Supplementary Material: Closed-loop Diffusion Planning over Multi-Modal Distributions for Robot Follow-Ahead

Author Names Omitted for Anonymous Review.

The Supplementary Material will first fill in the missing details on the cost function in the problem formulation in Sec. III (Sec. I). We will provide our diffusion model neural network implementation details, algorithm pseudocode, and discuss its advantages on robot system design (Sec. II). We will provide additional experiments on both human motion prediction and closed-loop path planning. We will also include details on real robot demonstration (Sec. III). Lastly, we will discuss the limitations of this paper (Sec. IV).

I. Problem Formulation

In this section, we introduce the specific cost function used for the RFA task.

Our cost function (Eq. 3) in the motion planning algorithm is designed mainly on two terms for the RFA task: viewing quality $\mathcal{L}_v$ and collision loss $\mathcal{L}_{col}$. In practice, since our occupancy map has noise, in Eq. 2, we set the weights for each loss to $lambda_v = 1$ and $\lambda_{col} = 20$.

a) Viewing Quality: We design our viewing quality metric for each time step as a function of the robot and the human's 2D pose, based on the Pixels-Per-Area (PPA) metric proposed by [1]. Assuming the camera is rigidly fixed on the robot. Given the robot 2D pose $\mathbf{X}_t = (x_t^R, y_t^R, \theta_t^R)$, and the human 2D pose $\tau_t = (x_t^H, y_t^H, \theta_t^H)$ at time t, PPA is defined as $\mathcal{C}_{ppa} = \cos\delta / \|\mathbf{X}_t - \tau_t\|$. $\delta = |\theta_t - \theta_h|$ is defined as the viewing direction difference. $\|\mathbf{X}_t - \tau_t\|$ is the distance between the robot and the human. Maximizing PPA yields a robot pose viewing from the normal direction (smaller δ), and viewing from a closer distance (smaller $d = \|\mathbf{X}_t - \tau_t\|$). The original work [1] maximizes the PPA for better viewing quality with a minimum (safety) distance d_{safe}. In our paper, we use the negative PPA values as our cost and add by one constant to ensure it's positive $\mathcal{L}_v = -\mathcal{C}_{ppa} + 1/d_{safe}$.

b) Collision: We define the collision cost as $\mathcal{L}_{col}(\mathcal{X}_{t_0:}, S) = \sum_t \mathbf{X}_t \cdot |\mathbf{S}|$. In other words, for each time step, for the human position $\mathbf{X}_t$, the collision cost would be 1 if the position is occupied or unknown; otherwise, 0.

II. Approach

In this section, we include details of our diffusion model implementation for predicting human future motion. We will discuss the insights behind our selection of sampling methods. We will also provide a detailed pseudocode for the **CLDP** algorithm and discuss its relationship to robot system design.

A. Network Architecture

In this paper, we implement two neural networks from state-of-the-art papers (**Dif-TR** [2] and **Dif-Unet** [3]) to learn the conditioned noise in the diffusion model. We show the architectures in Fig. 1. Both architectures share the same structure to encode the trajectory history and the local map. We use ResNet-18 [4] structure to extract the latent feature z_{map}. We use 1D convolution to extract the history trajectory (pose) embedding z_{traj}. We concatenate them as the latent conditioning feature z_{cond}. For **Dif-TR**, we encode the intermediate samples into the embedding space with d_{head} dimension and condition it on the features z_{cond} with the Multi-Head Attention (MHA) [5] mechanism. For **Dif-Unet**, we similarly encode the intermediate samples into the embedding space d_{feat} and go through n_{layer} conditional residual blocks, in each of which it's conditioned with FiLM layers. The dimensions for each layer and other parameters are displayed in the Table. I.

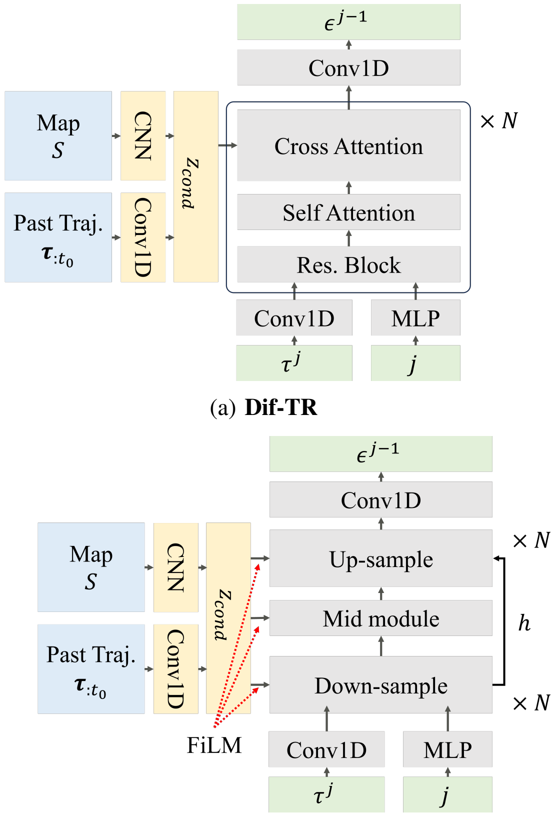
TABLE I: **Implementation details.**

Method	**Parameters**	**Value**
DDPM [6]	timesteps	100
	β_0	0.0001
	β_T	0.02
DDIM [7]	timesteps	100
	step size	5
	β_0	0.0001
	β_T	0.02
Method	**Layers**	**Dimension**
Conditioning	z_{map}	512
	z_{traj}	64
	z_{pose}	64
Dif-TR	d_{head}	64
	Num. of head	8
	Num. of MHA blocks	4
Dif-Unet	d_{feat}	64
	n_{layer}	8

B. Discussion on Sampling Methods

In this project, we choose the classical DDPM [6] method to sample the human future motion trajectories. We also experiment with other sampling methods, such as DDIM [7] or the perturbed DDPM method [2].

In the perturbed DDPM method, we define a few cost terms to guide the denoising process: 1) The smoothness of the human trajectory $\mathcal{C}_{smooth} = \sum_{t=t_0}^{t_{T-1}} |\ddot{\tau}_{t+1} - \ddot{\tau}_t|$ to



(a) **Dif-TR**

(b) **Dif-Unet**

Fig. 1: **Network architecture.** We implement two neural networks to predict the conditioned noise. Conditioning feature z_{cond} is extracted and used in cross-attention layer (**Dif-TR**) or FiLM layer (**Dif-Unet**)

avoid human's zig-zagging motion, 2) the collision loss $\mathcal{C}_{col} = \sum_t \tau_t \cdot |\mathbf{S}|$, and 3) the initial loss $\mathcal{C}_{init} = \tau_{t_0}$ to enforce the continuity between the predicted trajectory and the history. The summed cost is a linear combination among them $\mathcal{C}(\tau) = \eta_s \mathcal{C}_{smooth} + \eta_{col}\mathcal{C}_{col} + \eta_{init}\mathcal{C}_{init}$. In the denoising process, the trajectory sample is subtracted with one extra term with respect to the gradient to the summed cost shown as Eq. 1 below. Please refer to [2], [8] for more details.

$$\begin{aligned} \mu &= \mu_\phi(\tau^{(j)}, t, S, z_{cond}) \\ \tau^{(j-1)} &= \mathcal{N}(\tau^{(j)}; \mu + \lambda\Sigma\nabla_{\tau^{(j)}}(\mathcal{C}(\tau^{(j)}))|_{\tau^{(j)}=\mu}, \Sigma) \end{aligned} \quad (1)$$

However, experiments found that DDIM [7] inference is faster than DDPM, but they are less accurate. Meanwhile, perturbed DDPM does not significantly improve the prediction results while slowing the inference speed. Therefore, we choose DDPM as our sampling method.

C. Closed-loop Diffusion Planning Algorithm

In the planning task, we assume that time discretization is uniform. Control inputs are bounded by $0 < v < v_{max}$, and $|\omega| < \omega_{max}$. Here, we provide pseudo-code for our proposed (**CLDP**) method. We want to highlight that this scheme also aligns well with the robot system design perspective. While sampling the entire distribution (step 1) with a diffusion model is computationally expensive and slow, we can run it at a slower frequency and potentially deploy the module on a remote computing unit with higher computational power. Meanwhile, we can assign the probability updating and path planning steps (steps 3-7) on the onboard computer, which can be run at a high frequency and with less latency.

Algorithm 1 Closed-loop Diffusion Planning (**CLDP**)

```
Input: Human trajectory history $\tau_{:t_0}$, local map $S$, and the
       robot's current state $X_0$
1: $\{\hat{\tau}_{t_0:}\}$. Sample future human motion trajectories.
2: while waiting for a new sample set do
3:     $\tau_{t_k}$. Localize the human's position.
4:     $\lambda_i \leftarrow \exp\left(-\sum_{t=t_0\ldots t_k}\frac{1}{2\sigma_t^2}\|\tau_t - \hat{\tau}_t^i\|\right)$. Update the
       posterior probability for each sample.
5:     $\lambda_i \leftarrow softmax(\lambda_i)$. Normalize the probability
       among all samples.
6:     $\mathcal{U} \leftarrow \arg\min_{\mathcal{U}} \sum_{\tau^i \in \{\hat{\tau}_{t_0:}\}} \lambda^i \mathcal{J}(\tau^i, \mathcal{X}_{t_0:})$. Solve the
       minimization problem with the updated probability.
7:     Execute $\mathcal{U}$ with the first action $\mathbf{u}_{t_1}$
8: return
```

III. Experiments

In this section, we provide additional experimental details within three main aspects. 1) Junction dataset. 2) Additional experiments on human motion prediction and closed-loop diffusion planning. 3) Real robot experiments.

A. Junc dataset

In this paper, we create a dummy dataset, Junc (for 'Junction'), to evaluate our proposed method. The goal is to evaluate the model's performance when the ground truth human motion distribution is bimodal. The human is entering a perpendicular T-junction and about to turn left or right. We fix the velocity of human motion and the total trajectory length. We inserted two randomness into the dataset. 1) The human arrives at a different position at the current time step at the junction. 2) The human turns at a random position uniformly within an interval. We force humans to turn in either direction with equal probability. The distribution of human position at each time is visualized in the supplementary video.

B. Additional Results for Human Motion Prediction

a) Visualizing statistical distribution: To better illustrate the predicted trajectory distribution, we plot the statistical distribution of human positions at different time steps as a heatmap in Fig. 2. In other words, we plot the marginal probability map $p(\hat{\tau}_t)$ over each dataset, and compare our prediction results against the ground truth. We show from Fig. 2 that the diffusion model can represent the future distribution well among different datasets.

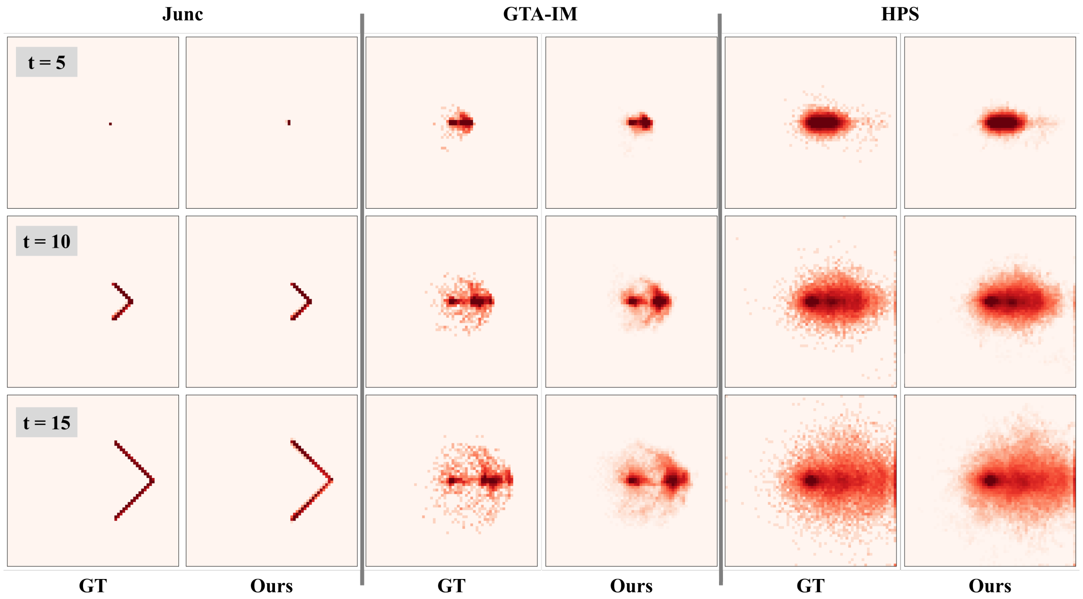


Fig. 2: **Human position distribution at each time.** We visualize the statistical distribution of the human position at time t. Each row represents a different time step. A darker red represents a higher probability. We show that diffusion models can accurately represent the future distribution of human motion.

b) Experiment on the human model: We also investigate further: What human model should we assume for human motion? Can we assume that the future motion is conditioned on the past few seconds or purely depends on the current state (Markovian)? To answer this question, we conducted two more experiments. Apart from predicting the trajectory conditioned on the past 2D trajectory, we also try predicting the trajectory, 1) only conditioned on the map (Markovian), named as *dif(map)*, and 2) conditioned on the full 3D-poses (skeleton key points), named as *dif(pose)*. We use **Dif-TR** as our sampling architecture.

We show from Table II that providing extra information, such as human trajectory history or human poses, does help the human motion prediction, compared to only conditioning on the map. There are two main takeaways: 1) Human motion is not Markovian. We can leverage some underlying patterns of past human trajectory to predict future trajectory. 2) Human body poses provide more information to predict human motion, which is consistent with the findings of the existing works [9], [10].

C. Additional Results for CLDP

a) Visualization of importance weight: We further show detailed examples of the closed-loop importance weight-adjusting process in the Fig. 3. We visualize the probability for each sample as the number at the end point of the trajectory. As shown from Fig. 3, as the human moves along the path, the posterior probability for each sample is updated, and wrong samples are rolled out.

b) Running time: We report the running time of the prediction and planning modules. In our experiment setup (Nvidia 4080 + Intel i7-13700K CPU), it takes 375.7ms for the diffusion model to sample 10 human future trajectories. In contrast, our robot path can be planned in 20.4ms using the MPPI [11] planner, which is 18+ times faster than the prediction module.

c) Additional Discussion: From both qualitative and quantitative results, we see that the closed-loop diffusion planning strategy performs better in two main aspects: 1) ability to leverage the underlying pattern of human motion and 2) agility under uncertainty. The experiments indicate that our **CLDP** method provides a middle space between the reactive planning and the predict-and-plan scheme (Fig. 4). We leverage some prior knowledge learned from the data. At the same time, our method does not entirely rely on prediction accuracy and maintains some agility.

D. Real robot Demonstration

We provide a real-world robot demonstration of the follow-ahead task. We built our robot on top of a Rover mobile robot base [12] with two RealSense D-series cameras (RGB-D). Our software stack consists of three main modules: 1) Robot SLAM and navigation, 2) human motion forecasting, and 3) robot path planning for RFA. All modules are built on top of the ROS system [13].

a) SLAM and navigation: We use RtabMap [14] for (3D) indoor environment mapping, based on RGB-D readings from the front camera. We use *robot_localization* [15] to merge IMU readings with visual odometry and localize the robot in 30Hz. We use the pre-built map during the execution time. Meanwhile, we maintain the local map in the actor frame with a size of 8m and a resolution of 64 × 64 for the human motion prediction. We also use the *move_base* package from the *navigation_stack* [16] to maintain the global costmap for robot navigation in $10Hz$.

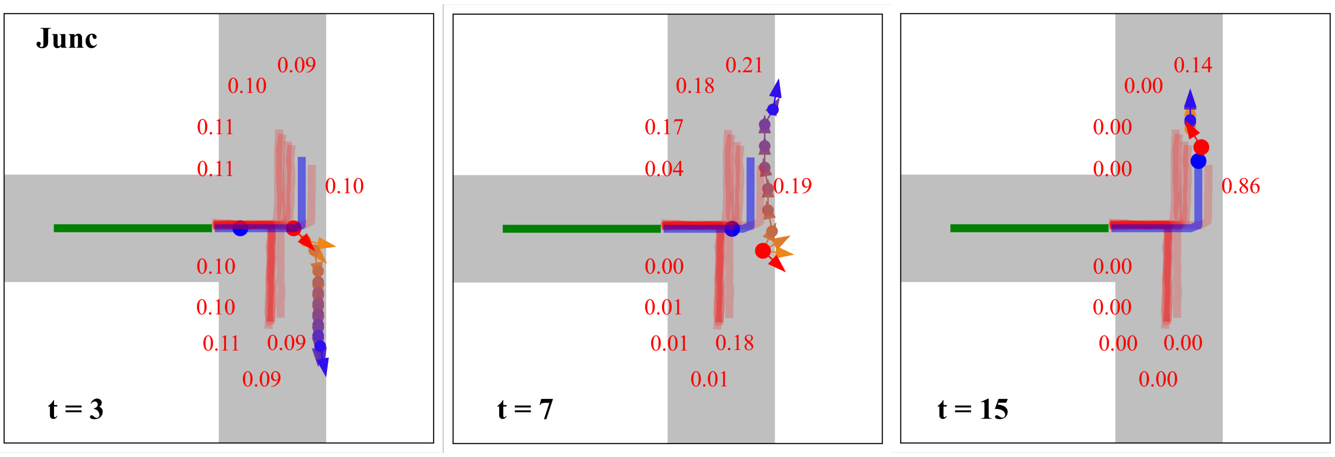


Fig. 3: **Closed-loop Diffusion Planning.** We visualize the normalized importance weight for each sample along the time horizon. The human position at time t is drawn as a blue dot. The importance weight for each sample is shown by the trajectory. Our method can filter out inaccurate prediction samples with posterior probability.

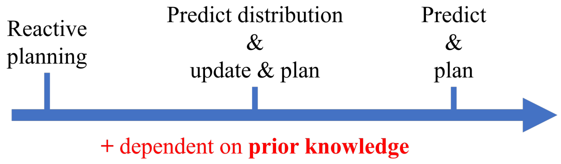

Reactive planning
Predict distribution & update & plan
Predict & plan
+ dependent on prior knowledge

Fig. 4: **Planning Spectrum**. We show the spectrum of planning strategies and their dependency on prior knowledge. One extreme is the reactive planning strategy, which does not rely on prior knowledge. The other extreme is the predict-and-plan scheme that fully relies on prediction accuracy. Our method positions in the middle of this spectrum.

b) Human motion forecasting: We use *Yolo-v8* [17] to detect the actor from the rear camera and use a Kalman Filter [18] to estimate the human 2D pose in 15Hz. Due to limited computational resources, we set the human motion prediction frequency from the diffusion model to 1 Hz, with a 3-second prediction horizon.

c) Robot path planning: We use log-MPPI [19] to solve the finite-horizon optimization problem in Eq. 7 at 40Hz. For each planning, we sample 2000 robot trajectories with standard variance for linear and rotational velocities $\sigma_v = 0.2$ and $\sigma_\omega = 0.1$.

Here, we show qualitative results for human motion prediction and robot planning in different scenarios (Fig. 6) in the real robot experiment, visualized in RViz [13]. We provide samples that the trained diffusion model (**Dif-TR** from **HPS** dataset) provides a complex predicted human motion distribution (red lines). Meanwhile, our **CLDP** algorithm can select the correct branch from a complex predicted human motion distribution with the help of the Bayesian filter. We include additional qualitative results in the supplementary materials.

IV. Limitation

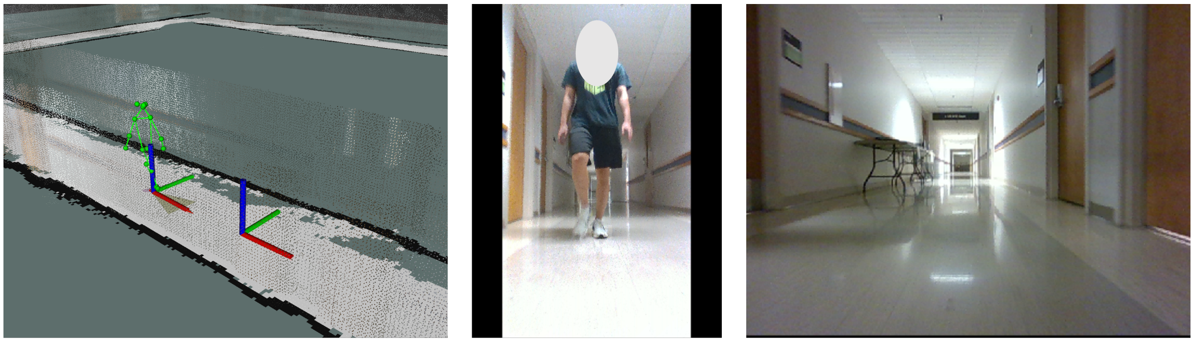
Returning to the assumptions we make throughout this paper, some important ones may limit our work to real-world applications. One strong assumption we make in our formulation (Sec. III.A) is that the motion distribution from the dataset represents the actual human behavior.

A. Human-Robot Interaction

One can argue that having a robot follow in front of a human may influence its motion pattern. For example, humans may switch lanes to avoid collisions with robots. In other words, the future motion should be conditioned on the robot motion as well, as $\hat{\tau}^i_{t_0:} \sim p(\tau_{t_0:}|\tau_{:t_0}, S, \mathcal{X}_{t_0:})$; This will make the interaction between the robot and the human a two-body system and make it more challenging to formulate. In such cases, game-theoretic formulations that optimize for worst-case performance might be more appropriate.

B. Distribution Coverage

Even without considering such interaction, the assumption that the future trajectory sample set can represent the future distribution may not always hold. Some of our failure planning cases occur when our diffusion model fails to provide samples close to the human's ground-truth trajectory, leading the robot to the wrong pose. This is a shared problem for all the prediction-based methods in the community. In such cases, it may make sense to switch to purely reactive behavior. We also want to highlight that, in addition to the amount of human motion data, its diversity and distribution are non-trivial.



(a) SLAM and human localization **(b) Front camera** **(c) Rear camera**

Fig. 5: **SLAM results and camera inputs** (a) We visualize the 3D mapping results as point clouds with an occupancy map, the robot localization result as the frame on the right, and estimated human 3D skeleton pose and 2D pose on the left. (b & c) Corresponding camera readings.

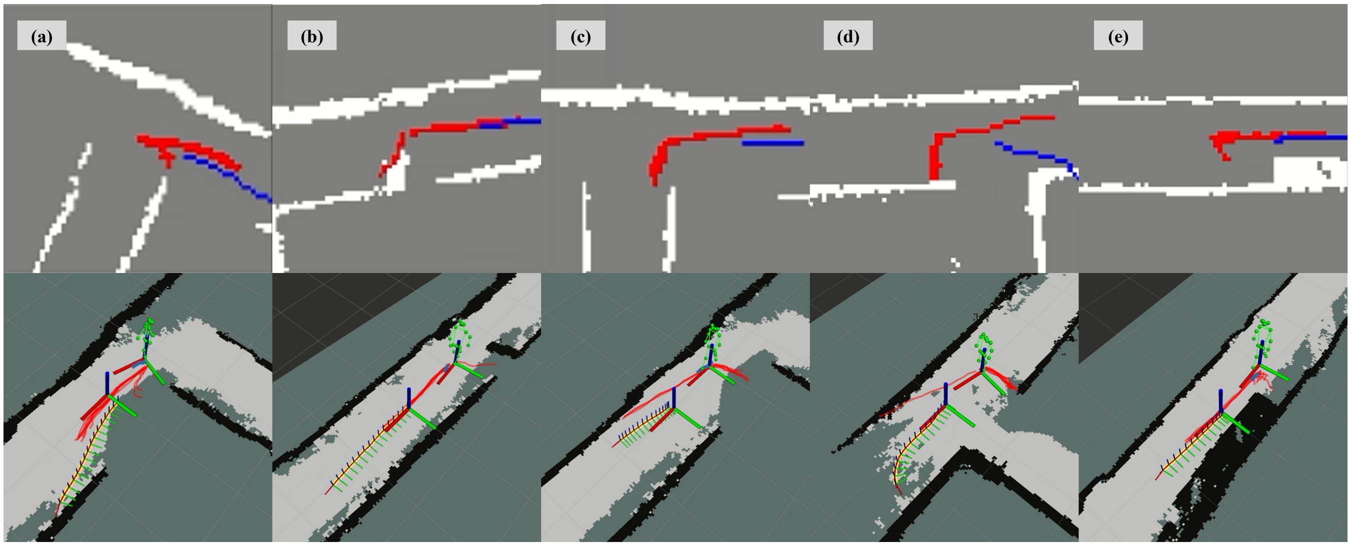


Fig. 6: **Real robot demonstration** We visualize five pairs of human motion prediction in the actor frame (top) and robot **CLDP** planning (bottom) results in 3rd-person view. Human trajectories are plotted as red lines in both figures. The planned robot path is plotted using a blue line in the top image and a yellow line with multiple frames in the bottom image. We also visualize the robot's current frame and the human skeleton pose for each sample.

References

[1] Q. Jiang and V. Isler, "Onboard View Planning of a Flying Camera for High Fidelity 3D Reconstruction of a Moving Actor," Jul. 2023, arXiv:2308.00134 [cs]. [Online]. Available: http://arxiv.org/abs/2308.00134

[2] S. Huang, Z. Wang, P. Li, B. Jia, T. Liu, Y. Zhu, W. Liang, and S.-C. Zhu, "Diffusion-based generation, optimization, and planning in 3d scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 16 750–16 761.

[3] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion Policy: Visuomotor Policy Learning via Action Diffusion," Jun. 2023, arXiv:2303.04137 [cs]. [Online]. Available: http://arxiv.org/abs/2303.04137

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Dec. 2015, arXiv:1512.03385 [cs]. [Online]. Available: http://arxiv.org/abs/1512.03385

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, \. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[6] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 6840–6851. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html

[7] J. Song, C. Meng, and S. Ermon, "Denoising Diffusion Implicit Models," Oct. 2022, arXiv:2010.02502 [cs]. [Online]. Available: http://arxiv.org/abs/2010.02502

[8] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep Unsupervised Learning using Nonequilibrium Thermodynamics," Nov. 2015, arXiv:1503.03585 [cond-mat, q-bio, stat]. [Online]. Available: http://arxiv.org/abs/1503.03585

[9] Q. Jiang, B. Susam, J.-J. Chao, and V. Isler, "Map-Aware Human Pose Prediction for Robot Follow-Ahead," Mar. 2024, arXiv:2403.13294 [cs]. [Online]. Available: http://arxiv.org/abs/2403.13294

[10] M. Mahdavian, P. Nikdel, M. TaherAhmadi, and M. Chen, "STPOTR: Simultaneous Human Trajectory and Pose Prediction Using a Non-Autoregressive Transformer for Robot Following Ahead," Sep. 2022, arXiv:2209.07600 [cs]. [Online]. Available: http://arxiv.org/abs/2209.07600

[11] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1433–1440.

[12] Rover Robotics, “Rover Robotics,” 2023. [Online]. Available: https://roverrobotics.com/

[13] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, and others, “ROS: an open-source Robot Operating System,” in *ICRA workshop on open source software*, vol. 3. Kobe, Japan, 2009, p. 5, issue: 3.2.

[14] M. Labbé and F. Michaud, “RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation,” *Journal of Field Robotics*, vol. 36, no. 2, pp. 416–446, 2019, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21831. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21831

[15] T. Moore and D. Stouch, “A Generalized Extended Kalman Filter Implementation for the Robot Operating System,” in *Proceedings of the 13th International Conference on Intelligent Autonomous Systems (IAS-13)*. Springer, Jul. 2014.

[16] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige, “The Office Marathon: Robust Navigation in an Indoor Office Environment,” in *International Conference on Robotics and Automation*, 2010.

[17] G. Jocher, A. Chaurasia, and J. Qiu, “YOLO by Ultralytics,” Jan. 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[18] G. Welch, G. Bishop, and others, “An introduction to the Kalman filter,” *Chapel Hill, NC, USA*, 1995, publisher: Chapel Hill, NC, USA.

[19] I. S. Mohamed, K. Yin, and L. Liu, “Autonomous Navigation of AGVs in Unknown Cluttered Environments: log-MPPI Control Strategy,” Jul. 2022, arXiv:2203.16599 [cs]. [Online]. Available: http://arxiv.org/abs/2203.16599