석 사 학 위 논 문
Master's Thesis

로봇팔의 경로 추적 문제에 대한 최적화 기법의 학습 기반 초기화

Learning-based Initialization of Trajectory Optimization for Redundant Manipulators' Path-following Problem

2022

윤 민 성 (尹 旻 星 Yoon, Minsung)

한 국 과 학 기 술 원

Korea Advanced Institute of Science and Technology

석 사 학 위 논 문

로봇팔의 경로 추적 문제에 대한 최적화 기법의 학습 기반 초기화

2022

윤 민 성

한 국 과 학 기 술 원

전산학부

로봇팔의 경로 추적 문제에 대한 최적화 기법의 학습 기반 초기화

윤 민 성

위 논문은 한국과학기술원 석사학위논문으로
학위논문 심사위원회의 심사를 통과하였음

2021년 12월 7일

심사위원장 윤 성 의 (인)

심 사 위 원 김 기 응 (인)

심 사 위 원 박 대 형 (인)

Learning-based Initialization of Trajectory Optimization for Redundant Manipulators' Path-following Problem

Minsung Yoon

Advisor: Sung-Eui Yoon

A dissertation submitted to the faculty of
Korea Advanced Institute of Science and Technology in
partial fulfillment of the requirements for the degree of
Master of Science in Computer Science

Daejeon, Korea
December 7, 2021

Approved by

Sung-Eui Yoon
Professor of School of Computing

The study was conducted in accordance with Code of Research Ethics[1].

[1] Declaration of Ethical Conduct in Research: I, as a graduate student of Korea Advanced Institute of Science and Technology, hereby declare that I have not committed any act that may damage the credibility of my research. This includes, but is not limited to, falsification, thesis written by someone else, distortion of research findings, and plagiarism. I confirm that my thesis contains honest conclusions based on my own careful research under the guidance of my advisor.

MCS 윤민성. 로봇팔의 경로 추적 문제에 대한 최적화 기법의 학습 기반 초기화. 전산학부 . 2022년. 19+iii 쪽. 지도교수: 윤성의. (영문 논문)
Minsung Yoon. Learning-based Initialization of Trajectory Optimization for Redundant Manipulators' Path-following Problem. School of Computing . 2022. 19+iii pages. Advisor: Sung-Eui Yoon. (Text in English)

초 록

본 논문은 운동학적으로 7 자유도 이상의 로봇 팔에 대한 3차원 공간상의 경로 추적 문제를 해결하는 것을 목표로 한다. 궤적 최적화 기법은 경로 추적 목표와 함께 운동학적 제약 조건 및 충돌 회피 조건을 충족하는 조인트 공간에 궤적을 최적화 기법을 통하여 생성하는 방법이다. 하지만, 경로 추적의 목표는 매 순간에 대해 많은 제약 조건이 부과되기에 초기 궤적에 따른 최적화의 결과가 극소값에 빠지기 쉽고, 이를 회피하기 위해서는 시간이 많이 걸리는 여러 번의 재시작이 필요하다. 이 문제를 개선하기 위해, 본 논문에서는 심층 강화 학습기법의 사용하여 학습된 신경망을 통하여 여러 제약조건들이 고려된 초기 궤적을 생성함으로써 최적화 기법의 성능을 향상시키는 방식을 제안한다. 신경망 학습을 위해 경로 추적, 충돌 회피 및 제약조건들에 대한 보상함수를 구성하였고, 추가적으로 영공간 상에 최적화를 위해 최적화된 궤적들로부터 모방 학습을 활용하였다. 본 프레임워크를 두 개의 대표적인 최적화 기법에 적용하여 수렴 결과에 최적성, 계산 효율성 및 다양한 경로에 대한 강인함의 향상을 확인하였다. 마지막으로 실제 로봇에 적용하여 최적화된 궤적에 효용성을 검증하였다.

핵 심 낱 말 모션 계획, 심층 강화 학습, 로봇 공학

Abstract

We aim to solve the problem of path following for kinematically redundant manipulators over SE(3). Trajectory optimization (TO) is a solution to generate a joint-space trajectory while satisfying physical constraints along with the path-following objective. Unfortunately, as many constraints are imposed over the objective, the optimization is prone to fall into local minima and requires time-consuming re-starts. To ameliorate this problem, we propose a learning-based initial-trajectory generation method that returns joint-space trajectories as good initial guesses for TO. Our method learns the kinematically feasible null-space motions following a target path over a multi-task reinforcement learning framework with demonstration guidance. We evaluate the proposed method and three baseline initial trajectory generation methods plugged into two representative TO frameworks. We show that our method boosts the performance of the optimization methods in terms of optimality, computational efficiency, and robustness. Finally, we verify the optimized trajectory quality using our initialization method by executing it on a real Fetch robot and show a better accurate and smooth tracking performance.

Keywords Motion Planning, Deep Reinforcement Learning, Robotics, Integrated Planning and Learning

Contents

List of Tables

List of Figures

Chapter 1. Introduction

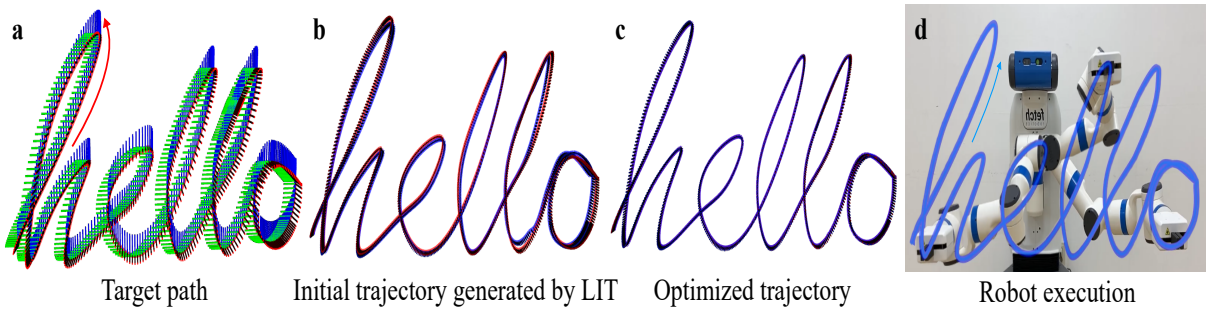


Figure 1.1: An exemplar path-following problem, the 'hello' word, of a kinematically redundant manipulator, Fetch from Fetch Robotics. (a) Given a target path (i.e., 6-D pose list), (b) our learning-based initial trajectory (LIT) generator quickly synthesizes a good initial trajectory. (c) Plugged into the trajectory optimizer, (d) the real robot smoothly follows the path using the optimized trajectory without violating any kinematic constraints. The colors of each figure are as follows. (a) Red, Green, and Blue: x-, y-, and z-axis for each pose, respectively. (b, c) Red-line: target paths, Blue-line: end-effector paths from the generated joint trajectories. (d) Blue-line: a tracked end-effector path of the optimized joint trajectory.

Accurate and fast path following of manipulators is an important issue for real-world tasks in manufacturing or other domains. Given a fully constrained (i.e., 6-dimensional) pose path, we aim to find a configuration-space trajectory for kinematically redundant manipulators (see Fig. 1.1) taking into consideration a variety of trajectory constraints, such as joint continuity, smoothness as well as potential collision in the environment. Although the redundant manipulators can provide a flexibility of solutions, the solution is generally not unique and cannot be represented as a closed-form.

Conventional methods, such as KDL [1] and Trac-IK [2], often use differential inverse kinematics (IK) to iteratively find a sequence of IK solutions maximizing the pose-matching objective. However, the local nature of solutions cannot explicitly consider the constraints over the path, though we can assign the constraint-related behavior in the null-space at each time step [3]. Instead, a recent solver, RelaxedIK [4], adopts a relaxed tracking objective by formulating an optimization problem that finds the closest one satisfying constraints. However, the method still suffers from its myopic solution that may result in no solution or local-minimum paths at the end. Alternatively, researchers introduce a global-search method that constructs a discrete layered graph with IK solutions along the path and finds the most kinematically feasible trajectory [5, 6, 7, 8]. The search is asymptotically optimal but computationally expensive due to the infinite number of IK solutions given an end-effector pose.

Recently, trajectory optimization (TO) has been widely adopted for generating a kinematically feasible trajectory by appending constraints along an end-effector path [9, 10, 11, 12, 13, 14]. In general, the non-convex optimization is likely to get stuck in local minima, depending on the quality of the initial guess, i.e., initial trajectory. Conventional initialization approaches have used 1) a linearly interpolated trajectory in configuration space [10, 11] or 2) a trajectory by greedily selecting IK solutions maximally satisfying the constraints [9, 14] considering a limited time budget. However, optimization results based on these are still prone to be sub-optimal due to the non-convex optimization landscape, requiring time-

costly restarts to avoid local minima. Thus, we need an efficient estimator to synthesize a good initial trajectory balancing generation time and satisfaction of constraints.

This work introduces a learning-based initial trajectory (LIT) generator that finds a warm-starting trajectory, as an initial guess of TO. Alongside the fast inference properties of neural networks, the core idea behind the method is that the kinematically feasible null-space motions from demonstrations can be informative for finding a global minimum in the manifold of TO solution space. To learn such motions, we formulate the trajectory generation problem as a finite-horizon Markov decision process (MDP) by defining a unified reward function composed of task, imitation, and constraint-relevant rewards. We then train the LIT generator with a variety of path-following problems with demonstrations by adopting multi-task reinforcement learning (RL) [15]. Our null-space imitation reward helps encourages the agent to quickly search kinematically feasible motion while resolving the conflict with the path-following task reward, raising the success rate.

We first demonstrate how we guide LIT to produce low-cost initial trajectories given complex path-following problems and evaluate its performance against the other three baseline methods in $5,000$ specific and $11,000$ randomized problems. We then show LIT can help to find better optimal solutions with minimal optimization time and minimal constraint-violation rate when plugging it to a state-of-the-art TO method, trajectory optimization of a redundant manipulator (TORM) [14]. We also show the generalization performance of LIT with another TO method, Trajectory Optimization for Motion Planning (TrajOpt) [10] and various environments. Finally, we verify the trajectory optimized by each initialization method on the real robot, Fetch from Fetch Robotics, and confirm that the trajectory optimized with the LIT initialization creates a more accurate and smooth tracking performance.

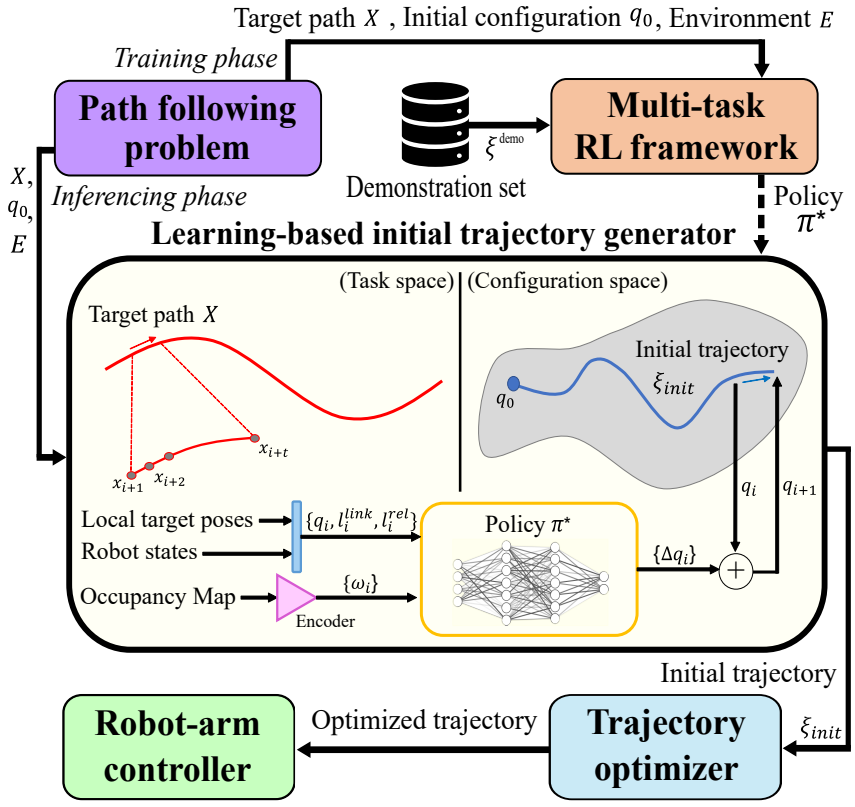


Figure 1.2: Overall framework of the proposed LIT method, which infers an initial guess (i.e., trajectory) for a path-following optimization problem.

Chapter 2. Trajectory Optimization for Path Following

Trajectory optimization, particularly path following for serial manipulators, is to transform an end-effector path into a joint path that satisfies various kinematic constraints such as joint continuity, smoothness, mechanical limit, collision, etc. Researchers often represent a pose path as a sequence of poses $X = [x_0, x_1, ..., x_{N-1}] \in \mathcal{X}$ evenly spaced in time, where N is the number of points in the path, $\mathcal{X}$ is the space of pose paths, and each pose is a pair of position ($\in \mathbb{R}^3$) and orientation ($\in SO(3)$). Likewise, the joint trajectory is a sequence of joint angles $\xi = [q_0, q_1, ..., q_{N-1}] \in \Xi$, where $q \in \mathbb{R}^d$ is a configuration of a d-degree-of-freedom (DoF) manipulator and Ξ is the Hilbert space of joint trajectories [10, 11, 14]. Note that d is greater than 6 in the case of a redundant manipulator in $SE(3)$.

There can be a number of kinematically feasible solution trajectories. To select the best one, TO requires setting an objective functional (i.e., cost function) $\mathcal{U} \in \mathbb{R}^+$, which often consists of multiple sub-objectives. For example, a state-of-the-art TO method, TORM, uses a unified objective functional for the path-following problem:

$$\mathcal{U}[\xi] = \mathcal{U}_{pose}[\xi] + \lambda_1 \mathcal{U}_{obs}[\xi] + \lambda_2 \mathcal{U}_{smooth}[\xi], \tag{2.1}$$

where $\mathcal{U}_{pose}$, $\mathcal{U}_{obs}$, and $\mathcal{U}_{smooth}$ are a pose error[1], an obstacle cost, and a joint smoothness along the trajectory ξ, respectively. λ_1, and λ_2 are constants. Further, the method uses many kinematics constraints such as joint position and velocity limits as well as constraints of singularity or collision avoidance. For more details, We refer the readers to [14].

TO algorithms are often sensitive to the quality of the initial trajectory ξ_{init} for the path following problem. Considering the objective functional $\mathcal{U}[\xi]$ represents the quality of the trajectory, we can design the problem of the trajectory initialization for TO as the minimization of $\mathcal{U}[\xi_{init}]$.

[1] In calculating the pose error, we use a weight of 0.17 for the rotational distance over the translational distance, used in [7, 14].

Chapter 3. Learning-Based Trajectory Initialization

We introduce a learning-based initial trajectory generation method, LIT, that finds a high-quality initial trajectory for TO via training a multi-task RL-based policy.

3.1 Problem Formulation

We first formulate an MDP $\mathcal{M}_X = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}_\mathcal{Z}, \mathcal{T}, \gamma \rangle_{X \sim \mathcal{P}(\mathcal{X})}$ given a sampled target path X from a distribution $\mathcal{P}(\mathcal{X})$ of target pose paths, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is a set of actions, $\mathcal{R}_\mathcal{Z} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a time-varying reward function, where $\mathcal{Z} = \{z \in \mathbb{Z} | 0 \leq z \leq N-1\}$ is a set of indices, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is a deterministic transition function, and $\gamma \in [0, 1)$ is a discount factor. The policy trained on the MDP $\mathcal{M}_X$ synthesizes a nearly constraints-satisfying trajectory ξ in the configuration space following the target path X. (see Fig. 1.2) On the other hand, since the manipulator has a restricted operation range in the task space with the arm's length, the target paths X from the distribution $\mathcal{P}(\mathcal{X})$ are within the same geometric space. To share the similarities between MDPs and obtain a unified policy, we define a multi-task reinforcement learning from the target path distribution $\mathcal{P}(\mathcal{X})$:

$$\underset{\pi}{\text{maximize}} \quad \mathbb{E}_{\substack{a_i \sim \pi(a_i|s_i) \\ \mathcal{M}_X \sim \mathcal{P}(\mathcal{M}_X|\mathcal{X})}} \left[\sum_{i=0}^{N-1} \gamma^i \mathcal{R}_i(s_i, a_i) \,\middle|\, \mu_0 \right], \tag{3.1}$$

where μ_0 is the initial configuration distribution at the initial pose x_0. Then, the objective is to find an optimal policy π^*, where $\pi^* : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_+$ maximizing the expected returns.

3.2 Scene-context based State and Action

We define a scene-context based state s_i that is a tuple, $s_i = (q_i, l_i^{link}, l_i^{rel}, \omega_i)$, at a time step i: $l_i^{link} \in \mathbb{R}^{(d+1,9)}$ is a list of arm links' frame poses including the gripper link. Note that here we represent each pose as a combination of a position vector ($\in \mathbb{R}^3$) and an orientation vector ($\in \mathbb{R}^6$) to enhance the learning performance of the neural network with continuous state-space representation [16]. $l_i^{rel} \in \mathbb{R}^{(t,9)}$ is a list of relative poses from the current end-effector pose to each target pose from step $i+1$ to step $i+t$. Here, t is a future time step to take far-sighted action to avoid local-minima.

$\omega \in \mathbb{R}^{dim(z_{vae})}$ is the scene context vector encoded from an occupancy map, which is constructed with the point clouds of the robot itself and the obstacles of the environment. This vector enables the policy to recognize self and external collisions. As shown in Fig. 1.2, we feed the map into a pre-trained encoder with a variational auto-encoder (VAE) structure [17] and then obtain a latent vector as the context vector input ω. In this work, we represent all of the geometric states with respect to the base-link frame of the robot. We also define an action as a configuration difference, $a_i = \Delta q_i \in \mathbb{R}^d$. In this work, $q_{i+1} = q_i + \Delta q_i$, since $\mathcal{T}$ is deterministic.

3.3 Guided Path-following Reward Signal

We introduce a multi-objective guided reward function, adopting the example-guided deep RL [18], to follow a target pose path while satisfying kinematic feasibility constraints:

$$\mathcal{R}_i = \mathcal{R}_{task,i} + \mathcal{R}_{im,i} + \mathcal{R}_{cstr,i}, \tag{3.2}$$

where the reward terms in the right-hand side represent task, imitation, and constraint-related rewards, respectively. Note that, for the sake of simplicity, we omit the arguments in the reward functions.

In detail, the task reward function $\mathcal{R}_{task,i}$ at the time step i is composed of position and orientation tracking rewards that encourage the agent to follow the target pose path $X = [x_0, x_1, ..., x_{N-1}]$. Let x_i be a target pose and $\hat{x}_i$ be an end-effector pose corresponding to a current joint configuration q_i at the time step i. Each pose is a tuple of position and quaternion, $(\hat{x}_i^{pos}, \hat{x}_i^{quat})$. Then, we define the position error e_i^{pos} and orientation error e_i^{quat} given q_i as:

$$e_i^{pos} = ||x_i^{pos} - \hat{x}_i^{pos}||_2, \tag{3.3}$$

$$e_i^{quat} = 2\cos^{-1}(|x_i^{quat} \cdot \hat{x}_i^{quat}|). \tag{3.4}$$

Then, instead of using the sum of negative errors as a combined reward, we normalize and reflect the relative importance of each error term e, similar to the parametric normalization [4], defining a function:

$$f(e, \mathbf{w}) = w_0 * \exp(-w_1 * e) - w_3 * e^2 \quad \in \mathbb{R}, \tag{3.5}$$

where $\mathbf{w} = [w_0, w_1, w_2] \in \mathbb{R}_+^3$ is a set of non-negative constants. Then, the task reward function is

$$\mathcal{R}_{task,i} = f(e_i^{pos}, \mathbf{w}^{pos}) + \mathbb{1}_{\sqrt{e_i^{pos}} \leq 5\,\mathrm{cm}} f(e_i^{quat}, \mathbf{w}^{quat}), \tag{3.6}$$

where $\mathbf{w}^{pos}$ and $\mathbf{w}^{quat}$ are user-defined parameters for each type of error. We activate the orientation reward when the current end-effector is within 5 cm of the target position to resolve the potential conflict between the two reward terms.

We also introduce the imitation reward $\mathcal{R}_{im,i}$ to make the agent learn the kinematically feasible postures depicted in the demonstration set ξ^{demo}. To resolve the potential conflicts between task and imitation rewards, we propose a null-space imitation reward function that projects the error between the current configuration q_i and the ith configuration in ξ^{demo} (i.e., $\xi^{demo}[i]$) to the null-space of the current configuration not to lower the path following performance while mimicking the demonstrated posture:

$$\mathcal{R}_{im,i} = f(e_i^{im}, \mathbf{w}^{im}), \tag{3.7}$$

$$e_i^{im} = ||(I - J(q_i)^\dagger J(q_i)) \cdot (\xi^{demo}[i] - q_i)||_2, \tag{3.8}$$

where $J(q_i)$ is the Jacobian matrix at the joint configuration q_i, $\dagger$ represents the Moore-Penrose inverse operation, I is an identity matrix, and $\mathbf{w}^{im}$ is a set of non-negative constants.

The last reward term is the constraint-related reward function $\mathcal{R}_{cstr,i}$ that penalizes collision, joint-limit violation, and early termination states as follows:

$$\mathcal{R}_{cstr,i} = \mathcal{R}_{C,i} + \mathcal{R}_{J,i} + \mathcal{R}_{E,i}, \tag{3.9}$$

where $\mathcal{R}_{C,i}(s_i) = -10 * \mathbb{1}_{collision}(s_i)$, and $\mathcal{R}_{J,i}(s_i) = -1 * \mathbb{1}_{q>q_{max} \bigcup q<q_{min}}(s_i)$. Here, we detect the collision between meshes using Flexible Collision Library [19]. To facilitate the training process, we early terminate the episode with the negative early termination reward $\mathcal{R}_E$, when the end-effector is more than 20 cm away from the target position, as proposed in [18]: $\mathcal{R}_{E,i}(s_i) = -3 * \mathbb{1}_{\sqrt{e_i^{pos}} > 20\,\mathrm{cm}}(s_i)$.

Chapter 4. Experimental Setup

We use a mobile manipulator, Fetch from Fetch Robotics, with a single 7-DoF arm ($d = 7$) for training and evaluation steps in both simulated and real-world experiments. Below we describe how to build pairs of the target path X and demonstration set ξ^{demo} for the MDPs formulation.

4.1 Generation of Target Paths and Demonstration Set.

We first collect 5000 valid end-effector poses within the range of $x : [0.2, 1.2] \times y : [-0.7, 0.7] \times z : [0.0, 1.2]$ (unit: m) by restricting the operation range to the task space. We consider a pose is valid when at least one IK solution satisfies collision-free and joint-limit constraints. From the valid poses, we randomly sample 5 to 8 poses as way-points and interpolate the positions at 0.5 cm distance interval along a B-spline curve. At each point, we also interpolate quaternion orientations using a spherical linear interpolation (Slerp). We then filter out if any intermediate pose $\{x_i \,|\, i \in [0, N-1]\}$ is not valid. Note that it does not guarantee that there is a feasible trajectory even when all intermediate poses are valid.

Based on the aforementioned procedure, we collect 5,000 paths in an environment without objects, and additional 10,000 paths from 500 randomly generated environments consisting of a table-shaped box and various objects on the box (see Fig. 4.1). By randomly sampling two valid initial configurations q_0 per path, we overall collect 30,000 pairs of the target path and optimized joint trajectory generated from TORM with a 120 s time budget. We use the collected pairs $\{X, \xi^{demo}\}$ for the training step below.

4.2 Training Details

We employed a soft actor-critic (SAC) with automatic entropy adjustment [20] to train a policy π maximizing Eq. (3.1) by extending the SAC code in Spinning-up RL library [21]. The policy and double Q networks are composed of 3 hidden layers with 1024 nodes per layer representing its parameter θ. Our LIT generates trajectories using the stochastic policy (i.e., diagonal Gaussian policy). In this work, we bounded the action within the range of $[-0.26, 0.26]$ (unit: rad) to enforce the generated trajectory naturally satisfies connectivity and smoothness; $\pi_\theta(a|s) \sim 0.26 * \tanh(\mathcal{N}(\mu_\theta(s), \Sigma_\theta(s)))$. We used Exponential Linear Unit (ELU) with $\alpha = 1.0$ as activation functions except the last layer.

In this paper, we empirically set the state space parameters as $t = 6$ and $dim(z_{vae}) = 32$, which is the size of scene-encoding latent space, and the normalization coefficients $\mathbf{w}$ of each reward term as $\mathbf{w}^{pos} = [2, 65, 30]$, $\mathbf{w}^{quat} = [2, 5, 0]$ and $\mathbf{w}^{im} = [1, 15, 0.5]$. As the parameters of SAC, we used the 0.99 discounting factor, the 0.995 polyak for target network update, the 3 target entropy, the 10^6 replay buffer size, the 4096 batch size, the 1×10^{-4} learning rate for the critic, 7×10^{-5} for policy, and 1×10^{-4} for the entropy regularization weight. We updated the weights 200 times every 10^4 steps using the Adam optimizer [22]. Adopting the reference-state initialization approach [18], we placed the agent at an initial state randomly sampling from the demonstration set ξ^{demo} and formulated the $\mathcal{M}_X$ with the paired target path X. Training of the policy required 3×10^7 simulation steps, which took approximately 144 hours on the standard desktop equipped with an Intel i9-9900K and a RTX 2080 Ti.

To train a VAE, we collected 3000 random scenes configured in the form of tables of various sizes and objects scattered on them. Following the approach [17], we trained the VAE for 500 epochs with

1×10^{-4} learning rate, 64 batch size, 1×10^{-6} weight decay, and 5×10^{-6} weight of Kullback-Leibler divergence loss term. The training took about 3 hours using the same optimizer and machine.

4.3 Evaluation Setup

We set 3 trajectory-initialization methods for the path following problem as our baselines:

- ***Linear***: *Linear* returns a linearly interpolated trajectory in the configuration space. Considering that the goal configuration is not given in the target path, we selected an IK solution at the last pose of the path having a minimum L_2 distance with an initial configuration q_0.
- ***Greedy*** [14]: *Greedy* extracts the sub-sampled poses from X with 10 intervals. Then, starting from an initial configuration q_0, this finds 150 random IK solutions at each next sub-sampled pose and search for and interpolates with the best IK solution minimizing the objective function (Eq. (2.1)).
- ***LIT_BC***: *LIT_BC* is another learning-based prediction method trained with a behavior-cloning (BC) framework [23], supervised learning for decision making, instead of RL. We trained the neural network naively mimicking the demonstration set ξ^{demo} with a mean squared error loss.

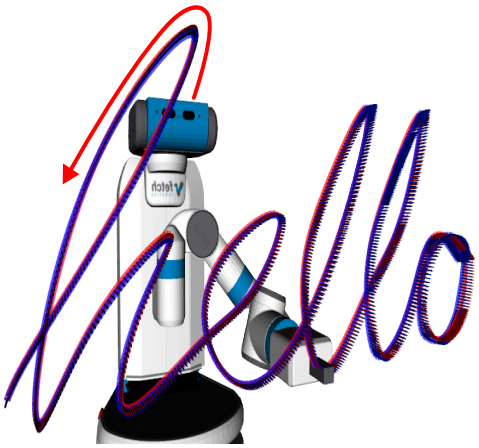
Note that we call our method *LIT_RL* for clarity.

To verify our method across the type of optimizers, we used TORM and TrajOpt. TORM iteratively optimizes and explores new initial trajectories to avoid local minima within 50 s. On the other hand, TrajOpt's update is performed using a quadratic solver, and thus one iteration takes from 3 s to 14 s, so we made one trajectory converged within 150 s.

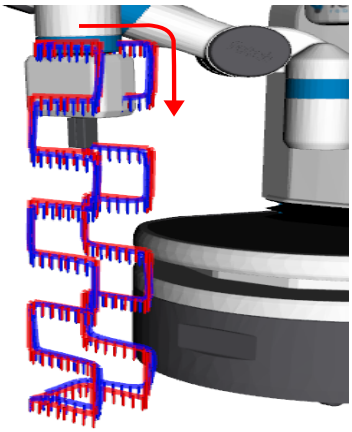
We used total five specific target paths for the comparison with baselines (see Fig. 4.1(a)-4.1(d)). Three paths X ('Hello', 'Rotation', 'Zigzag') are without external obstacles and two paths X ('Square', 'S') are with external obstacles. In the case of 'Hello', 'Zigzag', 'Square', and 'S', we fixed the orientation on the path. On the other hand, in the 'Rotation', we fixed the position on the path while varying the orientation in the range of $\pm 45^\circ$ along the direction of pitch and yaw axes. We collected 100 valid IK solutions, at the first pose x_0, per benchmark path to obtain reliable statistic results using various initial configurations q_0 since the prediction performance largely depends on the initial configurations [6].

We also evaluated the generality of the LIT methods by using 100 randomly generated target paths without external obstacles and 1000 random paths from 100 random scenes, where we sampled and interpolated the randomly sampled valid end-effector poses (see Fig. 4.1(e) and Fig. 4.1(f)). We call this benchmark set 'Random' below.

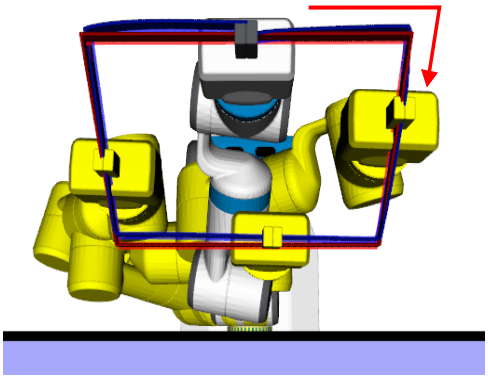
The number of points for each benchmark path is as follows: $N_{Hello} = 553$, $N_{Rotation} = 209$, $N_{Zigzag} = 227$, $N_{Square} = 320$, $N_S = 301$, and $N_{Random} \sim \mathcal{N}(626, 120)$.



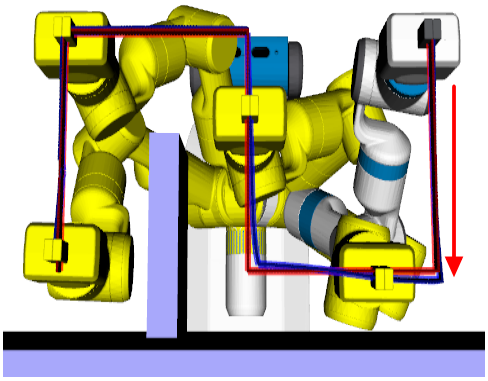
(a) Hello



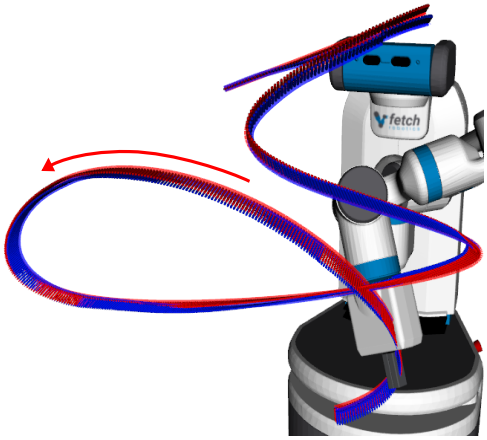
(b) Zigzag



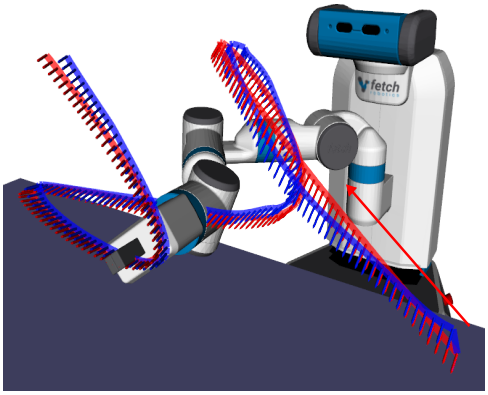
(c) Square



(d) S



(e) Random w/o obs



(f) Random w/ obs

Figure 4.1: Visualization of four specific and two random target paths used in evaluations. Red and blue lines are the target and generated end-effector paths, respectively. We compute blue lines via forward kinematics inputting initial joint trajectories from *LIT_RL*. In (c) and (d), the original color of the robot represents the initial configurations, and the yellow trails indicate that the generated trajectories satisfy collision-avoidance constraints in the environment.

Chapter 5. Results

We first perform quantitative and qualitative analyses of the initial trajectories from the proposed method and baselines in simulated environments. Fig. 5.1 shows the comparative analysis of the initial trajectory generation methods in terms of three quality metrics: objective functional value $\mathcal{U}[\xi]$ (Eq. (2.1)), constraint violation rate considering the collision-free and joint velocity limit violation constraints, and generation time for computational efficiency. We set the Δt between two subsequent configurations to 0.1 s and objective functions' coefficients as $\lambda_1 = 10$ and $\lambda_2 = 1.5$.

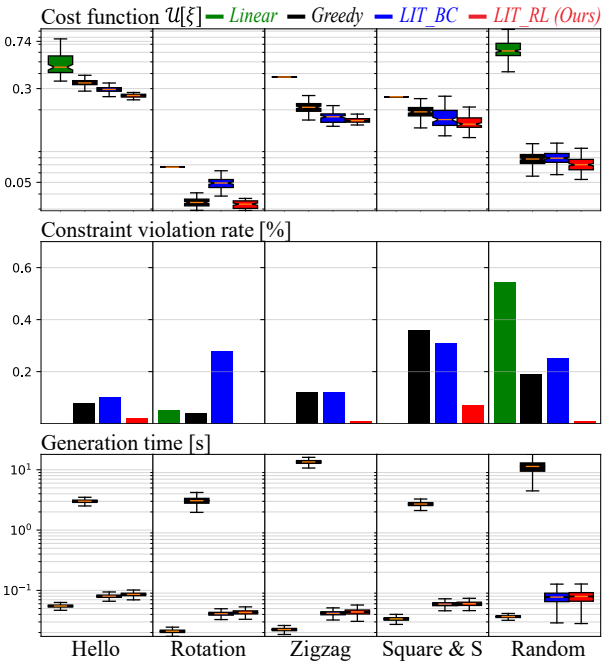
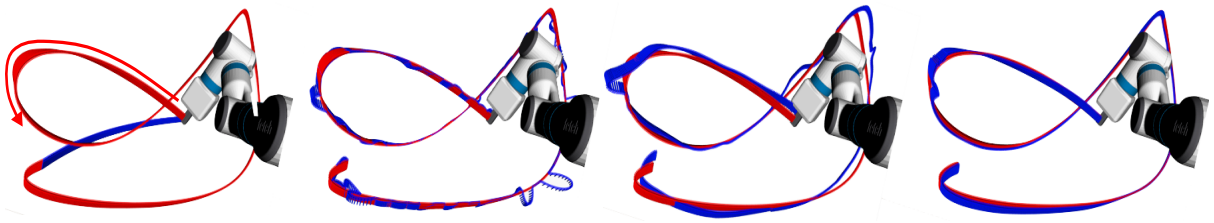


Figure 5.1: Comparative analysis of the four initial-trajectory generation methods in five types of simulated environments. The x- and y-axes are the type of benchmark problems and the performance metrics, respectively.

LIT_RL shows the lowest objective functional value in all benchmark problems; that is, the generated trajectory makes the balance between the sub-objectives of pose error, obstacle cost, and joint smoothness, and also shows the lower constraint violation rate than *Greedy* and *LIT_BC*. Specifically, as Fig. 5.2 qualitatively compares the generated trajectories of each method on one of the 'Random' problems, *Greedy* has the relatively small pose error and obstacle cost because it selects the best IK solution greedily. However, it has the worst performance in terms of joint smoothness since the continuity of the overall trajectory is not guaranteed. Most of the constraint violations of *Greedy* happened due to either no IK solution or the local minima solution with a limited number of IK solutions. On the contrary, the *Linear* has the best smoothness while having the worst pose error and obstacle cost among baselines since it does not consider the objectives represented in the task space. The learning-based methods naturally outperformed the *Greedy* method in generation time since the function approximation helps infer the entire path without time-costly searches. Among the two learning-based



(a) Linear $(0.694,\ 3.4 \cdot 10^{-8}, 0.084)$ (b) Greedy $(0.004, 1.4 \cdot 10^{-6}, 8.462)$ (c) LIT_BC $(0.03, 1.1 \cdot 10^{-6}, 0.198)$ (d) LIT_RL $(0.013, 1.0 \cdot 10^{-6}, 0.195)$

Figure 5.2: Qualitative results on one of the 'Random' problems ($N = 728$). Red and blue lines are the target and generated end-effector paths, respectively. The robot's posture indicates the initial configuration. Numbers within the parenthesis represent the average pose error, the average joint velocity (rad/s), and the generation time (s) in order.

methods, *LIT_RL* shows better performance concerning the objective value than the *LIT_BC* method, while generation time is almost the same as both methods use a neural network with the same structure. *LIT_BC* shows poor tracking performance with the limited demonstration set ξ^{demo} we prepared, while *LIT_RL* generalizes well to more diverse target paths by exploring the state space based on the reward function. Fig. 4.1 shows the initial trajectories synthesized by *LIT_RL* for each benchmark problem as qualitative results.

We further extended our experiment by combining each initialization method with two different trajectory optimization approaches: TORM and TrajOpt. Fig. 5.3 shows the convergence of pose errors over the optimization time. Our approach gave superior convergence performance for all experiments with lower pose errors, though the initial error is often higher than that of *Greedy*. The significantly low error indicates our method helps to find better optimum solutions via TO by providing better initial trajectories. In particular, this difference between our method and others is more prominent in TORM than TrajOpt because TORM is more sensitive to the initial guess due to the first-order optimization process.

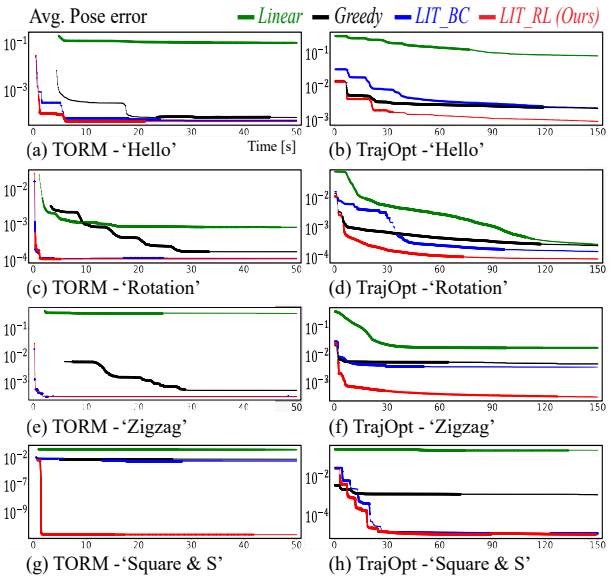


Figure 5.3: Average pose-error convergence of two optimization methods (i.e., TORM and TrajOpt) in four types of simulated benchmark problems during optimization time. We plugged in the proposed and three baseline initial trajectory generation methods into the optimization methods. The y-axis is the pose error in log scale, and the x-axis is the elapsed time (s).

Table 5.1 shows the success rate of the optimized solutions. *LIT_RL* improved the robustness of optimization methods by consistently maintaining the highest success rate in all benchmark sets. In particular, it showed a noticeable performance improvement compared to the *Greedy* in the 'Random' benchmark set. Randomly generated paths are more challenging to follow than the semantically generated ones, because the curvature of the path tends to be large, the path exists in more diverse regions, and the position and rotation change together along the path. Therefore, if the initial trajectory itself does not satisfy the continuity of the joint, it is difficult to converge to a feasible trajectory without falling into the local minima. To check only the convergence of each initial trajectory, we blocked TORM to iteratively explore the new trajectories within the time budget in this experiment. We assessed the optimized solution to be successful when the optimized trajectory satisfies all the constraints and the position and rotation errors are smaller than each threshold values shown in the Table 5.1.

We also investigate the effectiveness of the imitation reward on our framework comparing with that of the other reward combinations while all the constraint-related rewards $\mathcal{R}_{cstr}$ are same. Fig. 5.4 shows the comparison of learning curves (i.e., path-following success rate) when using four combinations of rewards: 1) $\mathcal{R}_{im}$, 2) $\mathcal{R}_{task}$, 3) $\mathcal{R}_{task} + \mathcal{R}_{im,L2}$, 4) $\mathcal{R}_{task} + \mathcal{R}_{im}$. Here, $\mathcal{R}_{im,L2}$ is a simple L_2-distance imitation reward where $e_i^{im} = ||\xi^{demo}[i] - q_i||_2$. Overall, $\mathcal{R}_{task} + \mathcal{R}_{im}$ resulted in the best success rate at the end (Red line). However, individual rewards such as $\mathcal{R}_{task}$ or $\mathcal{R}_{im}$ give lower performance than the proposed reward combination. This indicates the combination helps track the target pose and take kinematically feasible null-space posture captured in the demonstration. On the other hand, another

Path	TO	Method			
		Linear	*Greedy*	*LIT_BC*	*LIT_RL*
Hello	TORM	32.0	68.0	62.0	**87.0**
(0.1 cm, 0.1°)	TrajOpt	46.0	99.0	92.0	**100.0**
Rotation	TORM	93.0	98.0	87.0	**99.0**
(0.1 cm, 0.1°)	TrajOpt	99.0	**100.0**	**100.0**	**100.0**
Zigzag	TORM	3.0	74.0	94.0	**98.0**
(1 cm, 1°)	TrajOpt	73.0	77.0	87.0	**97.0**
Square & S	TORM	6.0	**100.0**	76.0	**100.0**
(1 cm, 1°)	TrajOpt	22.0	**100.0**	**100.0**	**100.0**
Random	TORM	2.3	19.0	61.0	**88.0**
(1 cm, 1°)	TrajOpt	6.5	90.0	85.0	**99.0**

Table 5.1: Comparison of two extended trajectory-optimization approaches with each trajectory initialization methods in terms of success rate (%). We consider an optimized trajectory is 'successful' if the trajectory satisfies kinematic feasibility constraints and the average of positional and rotational errors are lower than certain thresholds represented in the parentheses.

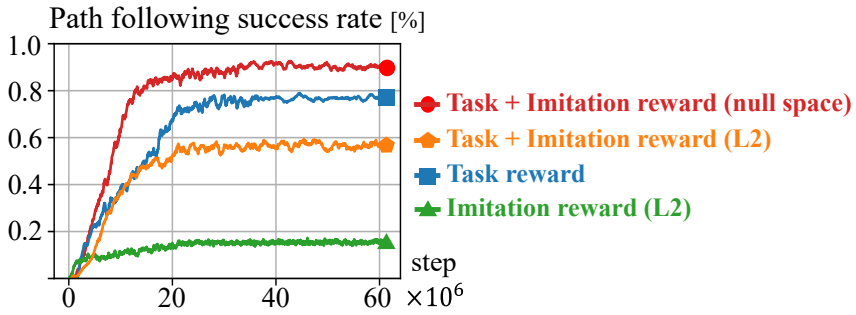


Figure 5.4: Comparison of learning curves from four combinations of reward functions. We measure the success rate by randomly constructing 20 problems at every 1×10^4 evaluation steps and consider one experiment successful when distances between the end effector pose and the target pose at all time steps are within 5 cm positionally and 3° rotationally without any collision.

similar combination, $\mathcal{R}_{task} + \mathcal{R}_{im,L2}$, led to lower performance (Orange line). This indicates that joint-space guidance from a non-optimal demonstration can help learn the kinematically better posture, but it conflicts with the accurate target pose tracking objective. Thus, this result shows the null-space projection in $\mathcal{R}_{im}$ helps to guide the internal posture without sacrificing the tracking performance.

Finally, we verified the optimized trajectory quality with *Greedy* and *LIT_RL* by executing it on the real robot (see Fig. 6.1) and confirmed that our method shows a more accurate and smooth tracking performance.

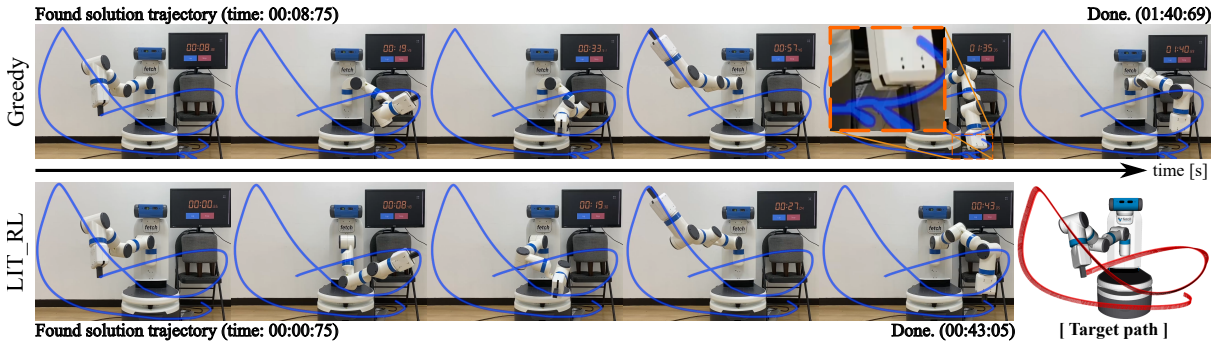


Figure 6.1: Demonstration on a randomly generated target path. Blue lines are the tracked end-effector paths where a Fetch manipulator is tracking trajectory-optimization results with the initial trajectories of *Greedy* [14] and *LIT_RL*. In the fifth frame of *Greedy*, the robot shows abrupt deviation from the target path and takes more time for *Greedy*. In this experiment, we set the Δt value large enough so that the greedy method could produce a discontinuous joint trajectory as a result. As a result, the discontinuity of the joint results in a large execution time as it sequentially tracks nodes in the trajectory.

Chapter 6. Conclusion

We presented a learning-based initial trajectory (LIT) generation method that quickly finds a low-cost initial trajectory for the better trajectory optimization of kinematically redundant manipulators. We formulate the 6-D pose path-following as a multi-task RL, which enables LIT to find a joint-space solution trajectory given a variety of path-following problems. In particular, by defining a unified reward function with a null-space imitation reward, we made the agent explore and learn kinematically feasible postures in demonstrations without conflicting with the task rewards.

We showed the high quality of our initialization method qualitatively and qualitatively. In conjunction with the two representatives TO methods, our initialization method resulted in boosted convergence speed and optimality with a higher success rate over diverse benchmark problems. In addition, We demonstrated the generalization performance and the anytime property of our learning-based method through a variety of simulated and real-world experiments.

The content written in this dissertation has been submitted to the ICRA conference in 2022 and is under the process of review.

Chapter 7. Related Work

7.1 Path-wise Inverse Kinematics (IK)

The inverse kinematics (IK) finds joint values of an articulated chain that produce the desired end-effector pose given in cartesian space. Various numerical methods for solving IK problems include jacobian, non-linear programming (NLP), and artificial intelligence-based methods [24].

In the case of a redundant manipulator, an infinite set of IK solutions are possible for one end-effector pose. Therefore, prior research attempted to simultaneously consider more task objectives, such as obstacle avoidance [25], joint limits, and kinematic singularity avoidance, by using redundancy in joint space, called null-space optimization [26].

Task priority IK [27, 28, 3] uses a null-space projection of the Jacobian matrix to consider tasks hierarchically in the order of priorities. On the other hand, *importance-based IK* [29, 4] formulates the objective function of NLP through the weighted sum of each task such that the IK solution is at the basin of the objective function.

The path-wise IK is extended to finding a list of joint configurations that satisfies the given list of end-effector poses. The constraints that all joint configurations should match the end-effector poses and be feasible (joint-level smoothness and non-collision) make the possible solution space discontinuous [7], resulting in jumping in the joint space in the middle of the joint configuration path with the local search method [2, 4].

RelaxedIK [4] considers joint continuity by adding an objective function related to smoothness with the previous joint values, but it is prone to be stuck into the local minima because only the immediately preceding joint is considered.

In order to make a very accurate and feasible joint path, some research [7, 5, 6] proposes off-line methods. [7] derived a joint path having minimal Fréchet distance with the reference end-effector poses in the task space using a variant of Dijkstra's graph search algorithm on a graph where IK solutions of each end-effector pose form vertices. STAMPEDE [5] efficiently formulated a discrete-space graph by creating vertices with locally optimal IK solutions at each end-effector pose through NLP. With the importance of the starting configuration to the quality of the path-wise IK solution, [6] made multiple candidate solutions from various starting configurations and selected the best one with a user preference. These methods can find a globally optimal solution if time is given enough, but in practice, there is a problem of hyper-parameters' resolution, such as the number of IK solutions for each end-effector pose, and it takes a long time because all solution spaces should be explored.

7.2 Trajectory Optimization for Path-wise IK

Trajectory optimization (TO) approaches have been extensively studied to quickly compute locally optimal trajectory connecting start to end configuration without collision [30]. CHOMP [11] extends the elastic band approach [31] to a functional gradient optimization method and finds a feasible trajectory by iteratively optimizing a functional trading-off between joint smoothness and obstacle avoidance. TORM [14] proposed a two-stage gradient descent technique for the problem of falling into local minima due to the conflict between constraints. STOMP [32] explored the solution space around the current trajectory

by injecting noise to the trajectory with the path integral method [33]. TrajOpt [10] and GuSTO [34] solved a sequential convex optimization problem with the convex relaxation techniques.

These methods usually use linear interpolation in joint space as an initial trajectory for the robot arm, completely independent of task space objectives, especially when all end-effector poses are fully constrained as in a path-wise IK problem. TORM [14] proposes a heuristic initialization method for the path-wise IK problem that creates an initial trajectory with joints selected from IK solutions for each segment of end effector poses. However, making an initial trajectory in the same homotopy set with a global optimum solution is still challenging because the joint continuity between segments is not considered. Recently, many neural-network-based approaches [35, 36, 37, 38] have been researched to warm-start the optimization with the learned initialization by leveraging off-line experience.

7.3 Data-driven Motion Generation

In the field of graphics, research to generate general motions from the collected motion capture data is an active research area. [39, 40, 41, 42, 43] have increased the generalization of motion by constructing neural network architectures to efficiently learn a pattern or mode of behavior repeated in the character's motion. However, although this seems realistic, it cannot be guaranteed whether the generated motion is dynamically feasible.

Therefore, methods using reinforcement learning techniques have emerged to make robust motions even in various unseen external perturbations in a dynamic environment [44, 45]. On the other hand, in reinforcement learning, when learning is performed with a sparse task reward, learning may proceed differently from the user's intention or may not be learned at all. Therefore, [18, 46] shape the reward function so that the agent can have a specific motion style while increasing the learning efficiency with a motion prior built from reference motion data.

Bibliography

[1] H. Bruyninckx, "Open robot control software: the orocos project", in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation*, 2001, vol. 3.

[2] Patrick Beeson and Barrett Ames, "TRAC-IK: An open-source library for improved solving of generic inverse kinematics", in *IEEE International Conference on Humanoid Robots*. IEEE, 2015, pp. 928–935.

[3] Bruno Siciliano, "Kinematic control of redundant robot manipulators: A tutorial", *Journal of intelligent and robotic systems*, vol. 3, no. 3, pp. 201–212, 1990.

[4] Daniel Rakita, Bilge Mutlu, and Michael Gleicher, "RelaxedIK: Real-time synthesis of accurate and feasible robot arm motion.", in *Robotics: Science and Systems*, 2018.

[5] Daniel Rakita, Bilge Mutlu, and Michael Gleicher, "STAMPEDE: A discrete-optimization method for solving pathwise-inverse kinematics", in *IEEE International Conference on Robotics and Automation*. IEEE, 2019, pp. 3507–3513.

[6] Pragathi Praveena, Daniel Rakita, Bilge Mutlu, and Michael Gleicher, "User-guided offline synthesis of robot arm motion from 6-dof paths", in *IEEE International Conference on Robotics and Automation*. IEEE, 2019, pp. 8825–8831.

[7] Rachel Holladay, Oren Salzman, and Siddhartha Srinivasa, "Minimizing task-space frechet error via efficient incremental graph search", *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1999–2006, 2019.

[8] Rosen Diankov, *Automated Construction of Robotic Manipulation Programs*, PhD thesis, Carnegie Mellon University, Robotics Institute, August 2010.

[9] Jingru Luo and Kris Hauser, "Interactive generation of dynamically feasible robot trajectories from sketches using temporal mimicking", in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 3665–3670.

[10] John Schulman, Jonathan Ho, Alex X Lee, Ibrahim Awwal, Henry Bradlow, and Pieter Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization", in *Robotics: Science and Systems*. Citeseer, 2013, vol. 9, pp. 1–10.

[11] Matt Zucker, Nathan Ratliff, Anca D Dragan, Mihail Pivtoraiko, Matthew Klingensmith, Christopher M Dellin, J Andrew Bagnell, and Siddhartha S Srinivasa, "CHOMP: Covariant hamiltonian optimization for motion planning", *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[12] Mrinal Kalakrishnan, Sachin Chitta, Evangelos Theodorou, Peter Pastor, and Stefan Schaal, "Stomp: Stochastic trajectory optimization for motion planning", in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 4569–4574.

[13] Chonhyon Park, Jia Pan, and Dinesh Manocha, "Itomp: Incremental trajectory optimization for real-time replanning in dynamic environments", in *Proceedings of the Twenty-Second International Conference on International Conference on Automated Planning and Scheduling*. 2012, p. 207–215, AAAI Press.

[14] Mincheul Kang, Heechan Shin, Donghyuk Kim, and Sung-Eui Yoon, "TORM: Fast and accurate trajectory optimization of redundant manipulator given an end-effector path", in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 9417–9424.

[15] Daniele Calandriello, Alessandro Lazaric, and Marcello Restelli, "Sparse multi-task reinforcement learning", 2014.

[16] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li, "On the continuity of rotation representations in neural networks", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5745–5753.

[17] Diederik P. Kingma and Max Welling, "Auto-encoding variational bayes", in *Proceedings of the Second International Conference on Learning Representations (ICLR)*, Apr. 2014.

[18] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills", *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–14, 2018.

[19] Jia Pan, Sachin Chitta, and Dinesh Manocha, "FCL: A general purpose library for collision and proximity queries", in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 3859–3866.

[20] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al., "Soft actor-critic algorithms and applications", *Proceedings of the 35th International Conference on Machine Learning*, 2018.

[21] Joshua Achiam, "Spinning Up in Deep Reinforcement Learning", 2018 [Online] Available: https://github.com/openai/spinningup.

[22] Diederick P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization", in *International Conference on Learning Representations (ICLR)*, 2015.

[23] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al., "End to end learning for self-driving cars", *arXiv preprint arXiv:1604.07316*, 2016.

[24] Samuel R Buss, "Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods", *IEEE Journal of Robotics and Automation*, vol. 17, no. 1-19, pp. 16, 2004.

[25] Anthony A Maciejewski and Charles A Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments", *The international journal of robotics research*, vol. 4, no. 3, pp. 109–117, 1985.

[26] Jürgen Hess, Gian Diego Tipaldi, and Wolfram Burgard, "Null space optimization for effective coverage of 3d surfaces using redundant manipulators", in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 1923–1928.

[27] Paolo Baerlocher and Ronan Boulic, "Task-priority formulations for the kinematic control of highly redundant articulated structures", in *Proceedings. 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications (Cat. No. 98CH36190)*. IEEE, 1998, vol. 1, pp. 323–329.

[28] Pasquale Chiacchio, Stefano Chiaverini, Lorenzo Sciavicco, and Bruno Siciliano, "Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy", *The International Journal of Robotics Research*, vol. 10, no. 4, pp. 410–425, 1991.

[29] Hyun Joon Shin, Jehee Lee, Sung Yong Shin, and Michael Gleicher, "Computer puppetry: An importance-based approach", *ACM Transactions on Graphics (TOG)*, vol. 20, no. 2, pp. 67–94, 2001.

[30] John T Betts, "Survey of numerical methods for trajectory optimization", *Journal of guidance, control, and dynamics*, vol. 21, no. 2, pp. 193–207, 1998.

[31] Sean Quinlan and Oussama Khatib, "Elastic bands: Connecting path planning and control", in *[1993] Proceedings IEEE International Conference on Robotics and Automation*. IEEE, 1993, pp. 802–807.

[32] Mrinal Kalakrishnan, Sachin Chitta, Evangelos Theodorou, Peter Pastor, and Stefan Schaal, "STOMP: Stochastic trajectory optimization for motion planning", in *IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 4569–4574.

[33] Hilbert J Kappen, "An introduction to stochastic control theory, path integrals and reinforcement learning", in *AIP conference proceedings*. American Institute of Physics, 2007, vol. 887, pp. 149–181.

[34] Riccardo Bonalli, Abhishek Cauligi, Andrew Bylard, and Marco Pavone, "Gusto: Guaranteed sequential trajectory optimization via sequential convex programming", in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 6741–6747.

[35] Somrita Banerjee, Thomas Lew, Riccardo Bonalli, Abdulaziz Alfaadhel, Ibrahim Abdulaziz Alomar, Hesham M Shageer, and Marco Pavone, "Learning-based warm-starting for fast sequential convex programming and trajectory optimization", in *2020 IEEE Aerospace Conference*. IEEE, 2020, pp. 1–8.

[36] Oliwier Melon, Mathieu Geisert, David Surovik, Ioannis Havoutis, and Maurice Fallon, "Reliable trajectories for dynamic quadrupeds using analytical costs and learned initializations", in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1410–1416.

[37] Oliwier Melon, Romeo Orsolino, David Surovik, Mathieu Geisert, Ioannis Havoutis, and Maurice Fallon, "Receding-horizon perceptive trajectory optimization for dynamic legged locomotion with learned initialization", 2021.

[38] Jeffrey Ichnowski, Yahav Avigal, Vishal Satish, and Ken Goldberg, "Deep learning can accelerate grasp-optimized motion planning", *Science Robotics*, vol. 5, no. 48, 2020.

[39] Daniel Holden, Taku Komura, and Jun Saito, "Phase-functioned neural networks for character control", *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–13, 2017.

[40] He Zhang, Sebastian Starke, Taku Komura, and Jun Saito, "Mode-adaptive neural networks for quadruped motion control", *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–11, 2018.

[41] Sebastian Starke, He Zhang, Taku Komura, and Jun Saito, "Neural state machine for character-scene interactions.", *ACM Trans. Graph.*, vol. 38, no. 6, pp. 209–1, 2019.

[42] Sebastian Starke, Yiwei Zhao, Taku Komura, and Kazi Zaman, "Local motion phases for learning multi-contact character movements", *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 54–1, 2020.

[43] Dario Pavllo, David Grangier, and Michael Auli, "Quaternet: A quaternion-based recurrent model for human motion", *arXiv preprint arXiv:1805.06485*, 2018.

[44] Ying-Sheng Luo, Jonathan Hans Soeseno, Trista Pei-Chun Chen, and Wei-Chao Chen, "Carl: Controllable agent with reinforcement learning for quadruped locomotion", *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 38–1, 2020.

[45] Kevin Bergamin, Simon Clavet, Daniel Holden, and James Richard Forbes, "Drecon: data-driven responsive control of physics-based characters", *ACM Transactions On Graphics (TOG)*, vol. 38, no. 6, pp. 1–11, 2019.

[46] Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa, "AMP: adversarial motion priors for stylized physics-based character control", *CoRR*, vol. abs/2104.02180, 2021.

Acknowledgments in Korean

막연하게 공학분야를 좋아하고 해당분야를 이끌어 보고 싶어 시작하게 된 연구자의 길에 있어 작은 한 챕터로써 석사과정을 마무리 짓게 되었습니다. 연구실 생활을 시작하며 논문도 읽고 발표, 미팅, 토론 등등을 경험하며 통하여 저 자신도 다양한 방면으로 성숙해진 것을 느낍니다. 분명 저 혼자만에 힘으로 여기까지 올 수 있었던 것은 아닙니다. 먼저 언제나 학생을 위해 주시고, 연구에 있어 엇나가지 않도록 많은 조언과 가르침을 주신 윤성의 교수님께 큰 감사의 말씀을 드립니다. 비단 교수님과 제자의 관계에 있어서 뿐만 아니라 인생 전반에 대해서도 배울 점이 많아 저에게는 큰 행운이었습니다. 또 짧으면 짧고 길면 긴 시간 동안 시간을 내주시어 다방면으로 큰 가르침을 주신 박대형 교수님께도 큰 감사의 말씀을 드립니다. 교수님께서 보여주신 애정에 보답하고자 더욱 열심히 준비하여 논문도 작성할 수 있었던 것 같습니다. 앞으로 시작하게 된 박사과정에 있어서도 논문의 X, Y, Z는 잊지 않도록 노력하고 교수님과 함께 더 재미있는 다양한 연구를 해보고 싶습니다.

2년간에 대학원 생활에 있어 SGVR 연구실에서의 생활은 너무도 행복하였습니다. 연구실 구성원들 모두가 친절하고, 각자의 연구분야를 이끌어가는 모습을 보며 큰 귀감이 되어 주셨습니다. 특히 매주 같이 미팅을 하며 다양한 피드백과 정보를 공유한 로봇팀(권용선, 김태영, 강민철, 안인규, 신희찬, 김민철, 문성주, 장충수, 장진혁, 유형열, 이세빈, 김진원)에게 감사하고 즐거웠습니다. 로봇을 좋아하는 사람들이 하나로 모여 서로가 서로를 이끌어 주는 모습이 너무 보기 좋았던 것 같습니다. GPU 대란으로 인한 장비 구매의 힘든 시기를 같이 해쳐나간 장비팀의 이미지 김재윤, 렌더링 김재윤, 조인영께도 감사드리고, 항상 뒤에서 행정을 봐주시어 학생들에 수고를 덜어주시는 김 선생님께도 큰 감사의 말씀을 드리고 싶습니다. 연구실의 동기이자 친구인 형열, 세빈, 진원에게는 특별한 감사의 말을 남깁니다. 길지는 않지만 같이한 순간순간들이 2년의 긴 터널을 너무나도 짧게 느껴지도록 합니다. 이제는 각자의 길을 선택하여 가지만 어딜 가든 잘할 것으로 믿고 있습니다. 모든 구성원 개개인간에 추억을 적기에는 너무나도 한정된 공간이 작게 느껴집니다. 좋은 사람들을 만날 수 있어서 너무 행복했고, 모두 다시 뵙겠습니다.

끝으로 항상 응원하고 지지해주시는 사랑하는 부모님 그리고 동생에게 감사의 마음을 전합니다. 제게는 그 무엇과도 바꿀 수 없는 선물입니다. 그에 대한 보답으로 작으나마 이 석사 학위 논문을 바칩니다. 감사합니다.

Curriculum Vitae in Korean

이　　름: 윤 민 성

학　　력

2011. 3. – 2014. 2.　서라벌 고등학교

2015. 3. – 2019. 2.　인하대학교 정보통신공학과 (학사)

2020. 3. – 2022. 2.　한국과학기술원 전산학부 (석사)

연 구 업 적

1. **Minsung Yoon**, Daehyung Park, and Sung-Eui Yoon, "Bias tree expansion using reinforcement learning for efficient motion planning", Korea Robotics Society Annual Conference (KRoC), 2021
2. Jinhyeok Jang, Heechan Shin, **Minsung Yoon**, Seungwoo Hong, Hae-Won Park and Sung-Eui Yoon, "Deep Neural Network-based Fast Motion Planning Framework for Quadrupedal Robot", *Machine Learning for Motion Planning* Workshop at ICRA 2021
3. **Minsung Yoon**, Mincheul Kang, Daehyung Park, and Sung-Eui Yoon, "Learning-based Initialization of Trajectory Optimization for Redundant Manipulators' Path-following Problem", IEEE International Conference on Robotics and Automation (ICRA) 2022 (under review)
4. Hyeongyeol Ryu, **Minsung Yoon**, Daehyung Park, and Sung-Eui Yoon, "Confidence-based Robot Navigation under Sensor Occlusion with Deep Reinforcement Learning", IEEE International Conference on Robotics and Automation (ICRA) 2022 (under review)