing proba-

bility distribution functions (pdf), implementation and algorithm details, as well as give an overview of a purely deep learning-based motion prediction framework.

Chapter 2

Problem Formulation

The goal of our approach is to make accurate human motion predictions while considering associated measurement uncertainty along with respecting kinematic and contextual constraints. We formulate the human motion prediction problem as an output of a GPR model in which we infuse the human kinematic model as well as physical workspace constraints. Similar to [13], we consider a sequence of observed motion as $\mathbf{S}(t) = \{\mathbf{s}_1, ..., \mathbf{s}_{T_O}\}$ where a frame $\mathbf{s}_t = \{\mathbf{j}_t^{(1)}, ..., \mathbf{j}_t^{(N)}\}$ denotes a pose at time step t with joints $\mathbf{j}_t^{(n)} \in R^q$, where q is the number of pose parameters, example: x,y,z, axis-angle representation, etc, and N is the number of joints tracked at every instant. These measurements are considered noisy and for an observed time horizon T_O. The goal of the prediction model is to generate an accurate joint probability distribution for the predicted $\{\hat{s}_{T_O+1}, ...\hat{s}_{T_O+T_P}\}$ where T_P is the prediction time horizon. We use a sliding window prediction framework, i.e. to predict $\hat{s}_{T_O+2}$, we use the sequence $\{\mathbf{s}_2, ..., \mathbf{s}_{T_O}, \hat{\mathbf{s}}_{T_O+1}\}$.

2.1 Inverse kinematics and joint angle space

We use inverse kinematics (IK) to transform the measured cartesian xyz space information to joint space. Given a pair of joint poses $(\mathbf{j}_t^{(n)}, \mathbf{j}_t^{(n+1)})$ in a kinematic chain, by defining the kinematic transformations and using forward kinematics, we can get the relation between the two joint poses as

$$\mathbf{j}_t^{(n+1)} = H_1 ... H_{p_n} \mathbf{j}_t^{(n)} = H \mathbf{j}_t^{(n)} \tag{2.1}$$

where each H_i denotes a Denavit-Hartenberg transformation matrix [17] which is a function of every θ in $\theta_t = \{\theta_{1,t}, \theta_{2,t}...\theta_{p,t}\} \in R^{p_n}$ for p_n number of joint angles that define the transformations from $\mathbf{j}_t^{(n)}$ to $\mathbf{j}_t^{(n+1)}$.

2.2 Constrained Gaussian Process Regression Models

Gaussian process regression models can be used to propagate uncertainty from one time step to the next. The human body is well-studied, and in the joint space, joint angles have linear constraints as well as higher-order derivative constraints that should be accounted for when making predictions on their motion[18]. To incorporate these into our prediction framework and account for the propagation of measurement noise across the model, for a given estimated $\hat{\theta} \in \theta_t$ we assume a truncated normal distribution prior to each joint angle defined as:

$$\hat{\theta} \sim \mathcal{TN}(\mu_\theta, \sigma_\theta^2, \theta_{lb}, \theta_{ub}) \tag{2.2}$$

where θ_{lb} and θ_{ub} are lower and upper bounds on θ and $\mu_\theta, \sigma_\theta^2$ are mean and variances obtained from the output of a Sparse Pseudo-input Gaussian Process (SPGP) model [19]. More accurate representation of θ can be estimated from methods like [20]. We define the SPGP as GP (Gaussian Process):

$$f_{\hat{\theta}} \sim GP(m, K) \tag{2.3}$$

where m and K are learned mean and covariance functions of the chosen representation space and dataset or set of observed trajectories, and θ_{lb} can be defined as:

$$\begin{aligned} \theta_{lb} &= \max(\theta_{lb}, \theta_{t-1} - \dot{\theta}_{ub}\Delta t) \\ \theta_{ub} &= \min(\theta_{ub}, \theta_{t-1} + \dot{\theta}_{ub}\Delta t) \end{aligned} \tag{2.4}$$

where $\dot{\theta}_{ub}$ is the upper bound on angular velocity and the latter part of the max function comes from a simple linear interpolation using maximum permissible joint velocity over a time step Δt. θ_{ub} can be defined in a similar way, and thus we get the bounds for our

truncated normal distribution.

2.3 Probability propagation and task space constraints

We intend to propagate the probability distributions from joint space back to task space since we are operating in task space R^3. We can use the Jacobian transformation to transform individual predicted probability distribution functions in joint angles to joint position probability distributions. Firstly, we can define the Jacobian as:

$$J = \begin{bmatrix} \frac{\partial X}{\partial \theta_1} & \frac{\partial X}{\partial \theta_2} & \cdots & \frac{\partial X}{\partial \theta_{p_n}} \\ \frac{\partial Y}{\partial \theta_1} & \frac{\partial Y}{\partial \theta_2} & \cdots & \frac{\partial Y}{\partial \theta_{p_n}} \\ \frac{\partial Z}{\partial \theta_1} & \frac{\partial Z}{\partial \theta_2} & \cdots & \frac{\partial Z}{\partial \theta_{p_n}} \end{bmatrix} \tag{2.5}$$

where $J \in R^{3\times p_n}$. For a given time t, We define $f_{X_jY_jZ_j}$ as the joint probability distribution function in the workspace:

$$f_{X_jY_jZ_j} = \frac{f_{\theta_1} \cdots f_{\theta_{p_n}}}{|J|} \tag{2.6}$$

since our underlying assumption is that joint angle motions are independent with respect to each other. We note that for certain kinematic chain configurations, the Jacobian J might not be a square matrix in which case, special methods exist to handle such issue like pseudo determinants, singular value decomposition, etc. We can augment J by introducing dummy variables and then integrating. Since calculating this integral may not always be feasible, Monte Carlo (MC) based samples from Eq. (2.2) and then the integrals are calculated using Reimann sums. Since the workspace itself can have physical constraints due to the possible existence of objects, etc. in the scene, it is important to modify $f_{X_jY_jZ_j}$ to account for inequality constraints:

$$\begin{aligned} x_{t,\min} \leq x_t \leq x_{t,\max} \\ y_{t,\min} \leq y_t \leq y_{t,\max} \\ z_{t,\min} \leq z_t \leq z_{t,\max} \end{aligned} \tag{2.7}$$

in the workspace coordinate frame where the minimum and maximum values for the constraints can be defined using static and dynamic obstacles in a workspace for a given prediction time t. The variables x_t, y_t, z_t can be obtained from cartesian-like representation from $j_t^{(n+1)}$ using the MC samples and rejecting them when Eq. (2.7) is not satisfied. A block diagram of this MC based rejection sampling approach leading to a joint pdf of the prediction is shown in Fig. 2.1. We normalize the resulting modified pdf using the sum of valid pdf values.

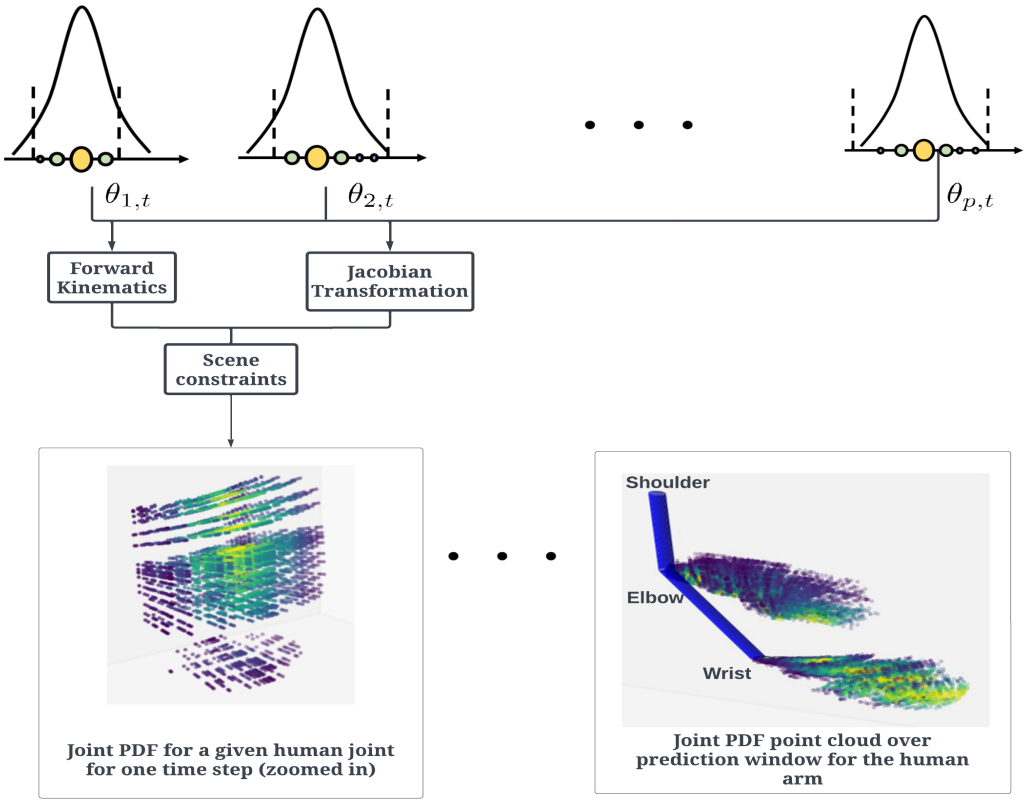

Figure 2.1: Constrained satisfaction-based Monte Carlo rejection sampling.

We note that our constrained human motion prediction, as highlighted in **Algorithm 1**, can be applied to any human motion prediction model that operates in joint space and can provide unconstrained uncertainty quantification of the predicted joint angle trajectories. Depending on the requirements of an application, and the level the sophistication desired on the modelling side, this model can be derived from Gaussian Processes, neural network-based models like spatio-temporal transformers [13], diffusion methods [4], etc. with an

added output of uncertainty quantification.

Algorithm 1 Constrained human motion prediction using Gaussian Process Regression for a pair of joints for one prediction time horizon

Offline:
▷ Tune hyperparameters of SPGP based on existing observed trajectories or datasets.
Online:
Input: $S(t)$ ▷ Observed motion sequence
$\theta_t = IK(j_t^{(n)}, j_t^{(n+1)})$ ▷ Obtain joint angles using inverse kinematics
for t = T_O to T_P **do**
  for i = 1 to p_n **do**
    $u_{\hat{\theta}_{t+1}^{(i)}}, \sigma_{\hat{\theta}_{t+1}^{(i)}} \leftarrow \mathcal{GP}(\theta_t^{(i)}, \theta_{t-1}^{(i)}, \theta_{t-2}^{(i)}...)$ ▷ GP one step sliding window prediction of mean, and variance
    $\hat{\theta}_{t+1}^{(i)} \sim \mathcal{TN}(u_{\hat{\theta}_{t+1}^{(i)}}, \sigma^2_{\hat{\theta}_{t+1}^{(i)}}, \hat{\theta}_{t+1,\min}^{(i)}, \hat{\theta}_{t+1,\max}^{(i)})$
  end for
  $\hat{x}_{t+1}, \hat{y}_{t+1}, \hat{z}_{t+1} = FK(\text{Samples from } \hat{\boldsymbol{\theta}}_{t+1})$ ▷ Use forward kinematics to transform back to task space
  $f_{XYZ} = \prod_{i=1}^{p_n} \hat{\theta}_{i,t+1} \frac{1}{|J|}$ ▷ Use jacobian method to obtain corresponding pdf in task space
  Reject $\hat{x}_{t+1}, \hat{y}_{t+1}, \hat{z}_{t+1}$ samples that violate constraints and corresponding f_{XYZ}.
  Normalize constrained f_{XYZ}
end for

Chapter 3

Implementation and evaluation

3.1 Dataset

To evaluate our approach, we use two relevant datasets that address our requirements: (i) the ANDY dataset[21] with industry like manual activities, (ii) the HA4M [22] assembly task dataset. These were selected since it is important to consider datasets in which the human interacts with the surroundings, especially ones that involve the execution of some task or activity. The ANDY dataset captures the skeleton data using inertial and optical motion capture sensors recorded at 240Hz and 120Hz respectively. The actions performed are labelled which thus allows the prediction framework to be tested on individual actions and consider workspace constraints like tables, shelves, etc objects in the scene. On the other hand, the HA4M dataset captures the skeleton data of different subjects performing an assembly task using the Microsoft®Azure Kinect Camera, where the skeleton joint poses are tracked using the Azure Kinect Body Tracking SDK. The assembly task consists of different actions to build an Epicyclic Gear Train and involves physical constraints to the workspace, example - table where the parts are placed, workspace constraints, etc.

3.2 Implementation

The performance of our framework is demonstrated by using a Gaussian Process Regression (GPR) model for the unconstrained prediction stage as it helps in quantifying the prediction

uncertainty. It assumes Gaussian distribution on the noise in the prior i.e. the observed joint angle trajectory.

We base our comparison on three methods of varying levels of representation and constraints:

3.3 Unconstrained xyz (GP xyz)

the hyperparameters of the GPR are tuned on data in which the input representation is past observed xyz joint positions, and the output is the predicted position along with the underlying uncertainty.

3.4 Unconstrained joint angles (GP J.A.)

the GPR is tuned with inputs in the joint angle space i.e. by applying inverse kinematics (IK) on observed joint positions to obtain joint angles, followed by GPR on the joint angles to obtain predicted joint angles, and propagating these back to xyz space using forward kinematics (FK).

3.5 Constrained prediction (GP Constr.)

similar to the unconstrained joint angle formulation but now we constrain the joint angle predictions as well as the forward kinematics output xyz values for collision avoidance or intersection with physical objects in the scene.

For the SPGP, we use a radial basis function (RBF) kernel and use an observed window of 200 milliseconds and a prediction window of 500 milliseconds. We consider trajectories from joints of both hands i.e. shoulder, elbow, and wrist. A simple kinematics model derived from [18] and PySwarms solver [23] for the inverse kinematics is used to get the joint space, and the joint angle and joint velocity constraints as used as defined in [18].

3.6 Simulations

HA4M [22]: For this particular dataset, we consider left and right arm (shoulder, elbow, and wrist) trajectories from 5 different subjects for training the SPGP and 7 other subjects for evaluating the constraints framework. The frame rate is 30Hz and a step size of 5 frames between every sliding prediction is used and both ends of the trajectories are trimmed to remove extended amounts of stationary behavior. For the constraints, the major one we consider is the table where parts are being picked and placed. In most frames, this was a simple linear constraint in the world z direction of the provided coordinate system. As for velocity constraints, we observe that the wrist motions do not exceed $1.5m/s$ and thus incorporate that as a constraint after transforming it in the joint space.

ANDY [21]: This dataset has an additional feature of labelled actions, and thus we evaluate our method only on specific actions - reaching, picking, carrying, and placing. These are the most relevant actions to the application of constrained based prediction. We again use arm trajectories and use a step size of 20 frames instead since the frame rate is 120Hz.

3.7 Evaluation Metrics

We use two metrics to evaluate our work. The first one is the mean per joint position error (MPJPE) on cartesian joint positions

$$\mathcal{L}_{\text{MPJPE}} = \frac{1}{N \cdot T_P} \sum_{n=1}^{N} \sum_{t=1}^{T_P} \left\| \mathbf{j}_t^{(n)} - \hat{\mathbf{j}}_t^{(n)} \right\|_2 \tag{3.1}$$

as popularly used in other works [16]. We use this metric specifically on hand motions i.e. shoulder, elbow and wrist trajectories in order to evaluate interactions with scene constraints.

The quantified uncertainty is evaluated using the negative log-likelihood (NLL) metric on the ground truth values $\mathbf{s}_t$ with respect to the constrained pdf:

$$\text{NLL} = \frac{1}{T_P} \sum_{t=1}^{T_P} -log\big(f_{XYZ}(\mathbf{s}_t)\big) \tag{3.2}$$

where the joint pdf f_{XYZ} is derived from Eq. (2.6).

3.8 Results

As used in previous methods [7], each GPR model was trained according to a given classified action in the ANDY dataset for each joint trajectory prediction, while on a subset of trajectories of 5 different people from HA4M. Using MPJPE from Eq. (3.1) on the expectation of the Monte Carlo (MC) based rejection sampled points for each method, we specifically evaluate our method for shoulder, elbow, and wrist trajectories of both arms for selected actions, and results are shown in Table 3.1

Table 3.1: MPJPE (in mm) using joints - shoulder, elbow, and wrist of both hands. The prediction window is 500ms. Our framework consistently outperforms unconstrained GPs in different representations.

Dataset	Action	GP xyz	GP J.A.	GP Constr.
ANDY	Reaching	223 ± 14	87 ± 5	**76 ± 5**
	Picking	268 ± 18	123 ± 7	**89 ± 4**
	Placing	255 ± 13	103 ± 8	**73 ± 5**
HA4M	-	173 ± 13	104 ± 7	**85 ± 9**

It is evident that moving from joint position space to joint angle space followed by adding constraints results in a better mpjpe error. We see a considerable reduction in error when changing representation space which can be attributed to both the incorporation of a kinematics model and enforcing constant bone lengths via the same. The next shift is from unconstrained joint space to constrained joint space. This shift is especially noticeable for the actions we tested because of how often we reach the edges of some joint constraints when performing them.

To evaluate the estimated joint pdf, we use the NLL metric from Eq. (3.2). We run our evaluation for 25 iterations of every method and plot the corresponding histograms on a log scale with the standard error on top of them as shown in Fig. 3.1.

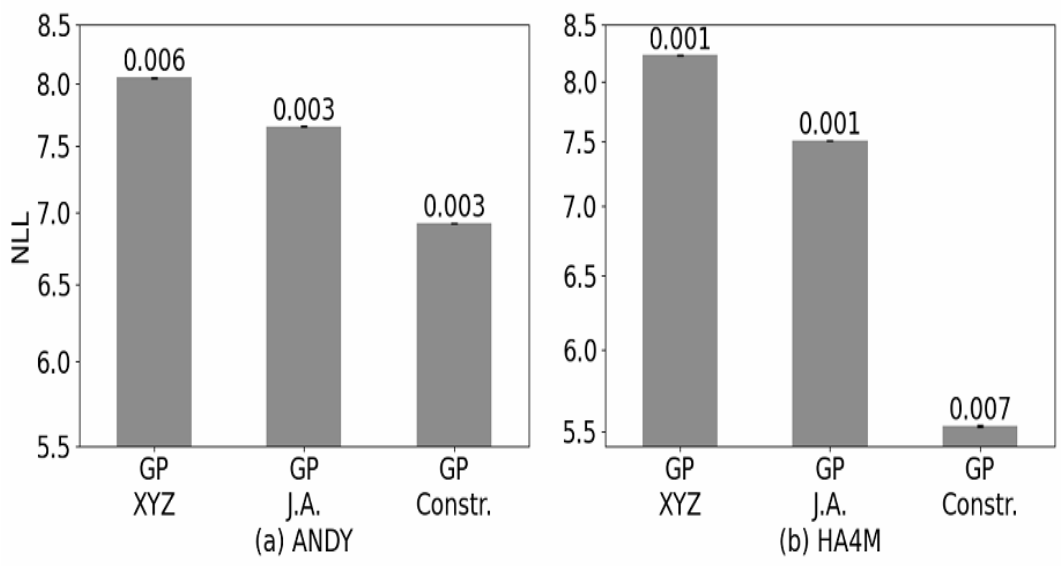

Figure 3.1: Comparing NLL of ground truth prediction with pdf for ANDY (left) and HA4M (right) for GP xyz, GP joint angles (J.A.) and GP constrained (constr.). Standard errors are on top of the bar plots.

It is again evident that changing representation space and adding constraints perform much better (15% and 32% from Fig. 3.1 ANDY and HA4M respectively) as the NLL is lowered, signifying a better probability value for the evaluated ground truth, thus meaning that rejecting points that violated constraints resulted in a considerable improved prediction pdf.

Chapter 4

Experiments

4.1 Experimental platform

Fig. 4.1 illustrates the experimental setup and a human-robot collaboration task scenario in which a UR5 robot arm is tasked with picking and placing a red, green, and blue cubes in order.

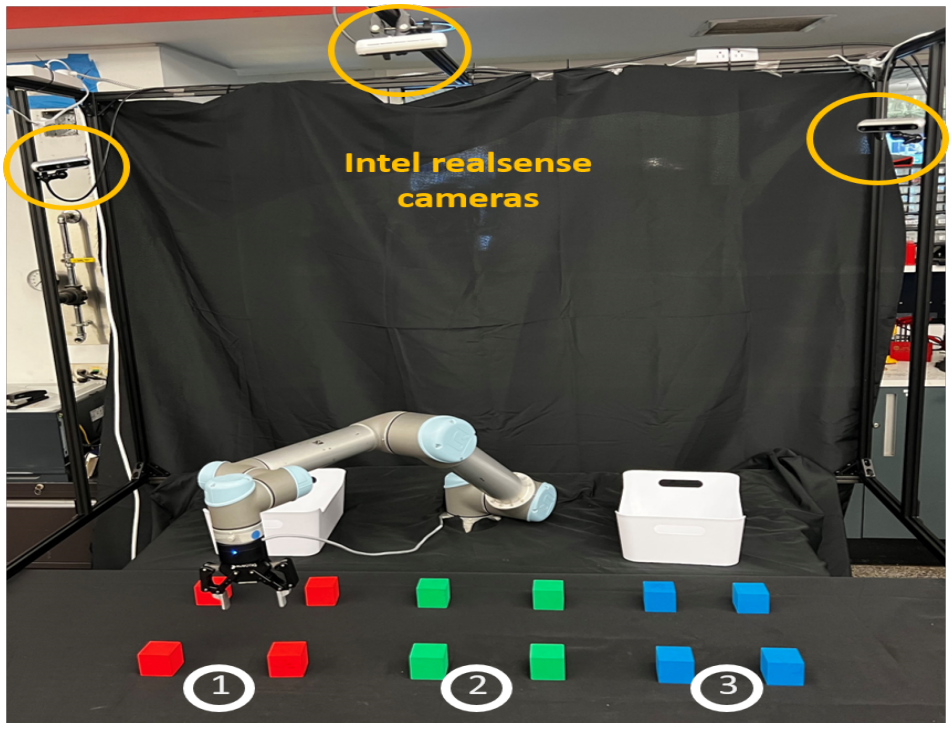

Figure 4.1: Experimental setup for human-robot collaboration.

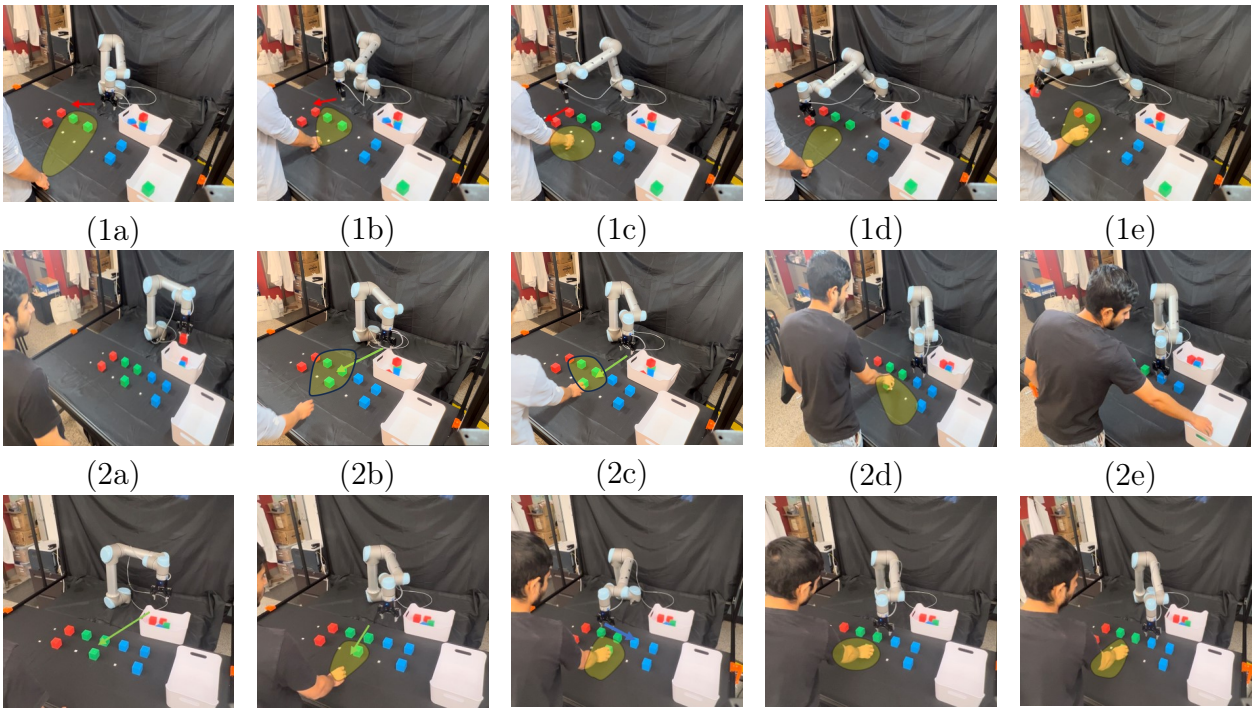

Figure 4.2: Experimental result. Each row represents a different demonstration where the prediction framework is leveraged for safe and efficient collaboration. The arrows and the corresponding color represent the robot planned motion direction as well as the object to grasp. The yellow region represents the predicted human wrist region. Both arrows and predicted regions are only for qualitative purposes. Row 1 (1a-e) avoiding predicted region - the robot is scheduled to grasp a red cube and the human is reaching out for a green cube; the robot safely navigates around the predicted region while executing the task at hand. Row 2 (2a-e) task re-planning - the robot is scheduled to grasp a green cube but the human predicted region indicates that the human is planning to grasp the same cube; the task is re-planned and the robot plans and grasps a blue block instead. Row 3 (3a-e) task and trajectory re-planning using prediction - similar to 2, the robot is scheduled to grasp a green cube and is executing trajectory for the same. As soon as a human is predicted to be in the green cubes space, the motion is re-planned and the robot grasps a blue cube instead.

As soon as a human enters the scene, image frames from the Intel RealSense cameras are passed through mmpose [24] to track human joint positions which are fused together and passed to both prediction and planning modules. Our constrained motion prediction framework is used to accurately predict the most probable regions where the human might be in the next 500ms. Depending on the predicted region, the grasp point or order is modified for successful and efficient completion of the task while safely navigating around the human. We use moveit [25], ROS Noetic and CHOMP [26] motion planner as the primary motion planner.

4.2 Demonstration

To demonstrate the real-time application of the proposed motion prediction framework, three different scenarios are used, as shown in Fig. 4.2. The associated video demonstrations can be found at youtube.com/@MITMechatronics/videos. The overall premise is to pick and place red, green, and blue cubes (in order), one at a time. Using the tracked human joint motions and the prediction framework, certain regions in space become unsafe and the robot has to avoid them, and also in the grasp order, if certain cubes become unsafe, then the grasp order is modified accordingly, under the assumption that the human might manipulate one or more of those cubes. While we only demonstrate one human in the scene, our framework can be successfully applied to multiple humans since for the application, only the predicted occupied region in space is relevant.

4.3 Conclusion

We introduce a novel constrained probability distribution prediction (CPDP) framework for human motion prediction that explicitly accounts for kinematic as well as scene constraints in order to predict more accurate probability distribution functions for a predicted motion trajectory. Our proposed framework is able to reason about the capabilities of the physical system at hand as well as account for any implausible predictions made when predicting the final trajectory pdf. We evaluate our framework on two task-relevant human motion datasets

and observe considerable improvements. We also implement it in a real-time human-robot collaboration application using a UR5 robot.

In future work, we intend to make use of CPDP on other existing prediction frameworks like [16] to demonstrate the added benefit of our framework to any human motion prediction module for a real-life application.

Appendix A

Normalization of PDFs

Given a finite set S of points, where each point p_i has an associated probability $P(p_i)$, the entire set traditionally conforms to the axiom that

$$\sum_{i=1}^{n} P(p_i) = 1,$$

signifying that the total probability across all points in the set is 1. If a subset S' of these points is removed, the remaining set, denoted as S'', does not maintain the aforementioned property. In order to normalize the probabilities for S'', one must adjust each associated probability to ensure the set once again adheres to the total probability axiom. This is accomplished by first computing the summation of the current probabilities within S'', denoted as

$$\sum_{i=1}^{m} P(p_i),$$

where m is the number of points in S''. Each individual probability $P(p_i)$ in S'' is then divided by this summation. The result is a new set of probabilities for S'' where the total is 1, thus providing a normalized probability distribution over the subset of remaining points.

Appendix B

Additional implementation details

Packages used for the demonstration:

1. Human pose detection: mmpose [24]. Checkpoint: deeppose_res50_coco_256x192_rle Config: td-reg_res50_rle-8xb64-210e_coco-256x192
2. UR5 motion planning: Moveit
3. ROS version: ROS noetic
4. Motion prediction python libraries: PyTorch, GPyTorch, Scikit, Numpy, PySwarm, Pandas, Pickle
5. Cameras: Intel realsense D455, 3 cameras positioned as shown in Fig. 4.1
6. Compute specifications: Graphics card: NVIDIA Titan TU 102, Operating system: Ubuntu 20.04

Fig. B.1 represents a flowchart of how different components of the ROS architecture are interconnected for a successful human robot collaboration demonstration. The /mainp_poses node is responsible for tracking the poses of objects in the scene that need to be picked and placed. 1, 2, and 3 nodes directly interface with the realsense cameras to to get rgb and depth images, and mmpose is integrated with them in order to output 2d keypoints with depth information from each node on to /human_pose1, /human_pose1, and /human_pose3 topics which are used by /human_pose_combiner to combine the 3 different pose readings of joints

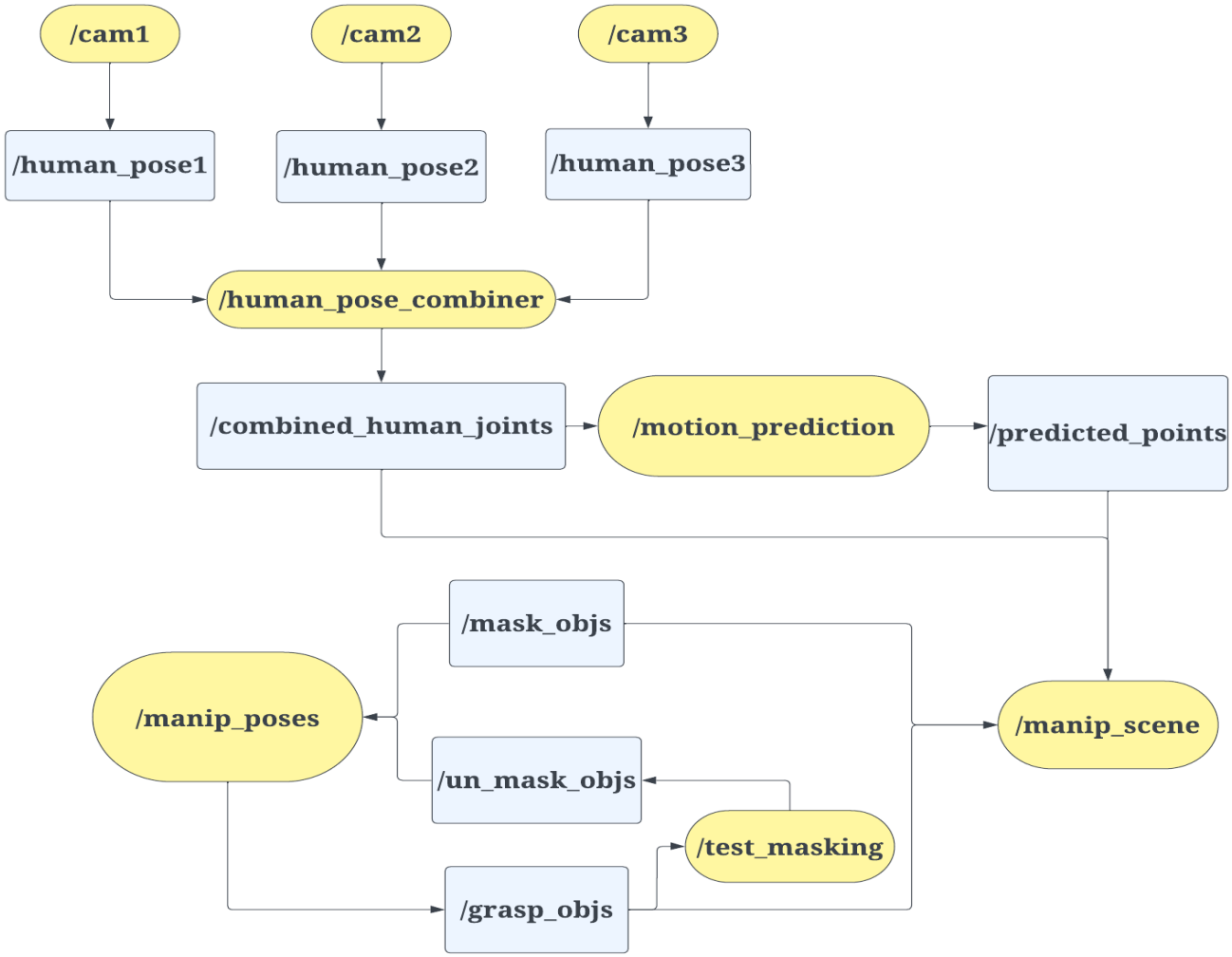

Figure B.1: ROS Topics and Nodes flowchart. Yellow blocks represent ROS nodes whereas blue blocks represent ROS topics.

into a single /combined_human_joints topic. This serves as a direct input to our prediction layer which keeps track of a window of human_joints information being published and uses the same for predicting a probabilistic point cloud of predicted motion for selected joints. The /manip_scene node is responsible for taking in human poses, predicted point cloud, as well as objects in the scene and publish this information in the form of a move_group scene as part of moveit. This helps in adding characteristics like collision, graspability, etc for the UR5 robot arm motion planning.

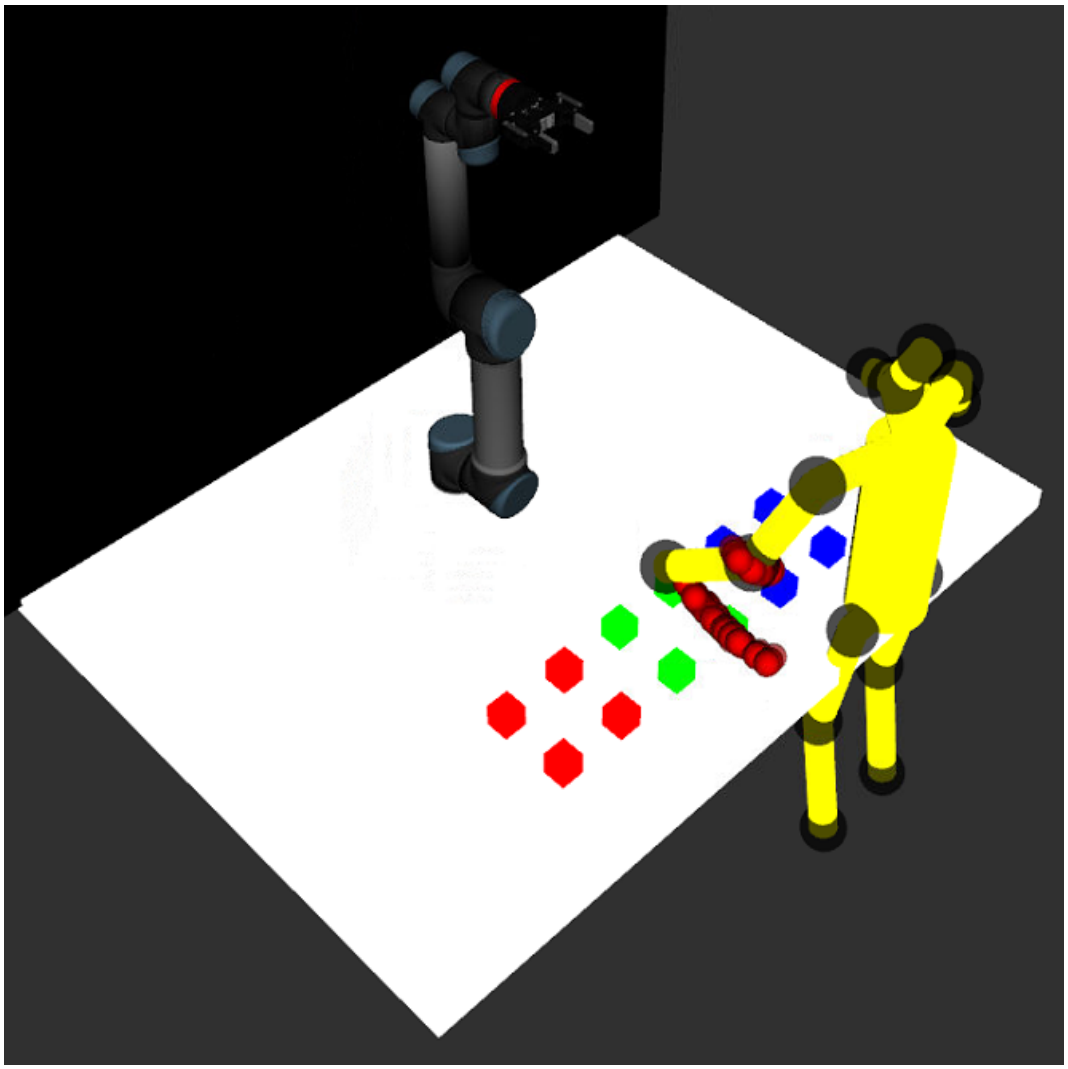

Figure B.2: Rviz visualization of the demonstration. Human is represented in yellow, with prediction being performed on left wrist and shoulder in the form of red point cloud. Objects to be picked and placed in order as displayed in red, green and blue colors, and the UR5 robot is shown in the scene. Two safety planes are added in the form of black and white colors shown.

For demonstration as well as our testing, we perform predictions considering the shoulder as the base joint. This can be altered depending on the level of sophistication needed as well as representation changes from the pose estimation layer.

In the demonstration, the goal of the robot is to grasp red, green, and blue blocks in order and repeat the same until all boxes have been picked and placed. Visual representation is shown in Fig. B.2 When a human enters the scene, the robot is tasked with making safe motions around the human as well as avoiding picking blocks that might be too close to the human and unsafe to pick. This is achieved by adding the prediction region as a collision object for the robot motion planner when planning motions. As for deciding grasp order, we can use a safe region distance metric to see if the scheduled block is too close to the predicted region and thus the robot can skip that block and move on to the next one.

Appendix C

Algorithm 1 details

Detailed version of Algorithm. 1. Constrained human motion prediction using Gaussian Process Regression for a pair of joints for one prediction time horizon.

1: **Offline**:

2: ▷ Tune hyperparameters of SPGP [19] based on existing observed trajectories or datasets. We used GPytorch [27], and combined a radial basis function kernel with an inducing point kernel to make the GP sparse. A complete set of observed trajectory is used for setting the inducing points and the remaining subset of training trajectories are used to tune the hyperparameters. We use an observed window corresponding to 0.167 seconds of motion to make a one-step prediction using the GP and keep sliding on the same. A different GP is tuned for each joint angle corresponding to each pair of joints of the body.

3: **Online:**

4: Input: $S(t)$ ▷ Observed motion sequence in task space i.e. x,y,z positions of each joint angle from t=0 to T_O

5: $\theta_t = IK(j_t^{(n)}, j_t^{(n+1)})$ ▷ Obtain joint angles using inverse kinematics. At a given time step, the past 0.167 seconds ($\frac{1}{6}^{th}$ of a second) is used as the initial input for a one-step prediction of each GP.

Detailed version of Algorithm. 1. (Continued)

for t = T_O to T_P **do**

  for i = 1 to p_n **do**

    $u_{\hat{\theta}^{(i)}_{t+1}}, \sigma_{\hat{\theta}^{(i)}_{t+1}} \leftarrow \mathcal{GP}(\theta^{(i)}_t, \theta^{(i)}_{t-1}, \theta^{(i)}_{t-2}...)$ ▷ GP one step sliding window prediction of mean, and variance of a given joint angle.

    $\hat{\theta}^{(i)}_{t+1} \sim \mathcal{TN}(u_{\hat{\theta}^{(i)}_{t+1}}, \sigma^2_{\hat{\theta}^{(i)}_{t+1}}, \hat{\theta}^{(i)}_{t+1,\min}, \hat{\theta}^{(i)}_{t+1,\max})$ ▷ Add min and max constraints on the joint angle for that time step to form a truncated normal distribution and thus account for constraints.

  end for

  $\hat{x}_{t+1}, \hat{y}_{t+1}, \hat{z}_{t+1} = FK(\text{Samples from } \hat{\boldsymbol{\theta}}_{t+1})$ ▷ Given a vector of normally distributed joint angles, sample and form a mesh of these to pass them through forward kinematics to obtain corresponding x,y,z values.

  $f_{XYZ} = \prod_{i=1}^{p_n} \hat{\theta}_{i,t+1} \frac{1}{|J|}$ ▷ When sampling, we make a note of probability of each joint angle, and thus the corresponding probvility values from the mesh are propagated using the jacobian method to obtain corresponding pdf in task space.

  Since we only want feasible x,y,z in task space, reject $\hat{x}_{t+1}, \hat{y}_{t+1}, \hat{z}_{t+1}$ samples that violate constraints and corresponding f_{XYZ}.

  Normalize constrained f_{XYZ} from the method described in the appendix.

end for

Appendix D

Video-based motion prediction

Deep learning has shown to be a powerful tool for predicting human motion and in this appendix, we will explore a step-by-step approach towards predicting future human poses in real-time using a single camera stream using purely deep learning-based methods. We will discuss some of the challenges in making this pipeline work in real-time as well as some of the shortcomings of the deep learning methods implemented.

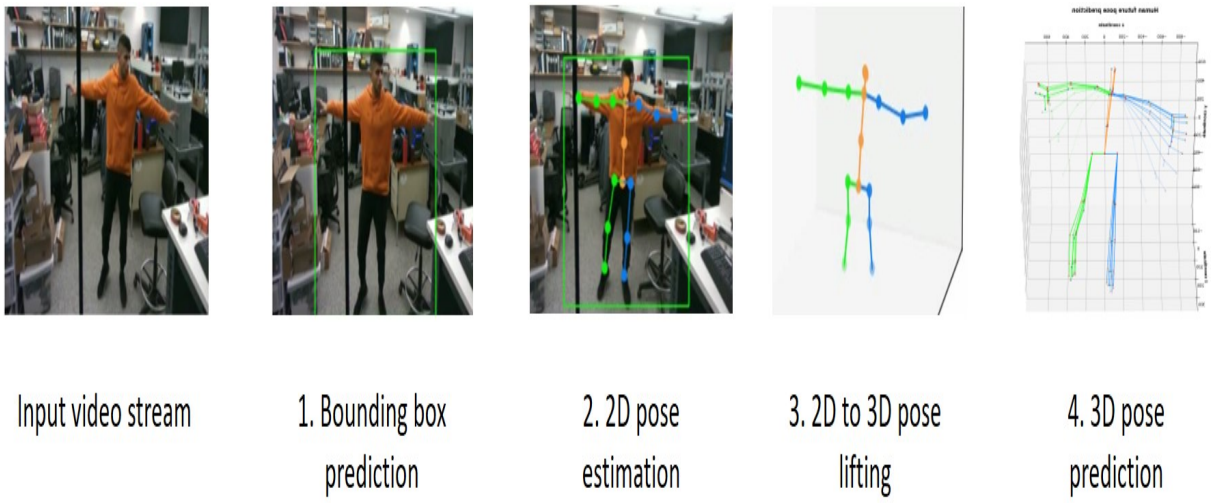

Figure D.1: Video-based human motion prediction pipeline. The first image on the left shows input video streams from the camera. The following images show the process in steps.

D.1 Human detection - DEtection TRansformer (DETR)

Figure D.2: Human bounding box detection an image.

In the human detection layer of our pipeline, our goal is to generate bounding boxes around detected humans in a given frame. Due to the real-time nature of our application, equal consideration is given to inference time and mean average precision (mAP)

Table D.1: Inference FPS and average precision errors

Model	Inference FPS	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster R-CNN+FPN	26	42.0	62.1	45.5	**26.6**	45.4	53.4
Deformable DETR	19	**43.8**	**62.6**	**47.7**	26.4	**47.1**	58.0
DETR	28	42.0	62.4	44.2	20.5	45.8	**61.1**

Based on the data from Table. D.1, the models have comparable Mean Average Precision (mAP) but since DETR [28] outperforms on inference speed, we choose the same for our implementation.

As shown in Fig. D.3, the main architecture of DETR can be broken down into a pipeline of three main components:

1. CNN Backbone: The conventional CNN backbone is used to learn a 2D representation of an input image.

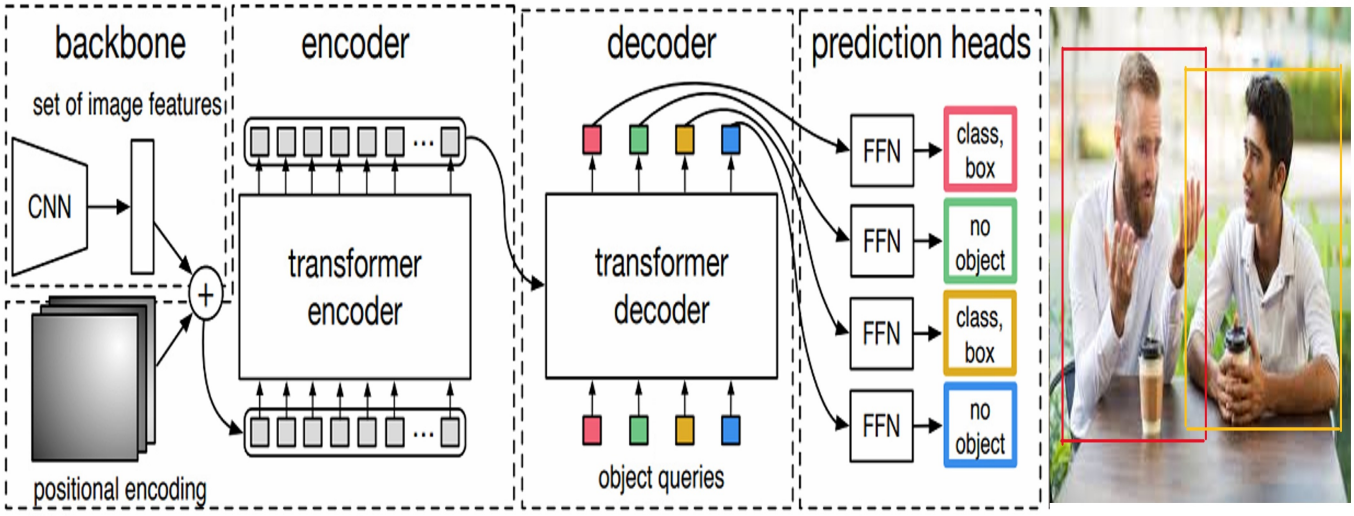

Figure D.3: The encoder-decoder architecture for generating a bounding box around humans.

2. Encoder-decoder transformer: The flattened representation from the CNN is combined with positional encoding and fed into a transformer encoder. The transformer decoder then takes a fixed number of learned encodings while attending to the encoder output.

3. Feed forward network (FNN): The output of the decoder is passed to a shared feed-forward network (FFN) that either predicts a detection or "no object" class.

D.2 2D pose detection - Video Pose Estimation via Neural Architectural Search (ViPNAS)

Human pose estimation is the process of detecting the pose of a person in a given image. For a given skeletal model, the goal is to detect key points (joints) of a human in a given image frame, followed by joining respective key points to generate a skeletal representation of the person. Again, we equally consider inference speed and accuracy trade-off to select the best model for 2D pose detection.

For fast online video pose estimation, while achieving a better trade-off between accuracy and efficiency, we select ViPNAS [29] for online pose estimation due to efficient pose estimation. The key to this architecture's efficiency is the allocation of different computational resources to different frames.

Figure D.4: Human pose estimation on a detected bounding box of a human.

Table D.2: Inference fps and average precision errors for different human pose estimation models.

Model	Inference FPS	AP
Resnet	29	0.72
ShuffleNet	**63**	0.6
HRNet	22	**0.75**
ViPNAS + MobileNet	54	0.7

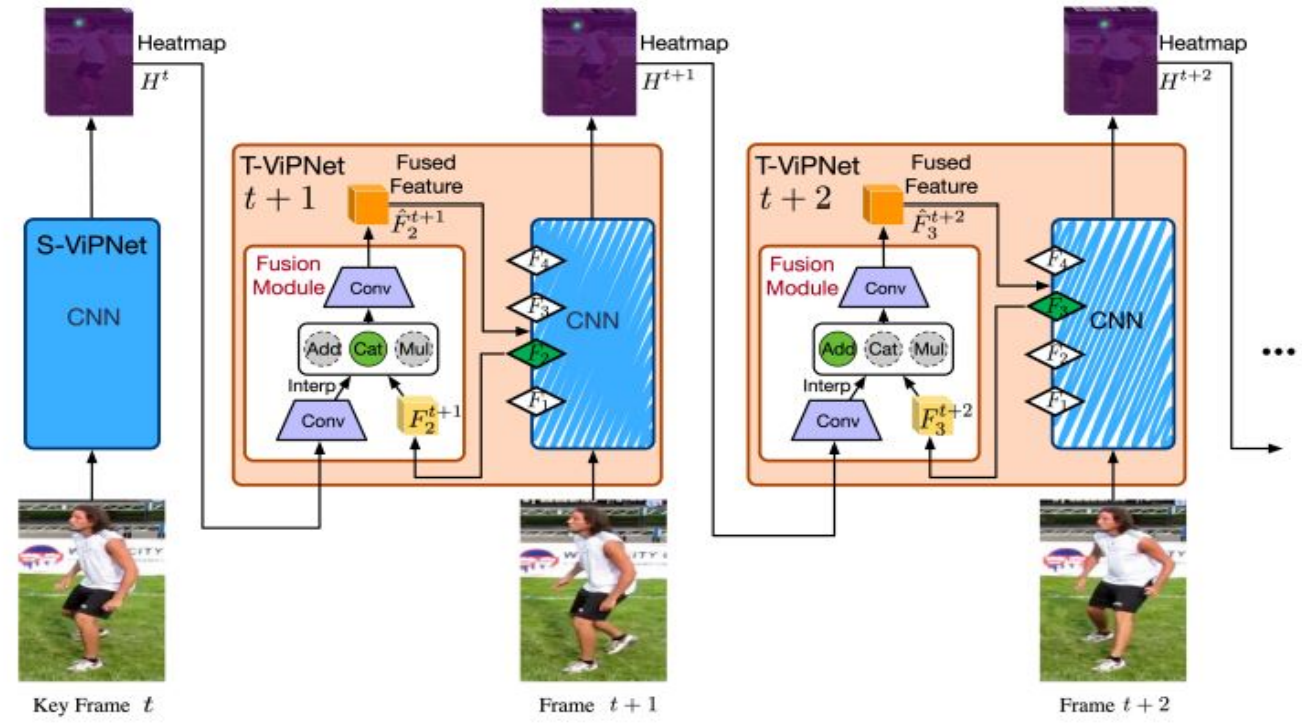

Figure D.5: Vipnas + MobileNet architecture for human pose estimation

As shown in Fig. D.5, the architecture can be decomposed into the following steps: 1. Select a set of T + 1 frames in a video 2. Out of the selected frames, select 1st frame as the key frame 1. Apply spatial video pose estimation network (S-ViPNet) on the same to localize human poses. 2. Use heatmaps to encode joint locations as Gaussian peaks. 3. For the non-key frames using temporal video pose estimation network (T-ViPNets): 1. Some CNN layers are used for feature extraction. 2. Fuse features of the current frame with heatmaps of the last frame 3. Pass fused features through remaining CNN layers to obtain heatmaps. The main analogy here is that poses in adjacent video frames are temporally correlated and thus lightweight models like T-VipNets can reasonably estimate joint locations with guidance from previous frames.

D.3 2D to 3D pose lifting - VideoPose3D

The goal of this layer is to infer the 3D poses given the 2D pose estimates from ViPNAS. VideoPose3D [30] is a fully convolutional architecture with residual connections and temporal

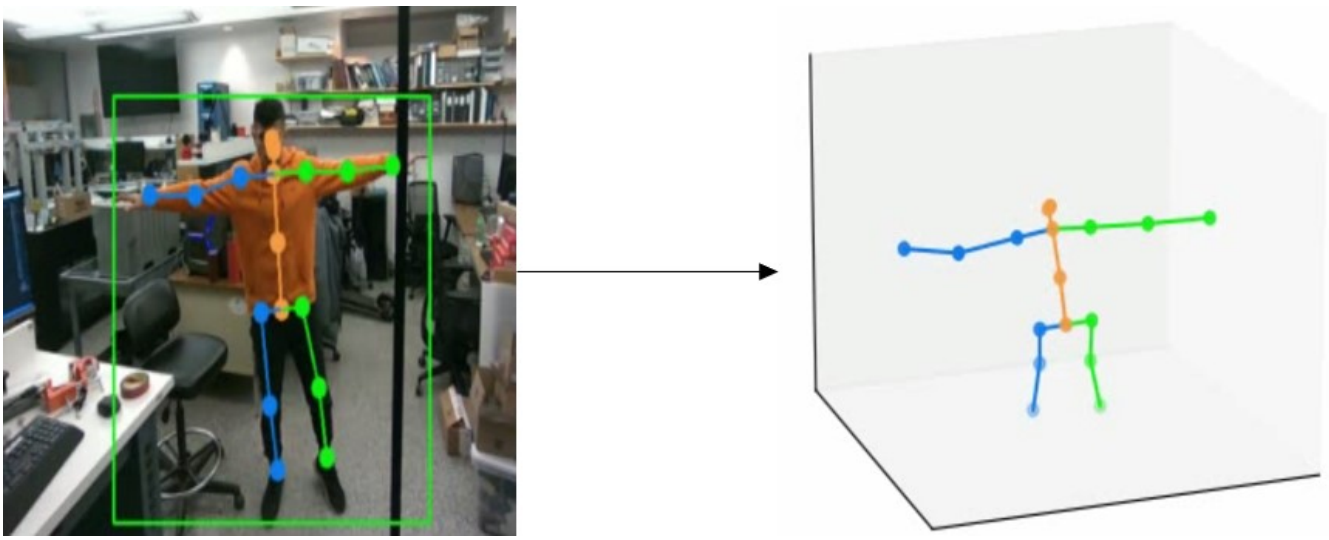

Figure D.6: Estimating 3D keypoints from 2D pose estimates.

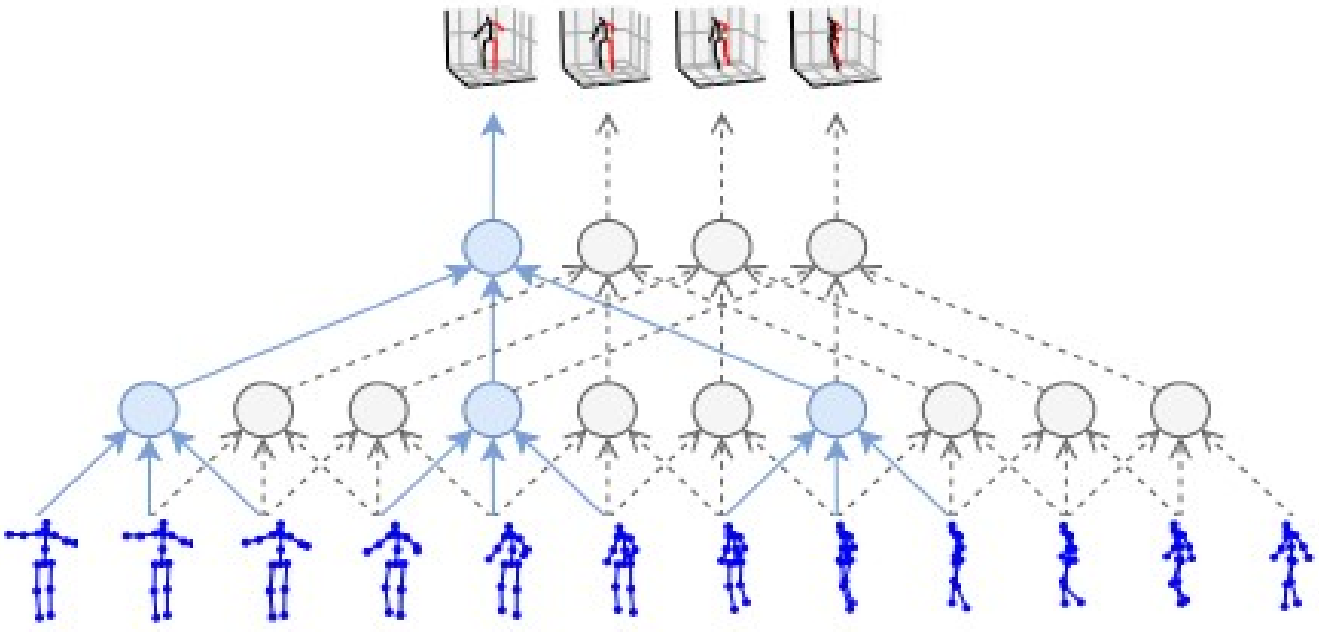

Figure D.7: VideoPose3D [30] temporal convolutions and residual connections.

convolutions as shown in Fig. D.7.

The reason for picking temporal convolutions over RNNs is:

1. Convolutional architecture offers precise control over temporal receptive fields, which the authors of the paper found important for 3D pose estimation.

2. Convolutional models enable parallelization over batch and time dimensions and also do not suffer from vanishing and exploding gradients.

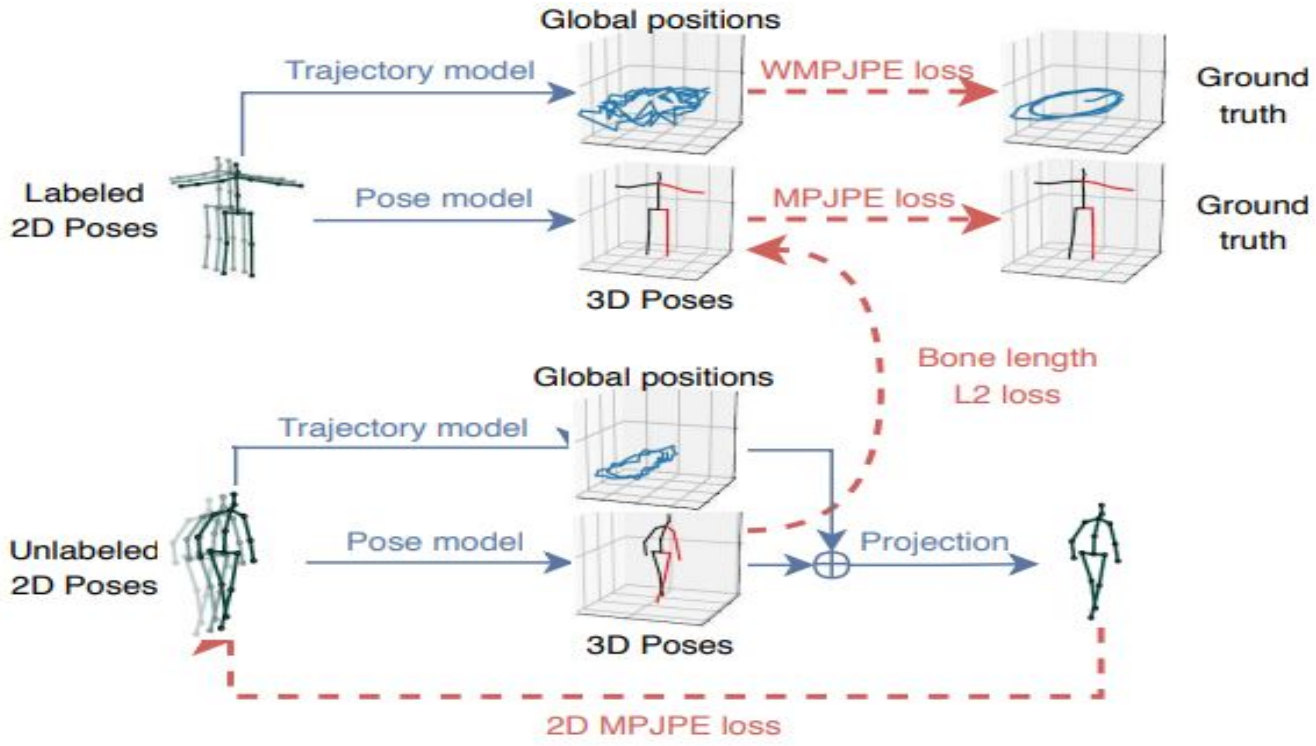

Figure D.8: Semi-supervised architecture for VideoPose3D [30].

To improve settings where labelled 3D ground-truth pose data is limited, a semi-supervised learning method is used as shown in Fig. D.8.

Essentially an unlabeled video with off the shelf 2D keypoint detector is used for a supervised loss function with back propogation loss term, and then the problem is setup as an auto encoding problem:

1. Encoder: 2D joint coordinates -> 3D pose estimation

2. Decoder 3D pose -> 2D joint coordinates

Lastly, since we require global position as well, the 3D trajectory of the person is regressed by a very similar model compared to the pose model.

D.4 3D pose prediction

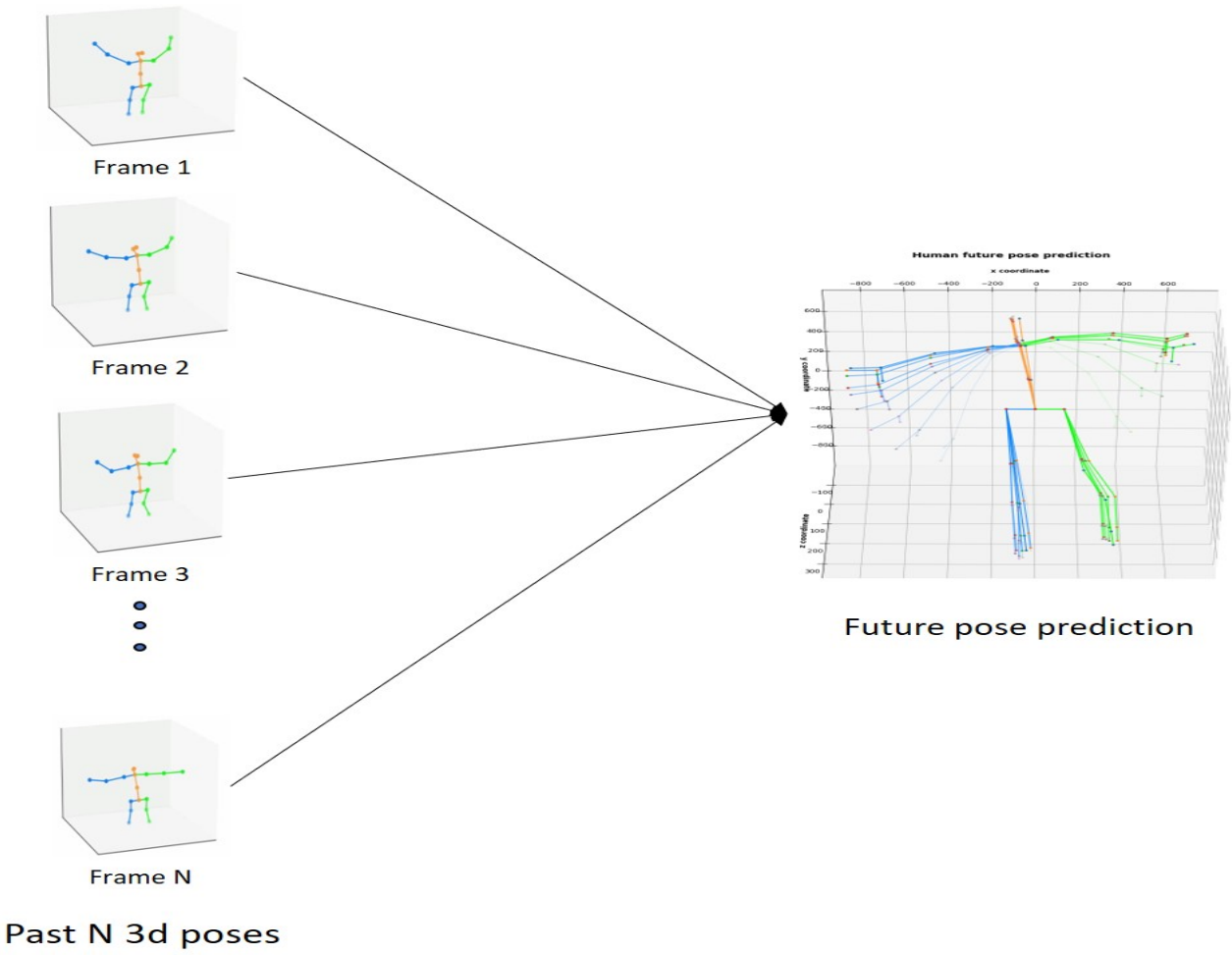

Figure D.9: 3D poses observed frames to prediction.

The goal of 3D pose prediction is given a past horizon of 3D joint poses, predict future joint poses over a given time horizon. For this task, we use the architecture called History Repeats Itself (HisRepItself) [31] which tackles human motion prediction via motion attention.

For this approach, we purely look at past frames over a certain horizon to predict future human poses over another horizon. The underlying hypothesis is since humans tend to

repeat motion across long time periods, sub-sequences in motion history can be discovered via motion attention.

This motion attention is then fed into the prediction model which consists of a Graph Convolution Network (GCN) to capture the spatial relationship of the human skeletal joints.

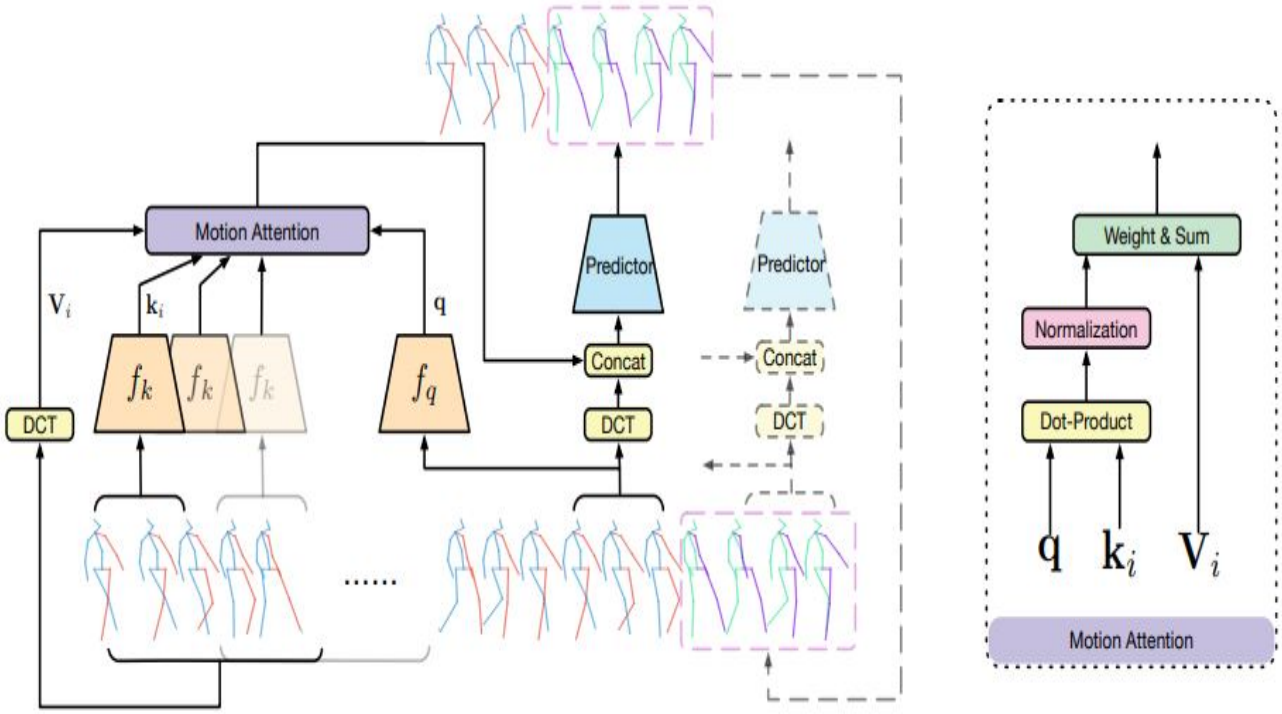

Figure D.10: HisRepItself motion prediction architecture

The visualization of the architecture is shown in Fig. D.10

The past poses are shown as blue and red skeletons and predicted ones and green and purple. For a given time frame (M consecutive poses), the Discrete Cosine Transforms are weighed using the computed attention score, and the weighted sum is combined with DCT coefficients of the last sub-sequence in the prediction layer for future predictions of human poses.

D.5 Inference results

Since the eventual goal of this approach is to leverage accurate human joint pose predictions in a human-robot collaborative setting, we need to give equal importance to accuracy and

inference speed. While we leave accuracy evaluations to test datasets, we analyze inference speeds for evaluating the application of this approach in a real-time setting.

The given pipeline structure is implemented on a live video stream coming from a webcam and inference speeds for each subtask are calculated as shown below:

Table D.3: Inference FPS for individual steps of the pipeline as well as combined FPS.

-	DETR	VIPNAS	Video Pose3D	Motion attention	Overall fps
Inference speed (FPS)	28	54	280	31	11

Fast inference speeds are crucial in a real-time human-robot collaboration task due to which recently we have been seeing a rise in well-engineered models like ViPNAS where heavy models are tied together with lightweight models in a key frame - non-key frame approach in order make these models fast while not compromising on accuracy.

D.6 Challenges and takeaways

The key takeaway from this pipeline-based approach for human motion prediction is the interpretability of the sub-tasks. By breaking down the goal into individual deep learning-based models instead of learning end to end, we can identify and debug errors individually and the integration of domain knowledge in every sub-task becomes easier.

Also due to the modular approach, it is easy to replace certain parts of the model depending on the task at hand and retrain certain parts instead of going through the pain of retraining the entire model. Analyzing the performance of individual components including attributes like inference speed makes it easier to highlight the pain points of an approach and select and fine-tune models accordingly, thus helping scale the architecture.

That being said, there are plenty of challenges that still need to be addressed in human motion prediction including some arising from the pipeline-based approach:

1. Robustness to missing joint positions and noisy data: It was noted that when using keypoint detections, oftentimes there were missing joints due to the human being in

a different orientation or partially outside the frame. Also, when 2D to 3D pose lifting, oftentimes the joint data would be noisy which led to overall bad human pose predictions. A few ways to make this approach more robust is by performing some kind of data imputation on missing joint data, and training the predictor with augmented data that has noise as well as missing joints, thus making sure such cases are not out of distribution.

2. Inference speed: As mentioned earlier, inference speed remains a challenge in the application of this setup in a real-time collaborative setting. Around 11fps of overall inference speed is still not fast enough for high-speed real-time tasks and hence inference speed remains a challenge.

3. Error and uncertainty propogation: We highlighted that there was some noise and uncertainty associated with inferring the human joint pose data which also included errors where there was false detection, thus throwing the predictor off in the downstream task of prediction. Also how uncertainties and errors from one block of the pipeline propagate to the other remains an open question.

4. Incorporating context-based information: As we noted in the prediction part, we solely rely on past human poses to predict future ones. In reality, this is not the case since a lot of times we use scene-based contextual information to drive one's motion.

5. Incorporating joint constraints and fusing physics-based models: Lastly, since this is a well-studied dynamics problem, one possibility to improve this approach is to fuse physics-based models as well as employ hard joint constraints in our model.

Thus we study a pipeline-based approach towards human motion prediction from a stream of camera frames in the form of a video. We highlight the pros and cons of this approach as well as our reasoning for picking specific models for each subtask. We also highlight potential research avenues as well as discuss what particular challenges we came across when experimenting with this approach in an actual setup.

References

[1] D. B. Chaffin, *On simulating human reach motions for ergonomics analyses*, 2002.

[2] J. Mainprice, R. Hayne, and D. Berenson, *Predicting human reaching motion in collaborative tasks using inverse optimal control and iterative re-planning*, IEEE, 2015.

[3] J. Mainprice, R. Hayne, and D. Berenson, *Goal set inverse optimal control and iterative replanning for predicting human reaching motions in shared workspaces*, 2016.

[4] H. Ahn, E. V. Mascaro, and D. Lee, *Can we use diffusion probabilistic models for 3d motion prediction?* 2023.

[5] W. Liu, X. Liang, and M. Zheng, *Task-constrained motion planning considering uncertainty-informed human motion prediction for human–robot collaborative disassembly*, 2023.

[6] V. Renganathan, S. Safaoui, A. Kothari, B. Gravell, I. Shames, and T. Summers, *Risk bounded nonlinear robot motion planning with integrated perception & control*, 2023.

[7] J. S. Park, C. Park, and D. Manocha, *I-planner: Intention-aware motion planning using learning based human motion prediction*, 2017. arXiv: 1608.04837 [cs.RO].

[8] J. Wang, A. Hertzmann, and D. J. Fleet, *Gaussian process dynamical models*, 2005.

[9] T. Tohme, K. Vanslette, and K. Youcef-Toumi, *Reliable neural networks for regression uncertainty estimation*, 2023.

[10] L. P. Swiler, M. Gulian, A. L. Frankel, C. Safta, and J. D. Jakeman, *A survey of constrained gaussian process regression: Approaches and implementation challenges*, 2020.

[11] E. Snelson, Z. Ghahramani, and C. Rasmussen, *Warped gaussian processes*, 2003.

[12] Z. Zhang, Y. Zhu, R. Rai, and D. Doermann, *Pimnet: Physics-infused neural network for human motion prediction*, 2022.

[13] E. Aksan, M. Kaufmann, P. Cao, and O. Hilliges, *A spatio-temporal transformer for 3d human motion prediction*, IEEE, 2021.

[14] V. Adeli, M. Ehsanpour, I. Reid, J. C. Niebles, S. Savarese, E. Adeli, and H. Rezatofighi, *Tripod: Human trajectory and pose dynamics forecasting in the wild*, 2021.

[15] V. Adeli, E. Adeli, I. Reid, J. C. Niebles, and H. Rezatofighi, *Socially and contextually aware human motion and pose forecasting*, 2020.

[16] T. Sofianos, A. Sampieri, L. Franco, and F. Galasso, *Space-time-separable graph convolutional network for pose forecasting*, 2021.

[17] J. Denavit and R. S. Hartenberg, *A kinematic notation for lower-pair mechanisms based on matrices*, 1955.

[18] A. M. Zanchettin, P. Rocco, L. Bascetta, I. Symeonidis, and S. Peldschus, *Kinematic motion analysis of the human arm during a manipulation task*, VDE, 2010.

[19] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, *Real-time human pose recognition in parts from single depth images*, Ieee, 2011.

[20] T. Tohme, M. Sadr, K. Youcef-Toumi, and N. G. Hadjiconstantinou, *Messy estimation: Maximum-entropy based stochastic and symbolic density estimation*, 2023.

[21] P. Maurice, A. Malaisé, C. Amiot, N. Paris, G.-J. Richard, O. Rochel, and S. Ivaldi, *Human movement and ergonomics: An industry-oriented dataset for collaborative robotics*, 2019.

[22] G. Cicirelli, R. Marani, L. Romeo, M. G. Domınguez, J. Heras, A. G. Perri, and T. D'Orazio, *The ha4m dataset: Multi-modal monitoring of an assembly task for human action recognition in manufacturing*, 2022.

[23] L. J. Miranda, *Pyswarms: A research toolkit for particle swarm optimization in python*, 2018.

[24] A. Sengupta, F. Jin, R. Zhang, and S. Cao, *Mm-pose: Real-time human skeletal posture estimation using mmwave radars and cnns*, 2020.

[25] S. Chitta, I. Sucan, and S. Cousins, *Moveit![ros topics]*, 2012.

[26] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, *Chomp: Gradient optimization techniques for efficient motion planning*, IEEE, 2009.

[27] J. Gardner, G. Pleiss, K. Q. Weinberger, D. Bindel, and A. G. Wilson, *Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration*, 2018.

[28] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, *Deformable detr: Deformable transformers for end-to-end object detection*, 2020.

[29] L. Xu, Y. Guan, S. Jin, W. Liu, C. Qian, P. Luo, W. Ouyang, and X. Wang, *Vipnas: Efficient video pose estimation via neural architecture search*, 2021.

[30] D. Pavllo, C. Feichtenhofer, D. Grangier, and M. Auli, *3d human pose estimation in video with temporal convolutions and semi-supervised training*, 2019.

[31] W. Mao, M. Liu, and M. Salzmann, "History repeats itself: Human motion prediction via motion attention," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, Springer, 2020, pp. 474–489.