Augmenting Human-Robot Collaboration Task by Human Hand Position Forecasting

Shyngyskhan Abilkassov[1][a], Michael Gentner[2,3][b] and Mirela Popa[1][c]
[1]*Department of Advanced Computing Sciences, Faculty of Science and Engineering, Maastricht University, P.O. Box 616, 6200 MD, Maastricht, The Netherlands*
[2]*Technical University of Munich, Munich, Germany*
[3]*BMW AG, Munich, Germany*

Keywords: Egocentric Vision, Human-Robot Collaboration.

Abstract: Human-Robot collaboration (HRC) plays a critical role in enhancing productivity and safety across various industries. While reactive motion re-planning strategies have proven useful, there is a pressing need for proactive control involving computing human intentions to enable efficient collaboration. This work addresses this challenge by proposing a deep learning-based approach for forecasting human hand trajectories and a heuristic optimization algorithm for proactive robotic task sequencing problem optimization. This work presents a human hand trajectory forecasting deep learning model that achieves state-of-the-art performance on the Ego4D Future Hand Prediction benchmark in all evaluation metrics. In addition, this work presents a problem formulation and a Dynamic Variable Neighborhood Search (DynamicVNS) heuristic optimization algorithm enabling robot to pre-plan their task sequence to avoid human hands. The proposed algorithm exhibits significant computational improvements over the generalized VNS approach. The final framework efficiently incorporates predictions made by the deep learning model into the task sequencer, which is evaluated in an experimental setup for the HRC use-case of the UR10e robot in a visual inspection task. The results indicate the effectiveness and practicality of the proposed approach, showcasing its potential to improve human-robot collaboration in various industrial settings.

1 INTRODUCTION

Human-robot collaboration (HRC) has gained significant importance in recent years due to the increasing demand for automation. In collaboration with humans, robots can improve productivity, efficiency, and safety in the manufacturing industry (Matheson et al., 2019). However, achieving effective HRC requires robots to comprehend and respond to human intentions and actions, which can be challenging in dynamic environments (Li et al., 2023).

Current research in HRC has primarily focused on developing reactive motion re-planning strategies that can adapt to dynamic environments (Li et al., 2021). While these reactive strategies have proven useful in different use cases ranging from domestic to industrial environments (Ajoudani et al., 2018), they have limitations in anticipating and planning for future changes in human-present environments. To overcome these issues, researchers have emphasized the significance of proactive control through the computation of human intentions (Fan et al., 2022; Li et al., 2023). By accurately predicting human intentions, robots can plan their actions in advance, enabling smoother and more efficient collaboration. In particular, forecasting human hand trajectories provides valuable information for robot motion planners to determine the appropriate task sequence.

Hand is a natural and ubiquitous mode of expression for the human and is an effective mean of communication between the human and the robot (Mukherjee et al., 2022). While several works have proposed using human hand trajectory estimation to augment robot planners, these works are often tailored to specific use cases, are limited in the diversity of used data (Mainprice and Berenson, 2013; Lyu et al., 2022) or by assuming deterministic and static human behavior (Cheng and Tomizuka, 2021). Developing an efficient and proactive robot motion planner

[a] https://orcid.org/0009-0002-5882-4275
[b] https://orcid.org/0000-0003-2319-8956
[c] https://orcid.org/0000-0002-6449-1158

Abilkassov, S., Gentner, M. and Popa, M.
Augmenting Human-Robot Collaboration Task by Human Hand Position Forecasting.
DOI: 10.5220/0012306300003660

In *Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2024) - Volume 4: VISAPP*, pages 262-269
ISBN: 978-989-758-679-8; ISSN: 2184-4321

necessitates the integration of efficient human hand forecasting and robot task sequencing modules.

Robotic task sequencing is the process of optimizing the sequence and order of task visits to improve the overall performance of the robotic system in industrial applications. Although various research has been focused on trajectory optimization (Ata, 2007), efficient control and end-effector path planning (Alatartsev et al., 2014), the popular approach is to model task sequencing problem as a Traveling Salesman Problem (TSP) (Alatartsev et al., 2015). The TSP is an NP-hard problem, which means that it is computationally infeasible to solve for large instances and thus an efficient algorithm is required to find optimal solutions in shortest time. To address the aforementioned challenges, this work proposes a deep learning-based framework for predicting human hand trajectories and integrating the predictions into robot task sequencer. The presented framework includes a state-of-the-art deep learning model trained on diverse human hand motion data that forecasts future hand trajectories. Previous approaches have attempted to solve the problem of hand position forecasting (Chen et al., 2022), but this work achieved higher evaluation results with the same memory requirements, by adapting the presented model's training procedure. The predicted trajectories are then utilized by the robot task sequencer to optimize the robot's task sequencing. A robot adaptive task sequencing module employs a Dynamic Variable Neighborhood Search heuristic algorithm to plan the most efficient task sequence. By forecasting future human hand positions, the robot can proactively adjust its plan to avoid interrupting human intentions, resulting in seamless and efficient collaboration. This work presents the following contributions:

- Generalized model that achieved state-of-the-art performance on future hand position forecasting task.
- Modified Travelling Salesman Problem formulation to model future hand position occupancy.
- Dynamic Variable Neighborhood Search heuristic algorithm to plan the most efficient task sequence.

2 RELATED WORK

Future Hand Position Forecasting. Predicting human hand trajectories is a complex task that requires understanding the intricate and dynamic nature of human movement. Recent works addressed the hand forecasting task, but they are limited by the scope of the applied datasets, which capture hand interactions in limited settings (Fan et al., 2018; Liu et al., 2020; Liu et al., 2022). Fan et al. introduced a two-stream convolutional neural network architecture that predicts future object and hand positions in video frames based on previous frames (Fan et al., 2018). Their innovation lies in forecasting the precise locations of objects in videos, a task not previously optimized for. However, the evaluation was limited to a specific dataset of human interactions, which may not fully represent daily hand interactions.

Liu et al. developed a deep-learning model for anticipating human-object interactions in egocentric videos, introducing the concept of motor attention to capture intentional hand movements (Liu et al., 2020). This model jointly predicts hand trajectories, interaction hotspots, and action labels, outperforming previous methods on benchmark datasets. However, it relies on manual annotations and may face scalability challenges with other datasets.

Liu et al. presented the Object-Centric Transformer (OCT) model to forecast future hand-object interactions in egocentric videos (Liu et al., 2022). This model captures object-centric interactions and hand-object motion dependencies, demonstrating improved performance on a new dataset generated from existing ones. This work's limitations include the dataset's limited size and the assumption that objects are static during interactions.

Robotics Task Sequencing. Simultaneously, the development of a proactive and dynamic task sequencer poses its challenges. Several works addressed the robotic task sequencing optimization problem based on the Traveling Salesman Problem formulation (Dubowsky and Blubaugh, 1989; Edan et al., 1991; Zacharia and Aspragathos, 2005; Kolakowska et al., 2014). However, these approaches have not addressed the dynamic time windows formulation due to the forecasting of human intentions. Dubowsky and Blubaugh introduced an ATSP-based method to plan time-optimal robotic manipulator motions for point-to-point tasks (Dubowsky and Blubaugh, 1989). They used dynamic programming to minimize task completion time but didn't consider multiple inverse kinematics solutions, limiting optimality. Edan et al. aimed to minimize robot cycle time while accounting for kinematics and dynamics (Edan et al., 1991). Their approach involved a TSP formulation and neural network solving but suffered from extensive computational time due to frequent cost matrix recalculations.

Zacharia et al. proposed an optimization-based TSP solution considering multiple inverse kinematics solutions using Genetic Algorithms (Zacharia and Aspragathos, 2005). While efficient, it is heuristic in

nature, so optimal solutions cannot be guaranteed.

Kolakowska et al.'s recent approach employs mathematical constraints and a corresponding solver for up to ten points (Kolakowska et al., 2014). However, the exponential computational growth with the number of points limits its applicability to large scenarios.

Human-Robot Collaboration Using Human Hand Position Prediction. Mainprice and Berenson introduced a framework for safe human-robot collaboration in shared workspaces (Mainprice and Berenson, 2013). They utilized a prediction algorithm to construct a 3D workspace occupancy space, enhancing safety and efficiency. However, the limited class and trajectory diversity restrict real-world applicability.

Landi et al. combined model-based and data-driven approaches to predict human arm movements (Landi et al., 2019). They tracked hand positions with a Kinect camera, fitting them to a model and incorporating neural networks. However, the focus on detecting reaching motions towards the robot and computational complexity are some of its limitations.

Cheng and Tomizuka proposed a hierarchical framework for long-term human hand trajectory prediction and action duration estimation (Cheng and Tomizuka, 2021). They employed a sigma-lognormal function and an online algorithm for parameter adjustment. However, their assumption of deterministic human actions may not hold in real-world scenarios.

Lyu et al. presented an efficient HRC pipeline encompassing trajectory prediction, target estimation, and robot trajectory generation (Lyu et al., 2022). While this pipeline ensures safe collaboration, it requires evaluation on more complex tasks and assumes a static environment, limiting its real-world applicability.

3 FUTURE HAND POSITION FORECASTING

In this section, the deep learning model used to predict future hand positions from a video sequence is presented and evaluated. This model is trained and evaluated on the Ego4D dataset and achieved state-of-the-art performance on Future Hand Position Forecasting benchmark.

3.1 Ego4D Dataset

Training deep learning models requires a diverse dataset that contains annotations of human hands in natural daily interactions. The Ego4D dataset (Grauman et al., 2022) is a large-scale egocentric video dataset and benchmark suite that contains data suiting that task. It consists of 3670 hours of recordings capturing daily life activities across 74 worldwide locations and nine countries. The dataset includes diverse occupational backgrounds and encompasses hand-object interactions in various environments. It provides hand annotations along with a hand position forecasting benchmark for evaluation. The benchmark focuses on predicting future hand positions in future key frames based on a recording of hands for consecutive 2 seconds. The benchmark defines five key frames: the contact frame (x_c), the pre-condition frame (x_p), and three frames preceding the pre-condition frame by 0.5s, 1.0s, and 1.5s, denoted as x_{p1}, x_{p2}, and x_{p3}, respectively.

3.2 Evaluation Metric

The evaluation metric is based on the Future Hand Prediction (FHP) benchmark presented as part of the Ego4D dataset (Grauman et al., 2022). The benchmark is based on two error metrics: Mean Key Frame Displacement Error (M.Disp.) and Contact Key Frame Displacement Error (C.Disp.). Mean Key Frame Displacement Error is defined in Eq. 1.

$$D_m = \frac{1}{n} \sum_{i \in H_t} ||h_i - \hat{h}_i|| \quad (1)$$

H_t denotes a set of visible hand positions in key frames, n denotes the size of the set H_t, h_i refers to the predicted hand position, while the $\hat{h}_i$ refers to the ground truth hand location. It should be noted that if the hand is not present in the annotation, zeros are padded into the $\hat{h}_i$.

The Contact Key Frame Displacement Error is defined in Eq. 2.

$$D_c = ||h_c - \hat{h}_c|| \quad (2)$$

h_c refers to the predicted hand position in the contact frame, while the $\hat{h}_c$ refers to the ground truth hand position in the contact frame.

It should be noted that the presented evaluation metric does not penalize the model if it gives predictions on frames without hands.

3.3 UniFormer Model

The UniFormer model is used for forecasting future hand positions. The UniFormer is a unified transformer architecture for efficient spatiotemporal representation learning from high-dimensional videos (Li et al., 2022). It addresses the challenges of spatiotemporal redundancy and dependency by combining the strengths of 3D CNNs and vision transformers. The

UniFormer architecture serves as a backbone in all experiments conducted in this work. The UniFormer is used as the feature extraction backbone and is followed by a linear mapping function as a regressor.

3.4 Multitask Learning Loss

This work utilizes a multi-task learning loss to facilitate the learning of future hand positions and occupancy representations. Firstly, the regression task involves predicting hand positions in spatial coordinates for both hands in five key frames. Each prediction target vector consists of 20 values, where two represent the left and right hand, two represent the x and y coordinates, and five represent the key frames. In addition to those annotations present in the dataset, this work introduces annotations for learning the contact delay time c_t. This is accomplished by calculating the difference between the ground truth contact frame time and the frame time of the first key frame. Consequently, the model becomes capable of learning and predicting the precise time delay between the pre-condition and contact frames. The resulting target representation vector comprises 21 values employed in the regression task.

The regression component measures the Smooth L1 loss between the predicted and ground truth values. Subsequently, the losses for all 21 points are summed and divided by the total sum of mask values. This loss component encourages the model to generate accurate predictions for all values at the expense of producing false positives and is defined in Equations 3 and 4.

$$l_i = \begin{cases} 0.5 \cdot (h_i - \hat{h}_i)^2, & \text{if } |h_i - \hat{h}_i| < 1 \\ |h_i - \hat{h}_i| - 0.5, & \text{otherwise} \end{cases} \tag{3}$$

$$\text{RLoss} = \frac{\sum_{i=1}^{N} l_i}{\sum_{i=1}^{N} m_i} \tag{4}$$

The classification component introduces an additional branch in the model output that predicts the existence mask of the hand in a future key frame or contact with the object. This branch outputs a single value between 0 and 1, representing the probability of the target's existence. The ground truth for this branch is a binary value of 1 (indicating target existence) or 0 (indicating target non-existence). The binary cross-entropy loss is computed by comparing the predicted existence probability with the ground truth existence label as shown in Equation 5.

$$\text{CLoss} = -\sum_{i=1}^{N} \hat{m}_i \log(m_i) \tag{5}$$

The total loss is then formulated as a combination of the regression loss and the existence loss, suitably weighted. The following equation represents the computation of the total loss:

$$\text{TotalLoss} = \alpha \cdot \text{RLoss} + \beta \cdot \text{CLoss} \tag{6}$$

In Equation 6, the weights α and β are assigned to balance the contributions of the regression and the binary cross-entropy loss, respectively.

3.5 EgoVLP Pretraining

In this work, clip-level annotations from EgoVLP are utilized for model pretraining. Specifically, the verb-filtered clip-level annotations provided by Chen et al. (Chen et al., 2022) are employed. Authors highlight that video-language pretraining based on the entire Ego4D dataset significantly improves model performance on various downstream tasks. Their work suggests that using pretrained backbones directly yields inferior performance due to the distribution gap between egocentric and exocentric data.

4 DYNAMIC TASK SEQUENCING

In this section the Dynamic Traveling Salesman Problem with Time Windows problem formulation along with the heuristic optimization algorithm is presented. The trajectories predicted by the deep learning model are utilized by the Dynamic Variable Neighborhood Search heuristic algorithm to plan the most efficient task sequence. By forecasting future human hand positions, the robot can proactively adjust its plan to avoid interrupting human intentions, resulting in a seamless and efficient collaboration.

4.1 Dynamic Traveling Salesman Problem with Time Windows Formulation

This work presents a formulation of the Dynamic Traveling Salesman Problem with Time Windows (D-TSPTW), introducing a modification to the traditional TSPTW problem by incorporating time variability into node parameters. The introduction of time variability in the D-TSPTW implies that the problem graph $G = (V, A)$, along with its node parameters, can undergo changes over time. The main difference with a well-established Time-Dependent Traveling Salesman Problem with Time Windows (TD-TSPTW) is that D-TSPTW assumes changes in the customer configurations, while TD-TSPTW assumes changes in travel times (Vu et al., 2020).

To define the problem, specific restrictions are imposed. Firstly, the graph structure and node parameters remain unchanged during the execution of one cycle. Here, a cycle is defined as a sequence of actions performed by an agent, which includes servicing a node, traveling to another node, and waiting for the start time of the next node. Additionally, D-TSPTW assumes that only the following parameters can change between each cycle: the number of present nodes, their start times, and due times. Node coordinates and service times are assumed to remain constant throughout the execution of the task.

The D-TSPTW task is formalized as a tour on a time-expanded network denoted as $\mathcal{D} = (\mathcal{N}, \mathcal{A})$. The time in this notation is split into discretized cycles, as described earlier. The node set $\mathcal{N}$ consists of nodes (i,t), where $i \in N$ and $t \in [a_i, b_i]$, representing the time window for node i. The travel arc set $\mathcal{A}$ comprises travel arcs represented as $((i,t),(j,\hat{t}))$, where $i \neq j$, $(i,j) \in \mathcal{A}$, $t \geq a_i$, and $\max(a_i, t + \tau_{ij}(t)) \leq \hat{t} \leq b_i$.

The binary variable x_a is equal to 1 if and only if the arc $a = ((i,t),(j,\hat{t}))$ is visited. Furthermore, $c_a = c_{ij}(t)$ denotes the non-negative travel cost of arc a. Thus, the optimization task can be defined as an integer programming problem, as shown in Equation 7.

$$\begin{aligned} \min \quad & \sum_{a \in \mathcal{A}} c_a x_a \\ \text{s.t.} \quad & \sum_{a \in \mathcal{A}} x_a = 1 \\ & \hat{t} \geq t + c_a - M(1 - x_a) \quad \forall a \in \mathcal{A} \\ & a_i \leq t \leq b_i \quad \forall i \in V \\ & x_a \in \{0,1\} \quad \forall a \in \mathcal{A} \end{aligned} \tag{7}$$

The objective function in Equation 7 aims to minimize the total cost of the tour. The first constraint ensures that each arc in the set $\mathcal{A}$ is visited exactly once. The second constraint enforces the requirement that the visiting time of node j must be greater than or equal to the sum of the visiting time of node i and the associated arc cost.

This work assumes that the number of cycles is 5, where each cycle correspond to a particular time in the future as denoted in each key frame definition in section 3.1.

4.2 Dynamic Variable Neighborhood Search Heuristic Algorithm

A heuristic solver for the Dynamic Traveling Salesman Problem with Time Windows (D-TSPTW) is now presented. The motivation for this work comes from the two-stage heuristic presented by da Silva and Urrutia (Da Silva and Urrutia, 2010) for the traditional TSPTW problem. Their approach combines the Variable Neighborhood Search (VNS) for generating feasible solutions and the Variable Neighborhood Descent (VND) for optimizing local solutions. The algorithm they presented demonstrated computational efficiency compared to state-of-the-art approaches and achieved superior results in terms of execution time on most of the benchmark instances (Da Silva and Urrutia, 2010).

The approach proposed by da Silva and Urrutia is limited to the static TSPTW problem and does not account for variations in node parameters as introduced in the extended D-TSPTW problem formulation. This approach can be applied in a straightforward manner and be used to solve the D-TSPTW problem by tackling each cycle individually. This approach will be used as a baseline in our experiments.

To improve the optimization performance, this work proposes an extension of the GVNS optimization algorithm to solve the D-TSPTW problem called DynamicVNS. The proposed algorithm is designed to tackle the dynamic nature of the problem and adapt to changing node parameters. The main difference with the algorithm proposed by da Silva and Urrutia is in the construction phase. The `GetFeasibleSolution()` algorithm, shown in Algorithm 1, is responsible for obtaining a feasible solution in the construction phase for the D-TSPTW problem. It takes the objective function $f(\cdot)$ and the number of nodes n as inputs, and optionally, a previous solution *previousSolution*. The algorithm repeatedly searches for feasible solutions, employing perturbation and local search operation using `Local1Shift()` function until a feasible solution is found. If *previousSolution* is provided, the algorithm starts with this solution. Otherwise, it begins with a new solution using the `NewSolution()` function, which generates a path depending on an initialization strategy. The initialized path can be sorted by customers' ready or due times or randomly shuffled. The perturbation is controlled by the variable k, which is increased iteratively to explore different neighborhoods until a feasible solution is found or until k_{max} is reached. As a result, each consecutive expanded problem formulation is solved until the last one.

5 RESULTS

This section begins by showcasing the outcomes of the UniFormer model training and evaluation experiments. Following that, it provides an explanation

Data: $f(\cdot)$, n, $previousSolution = None$
Result: x
repeat
  $k \leftarrow 1$;
  if $previousSolution \neq None$ **then**
    $x \leftarrow previousSolution$;
  else
    $x \leftarrow NewSolution(n)$;
  end
  $x \leftarrow Local1Shift(x)$;
  while x *is unfeasible and* $k < k_{max}$ **do**
    $x' \leftarrow Shake(x,k)$;
    $x^* \leftarrow Local1Shift(x')$;
    if $f(x^*) < f(x)$ **then**
      $x \leftarrow x^*$;
      $k \leftarrow 1$;
    else
      $k++$;
    end
  end
until x *is feasible*;

Algorithm 1: GetFeasibleSolution() algorithm.

of the DynamicVNS heuristic optimization algorithm experiments. The section concludes by presenting the HRC setup, including both simulation and real robot experiments.

5.1 UniFormer Model Training

The UniFormer model was firstly pretrained on the video-language EgoClip dataset and then fine-tuned for the future hand prediction task. The UniFormer model is fine-tuned on the future hand prediction task with the following configuration. Training is performed with a batch size of 16 and 8 segments per sample. The input size and short side size were set to 320. Each input consisted of 4 frames. The optimizer used is AdamW, with a learning rate of 1e-3 and beta values of 0.9 and 0.999. Weight decay is set to 0.05. The training process spanned 30 epochs, with 5 warm-up epochs. During testing, 2 segments were used, and 3 crops were taken per segment. The sampling strategy involved sampling 8 frames in 30 temporal views, which showed best performance across mutliple experiments. Moreovoer, experiments showed that better evaluation results on the validation sets could be achieved by reducing the learning rate to 10e-4, removing abnormal videos, and employing early stopping. Finally, the model was trained using a multi-task learning approach, incorporating the best learning parameters and sampling strategy. The learning task employed the multi-task loss defined in Equations 3, 4, 5, and 6.

Despite the evaluation benchmark not explicitly considering the computed multi-task output, the model achieved state-of-the-art performance on the FHP benchmark. The final model's performance is compared with previous state-of-the-art approaches in Table 1, demonstrating superior results on the test and validation data. All presented UniFormer-based results use the (320, 8, 30) sampling strategy for common evaluation.

5.2 Qualitative Results

Figure 1 showcases qualitative results from the final trained model on sample video snippets extracted from the validation set. The figure depicts the predictions made on five key frames ($x_{p3}, x_{p2}, x_{p1}, x_p, x_c$) in sequential order. Green dots represent ground truth hand positions, while blue dots depict the model's predictions. The presented visualization illustrates that the model can clearly model human hand positions in future frames with varying people, environments, and lighting conditions. The predicted trajectory clearly follows the intended human trajectory with minor differences.

Despite minor differences, the model's overall performance demonstrates its ability to accurately predict human hand positions in various scenarios. These qualitative results provide valuable insights into the model's capabilities and highlight its potential for real-world applications.

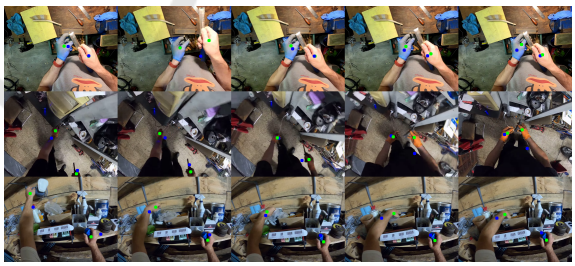

Figure 1: Qualitative results showing predictions made by the final model on the 5 key frames in sequential order (x_{p3}, x_{p2}, x_{p1}, x_p, x_c). Ground truth hand positions are represented in green dots, and predictions made by the model are represented in blue dots.

5.3 DynamicVNS Implementation and Evaluation

The DynamicVNS algorithm, described in the methodology section, was implemented in Python 3.10. The algorithm's performance was evaluated using three different route initialization strategies: random shuffling (`Random`), sorting based on node due

Table 1: Evaluation of the best-performing models on the test set and comparison with previously presented state-of-the-art.

Set	Method	Left Hand		Right Hand	
		M.Disp. ↓	C.Disp. ↓	M.Disp. ↓	C.Disp. ↓
Test	I3D (Grauman et al., 2022)	52.98	56.37	53.69	56.17
	UniFormer (Chen et al., 2022)	43.85	53.33	46.25	53.38
	UniFormer + Multi-Task Loss	**43.00**	**53.47**	**44.36**	**53.42**

Table 2: Comparison experiments of the DynamicVNS and GVNS algorithm on time unrolled sequencing tasks using different initial path generation strategies.

#	Random		Due Time		Ready Time	
	DynamicVNS	GVNS	DynamicVNS	GVNS	DynamicVNS	GVNS
mean	**0.326**	4.814	**0.563**	28.200	**0.341**	0.346
std	0.00179	0.56678	0.00976	0.24479	0.00846	0.00406

time b_i (`Due Time`), and sorting based on node ready time a_i (`Ready Time`). The execution time of the DynamicVNS algorithm was recorded for each initialization strategy and compared to the GVNS (Da Silva and Urrutia, 2010). Table 2 presents the performance differences for each strategy. Each strategy-algorithm pair was evaluated ten times on the same benchmark instance, and the execution time was recorded. The table 2 displays the mean and standard deviation of the computational times across all runs.

The results clearly show that DynamicVNS outperforms the GVNS algorithm on all task instances while generating solutions with the same travel cost. The algorithm exhibits a noticeable speedup when using the random and due time initialization strategies, but the speedup is smaller when using the ready time initialization strategy. The superior performance of DynamicVNS can be attributed to its inherent design, which preserves and utilizes temporal information from previous runs. It should be noted that while the random shuffling method shows the fastest execution time in the presented table, it was not able to find the optimal solutions in some of the benchmark instances. Therefore, the ready-time initialization strategy offers the best speed and robustness trade-off.

5.4 HRC Experiments

Robotic task sequencing experiments were conducted in simulation and on a real robot setup. Firstly, the simulation environment in the Gazebo simulation software was constructed. The simulation environment consists of a UR10e robot with a mounted camera and a table with four objects on top. The simulation environment can be seen in Figure 2. Initial experiments with pre-recorded hand movements were conducted in the simulation environment to validate robot task-planning behavior.

The real setup with the same configuration can be seen in Figure 3. The only difference is the overhead-mounted camera that captures the hand movements and sends the images to the camera stream. Multiple interaction experiments were conducted to confirm the feasibility of the proposed framework for real-time HRC application. Experiments have shown that the forecasting algorithm can accurately predict hand positions, and the task sequencing algorithm produces explainable task sequences that do not interfere with human intentions.

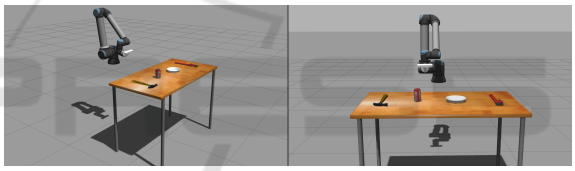

Figure 2: Simulation environment setup consisting of a simulated UR10e robot and tabletop objects.

Figure 3: Real setup consisting of UR10e robot, overhead mounted camera, and tabletop objects.

6 CONCLUSIONS

The comprehensive framework developed in this paper provides an overview of the development of proactive robotic planner. By leveraging deep models for future hand position forecasting and integrating them with task sequencing algorithms, safer and more productive human-robot interactions can be facilitated in diverse real-world applications. As a di-

rection for future work, further refinements and optimization of the framework can be explored alongside more sophisticated travel cost estimation based on robot motion profiles, integration of obstacle avoidance modules, and more use-case experiments. Moreover, investigations into end-to-end approaches of human intention forecasting for robot task sequencing may prove to be effective.

REFERENCES

Ajoudani, A., Zanchettin, A. M., Ivaldi, S., Albu-Schäffer, A., Kosuge, K., and Khatib, O. (2018). Progress and prospects of the human–robot collaboration. *Autonomous Robots*, 42:957–975.

Alatartsev, S., Belov, A., Nykolaychuk, M., and Ortmeier, F. (2014). Robot trajectory optimization for the relaxed end-effector path. In *2014 11th Int. Conf. on Informatics in Control, Automation and Robotics (ICINCO)*, volume 1, pages 385–390. IEEE.

Alatartsev, S., Stellmacher, S., and Ortmeier, F. (2015). Robotic task sequencing problem: A survey. *Journal of intelligent & robotic systems*, 80:279–298.

Ata, A. A. (2007). Optimal trajectory planning of manipulators: a review. *Journal of Engineering Science and technology*, 2(1):32–54.

Chen, G., Xing, S., Chen, Z., Wang, Y., Li, K., Li, Y., Liu, Y., Wang, J., Zheng, Y.-D., Huang, B., et al. (2022). Internvideo-ego4d: A pack of champion solutions to ego4d challenges. *arXiv preprint arXiv:2211.09529*.

Cheng, Y. and Tomizuka, M. (2021). Long-term trajectory prediction of the human hand and duration estimation of the human action. *IEEE Robotics and Automation Letters*, 7(1):247–254.

Da Silva, R. F. and Urrutia, S. (2010). A general vns heuristic for the traveling salesman problem with time windows. *Discrete Optimization*, 7(4):203–211.

Dubowsky, S. and Blubaugh, T. (1989). Planning time-optimal robotic manipulator motions and work places for point-to-point tasks. *IEEE Transactions on Robotics and Automation*, 5(3):377–381.

Edan, Y., Flash, T., Peiper, U. M., Shmulevich, I., and Sarig, Y. (1991). Near-minimum-time task planning for fruit-picking robots. *IEEE transactions on robotics and automation*, 7(1):48–56.

Fan, C., Lee, J., and Ryoo, M. S. (2018). Forecasting hands and objects in future frames. In *Proc. of the European Conf. on Computer Vision (ECCV) Workshops*.

Fan, J., Zheng, P., and Li, S. (2022). Vision-based holistic scene understanding towards proactive human–robot collaboration. *Robotics and Computer-Integrated Manufacturing*, 75.

Grauman, K., Westbury, A., Byrne, E., Chavis, Z., Furnari, A., Girdhar, R., Hamburger, J., Jiang, H., Liu, M., Liu, X., et al. (2022). Ego4d: Around the world in 3,000 hours of egocentric video. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, pages 18995–19012.

Kolakowska, E., Smith, S. F., and Kristiansen, M. (2014). Constraint optimization model of a scheduling problem for a robotic arm in automatic systems. *Robotics and Autonomous Systems*, 62(2):267–280.

Landi, C. T., Cheng, Y., Ferraguti, F., Bonfè, M., Secchi, C., and Tomizuka, M. (2019). Prediction of human arm target for robot reaching movements. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5950–5957.

Li, K., Wang, Y., Gao, P., Song, G., Liu, Y., Li, H., and Qiao, Y. (2022). Uniformer: Unified transformer for efficient spatiotemporal representation learning. *arXiv preprint arXiv:2201.04676*.

Li, S., Han, K., Li, X., Zhang, S., Xiong, Y., and Xie, Z. (2021). Hybrid trajectory replanning-based dynamic obstacle avoidance for physical human-robot interaction. *Journal of Intelligent & Robotic Systems*, 103:1–14.

Li, S., Zheng, P., Liu, S., Wang, Z., Wang, X. V., Zheng, L., and Wang, L. (2023). Proactive human–robot collaboration: Mutual-cognitive, predictable, and self-organising perspectives. *Robotics and Computer-Integrated Manufacturing*, 81:102510.

Liu, M., Tang, S., Li, Y., and Rehg, J. M. (2020). Forecasting human-object interaction: joint prediction of motor attention and actions in first person video. In *Computer Vision–ECCV 2020*, pages 704–721.

Liu, S., Tripathi, S., Majumdar, S., and Wang, X. (2022). Joint hand motion and interaction hotspots prediction from egocentric videos. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, pages 3282–3292.

Lyu, J., Ruppel, P., Hendrich, N., Li, S., Görner, M., and Zhang, J. (2022). Efficient and collision-free human-robot collaboration based on intention and trajectory prediction. *IEEE Trans. on Cognitive and Developmental Systems*.

Mainprice, J. and Berenson, D. (2013). Human-robot collaborative manipulation planning using early prediction of human motion. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 299–306. IEEE.

Matheson, E., Minto, R., Zampieri, E. G., Faccio, M., and Rosati, G. (2019). Human–robot collaboration in manufacturing applications: A review. *Robotics*, 8(4):100.

Mukherjee, D., Gupta, K., Chang, L. H., and Najjaran, H. (2022). A survey of robot learning strategies for human-robot collaboration in industrial settings. *Robotics and Computer-Integrated Manufacturing*, 73:102231.

Vu, D. M., Hewitt, M., Boland, N., and Savelsbergh, M. (2020). Dynamic discretization discovery for solving the time-dependent traveling salesman problem with time windows. *Transportation science*, 54(3):703–720.

Zacharia, P. T. and Aspragathos, N. (2005). Optimal robot task scheduling based on genetic algorithms. *Robotics and Computer-Integrated Manufacturing*, 21(1):67–79.