2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Unified Control Framework for Real-Time Interception and Obstacle Avoidance of Fast-Moving Objects with Diffusion Variational Autoencoder

Apan Dastider[1], Hao Fang[1] and Mingjie Lin[1]

***Abstract*— Real-time interception of fast-moving objects by robotic arms in dynamic environments poses a formidable challenge due to the need for rapid reaction times, often within milliseconds, amidst dynamic obstacles. This paper introduces a unified control framework to address the above challenge by simultaneously intercepting dynamic objects and avoiding moving obstacles. Central to our approach is using diffusion-based variational autoencoder for motion planning to perform both object interception and obstacle avoidance. We begin by encoding the high-dimensional temporal information from streaming events into a two-dimensional latent manifold, enabling the discrimination between safe and colliding trajectories, culminating in the construction of an offline densely connected trajectory graph. Subsequently, we employ an extended Kalman filter to achieve precise real-time tracking of the moving object. Leveraging a graph-traversing strategy on the established offline dense graph, we generate encoded robotic motor control commands. Finally, we decode these commands to enable real-time motion of robotic motors, ensuring effective obstacle avoidance and high interception accuracy of fast-moving objects. Experimental validation on both computer simulations and autonomous 7-DoF robotic arms demonstrates the efficacy of our proposed framework. Results indicate the capability of the robotic manipulator to navigate around multiple obstacles of varying sizes and shapes while successfully intercepting fast-moving objects thrown from different angles by hand. Complete video demonstrations of our experiments can be found in https://sites.google.com/view/multirobotskill/home.**

***Index Terms*— Dynamic Interception, Obstacle Avoidance, Diffusion Variational Autoencoder, 7-DoF Robotic Arm**

I. Introduction

Intercepting flying objects presents unique challenges distinct from typical robotic reaching tasks extensively studied in controlled laboratory or simulation environments [1]. Specifically, flying objects tend to exhibit high speed and follow nonlinear trajectories [1]–[4]. Additionally, workspaces are often cluttered with obstacles that are constantly moving and dynamically altering their geometric configurations [5]. Consequently, existing methodologies often result in the prohibitively high cost of accommodating any perceptual latency [6]. In other words, executing real-time, high-speed reaching maneuvers with intelligent robots in complex environments laden with dynamic obstacles becomes exceedingly challenging [6].

Numerous robotic motion planning algorithms have emerged in recent years, aiming to enhance the efficacy of intercepting flying objects. Early algorithms such as RRT [7]–[9] were tailored for static objects in relatively simple, obstacle-free environments. However, these approaches and their variants [10]–[13] encounter limitations in more complex obstacle environments. Traditional robotic control algorithms resort to dynamical-system-based methods [14], [15], wherein obstacles are depicted as the workspace constraint, and later the optimal control theory is employed to generate robot commands. Unfortunately, as workspace dimensions increase and constraints become more nonlinear, solutions to optimal control problems gradually become unattainable. Furthermore, extending traditional dynamical-system-based methods to handle dynamical obstacles proves challenging, necessitating the integration of time-varying workspace constraint into the optimal control formulation [16].

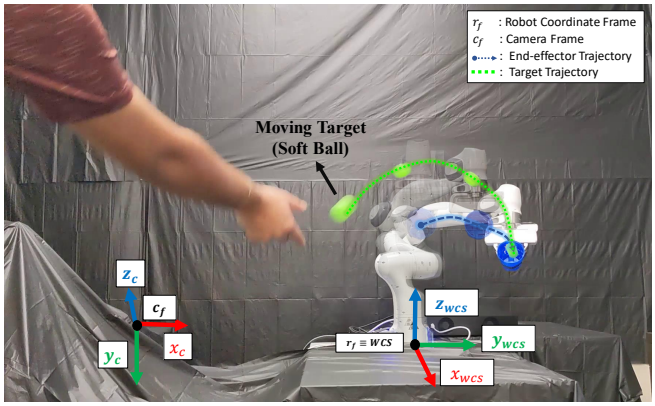

Fig. 1: Interception of a moving object by a 7-DoFs Robotic Manipulator.

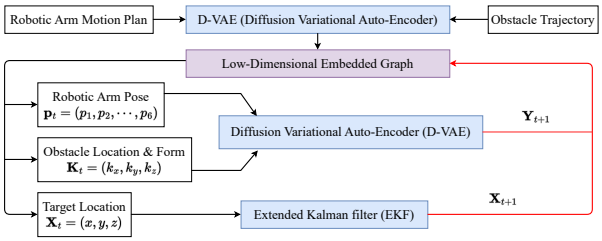

Fig. 2: Block diagram of our proposed unified control framework. More algorithm details can be found in Methods section Fig. 3.

Recent advances in deep learning have facilitated the realization of flying object interception and obstacle avoidance within an end-to-end learning framework [17]–[19]. A noteworthy innovation is the utilization of deep neural networks for dimensionality reduction in the operational space, thereby expediting the synthesis of reaching trajectories. For instance, [20] devised a path-planning algorithm on the latent workspace manifold and introduced an obstacle avoidance scheme by modifying ambient metrics. Motion planning networks (MPNet) [11] employ sampling-based motion planning and obstacle avoidance via learned latent space networks. Recent work DAMON [21] leveraged the variational autoencoder (VAE) structure to learn low-

[1]Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL, 32816, USA (E-mail: apan.dastider@ucf.edu)

dimensional embeddings, upon which motion trajectories were planned. However, relying solely on deep learning techniques may not fully address interception challenges, particularly given the high-speed motion of the target object. Firstly, the nonlinear and unpredictable trajectory of the target object could confound purely deep learning-based algorithms due to distribution shifts between training and test data [22]. Secondly, re-synthesizing robotic commands based on real-time object positions allows for only brief reaction times, potentially insufficient for a single forward process in command synthesis [23]. Consequently, there is a pressing need for more *unified* and *computationally efficient* robotic motion planning algorithms to tackle these challenges effectively.

In this paper, we introduce a unified control framework utilizing diffusion variational autoencoder (D-VAE) for real-time dynamic object interception and collision avoidance (see Figure 2). The significant contributions of our work are outlined as follows:

- Development of a diffusion variational autoencoder (D-VAE) to model the complex high-dimensional state vector with a two-dimensional latent manifold representation. This reveals the underlying low-dimensional dynamics of both the robotic manipulator and environmental obstacles. One unique aspect of our approach is that both robotic arm motion dynamics and moving obstacles are integrated into a unified model.
- Construction of a densely-connected graph network over the learned latent manifold. This network facilitates efficient robotic motion planning by identifying the shortest path routes through graph traversing while ensuring collision-free motion.
- Integration of the extended Kalman filter (EKF) to provide accurate real-time estimation of moving objects within the reachable workspace of the robotic arm.
- Generation of decoded control commands to drive real-time motion of robotic motors. These commands effectively ensure avoidance of moving obstacles and achieve high accuracy in intercepting fast-moving objects.

We validate our proposed method through experiments conducted in both computer simulation environments and real-world scenarios with robotic arms. Our results demonstrate promising potential for the future generalization of robotic motion planning algorithms to effectively handle dynamic moving obstacles while intercepting moving targets in real time.

II. Related Work

A. Robotic motion planning algorithm

Current typical robotic motion planning algorithms include three perspectives: 1) RRT and its variants such as L2RRT and Dynamic RRT* [10], [12]; 2) dynamical-system-based methods [14] 3) neural-network based approaches [19], [24]. For example, [20] developed a path-planning algorithm on the latent workspace manifold and introduced an obstacle avoidance scheme by modifying the ambient metrics. Motion planning networks (MPNet) [11] use sampling-based motion planning and obstacle avoidance through learned latent space networks.

B. Variational autoencoder

Variational autoencoder (VAE) is a powerful neural network architecture in many fields such as generative AI for robotics [25], dimensionality reduction for computation efficient motion planning [21], task conditioned movement leaning [26]. VAEs aim to optimize the network parameters by maximizing the evidence lower bound (ELBO) using the reparameterization trick. In our work, we augmented a VAE with techniques from the diffusion map operation for smooth robotic motion planning in low-dimensional manifold space.

C. Kalman filter

The Kalman filter, named by Rudolf E. Kalman, was developed back in 1961 and later was widely applied in modern society such as robotics [27], time series analysis [28], neural engineering [29], [30]. The basic idea of the Kalman filtering process is to use the linear control dynamical equations and sequences of output measurement to recursively estimate the underlying system states in real time. More recently, many variants of the Kalman filter were also proposed to deal with nonlinear control dynamics such as extended Kalman filter (EKF), unscented Kalman filter, and adaptive Kalman filter [31].

III. Methods

Our method consists of three major components: a) a diffusion variational autoencoder to learn a two-dimensional latent manifold representation of the observed high-dimensional robotic state vector (see section III-A); b) a dense-connected graph network over the learned two-dimensional latent manifold to facilitate robotic motion planning with the shortest path routing (see section III-B); c) an extended Kalman filter (EKF) for the accurate real-time estimation of the moving object in the reachable workspace of the robotic arm (see section III-C). Figure 3 depicts the comprehensive structure of our proposed method.

A. Diffusion Variational Autoencoder

We develop a diffusion variational diffusion autoencoder (D-VAE) to efficiently fuse the advantage of variational autoencoder and diffusion maps for robot motion planning, which learns a two-dimensional latent manifold representation of the observed high-dimensional robotic state vector. Consider a probability space $\mathbb{X}$ and random samples $\{x_i\}_{i=1}^{N}$ from the probability space (N denotes the total number of samples), i.e., each $x_i \in \mathbb{X}$. Once we define Gaussian diffusion kernel $k_d(x_i, x_j)$, the D-VAE automatically encodes geometrical properties of the Euclidean space $\mathbb{X}$ into low-dimensional manifold space $\mathcal{M}_Z := \widetilde{\psi}_\Gamma(\mathbb{X})$.

The diffusion map learner is designed as a deep neural network-based approximator parameterized by Γ to model the diffusion map feature space $\widetilde{\psi}_\Gamma(\cdot)$. Given the similarity affinity function between any x_i and x_j, the loss function of the diffusion map learner is defined by,

$$\mathcal{L}_{\text{diffusion map}} = \mathbb{E}[k_d(x_i, x_j)||z_i - z_j||_2^2] \tag{1}$$

where, $z_i, z_j \in \mathcal{M}_Z$ and Γ denotes the parameters for the mapping, $\mathcal{M}_Z := \widetilde{\psi}_\Gamma(\mathbb{X})$. This loss function ensures that similar points in high-dimensional space are also in close proximity in lower-dimensional latent space.

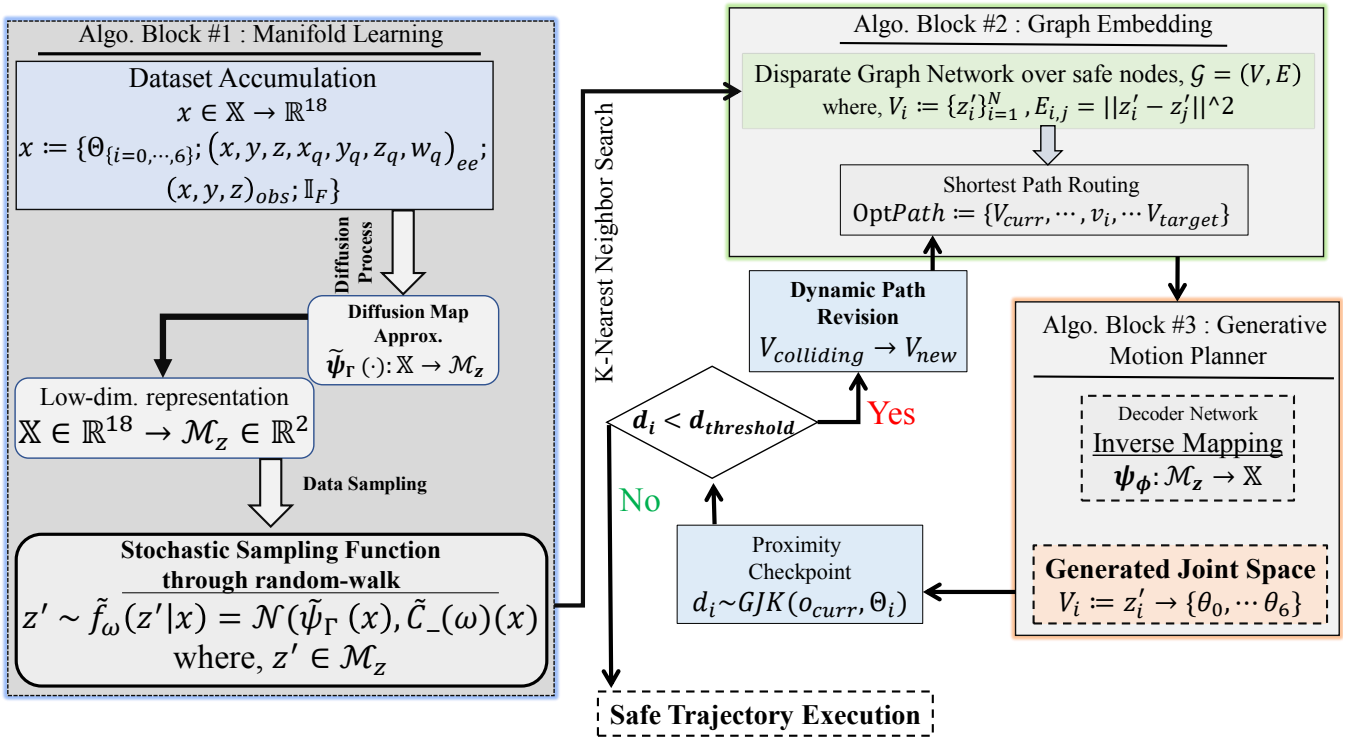

Fig. 3: Overall algorithm blocks of our proposed framework. We first encode the high- dimensional data to a two-dimensional latent manifold using diffusion variational autoencoder (see left Block 1). Then. we construct an offline dense connected trajectory graph (see Block 2). We then leverage a graph-traversing strategy on the constructed offline dense graph to generate the encoded robotic motor control commands (also see Block 2). We last decode the generation of encoded control commands for the real-time motion of robotic motors (see Block 3).

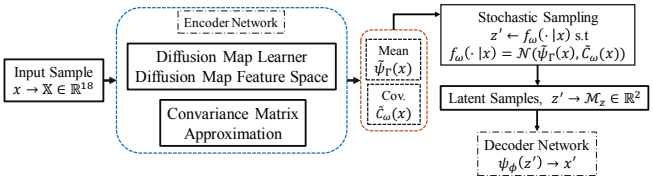

Fig. 4: The neural network architecture of our proposed diffusion variational encoder (D-VAE). The key difference of our proposed D-VAE with traditional VAE is the Encoder part, where we use the diffusion map to learn lower dimensional embedding efficiently.

Secondly, we learn a stochastic sampling function $\tilde{f}_\omega(z'|x) \to z'$ which is defined by a Gaussian normal distribution, $\mathcal{N}(\tilde{\psi}_\Gamma(x), \tilde{C}_\omega(x))$ following [32]. Here, $\tilde{C}_\omega(x)$ defines the covariance matrix of the random walk on manifold space $\mathcal{M}_Z$ for any sample x_i, and is parameterized by ω. The inverse diffusion mapping is defined by $\psi_\phi := \mathcal{M}_z \to \mathbb{X}$ and the learning parameter space for reconstruction is defined by ϕ. Thus, the empirical loss function for D-VAE can be expressed as

$$\mathcal{L}_{\text{D-VAE}} = -D_{KL}(f_\omega(z'|x)||p_\phi(z'|x)) + \mathcal{L}_{\text{decoder}}. \quad (2)$$

where the first term denotes the KL divergence from true diffusion probabilities and the second term defines the reconstruction error. Specifically, the decoder part for the inverse transformation $\psi_\phi := \mathcal{M}_z \to \mathbb{X}$ is from the low-dimensional embedded space $\mathcal{M}_z$ back to the high-dimensional robotic operation space $\mathbb{X}$. To simplistically and effectively train the decoder model, we use the following L2-norm reconstruction error function

$$\mathcal{L}_{\text{decoder}} \approx \arg\min_{\omega,\phi} ||x - \psi_\phi(z')||_2^2 \quad (3)$$

where $z' \sim \tilde{f}_\omega(\cdot|x)$. Using this decoder, we essentially build a bijective mapping, where we can generate robotic motion control commands Θ_i' through the low-dimensional embedding z_i'. Using $\mathcal{L}_{\text{diffusion map}}$ and $\mathcal{L}_{\text{D-VAE}}$ together, we essentially learn a smooth latent manifold representation, where the original distance metric is preserved.

After we construct the architecture of D-VAE, we next apply it to reduce the dimension of our high-dimensional robotic data, which consists of 18 floating numbers. Specifically, each sample is defined by $[\theta_0, \cdots, \theta_6; \{x,y,z,x_q,y_q,z_q,w_q\}_{ee}; \{x,y,z\}_o; \mathbb{I}_F]$, where θ_is determining the exact full joint-space pose of a robotic arm; $\{x,y,z,x_q,y_q,z_q,w_q\}_{ee}$ representing end-effector's coordinates and orientations in quaternion; $\{x,y,z\}_o$ defining the location of a point obstacle; $\mathbb{I}_F$ which is boolean collision flag. Essentially, the above 18 dimensions describe both the complicated robotic dynamics and geometrical properties of environmental obstacles. After sending high-dimensional robotic data into D-VAE, we stored the output low-dimensional embedded data $\{z_i'\}_{i=1}^N \in \mathcal{M}_z$ in two-dimensional space by considering only the first two dominant components [33], which later would be utilized to form a connected network in two-dimensional space for real-time motion planning (see next section).

B. Densely-Connected Graph Network and Shortest Path Routing

Diffusion map-based graph construction and optimal routing through connected graphs for real-world navigation and motion planning has been well established with a view to learning obstacle avoidance [34] and visual navigation [35]. However, such graph-based planning has been largely employed in mobile robots maneuvering on planar Euclidean space. Using the established D-VAE, we achieve the low-dimensional embedding via $\mathbb{X} \rightarrow \mathcal{M}_z$, where we can build graph $\mathcal{G}$ using z_i' in the embedding space $\mathcal{M}_z$ as

$$\mathcal{G} = \text{graph}(z_i')_{i=1}^{N} \in \mathbb{R}^2, \tag{4}$$

which encodes the geometric properties of $\mathbb{R}^3 \times \mathcal{S}^3$ space of a redundant robotic manipulator operating against point obstacles in $\mathbb{R}^3$. Therefore, graph $\mathcal{G} = (V, E)$ consists of nodes $V_{\{i=1,\cdots,N\}} \in \mathcal{M}_z$, which is associated with high-dimensional samples $x_i \in \mathbb{X}$; edge E in the graph can be computed by Euclidean distance between two points in the low-dimensional space. Next, we develop a shortest path routing algorithm on the constructed graph $\mathcal{G} = (V, E)$. Notice that each low-dimensional node is labeled either as "collision" or "collision-free" based on the corresponding indicator $\mathbb{I}_F$. Thus, we apply the nearest neighbor search algorithm (greedy algorithm) on each "collision-free" node to perform the shortest routing algorithm for reaching a desired target node. We also adaptively update the previous routing to avoid any dynamic obstacles appearing in real time.

C. Extended Kalman filter (EKF) for real-time estimation of the moving object

To predict the desired position of the target node, we use the well-known Extended Kalman filter (EKF) for real-time estimation of the moving object. Here, we consider a vector $s \in \mathbb{R}^n$ consisting of n features on each frame captured using the image-based visual servoing system, where features can be determined as pixel coordinates (f_x, f_y) on RGB frames and depth values Z of respective coordinate from aligned depth frames. In detail, we have a bi-directional system integrating a eye-to-hand system and a hand-to-eye system. The interaction are described by the following dynamics, (also see online materials for more detailed derivations)

$$\dot{x} = \begin{bmatrix} L_g(x_1, x_2)^{c_f}\mathbb{T}_{r_f} J_{ee} u - v_o \\ L_{Z_g}(x_1, x_2)(^{c_f}\mathbb{T}_{r_f} J_{ee} u - L_g^{\dagger}(x_1, x_2) v_o) \end{bmatrix} + w_t, \tag{5}$$

where $x(t) = [s(t), z(t)] = [x_1(t), x_2(t)]$ where $s(t) := (f_x, f_y)$ and focal length, λ of the vision sensor is known from the intrinsic properties of the sensor; control input, $u = \dot{\Theta}$, hand-to-eye compatible depth Jacobian, $L_{Z_g} = -L_Z \mathbb{H}_{Rt}$ and $\dot{s}_o = v_o$ the motion of feature calculated empirically from consecutive feature points and frame rate of the sensor. $L_g^{\dagger}$ is the Moore-Penrose pseudo-inverse of the interaction matrix and $w_t \sim \mathcal{N}(0, Q_k)$ is additive Gaussian noise.

Having the above explicit dynamics (5), in the *prediction* phase of state estimation with known $f(x_{k|k}, u_k)$ and its covariance matrix, $P_{k|k}$ at any time-step, k, we first required to linearize and discretize the system dynamics by first-order Taylor series approximation at each period ΔT by following Jacobians [36], [37] and updating the system dynamics and uncertainty matrix:

$$\begin{aligned} \bar{F} &= \tfrac{\delta f(x,u)}{\delta x}|_{x=x_k, u=u_k} \\ x_{k+1|k} &= x_{k|k} + f(x_{k|k}, u_k)\Delta T \\ P_{k+1|k} &= F_k P_{k|k} F_k^T + Q_k \text{ where } F_k \approx \mathbb{I} + \bar{F}\Delta T \end{aligned} \tag{6}$$

In our study, the measurements y_k are image point features of dynamically moving objects in consecutive frames. When new measurements arrive, the *update* phase improves the prediction outcomes of system dynamics through the following computations,

$$\begin{aligned} K_{k+1} &= P_{k+1|k} C_{k+1}^T (C_{k+1} P_{k+1|k} C_{k+1}^T + R_{k+1}) \\ x_{k+1|k+1} &= x_{k+1|k} + K_{k+1}(y_k - C_{k+1} x_{k+1}|k) \\ P_{k+1|k+1} &= (\mathbb{I} - K_{k+1} C_{k+1}) P_{k+1|k} \end{aligned} \tag{7}$$

where, C_k is the measurement sensitivity matrix and K_k is the Kalman gain. Once the filter converges, we can accurately predict the target's future state. Also, the coordinate values in (x, y) plane of c_f can be computed by using the projection model $(f_x, f_y) \rightarrow (X = f_x Z/\lambda, Y = f_y Z/\lambda)$. Therefore, we can in real-time estimate the future 3D coordinates of target location in robot's coordinate frame by ${}^{r_f}\mathbb{T}_{c_f}(x, y, z)_{c_f}$.

To this end, we complete the design of our algorithm. A summary of the proposed algorithm can be found in the algorithm 1 and figure 3.

Algorithm 1: Unified Control Algorithm

Input:
Threshold Obstacle Distance $= \lambda_{obs}$
Threshold Target Pose Change $= \lambda_{obj}$
Initial State $= x_o$
Current Goal State $= x_g$
Connected Graph Network, $G = (V, E)$
Output:
Optimal Control Sequence, Θ^*
Function
Unified Control Algorithm$(x_o, x_g, \lambda_{obs}, \lambda_{obj}, G)$*:*
  `GET :`
  Initial Node on G, $V_o \leftarrow \widetilde{f}_\omega(\cdot|x_o)$
  Current Goal Node on G, $V_g \leftarrow \widetilde{f}_\omega(\cdot|x_g)$
  Initial Path, $P_{opt} \leftarrow GraphTraverse(V_o, V_g)$
  while *not intercepted or* $t \leq t_{allowed}$ **do**
    `CALCULATE:`
    Distance, $d_{obs} \leftarrow GJK(robot, obstacle)$
    Estimated Target Pose, $O_{curr} \leftarrow EKF(\cdot)$
    if $d_{obs} < \lambda_{obs}$ *or* $||O_{curr} - O_{old}||_2 > \lambda_{obj}$ **then**
      `UPDATE:`
      Replanned Path,
      $p'_{opt} \leftarrow GraphTraverse(V_{curr}, V_g^{new})$
    else
      `EXECUTE:`
      Inverse Mapping, $\mathcal{M}_z \rightarrow \mathbb{X} \sim V_i \rightarrow x_i$
      Robot Control , $\Theta_i \rightarrow \{\theta_0, \cdots, \theta_6\}$

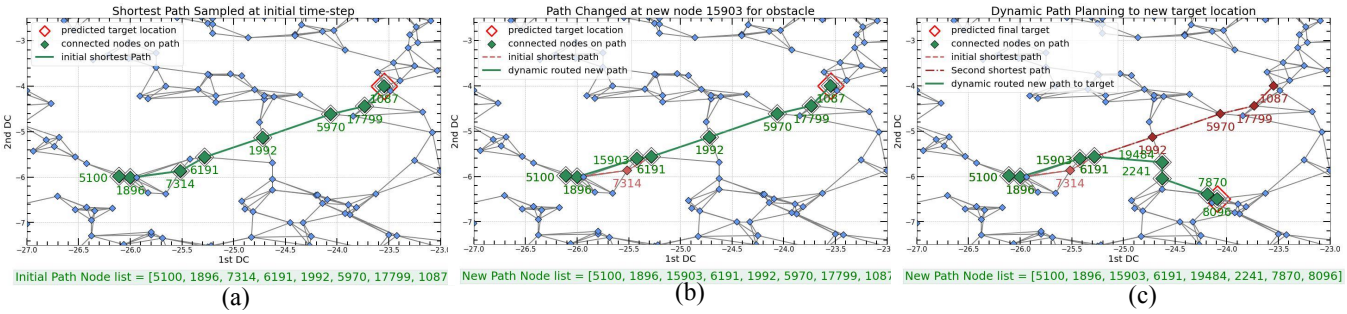

Fig. 5: Dynamic Routing on graph. (a) Initial trajectory (b) New trajectory to avoid obstacles, (c) Dynamic trajectory revising to intercept objects.

D. Evaluation metrics

To effectively evaluate the performance of our proposed method, we define the following evaluation metrics. First, we use the successful ratio to measure the completeness of intercepting flying objects as

$$R_s = \frac{S}{N} \times 100\%, \tag{8}$$

where S counts the successful tasks and N denotes the total number of tasks that we performed. Next, we record the experimental time T to indicate the computational costs for successfully achieving tasks. Last, we use the L_2 distance to quantify the difference between the oracle trajectory of the target moving objects, τ_{oracle} and our prediction trajectory of the target moving objects. τ_{pred} to prove the efficiency of EKF,

$$D = ||\tau_{\text{oracle}} - \tau_{\text{pred}}||_2. \tag{9}$$

IV. Platform Overview

1) Simulation platform

To learn the low-dimensional embedding, we prepare the training dataset using the simulation model of 7-DoFs Franka Emika Panda robotic arm inside the simulation platform of Python Robotics Toolbox (RTB) [38]. Each collected sample has 18 numerical floating values, lying in 18 dimensional space–7 joint angles $\{\theta_i\}_{i=0,..,6}$, 7 pose variables of end-effector containing pose coordinates and quaternion orientations in $\mathbb{R}^3 \times \mathcal{S}^3$ space, location of point obstacle $\{x, y, z\}_o$ and collision flag $\mathbb{I}_F$. Next, we sent uniformly generated random control signals within the safety ranges of joint angles to the simulation model to exclude data bias and ensure enough data variance in preparing training dataset. The obstacle locations are also generated randomly via binary distribution to maximize numbers of "*collision and collision-free*" samples. To have a dense sampling of the manipulator's operation space, we gather in total of 200k samples in our dataset for training the diffusion variational autoencoder. We assumed that by dense sampling, a collision-free trajectory for reaching the target object location is feasible.

2) Hardware platform

For the demonstrations on the real-hardware platform, we utilize the 7-DoF Franka Emika Panda robotic manipulator (see Figure 1). The base link of the robot arm is fixed on the top of the table. Our object tracking and pose estimation algorithm, and adaptive trajectory planning pipeline run on a QUAD GPU server equipped with the Intel Core-i9-9820X processor through straightforward Python implementation. For obstacle detection and localizing dynamic target in $\mathbb{R}^3$ space, we incorporate Intel RealSense Depth Camera D435i and extracted the depth information to obtain coordinates $\{x, y, z\}_{target}$ in a three-dimensional camera coordinate frame. We also track the current obstacle location through depth sensing and extract the geometry of the obstacle with a safety barrier around it. Instantly, we transform this location into the robot's coordinate system to catch the thrown target smoothly and timely. These continuous transformations and feedbacks from depth sensing expedited the optimum routing on the $\mathbb{R}^2$ manifold graph for parallel smooth execution of obstacle avoidance and target reach task. To facilitate low-latency data communication between the actual controller and Franka controller interface, we design the whole system inside the Robot Operating System (ROS) using the Franka Integrated Library – Libfranka and Franka ROS.

V. Results

We use comprehensive experiments in both computer simulations and hardware platforms to evaluate our proposed method. Specifically, our experiments seek to investigate the following questions:

1) Can our unified control framework enable the robotic manipulator to smoothly avoid obstacles and reach a target object location in simulation platforms?
2) How is the flying object intercepting performance in terms of successful rate compared to the state-of-the-art robotic motion planning algorithms?
3) Can our diffusion variational autoencoder learn a lower-dimensional manifold such as $\mathbb{R}^2$ while remaining disparity between the collision-free and collision samples?
4) Can we concurrently localize the fast-flying object through image-based visual servoing on consecutive frames and robustly predict the future position using the proposed extended Kalman filter (EKF)?

A. Intercepting flying objects with obstacle avoidance in both simulation and hardware platforms

We start to answer the first question by running computer simulations for intercepting flying objects under obstacle environments. Fig. 6 gives an example illustration on how the robotic arm tries to capture the dynamic objects while replanning the routes so that it can avoid the potential collision. We observed that the robotic arm was lifted up to change the moving trajectory as it was close to the

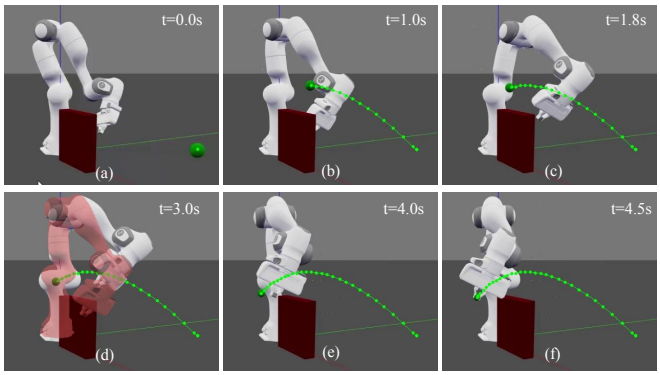

Fig. 6: (a) Initial configuration. (b)-(c) Robot initializes to move to the predicted location, (d) Robot reroutes through a new path over the red obstacles, (in red shaded – collision with obstacles without real-time planning), (e)-(f) Robot catches the thrown ball at the final target location.

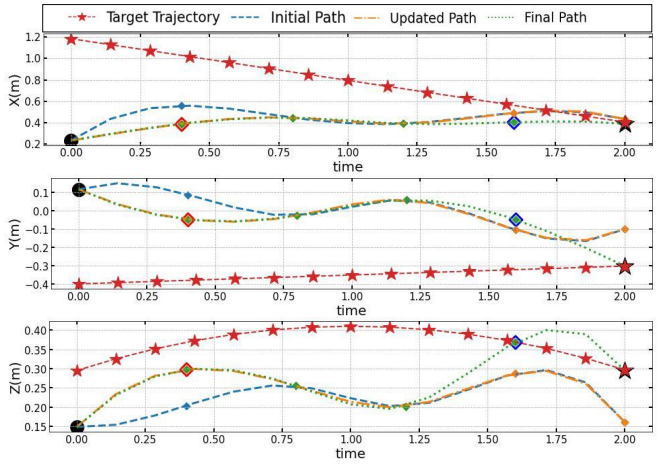

Fig. 7: The robot end-effector trajectory changes for moving obstacle location and updated intercepting location. Here, ◊ represents the trajectory reroute location for obstacle point and ◊ denotes the time-step point where the intermediate path gets upgraded for updated intercepting point. ⋆ denotes the real target location for the thrown target object.

obstacles (Fig. 6 c, d, and e), which ultimately intercept the objects without touching any obstacles (Figure 6 f). Also, we observed that the replanned tracking trajectory is smooth without any rigid movement (Fig. 7). Further, our replanned tracking trajectory shows how the position of the robotic arm dodges the obstacles and reroutes through a new path to catch the moving target in three-dimensional space (see Fig. 7). The above simulation illustration example gives us the first confidence about the potential advantages of our proposed framework. Next, we tested our proposed framework on the hardware platform for intercepting a flying ball (Fig. 1). To emphasis the ability of real-time obstacle avoidance, we introduce the real-time obstacles on purpose during the interception tasks. It turned out that our robotic arm can react to the real-time obstacles by replanning a new collision-free trajectory, which finally intercepted the ball. As a result, both computer simulations and hardware evaluation examples gave us strong evidence about the superior performance of our proposed framework.

B. *Comparison with the state-of-the-art robotic motion planning algorithms*

In this section, we compare our proposed framework with other robotic motion planning algorithms for the task of intercepting flying objects. Here, we focus on the successful ratio of catching flying objects and compare it with our previous proposed method DAMON as well as other methods such as MpNet, L2RRT and RRT variants. We start with simple cases where we ask the robotic arms to track a static target. Since the target is stationary over time, all the planning algorithms only need to avoid the obstacles considering the possible collision during the motion planning process. As a result, all the successful ratios are more than 80%, where our proposed method and our prior work DAMON achieved more than 95%. We argue this superior performance is due to the power of dimensionality reduction using the neural network architecture of variational autoencoder (see next section for more detailed analyses of D-VAE). On the contrary, several RRT variants calculate the trajectory in the original high-dimensional space which increases the planning difficulties and drops in the successful ratio (see Figure 8(a)). Next, we move the tracking performance for flying objects. It was obvious that all the RRT-based methods can not achieve good successful ratios, which dropped below 70%. Our prior work DAMON did the best among the SOTA algorithms, which was almost 80%. However, our proposed method achieved more than 94% successful ratio, which is significantly higher than SOTA algorithms. Compared to the DAMON, we argue the great improvement may be caused by our real-time object estimation mechanisms, i.e., use EKF to calculate nonlinear trajectory prediction, which reduces potential collision in advance. Moreover, D-VAE generates very smooth latent space in $\mathbb{R}^2$ which allows for fast and efficient motion planning computation through obstacle-free closest points in joint-space. In Fig. 8(b), we showed that the computation time for ours proposed method for successful motion planning is the lowest among all other methods. The above experimental results confirmed that our proposed framework can lead to great improvement in tracking both static and dynamic objects while avoiding obstacles in real time.

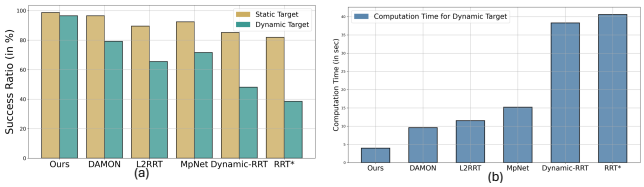

Fig. 8: (a) Successful ratio comparison between different state-of-the-art robotic motion planning algorithms for both tracking static objects and flying objects. The results for static objects are in green and the flying objects are in blue. (b) Computation time comparison for dynamic target interception.

C. *Two-Dimensional embedding using D-VAE and full embedding graph traversal*

The prior section argues the improvement of the successful task completion is partly due to the power of D-VAE for dimensionality reduction and smooth latent space. Especially, we can easily build the graph in lower dimensions and use graph traversal algorithms for trajectory planning with lower computational cost. From high-dimensional data samples of 18 numeric values, we use D-AVE for creating the representation of embedding manifold in $\mathbb{R}^2$ space. Therefore, we run the first set of ablation studies by comparing different

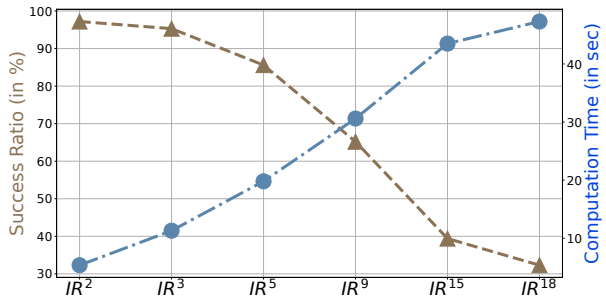

Fig. 9: Comparison of the successful ratio (light brown) and computation time (blue) using different embedding dimensions.

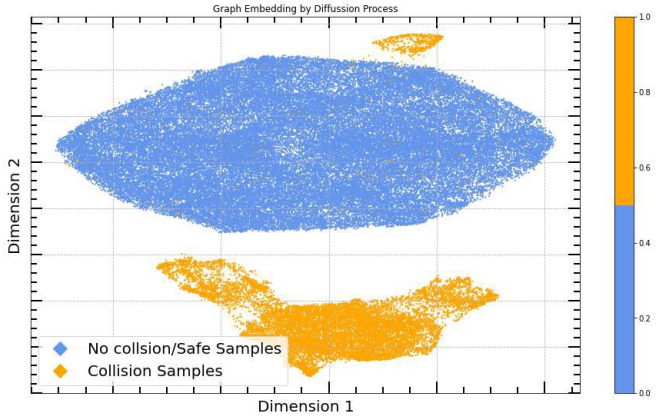

Fig. 10: Low dimensional embedding of "collision-free" samples (in blue color) or "colliding" samples (in orange color).

embedding dimensions of our proposed algorithms. Fig. 9 shows that our methods achieved higher successful ratio with lower computational time for graph traversal in $\mathbb{R}^2$ space compared to the case in which it operated in $\mathbb{R}^{18}$ space. (Successful ratio: $\mathbb{R}^2$ 97.23% v.s. $\mathbb{R}^{18}$ 32.38% and Computational time: $\mathbb{R}^2$ 5.36s v.s. $\mathbb{R}^{18}$ 47.61s). Further, we consistently observed that the lower dimension ensures less computational time and higher successful ratio across a wide range of embedding dimensions. The above study confirmed that using D-AVE can improve the successful ratio and greatly decrease the computational cost.

In detail, our D-VAE can effectively segregate between two binary labels of either "collision-free" samples or "colliding" samples, which greatly capture the geometric properties of the interaction of robotic systems and environment obstacles (see Fig. 10). To visualize the trajectory generation process when our original path anticipates a potential obstacle, we provide the following illustration example in Fig. 5. We observed our methods automatically avoid the obstacle with almost the minimum change of the original path, which partially explains the lower computational time (Fig. 5). The smooth latent space allows our method to adaptively reroute through the closest nodes in graph to reach new target location while avoiding any possible collisions.

D. Real-time object estimation using extended Kalman filter

Having established the power of D-VAE, we lastly illustrate the importance of the extended Kalman filter (EKF). Using our proposed EKF for the real-time estimation of the

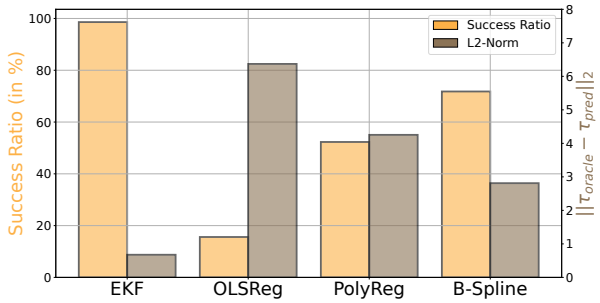

Fig. 11: Successful ratio using different estimation methods

flying objects (see section III-C for details), we realize a precise position estimation of the objects. With new target locations, the target node in the graph is also updated for better interception. We observed that without EKF, the trajectory tracking deviation gradually accumulated, which finally led to a huge deviation and possibly caused the failure of intercepting the flying target objects. On the contrary, the EKF can modify the tracking trajectory by executing the real-time estimation of target objects, which successfully tracks the oracle moving trajectory with smallest tracking errors. To further confirm our observations, we run comprehensive experiments and also compare our EKF with other linear and polynomial estimation algorithms such as linear/polynomial regression, and nonlinear estimation algorithms such as the B-spline algorithm. First, we observed that using static estimation algorithms causes large estimation errors as they only use current position information to predict. On the contrary, the EKF has nonlinear and recursive mechanisms, which utilize both current and past position information for the prediction of future object locations. In detail, our EKF achieved cumulatively 0.67 estimation errors, which outperformed other algorithms OLSReg (Ordinary Least Squares Linear Regression): 6.37; PloyReg (Polynomial Regression): 4.25; B-Spline: 2.81. Therefore, by integrating EKF, our method achieved the higher success ratio 98.6%, which significantly outperformed other algorithms OLSReg: 15.6%; PloyReg: 52.3%; B-Spline: 71.8%. The above ablation study confirmed that the real-time nonlinear and recursive estimation of the target moving object is necessary for improving the final successful ratio.

VI. Discussion and Conclusion

In our work, we propose a unified control framework using diffusion variational autoencoder (D-VAE) for intercepting flying objects with dynamic obstacles. The key advantage of using D-VAE is the feasibility of dimensionality reduction from the original 18 dimensional space to a two-dimensional manifold, which learns lower dimensional embedding and efficiently reduces the redundant robotic manipulator parameters. On the two-dimensional embedding, we construct a dense-connected graph and develop a path routing algorithm, which enables the fast synthesis of control commands in real time. Future work can incorporate more advanced graph traversing algorithms for improving the path routing efficiency with minimum energy cost requirement [39]. Our work borrows the EKF for the real-time position prediction of target objects. Using more advanced Kalman filters

from the adaptive dynamical system approaches is another future direction [40]. We demonstrate the effectiveness of our proposed algorithms on both computer simulations and autonomous 7-DoF robotic arms. Our results hold promise for the future generalization of robotic motion planning algorithms to handle dynamic moving obstacles while intercepting a moving target in real time. Giving the generalizablity of our proposed framework, we envision future validation on more complicated robotic systems such as dexterous in-hand manipulation [41] and robotic dogs [42].

REFERENCES

[1] M. Q. Mohammed, S. C. Chua, and L. C. Kwek, "Comprehensive review on reaching and grasping of objects in robotics," *Robotica*, vol. 39, pp. 1849 – 1882, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:234067726

[2] A. Dastider, H. Fang, and M. Lin, "Retro: Reactive trajectory optimization for real-time robot motion planning in dynamic environments," *arXiv preprint arXiv:2310.01738*, 2023.

[3] S. Kim, A. Shukla, and A. Billard, "Catching objects in flight," *IEEE Transactions on Robotics*, vol. 30, no. 5, pp. 1049–1065, 2014.

[4] N. Marturi *et al.*, "Dynamic grasp and trajectory planning for moving objects," *Autonomous Robots*, vol. 43, no. 5, p. 1241–1256, Aug. 2018. [Online]. Available: http://dx.doi.org/10.1007/s10514-018-9799-1

[5] J. R. Sánchez-Ibáñez *et al.*, "Path planning for autonomous mobile robots: A review," *Sensors*, vol. 21, no. 23, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/23/7898

[6] D. Falanga, S. Kim, and D. Scaramuzza, "How fast is too fast? the role of perception latency in high-speed sense and avoid," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1884–1891, 2019.

[7] S. M. LaValle, "Rapidly-exploring random trees : a new tool for path planning," *The annual research report*, 1998. [Online]. Available: https://api.semanticscholar.org/CorpusID:14744621

[8] O. Salzman and D. Halperin, "Asymptotically near-optimal rrt for fast, high-quality motion planning," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 473–483, 2016.

[9] C. Zito *et al.*, "Two-level rrt planning for robotic push manipulation," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 678–685.

[10] B. Ichter and M. Pavone, "Robot motion planning in learned latent spaces," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2407–2414, 2019.

[11] A. H. Qureshi *et al.*, "Motion planning networks: Bridging the gap between learning-based and classical motion planners," *IEEE Transactions on Robotics*, vol. 37, no. 1, pp. 48–66, 2021.

[12] O. Adiyatov and H. A. Varol, "A novel rrt*-based algorithm for motion planning in dynamic environments," in *2017 IEEE International Conference on Mechatronics and Automation*, 2017, pp. 1416–1421.

[13] K. Wei and B. Ren, "A method on dynamic path planning for robotic manipulator autonomous obstacle avoidance based on an improved rrt algorithm," *Sensors*, vol. 18, no. 2, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/2/571

[14] S. M. Khansari-Zadeh and A. Billard, "A dynamical system approach to realtime obstacle avoidance," *Autonomous Robots*, vol. 32, no. 4, pp. 433–454, 2012.

[15] S. Stavridis, D. Papageorgiou, and Z. Doulgeri, "Dynamical system based robotic motion generation with obstacle avoidance," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 712–718, 2017.

[16] R. Raveendran, A. D. Mahindrakar, and U. Vaidya, "Dynamical system approach for time-varying constrained convex optimization problems," *IEEE Transactions on Automatic Control*, 2023.

[17] A. Amiranashvili *et al.*, "Motion perception in reinforcement learning with dynamic objects," in *Proceedings of The 2nd Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, A. Billard, A. Dragan, J. Peters, and J. Morimoto, Eds., vol. 87. PMLR, 29–31 Oct 2018, pp. 156–168. [Online]. Available: https://proceedings.mlr.press/v87/amiranashvili18a.html

[18] D. Hafner *et al.*, "Learning latent dynamics for planning from pixels," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 2555–2565. [Online]. Available: https://proceedings.mlr.press/v97/hafner19a.html

[19] C. Zhou, B. Huang, and P. Fränti, "A review of motion planning algorithms for intelligent robots," *Journal of Intelligent Manufacturing*, vol. 33, no. 2, p. 387–424, Nov. 2021. [Online]. Available: http://dx.doi.org/10.1007/s10845-021-01867-z

[20] H. B. Mohammadi *et al.*, "Learning riemannian manifolds for geodesic motion skills," in *Robotics: Science and Systems*, 2021. [Online]. Available: https://doi.org/10.15607/RSS.2021.XVII.082

[21] A. Dastider and M. Lin, "Damon: Dynamic amorphous obstacle navigation using topological manifold learning and variational autoencoding," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 251–258.

[22] Q. Wu *et al.*, "Handling distribution shifts on graphs: An invariance perspective," *arXiv preprint arXiv:2202.02466*, 2022.

[23] D. Carneiro, F. Silva, and P. Georgieva, "Robot anticipation learning system for ball catching," *Robotics*, vol. 10, no. 4, 2021. [Online]. Available: https://www.mdpi.com/2218-6581/10/4/113

[24] L. Dong *et al.*, "A review of mobile robot motion planning methods: from classical motion planning workflows to reinforcement learning-based architectures," *Journal of Systems Engineering and Electronics*, vol. 34, no. 2, pp. 439–459, 2023.

[25] B. Ivanovic *et al.*, "Multimodal deep generative models for trajectory prediction: A conditional variational autoencoder approach," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 295–302, 2021.

[26] M. Noseworthy *et al.*, "Task-conditioned variational autoencoders for learning movement primitives," in *Proceedings of the Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, L. P. Kaelbling, D. Kragic, and K. Sugiura, Eds., vol. 100. PMLR, 30 Oct–01 Nov 2020, pp. 933–944. [Online]. Available: https://proceedings.mlr.press/v100/noseworthy20a.html

[27] C. Urrea and R. Agramonte, "Kalman filter: Historical overview and review of its use in robotics 60 years after its creation," *Journal of Sensors*, vol. 2021, p. 1–21, Sep. 2021.

[28] X. Li *et al.*, "Applications of nbsp;kalman filtering in nbsp;time series prediction," in *Intelligent Robotics and Applications: 15th International Conference, ICIRA 2022, Harbin, China, August 1–3, 2022, Proceedings, Part III*. Berlin, Heidelberg: Springer-Verlag, 2022, p. 520–531. [Online]. Available: https://doi.org/10.1007/978-3-031-13835-5_47

[29] H. Fang and Y. Yang, "Predictive neuromodulation of cingulo-frontal neural dynamics in major depressive disorder using a brain-computer interface system: A simulation study," *Frontiers in Computational Neuroscience*, vol. 17, p. 1119685, 2023.

[30] H. Fang *et al.*, "Robust adaptive deep brain stimulation control of non-stationary cortex-basal ganglia-thalamus network models in parkinson's disease," *bioRxiv*, pp. 2023–08, 2023.

[31] M. Khodarahmi and V. Maihami, "A review on kalman filter models," *Archives of Computational Methods in Engineering*, vol. 30, no. 1, p. 727–747, Oct. 2022. [Online]. Available: http://dx.doi.org/10.1007/s11831-022-09815-7

[32] H. Li *et al.*, "Variational diffusion autoencoders with random walk sampling," in *ECCV*, 2020.

[33] J. De la Porte *et al.*, "An introduction to diffusion maps," in *Proceedings of the 19th symposium of the pattern recognition association of South Africa (PRASA 2008), Cape Town, South Africa*, 2008, pp. 15–25.

[34] S. Hong, J. Lu, and D. P. Filev, "Dynamic diffusion maps-based path planning for real-time collision avoidance of mobile robots," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 2224–2229.

[35] O. Kupervasser *et al.*, "Using diffusion map for visual navigation of a ground robot," *Mathematics*, vol. 8, no. 12, 2020.

[36] A. A. Oliva *et al.*, "Towards dynamic visual servoing for interaction control and moving targets," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 150–156.

[37] S. J. Julier and J. K. Uhlmann, "New extension of the kalman filter to nonlinear systems," in *Defense, Security, and Sensing*, 1997.

[38] P. Corke and J. Haviland, "Not your grandmother's toolbox – the robotics toolbox reinvented for python," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 11 357–11 363.

[39] Y. Zhang *et al.*, "Fsgraph: fast and scalable implementation of graph traversal on gpus," *CCF Transactions on High Performance Computing*, vol. 5, pp. 277 – 291, 2023.

[40] H. Fang and Y. Yang, "Designing and validating a robust adaptive neuromodulation algorithm for closed-loop control of brain states," *Journal of Neural Engineering*, vol. 19, no. 3, p. 036018, 2022.

[41] O. M. Andrychowicz *et al.*, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020. [Online]. Available: https://doi.org/10.1177/0278364919887447

[42] Y. Zhao *et al.*, "Intelligent control of multilegged robot smooth motion: A review," *IEEE Access*, vol. 11, pp. 86 645–86 685, 2023.