Path Following for Robot Manipulators Using Gyroscopic Forces

by

Nan Wei

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2017

Abstract

This thesis deals with the path following problem the objective of which is to make the end effector of a robot manipulator trace a desired path while maintaining a desired orientation. The fact that the pose of the end effector is described in the task space while the control inputs are in the joint space presents difficulties to the movement coordination. Typically, one needs to perform inverse kinematics in path planning and inverse dynamics in movement execution. However, the former can be ill-posed in the presence of redundancy and singularities, and the latter relies on accurate models of the manipulator system which are often difficult to obtain.

This thesis presents an alternative control scheme that is directly formulated in the task space and is free of inverse transformations. As a result, it is especially suitable for operations in a dynamic environment that may require online adjustment of the task objective. The proposed strategy uses the transpose Jacobian control (or potential energy shaping) as the base controller to ensure the convergence of the end effector pose, and adds a gyroscopic force to steer the motion. Gyroscopic forces are a special type of force that does not change the mechanical energy of the system, so its addition to the base controller does not affect the stability of the controlled mechanical system. In this thesis, we emphasize the fact that the gyroscopic force can be effectively used to control the pose of the end effector during motion. We start with the case where only the position of the end effector is of interest, and extend the technique to the control over both position and orientation. Simulation and experimental results using planar manipulators as well as anthropomorphic arms are presented to verify the effectiveness of the proposed controller.

Acknowledgements

I would like to first thank my supervisor, Professor Soo Jeon, for the opportunity to work in the field of robotics and control. His guidance throughout the past couple of years is greatly appreciated, and this thesis would not have been possible without him.

I would also like to thank my thesis committee members, Professor Christopher Nielsen and Professor Baris Fidan, for their time and support. I am grateful to Professor Dong Eui Chang for his valuable course on nonlinear control theory which is a source of inspiration for this thesis.

Dedication

To my family and friends.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ADRC	Active Disturbance Rejection Control 4
APE	Artificial Potential Energy 7, 20, 22, 23, 29, 30, 33, 35, 36, 38, 42–46, 48, 51, 52, 55
CNS	Central Nervous System 3, 5, 6, 17
DoF	Degrees of Freedom 5–7, 9, 12, 17, 24, 30–33, 35–38, 40–43, 46, 47, 49, 52–54, 66, 69, 71
EEF	End Effector 2, 3, 5–8, 10–13, 17, 19, 20, 22–24, 26, 27, 30, 31, 33, 35–37, 40–44, 46–50, 55, 66
MPC	Model Predictive Control 8, 9
PD	Proportional-Derivative 2, 4
PID	Proportional-Integral-Derivative 2, 4, 7, 38
PVFC	Passive Velocity Field Control 10, 16
TFL	Transverse Feedback Linearization 9, 10
WAM	Whole Arm Manipulator 5, 35, 36, 38, 46, 47, 49, 52–54

Chapter 1

Introduction

Motion control can be broadly categorized into three classes:

- **Set-point regulation** has the objective of asymptotically driving the output of the system $y(t)$ to a desired set-point $\bar{y}$.
- **Trajectory tracking** takes in a time-history of the desired output known as the reference trajectory $\bar{y}(t)$, and the goal is to asymptotically drive $y(t)$ to the reference trajectory.
- **Path following** ignores the temporal specification of the reference trajectory. Instead, it is concerned with controlling the system so that $y(t)$ converges to a desired path $\mathcal{P}$ in the output space.

Consider a mobile robot trying to reach a target location. If the robot is free to travel in the output space, one can treat this simply as a set-point regulation problem. In the presence of obstacles, a collision-free path needs to be identified first before any control action can be taken. After that, path following control strategies can be employed to ensure that the robot stays on this desired path while traveling towards the target. However, path following control is relatively rarer compared to the other two types [1]. Typically, the desired path is converted into a reference trajectory by enforcing a time parameterization, and then one can use trajectory tracking control schemes to realize the control objective. Though the tracking controller can be tuned to obtain high accuracy, the time parameterization can lead to infeasible trajectories and is difficult to modify in real-time in a dynamic environment [2].

This thesis is concerned with the path following control for robot manipulators in the task space. We first note that the term "path" here refers to both the position and orientation. In other words, the objective is to control the End Effector (EEF) so it stays on or close to a set of desired poses during the motion.

1.1 Background

The conventional approach for manipulator motion generation is depicted in Figure 1.1 [3, 4, 5]. One almost always starts with the goal pose of the EEF. The first step is to plan a path between the initial pose and the target pose, and then parametrize it with respect to time to obtain a reference trajectory. This reference trajectory is described in the task space, which is often Cartesian coordinates for position, and Euler angles or unit quaternions for orientation. The low-level control command for the robot, however, exists in a different coordinate; usually the control is over the torque exerted by the motors at each joint, or equivalently the motor voltages or currents. As a result, it seems inevitable that one needs to convert the reference trajectory in task space into one in joint space, i.e., desired joint angles over time. Only after this step of inverse kinematics can one apply the appropriate control law to track the reference trajectory in joint space and generate the motion. This hierarchical approach has been widely implemented in manufacturing applications such as spray painting [6], welding, and part assembly [7]. The trajectory planning and inverse kinematics steps are carried out offline first, and then real-time control is implemented to track the trajectory repeatedly. Commonly, a high gain joint space Proportional-Derivative (PD) or Proportional-Integral-Derivative (PID) control scheme is used. One may also apply inverse dynamics for better accuracy if the dynamical model of the system is known.

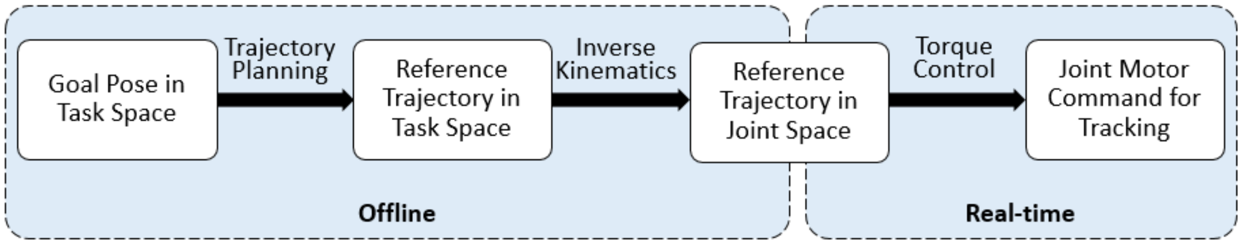

Goal Pose in Task Space
Trajectory Planning
Reference Trajectory in Task Space
Inverse Kinematics
Reference Trajectory in Joint Space
Torque Control
Joint Motor Command for Tracking
Offline
Real-time

Figure 1.1: Typical approach for robot manipulator motion generation

On the other hand, as robots embrace increasingly wider applications outside of the factory floors, the various tasks they perform are no longer repetitive, and the uncertain and dynamic nature of the environment in which the they operate may need on-line adjustments of the desired path. Compared to industrial robots, general purpose robots usually have lower requirements for precision, but the ability to adapt to a changing circumstance is critical.

To cope with this different demand, we take inspiration from human motion generation. Although the academic world has yet agreed to a conclusion on how the human Central Nervous System (CNS) functions to coordinate movements, numerous studies in neuroscience suggest that the primary control focus is placed on the task space variables [8, 9]. Indeed, it has been well-known that human reaching motions have invariant patterns in the task space, characterized by quasi-straight path and bell-shaped velocity profile at the hand [10, 11]. Consistent with these observations is a task space reaching (set-point regulation) controller named the transpose Jacobian control which was introduced by Takegaki and Arimoto in 1981 [12]. This controller in essence places a virtual spring at the EEF so that it converges to the equilibrium or target point. The beauty of this control scheme is that the reaching convergence is guaranteed without inverse kinematics calculations as shown in Figure 1.2, so the offline computation is avoided. Compared to the traditional approach in Figure 1.1, one can perform on-line adjustments by simply updating the target pose. The downside, however, is that this set-point regulation controller is incapable of controlling the intermediate path unless one manually tunes the gains by trial-and-error. This problem can be addressed by applying inverse dynamics to compensate for the nonlinearities, but this requires accurate *a priori* knowledge of the dynamical model.

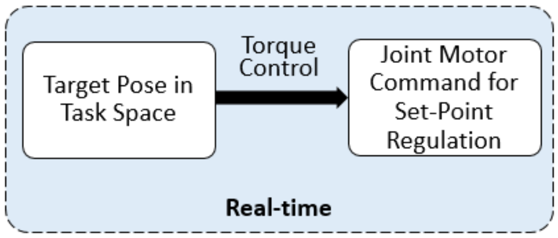

Target Pose in Task Space
Torque Control
Joint Motor Command for Set-Point Regulation
Real-time

Figure 1.2: Schematic of the transpose Jacobian control

This thesis intends to identify an inverse-free control law that not only guarantees the convergence of EEF pose but also shapes its path during the motion. More specifically, a modification to the transpose Jacobian controller is proposed by adding a gyroscopic force as an auxiliary control action. The additional term does not affect the stability of the original closed-loop system, but can act as a steering factor when the robot manipulator is in motion even though inverse dynamics is not performed. This gyroscopic force is also parametrized directly in the task space, so the overall controller, just like the original transpose Jacobian controller, is free from any inverse transformations.

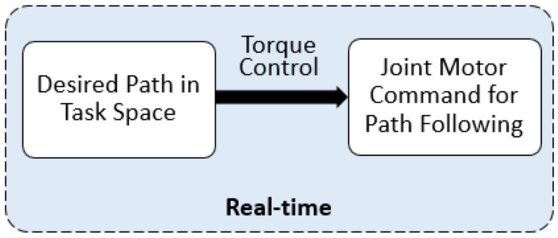

Desired Path in Task Space
Torque Control
Joint Motor Command for Path Following
Real-time

Figure 1.3: Schematic of the proposed control law

1.2 Related Works

1.2.1 Joint Space Control Schemes

Joint space control, illustrated in Figure 1.1, is the most common approach for robot manipulator motion generation, with PID control being the dominant method. To date, the PID control law has been developed for roughly a century and is used in virtually all industries [13]. Its rise in the last century was largely contributed by its simplicity, as one can implement the feedback by using a simple analog circuit. In this perspective, Han argued that PID "cannot fully take advantage of the new compact and powerful digital processors" [13]. Indeed, in most cases for robot manipulators, the PID gains are designed as diagonal matrices, which means that each joint is independently tracking its own target, and the sense of coordination between multiple joints is then lost.

Common alternatives to PID include PD control with gravity compensation, computed torque control, inverse dynamics, adaptive control, and robust control. Details on these algorithms are out of the scope of this thesis, and the interested reader is referred to [14, 15, 16, 17] for more discussions. The recent rise of big data also leads to developments in data-driving approaches [18, 19, 20]. Active Disturbance Rejection Control (ADRC), which treats the nonlinear dynamics as disturbances and acts to estimate and cancel them online, has received some attention as well [21, 22].

As mentioned in Section 1.1, performance aside, these joint space control algorithms all require an offline inverse kinematics step prior to the implementation of real-time torque control. Various efficient methods for the inverse kinematics problems have been proposed. One can find analytical solutions of the inverse kinematic problem for certain types of robot manipulators [23, 24, 25]. Alternatively, machine learning algorithms can also be used to learn the inverse kinematics mapping [26, 27, 28]. Efficient numerical methods that speed up the calculation are also developed [29, 30, 31]. The improvements of computer power lessens the concern on the computation time needed for this step as well.

Despite these efforts, the inverse kinematics problem is notorious for two issues: singularity and redundancy. Singularity occurs when the manipulator enters a configuration at which one or more Degrees of Freedom (DoF) are lost, and the EEF loses the ability to move in certain directions. At singular configurations, the Jacobian matrix becomes rank-deficient, so inverse kinematics using the inverse or pseudo-inverse of the Jacobian matrix will experience numerical instability. Redundancy refers to when DoF of the manipulator is greater than the dimension of the task space. For instance, rigid-body motion in 3D has a dimension of 6 while human arms have 7 DoF: 3 at the shoulder, 1 at the elbow, and 2 at the wrist [32]. Most anthropomorphic manipulators, including the Whole Arm Manipulator (WAM) from *Barrett Technologies*, Baxter from *Rethink Robotics* and the *Mitsubishi* PA10 robot, adopt the same structure. Redundancy leads to dexterity in motion because the extra DoF allows humans and robots to be flexible when performing a task [33]. When it comes to motion control, however, redundancy makes the inverse kinematic problem ill-posed because there often exists infinitely many solutions. This problem was first articulated by Bernstein in the mid-20th century [34, 35]. In order to force it into a unique solution, a redundancy resolution method needs to be in palce, usually in the form of some constraints or as the result of certain optimization [36, 37]. Researchers in the neuroscience community have proposed different cost functions to be minimized that can explain the patterns in human reaching motion: jerk [38], torque change [39], or positional variance in the presence of signal-dependent noise [40].

In addition to computation time and cost, it still remains a question whether optimization is indeed the underlying principle of the CNS. In the early 20th century, Bernstein [41] designed a motion capture system to analyze the repetitive motion of a blacksmith hitting the chisel with a hammer. By placing light bulbs on the blacksmith and taking photographs at a fast rate, he observed that the variability of the hammer position across different trials is considerably smaller than the variability of each joint angle [42, 41]. This suggests that instead of seeking a unique solution, the CNS might be actively exploiting the redundancy by coordinating all joints to compensate for errors in the task space in real-time. In light of this, Latash argues that the term "motor redundancy" is incorrectly used since it suggests that the extra DoF is a burden that needs to be treated via some redundancy resolution method; instead, she proposes the term "motor abundance" that better captures the synergies between all joints [42].

In order to be consistent with most literatures in the robotics field, this thesis will continue using the term "redundancy," but the readers are advised to bear in mind the distinction made by Latash.

1.2.2 Operational Space Framework

The operational space framework is a popular approach developed by Khatib in the 1980s [43, 44], motivated by the fact that the task description is in a different coordinate than the joint space. In this method, a lower level joint space control is in place to cancel out the nonlinear dynamics of the manipulator, and by finding the appropriate mapping between the joint space torque and task space force, one can make the EEF behave like a unit mass [45]. Then one can simply treat the system as a mobile agent and use any task space position control algorithm at the outer loop. Applying the same principle at points on the manipulator other than the EEF, real-time collision avoidance can also be achieved [43]. The structure of the operational space motion control is demonstrated in Figure 1.4.

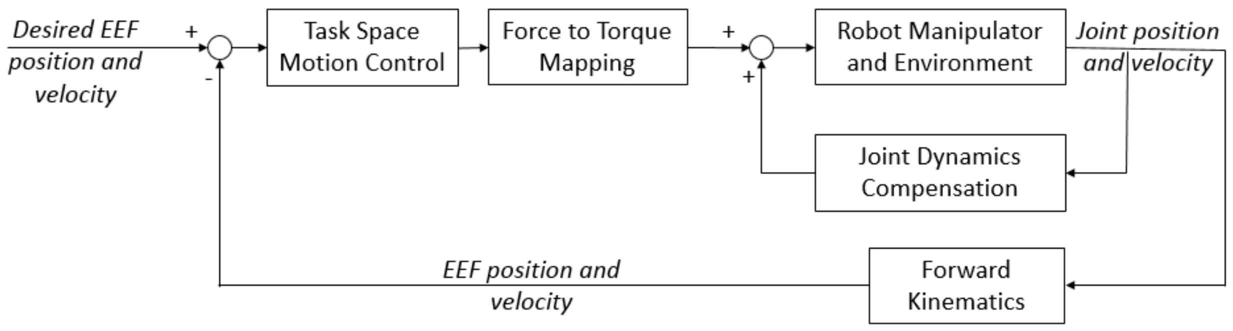

Desired EEF position and velocity
+
-
Task Space Motion Control
Force to Torque Mapping
+
+
Robot Manipulator and Environment
Joint position and velocity
Joint Dynamics Compensation
EEF position and velocity
Forward Kinematics

Figure 1.4: Structure of the operational space control

The operational space approach is useful not only in motion control but also force control in the task space, which makes it well-suited for applications such as human-robot interaction and object manipulation. The performance of this framework heavily depends on the accuracy of the joint dynamics compensation. As Khatib stated in [45], a poor estimate of the Coriolis and centrifugal forces may lead to negative damping, in which case the system becomes unstable even in the absence of any outer loop commands. Additionally, though the end user may treat the EEF as a mobile agent, an inverse or pseudo-inverse of the Jacobian matrix is needed at the lower level controller to achieve this behavior, so the problem with singularity still exists. At singular conditions, the EEF loses one or more directions of motion; in other words, the perceived inertia in the task space is infinite in these directions. As a result, the force to torque mapping step in Figure 1.4 will lead to very large torques that cause the manipulator to be unstable or even dangerous when human workers are nearby.

1.2.3 Energy Shaping and the Transpose Jacobian Control

Takegaki and Arimoto introduced a simple and inverse-free reaching controller in 1981 [12]. Inspired by the fact that without external forces, a manipulator will always converge to a configuration that has the lowest potential energy, this control law cancels out the gravity exerted on the system and replaces it by an Artificial Potential Energy (APE) in the task space that has a unique minimum at the target position. With the aid of damping, the EEF is guaranteed to converge to this target position. This is often believed to be the origin of a class of control laws known as energy shaping control or method of controlled Lagrangian [46, 47], which later also accepts wider applications such as navigation and obstacle avoidance using APE [43]. Because this control law uses the transpose of the manipulator Jacobian to convert task space force to joint torques, it is often referred to as the transpose Jacobian control. In addition to reaching control, the transpose Jacobian controller is used in fields such as impedance control [48, 49] since the EEF exhibits spring-like behavior under this control law. It is also popular in visual servoing [50, 51] thanks to the simplicity of the algorithm.

It was later found that the orientation of the EEF can also be regulated by the same controller, and a simulation with a 6-DoF space manipulator is presented in [52]. One major advantage of this controller over conventional PID servo control scheme is that convergence is guaranteed without the need to perform inverse kinematics calculations for both non-redundant and redundant manipulators. Therefore, it was argued that the transpose Jacobian control in some ways provides a solution to Bernstein's DoF problem.

Arimoto also showed that the transpose Jacobian control can generate skilled motions,

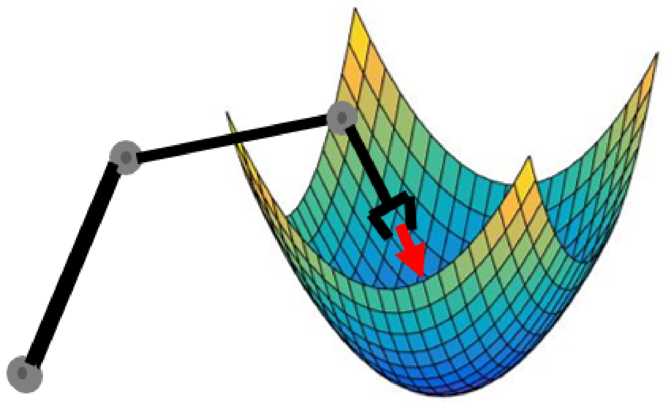


Figure 1.5: Illustration of APE in the transpose Jacobian control

such as a human reaching motion in which the hand follows a quasi-straight line with bell-shape velocity profile [53, 54, 55]. However, a systematic procedure to tune the control gains has not been developed, and the same set of gains do not work equally well for different trajectories even with the same manipulator. In other words, despite the convergence of the EEF to the goal position, its intermediate path profile is not controlled, and the transpose Jacobian control is generally categorized as a set-point regulation controller.

1.2.4 Path Following Controls for Robot Manipulators

As mentioned in the beginning of this chapter, the typical approach towards a path following problem is to re-formulate it as a trajectory generation and tracking problem, by first parameterizing the path as a function of time. However, the timing of the desired states is not always critical for the task, and this time parametrization may lead to undesired consequences such as infeasible trajectories and actuator saturations. Taking machining for example, usually the main objective is to have the cutting tool to follow a contour precisely; parameterizing the path as a function time without considering the dynamical constraints could result in unnecessarily demanding trajectories that compromise the contour following accuracy.

Therefore, an offline constrained optimization is usually performed to convert a path into a trajectory to ensure that the resulting trajectory is feasible. The authors in [56] proposed a nonlinear Model Predictive Control (MPC) scheme directly for the path following problem and showed that by simplifying the model, the control can be implemented in real-time at up to $1kHz$. However, it should noted that the examples given were writing letters or drawing patterns at a very slow speed; for instance, the allowable time to write the word "hello" was $20s$. It is unclear how the controller will perform under faster operations. On top of the computation demand, one needs a fairly accurate dynamical model of the manipulator in order to implement MPC, and in many cases this is a difficult problem by itself, especially when the DoF is high.

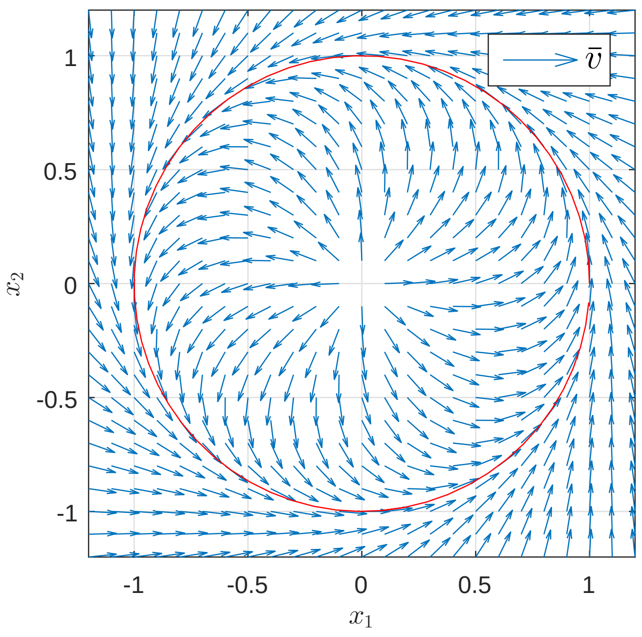


Figure 1.6: Example of a desired velocity field for a circular path

The Transverse Feedback Linearization (TFL) method [57, 58, 59] for path following relies on a step of feedback transformation. In the new coordinate frame, the dynamics can be separated into 2 sub-systems, one along the desired path (tangential dynamics) and the other off the desired path (transversal dynamics). Then one can design controls for the 2 sub-systems to exhibit desired behaviors. TFL converts the desired path into an attractive and invariant set in the transversal dynamics, and the closed-loop tangential states will travel in the desired direction. In the case of redundant manipulators, the redundancy can be exploited to account for joint angle limits and to enforce boundedness of the internal dynamics [60]. TFL is a powerful approach for a broader class of mechanical systems not limited to torque-controlled robot manipulators. Though it has be shown that the outer loop control can be designed to make the TFL robust to model uncertainties [61], severe modeling errors will degrade the closed-loop performance. The complicated form of TFL also makes it difficult to implement, limiting its acceptance in applications.

Alternatively, path following can be formulated as a velocity field control problem. For instance, if the task is to trace a circle repeatedly, one can form a desired velocity field in Figure 1.6. If the control law can be shown to make the EEF velocity converge to the desired velocity $\bar{v}$, then equivalently convergence to the path is automatically guaranteed. The Passive Velocity Field Control (PVFC) was developed based on this idea [62, 63]. The controller design starts with augmenting the manipulator plant with a fictitious flywheel which acts as an energy reserve. PVFC works by transferring energy between the actual manipulator plant and this fictitious system while steering the direction of velocity. A major advantage of PVFC comes from the passivity of the closed-loop system, which makes it a good choice in some machining applications where excessive damage to the tool should be avoided, or in safety-critical applications such as human-robot interactions. In spite of the path following performance presented, this controller is complex as it requires the inertia matrix and Coriolis and centrifugal matrix as inputs. Additionally, one still needs to perform inverse kinematics to map the desired EEF velocities to desired joint motions.

1.3 Contributions and Organizations

This thesis takes on the manipulator path following problem and aims to design an inverse-free controller that is computationally efficient and simple to implement. In doing so, the transpose Jacobian control is selected as the base control law for the purpose of reaching (or set-point regulation), and a gyroscopic force is added as an auxiliary input for path following during motion.

The simple form of the controller presented in this thesis comes with its disadvantages. A prominent one is that though it has been verified that the controller drives the EEF to move along the desired path both in simulation and experiments, it cannot guarantee that the desired path is an invariant set of the closed-loop system like the other control laws mentioned in Section 1.2.4. However, if the performance is not satisfactory, one can simply increase the gain for the gyroscopic term for better path following result. This is a safe change because regardless of its magnitude, the gyroscopic force does not change the mechanical energy of the system.

The rest of this thesis is organized as follows. In Chapter 2, kinematic and dynamic modeling of a robot manipulator is presented, followed by an introduction of gyroscopic forces. Based on our published works in [64, 65] on path following of the EEF position, we verify the effectiveness of the controller via experiments in Chapter 3. We then extend the work to consider both the position and orientation of the EEF Chapter 4. Conclusions and future works are presented in Chapter 5.

Remark 1. Please note that the term "force" used in this thesis refers to generalized force vectors and is not limited to those with a unit of newton. It includes forces and moments in the task space as well as torques in the joint space.

Chapter 2

Mathematical Preliminaries

2.1 Modeling of Robot Manipulators

2.1.1 Kinematics

In this thesis, we use $Q \subseteq \mathbb{R}^n$ to denote the configuration space of an n DoF robot manipulator. The position vector which describes the configuration of the manipulator is denoted by $q = (q_1, ..., q_n)^T \in Q$; in other words, q represents the set of joint angles of the manipulator.

To keep our analysis general, we assume that the EEF has translational and rotational motions in 3D space. We denote the EEF pose in the task space by elements in the special Euclidean group $SE(3)$ as

$$g(q) = \begin{bmatrix} R(q) & x(q) \\ 0 & 1 \end{bmatrix} \tag{2.1}$$

where $x(q) \in \mathbb{R}^3$ represents the position with respect to the inertial frame, and $R(q) \in SO(3)$ is an orthonormal matrix that describes the orientation of the EEF. Vector parameterizations of the orientation are available, such as Euler angles and unit quaternions, but for reasons that will be stated in Chapter 4, we resolve to directly working with the matrix $R(q)$.

The velocity of the EEF can be represented in twist form as [14]

$$\xi = \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \dot{x} \\ (\dot{R}R^T)^\vee \end{bmatrix} = \begin{bmatrix} J_p(q) \\ J_o(q) \end{bmatrix} \dot{q} \tag{2.2}$$

where $J_p(q) = \frac{\partial x(q)}{\partial q}$ is the Jacobian matrix for position, and $J_o(q)$ is the Jacobian that maps the joint velocity to the angular velocity of the EEF with respect to the inertial frame. The vee operator $\bullet^\vee$ is the inverse mapping of wedge operator $\widehat{\bullet}$ which is defined as

$$\widehat{a} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{2.3}$$

for $a \in \mathbb{R}^3$.

The term forward kinematics refers to the problems of finding the pose and twist of the EEF given a set of joint position and velocity. A unique solution always exists and this problem is trivial given (2.1) and (2.2). Inverse kinematics is the problems of finding a set of joint position and velocity that correspond to a given EEF pose and twist. Because a minimal representation of $SE(3)$ is defined by 6 variables, there could be infinitely many solutions if $n > 6$. Additionally, there are cases where no solutions exist, when the desired pose is outside of the workspace of the manipulator or when the Jacobian is singular, which can make the desired twist infeasible.

2.1.2 Dynamics

The dyanmics of a rigid robot manipulator is described by the Euler-Lagrange equation:

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}}\right) - \frac{\partial \mathcal{L}}{\partial q} = \tau \tag{2.4}$$

where $\tau \in \mathbb{R}^n$ is the external force vector and $\mathcal{L} = T(q, \dot{q}) - U(q)$ is the Lagrangian of the system with kinetic energy $T = \frac{1}{2}\dot{q}^T M(q)\dot{q} \in \mathbb{R}$ and potential energy $U(q) \in \mathbb{R}$. In the robotics community, equation (2.4) is commonly expressed as:

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + N(q) = \tau \tag{2.5}$$

where $M(q)$ is the mass (or inertia) matrix which is symmetric and positive definite, $C(q, \dot{q})\dot{q}$ is the Coriolis and centrifugal forces, and $N(q) = \frac{\partial U(q)}{\partial q}$ is the gravity term. We will omit the arguments q and $\dot{q}$ in the remainder of this thesis to avoid clutter. However, it needs to be kept in mind that the term $C\dot{q}$ is in fact quadratic in velocity as

$$C_{ij}\dot{q}_j = \frac{1}{2}\Big(\frac{\partial M_{ij}}{\partial q_k} + \frac{\partial M_{ik}}{\partial q_j} - \frac{\partial M_{jk}}{\partial q_i}\Big)\dot{q}_j\dot{q}_k$$

In the above equation, we employed the Einstein summation convention, where a repeating index implies summation over that index. For instance, the term on the left-hand side, $C_{ij}\dot{q}_j$, implies it is summed over j just like one would do when post-multiplying the matrix C by the vector $\dot{q}$. This notation will be used repeatedly in the remainder of this thesis.

Among the properties of a robot manipulator, an important one is the skew-symmetry of the matrix $S = \dot{M} - 2C$:

$$\begin{aligned} S_{ij} &= \dot{M}_{ij} - 2C_{ij} \\ &= \frac{\partial M_{ij}}{\partial q_k}\dot{q}_k - \Big(\frac{\partial M_{ij}}{\partial q_k} + \frac{\partial M_{ik}}{\partial q_j} - \frac{\partial M_{jk}}{\partial q_i}\Big)\dot{q}_k \\ &= \Big(\frac{\partial M_{jk}}{\partial q_i} - \frac{\partial M_{ik}}{\partial q_j}\Big)\dot{q}_k \end{aligned}$$

It is obvious that $S_{ij} = -S_{ji}$, so $S = -S^T$ is skew-symmetric. This property is closely related to what is known as the passivity property of the robot manipulator.

Definition 2.1 (Passivity). *A dynamic system with input u and output y is passive with respect to the supply rate $s(u, y) = u^T y$ if for all $t_1 > t_0$,*

$$\int_{t_0}^{t_1} s(u, y)dt \geq -c^2 \tag{2.6}$$

where $c \in \mathbb{R}$ depends on the initial condition [63].

In the context of a robot manipulator, the passivity can be proved using the joint velocity $\dot{q}$ as the output and joint torque τ as the input. Let $E = T + U$ be the total energy of the system, i.e.

$$E = \frac{1}{2}\dot{q}^T M(q)\dot{q} + U(q)$$

Its time derivative, $\dot{E}$ is

$$\begin{aligned} \dot{E} &= \dot{q}^T M\ddot{q} + \frac{1}{2}\dot{q}^T\dot{M}\dot{q} + \dot{q}^T N \\ &= \dot{q}^T(\tau - C\dot{q} - N) + \frac{1}{2}\dot{q}^T\dot{M}\dot{q} + \dot{q}^T N \\ &= \dot{q}^T\tau + \frac{1}{2}\dot{q}^T \underbrace{(\dot{M} - 2C)}_{\text{skew-symmetric}}\dot{q} \\ &= \dot{q}^T\tau \end{aligned} \tag{2.7}$$

Substituting this into (2.6), we have

$$\int_{t_0}^{t_1} s(\tau, \dot{q})dt = \int_{t_0}^{t_1} \dot{E}dt = E(t_1) - E(t_0) \geq -E(t_0)$$

where the inequality comes from the fact that $E(t_1) \geq 0$. Setting $c = \sqrt{E(t_0)}$ proves the passivity of the system. In dissipative systems, this total energy E is referred to as the storage function. The interpretation of the passivity property is that, the rate of change in the total energy is governed by $\dot{q}^T\tau$ if we ignore the effect of friction and disturbances. In the absence of any external forces, the total energy of a robot manipulator will stay constant.

2.2 Overview of Gyroscopic Forces

The passivity property naturally leads us to gyroscopic forces, a concept introduced by Thomson (Lord Kelvin) and Tait (1879) [66]. This section will first give the definition of gyroscopic forces, followed by a brief review on the use of such forces in robot manipulator control.

2.2.1 Definition and Examples

Definition 2.2 (Gyroscopic Force)**.** *A non-zero input force τ is gyroscopic if it does not cause any energy change to a system in motion [67].*

For a robot manipulator, Definition 2.2 means that $\dot{q}^T\tau = 0$ at all times, or equivalently that a gyroscopic force is always orthogonal to the joint velocity.

Corollary 2.1. *Any force in the form of $\tau = W\dot{q}$, with $W = -W^T$, is a gyroscopic force.*

Corollary 2.1 can be easily proved by observing that $\dot{q}^T W\dot{q} = 0$ for any skew-symmetric matrix W. It immediately follows from the skew-symmetry property that the force $(\frac{1}{2}\dot{M} - C)\dot{q}$ is gyroscopic.

Let us for a moment forget about robot manipulators and consider a simple example with a point mass moving in a horizontal circle at a constant speed, as illustrated in Figure 2.1. It is obvious that the total energy of the point mass stays constant, and the centripetal force acting on the mass is always orthogonal to the velocity. Therefore, the centripetal force is a perfect example of a gyroscopic force. As opposed to $(\frac{1}{2}\dot{M} - C)\dot{q}$ whose direction varies along the trajectory, this suggests that gyroscopic forces can be formulated with strong directionality. In this example, it naturally shows up as a steering force.

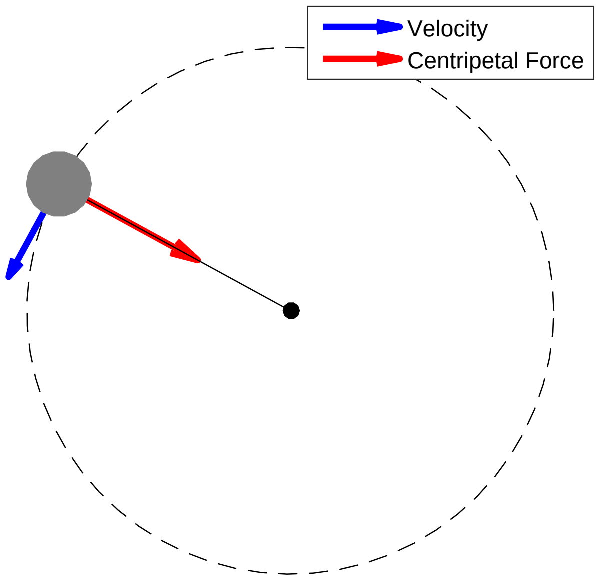


Figure 2.1: Centripetal force as an example of gyroscopic force

2.2.2 Existing Works Involving Gyroscopic Forces

The idea of using gyroscopic forces to steer motion has been explored in the context of mobile agents for obstacle avoidance and formation control [68, 69]. On the other hand, the use of these forces in robot manipulators is somewhat rare. One example includes the work on controlled Lagrangian by Chang, who used gyroscopic forces as a part of the feedback control to stabilize under-actuated systems through energy and force shaping [70, 46, 47]. This method relies on solving complicated partial differential equations known as the matching conditions. As opposed to controlled Lagrangian methods with no force shaping, gyroscopic forces allows one to expand the region of convergence and enjoy an extra flexibility to tune the performance [70]. The PVFC design also incorporates the idea of gyroscopic forces [62, 63], though the control is gyroscopic in the augmented system including the fictitious states but not in the manipulator system itself.

Since energy shaping controls such as the transpose Jacobian controller use the total energy of the system as the Lyapunov function for proof of convergence [12], the addition of a gyroscopic force to the controller will not affect the convergence. In light of [68] which used gyroscopic forces as a steering force for mobile agents to avoid obstacles, one can use gyroscopic forces in manipulator control to constrain the path profile of the EEF without modifying the transpose Jacobian control gains.

Another distinction between the use of gyroscopic forces in this thesis and that in the works mentioned above is that, we propose the design of a gyroscopic force directly in the task space to avoid any inverse transformations. A gyroscopic force in the task space is more generally known as a reciprocal wrench. The concept of reciprocal screws have been widely used in areas such as grasp analysis [71] and mobility analysis of parallel manipulators [72] and other complex kinematic systems [73]. By considering the force or geometric constraints, these studies generally aim to determine the DoF and/or possible directions of motion of a system. Our approach is somewhat the opposite way: given the position and velocity (or twist) of the EEF, we look to find a reciprocal wrench that can lead to the desired motion.

To the best of the author's knowledge, the most similar use of gyroscopic force to that in this thesis lies in the field of neuroscience when studying human motor controls. In [74, 75, 76], experiments were conducted for the human planar reaching motion when subjects hold the EEF of a robot manipulator. The baseline setting is when the manipulator is not supplying any torque, and in some trials, a perturbing force is exerted by the EEF in the form of

$$F = K \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \dot{x}$$

where $K \in \mathbb{R}$ is the gain term that adjusts the magnitude of the perturbation. By doing so, researchers were able to gain insights on how the human CNS adapts to the environment. It is obvious that by Corollary 2.1, this perturbation is in fact a gyroscopic force in the task space. In the field of neuroscience, instead of gyroscopic forces, this type of force field is more commonly known as a viscous curl field because it is indeed a divergence-free curl field in the space of the EEF velocity, as illustrated in Figure 2.2. Though it was not explicitly stated, an important reason for selecting such type of perturbation over others is its energy-conserving nature. In other words, adding a gyroscopic force is a safer operation compared to perturbations that may inject more energy into the system. Such property is especially important in these experiments since human subjects are involved.

As a result of this perturbation, it was observed that human subjects initially had

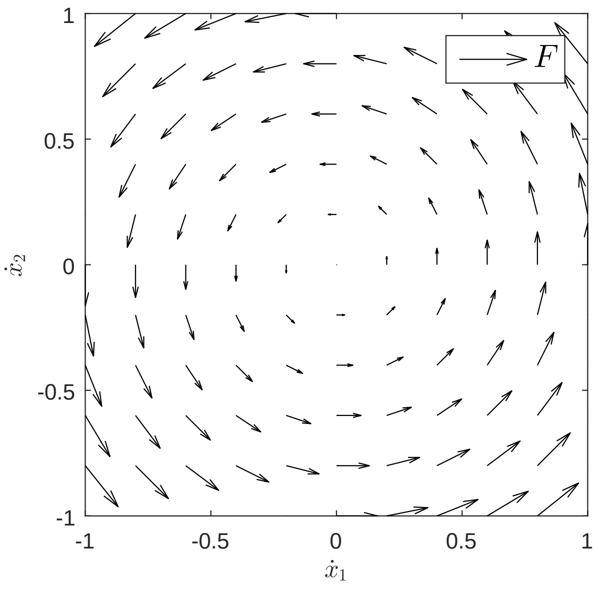


Figure 2.2: Viscous curl field of the perturbing force

significant deviations from the straight line path, but as more trials are conducted, they gradually adapted to the presence of such force field and learned to compensate for it so that the resulting path becomes quasi-straight again. The methods proposed in this thesis can be understood as the reverse of this perturbation: if a deviation from the desired path is present, gyroscopic forces can assist the control to correct this deviation.

Chapter 3

Position Control of the End Effector

In this chapter, we consider the case where only the position of the EEF during motion is of interest. In other words, we are concerned about $x(p)$ but not $R(p)$ of the forward kinematics. This is consistent with the control objective of most path following problems. The straight-line property of human reaching motions can be reproduced with position control. By assuming that a tool is rigidly attached to the EEF or that the EEF has a wrist with independent orientation control, such position control can lead to many useful applications such as drawing and handwriting [56, 53].

We will first present a brief review of the transpose Jacobian control in Section 3.1. The formulation of gyroscopic forces will be discussed in Section 3.2, followed by simulation and experimental results in Sections 3.3, 3.4, and 3.5.

3.1 Transpose Jacobian Control

Consider a pendulum with its center of rotation fixed to a vertical wall, and in the absence of any other forces, we know that it will always converge to the downward pointing configuration at which the potential energy is at the minimum. The vertically upward configuration is also an equilibrium point, but it is unstable and a very small disturbance will cause it to collapse. Now, if a motor is available at the center of rotation to perfectly cancel out the torque due to gravity, then the pendulum will behave as if it is placed on a horizontal plane, and the equilibrium can be any configuration. If the initial velocity is zero, then the pendulum will remain at its initial position. For non-zero initial velocities, it will theoretically keep moving at a constant speed, but friction will cause it to stop eventually. If the objective is to regulate the position to a set-point, we can then add a gravity-like input at the motor to create an APE that has a unique minimum at the desired position. The closed-loop system will behave as if the gravitational field is rotated favoring the set-point. The only exception is when the pendulum starts with an initial position exactly opposite to this set-point. Similar to an inverted pendulum, this is an unstable equilibrium which can be avoided by small perturbations.

The logic described in the example above is exactly the same thinking behind the transpose Jacobian control. Given a robot manipulator with Lagrangian $\mathcal{L} = T(q, \dot{q}) - U(q)$, we seek to synthesize a controlled Lagrangian $\hat{\mathcal{L}} = T(q, \dot{q}) - \hat{U}_p(x)$ by replacing the original potential energy $U(q)$ with an artificial one $\hat{U}_p(x) = \hat{U}_p(x(q))$ which is parameterized by the deviation of the task space coordinate $x = p(q)$ from its desired value, denoted by $\bar{x}$ in this thesis. After that, we inject the viscous damping term for stabilization. The resulting control law is given by [12, 55]

$$\tau = N(q) + \frac{\partial}{\partial q}\hat{U}_p(x) - C_d\dot{q} \tag{3.1}$$

where $C_d \in \mathbb{R}^{n\times n}$ is the damping matrix which is positive definite (and usually diagonal). The purpose of the first term in (3.1) is to compensate for gravity so that the original potential energy is cancelled out. The second term is the artificial gravitational force that causes the closed-loop system to have the APE. Commonly, the artificial potential function is designed in a simple quadratic form, $\hat{U}_p(x) = \frac{1}{2}\left(\bar{x} - x\right)^T K_p \left(\bar{x} - x\right)$, for some gain matrix $K_p \in \mathbb{R}^{3\times 3}$ which is positive definite (and usually diagonal). The resulting transpose Jacobian control law is then

$$\tau = N(q) + J_p(q)^T K_p(\bar{x} - x) - C_d\dot{q} \tag{3.2}$$

where $J_p(q)$ is the Jacobian matrix for the EEF position from (2.2). This control law is named the transpose Jacobian control because of the use of $J_p^T(q)$. Figure 3.1 shows 2 intuitive interpretations of this controller. It effectively causes the EEF to travel in a potential field towards the minimum point. Alternatively, one can think of it as placing a virtual spring and damper directly at the EEF with the target position as the equilibrium point. We formalize the convergence of the transpose Jacobian control as a theorem here:

Theorem 3.1. *For the closed-loop system with a robot manipulator* (2.5) *and the transpose Jacobian control law* (3.2)*, the equilibrium points characterized by $x(q) = \bar{x}$ and $\dot{q} = 0$ are asymptotically stable.*

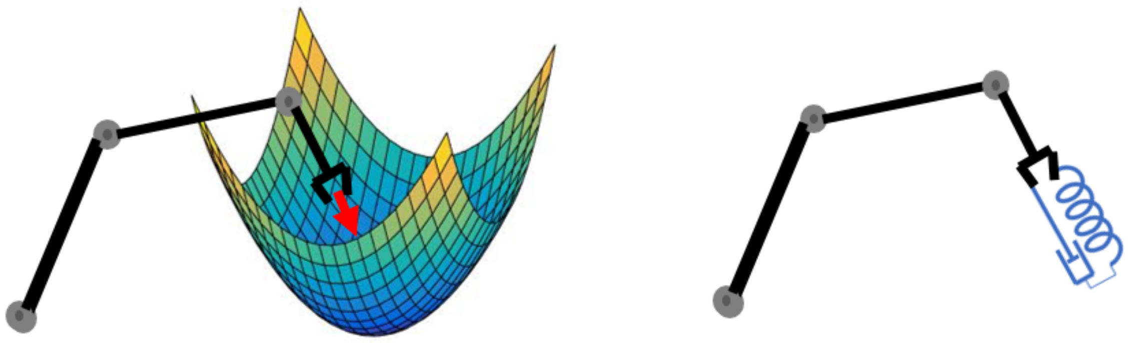


Figure 3.1: Interpretations of the transpose Jacobian control: potential energy field (left) and task space spring-and-damper (right)

Proof. We first assume that the system is non-redundant; in other words, the dimension of the task space equals to n. If $\bar{x}$ is within the workspace of the manipulator, there exists an isolated joint configuration $q_d \in Q$ satisfying $x(q_d) = \bar{x}$ [77]. Consider the total artificial energy as the Lyapunov function candidate:

$$V = T(q, \dot{q}) + \hat{U}_p(x(q)) = \frac{1}{2}\dot{q}^T M \dot{q} + \frac{1}{2}(\bar{x} - x)^T K_p (\bar{x} - x) \tag{3.3}$$

Note that the first term is the kinetic energy of the manipulator and is positive definite with respect to $\dot{q}$. The second term is positive definite with respect to $\bar{x} - x$ but not q. However, because q_d is an isolated minimum point of $(\bar{x} - x(q))^T K_p (\bar{x} - x(q))$, we conclude that the second term is locally positive definite with respect to $q_d - q$ [77]. Its time derivative is as follows:

$$\begin{aligned}
\dot{V} &= \dot{q}^T M \ddot{q} + \frac{1}{2}\dot{q}^T \dot{M} \dot{q} - \dot{x}^T K_p(\bar{x} - x) \\
&= \dot{q}^T (\tau - C\dot{q} - N + \frac{1}{2}\dot{M}\dot{q}) - (J_p \dot{q})^T K_p(\bar{x} - x) \\
&= \dot{q}^T \Big(N + J_p^T K_p(\bar{x} - x) - C_d\dot{q} - C\dot{q} - N + \frac{1}{2}\dot{M}\dot{q}\Big) - \dot{q}^T J_p^T K_p(\bar{x} - x) \\
&= \dot{q}^T \Big(-C_d\dot{q} + (\frac{1}{2}\dot{M} - C)\dot{q}\Big) \\
&= -\dot{q}^T C_d \dot{q} \quad \leq 0
\end{aligned} \tag{3.4}$$

where the skew-symmetry property of $\frac{1}{2}\dot{M} - C$ was used for the last equality. At this point, we are left with $\dot{V} = 0$ for any states with $\dot{q} = 0$, so convergence is not proved just yet. We invoke LaSalle's invariance principle [78] by analyzing the closed-loop system dynamics on the set of states characterized by $\dot{q} = 0$:

$$\begin{aligned} M\ddot{q} + C\dot{q} + N = \tau &= N + J_p^T K_p(\bar{x} - x) - C_d\dot{q} \\ M\ddot{q} &= J_p^T K_p(\bar{x} - x) \\ \ddot{q} &= M^{-1} J_p^T K_p(\bar{x} - x) \end{aligned} \tag{3.5}$$

Since $\ddot{q} = 0$ is satisfied only when $x = \bar{x}$ (or locally $q = q_d$), we conclude that the equilibrium points with $q = q_d$ and $\dot{q} = 0$ are locally asymptotically stable.

In the proof above, we used the fact that for a non-redundant manipulator, the transformation between joint space and task space is locally injective (or one-to-one). Such property does not hold for a redundant manipulator, in which case the APE is not positive definite with respect to the joint position. To deal with this complication, Arimoto introduced the notion of "stability on a manifold" and "transferability to a submanifold" to prove the local asymptotic stability of the closed-loop system, and the interested reader can refer to [53, 54] for a detailed proof. □

Remark 2. Though not explicitly stated in Theorem 3.1, from the author's experience with simulation and experiment, the closed-loop system in general exhibits almost global asymptotic convergence of x to $\bar{x}$, i.e. $\lim_{t\to\infty} x(t) = \bar{x}$. The following are some of the special scenarios where convergence may not be achieved:

- $\bar{x}$ is outside of the work space of the robot manipulator. This condition can be easily verified so it should not be a concern. If no joint angle limit is present, then the transpose Jacobian control will drive the manipulator to a singular configuration that minimizes the $\|x - \bar{x}\|_2$.

- If the initial condition starts from an equilibrium point in the task space other than $x(q_0) = \bar{x}$, in which case $\frac{\partial}{\partial q}\hat{U}_p(x(q_0)) = 0$. Provided that the mapping $\hat{U}_p(x(q)) : \mathbb{R}^n \to \mathbb{R}$ is of class C^∞ (because both $x(q)$ and $U_p(x)$ are smooth), we invoke Sard's Theorem [79] to conclude that the critical values of this mapping is of measure zero. In other words, these initial configurations are rare in the sense that they occupy zero volume in the space of $\mathbb{R}^n$. Analogous to the unstable equilibrium of an inverted pendulum, these initial conditions are unstable, and a small perturbation will cause the EEF to leave these equilibrium points.

Remark 3. Although here we only showed the proof for the APE in the quadratic form $\hat{U}_p(x) = \frac{1}{2}(\bar{x} - x)^T K_p (\bar{x} - x)$, the same applies to all smooth APEs with a global minimum at $x = \bar{x}$ as long as the control law is derived using (3.1).

In the presence of large friction force in some manipulator systems, it was noted that (3.2) sometimes cannot provide enough joint torque command to overcome the Coulomb friction, leaving a non-zero steady state error. This issue can be mitigated with larger gain values in K_p, but in cases where the EEF starts from a position far from its target, the control law (3.2) will saturate the joint torque limits. To address this problem, the authors of [77] recommended an alternative form of the APE

$$\hat{U}_p(x) = \sum_{i=1}^{3} \frac{k_i}{\lambda_i} \ln[\cosh(\lambda(\bar{x}_i - x_i))] \tag{3.6}$$

where k_i is the ith diagonal element of a positive definite and diagonal matrix K_p, and λ is a positive constant. This results in the following control law:

$$\tau = N(q) + J_p(q)^T K_p \begin{bmatrix} \tanh(\lambda(\bar{x}_1 - x_1)) \\ \tanh(\lambda(\bar{x}_2 - x_2)) \\ \tanh(\lambda(\bar{x}_3 - x_3)) \end{bmatrix} - C_d \dot{q} \tag{3.7}$$

where the hyperbolic tangent function is utilized to constrain the torque command when the x is far from its target. When the value of λ is large, the second term in (3.7) is essentially constant prior to the mapping by $J_p(q)^T$, and only decreases until x enters a small neighborhood near $\bar{x}$. If $k_1 = k_2 = k_3 = k$, one can understand (3.6) as a smooth approximation of an APE linear in the position error, i.e.

$$\hat{U}_p(x) = \sum_{i=1}^{3} \frac{k_i}{\lambda_i} \ln[\cosh(\lambda(\bar{x}_i - x_i))] \approx k\sqrt{3}\|\bar{x} - x\|_2 \quad \text{for large } \lambda\|\bar{x} - x\|_2$$

The beauty of the transpose Jacobian control law is that the convergence of the EEF position is guaranteed though no inverse transformation is needed. It does not require an accurate dynamical model of the manipulator to perform inverse dynamics calculation as in computed torque control schemes. Inverse kinematics is also avoided so the ill-posedness is not an issue. Transpose Jacobian also enjoys the property of being numerically stable, especially near singular configurations when compared to using the inverse or pseudo-inverse of the Jacobian matrix.

Despite the advantages of the transpose Jacobian control, it can only guarantee the convergence of the EEF pose, while the intermediate path profile is not directly controlled. Although the authors showed in [53, 54, 55] that motion profiles similar to human reaching can be reproduced by carefully selecting the control gains K_p and C_d, the gain tuning is more of a trial-and-error process rather than a systematic approach. Given a new set of initial configuration and final pose, one has to re-tune the gains to achieve similar performances. Simulation results of using the same transpose Jacobian control gain for a 3-DoF robot manipulator are plotted in Figure 3.2. The manipulator starts from different initial configurations but the target EEF location is the same for all trials. The initial EEF position is marked by 'x' and the target is marked by 'o'. Observe that the path profiles can be drastically different though the same gain is used. The case where the EEF starts from $x = \mathrm{col}(-0.52, 0.35)m$ has a path profile that is closer to a quasi-straight line, but the other trials have almost unpredictable results in terms of the EEF path.

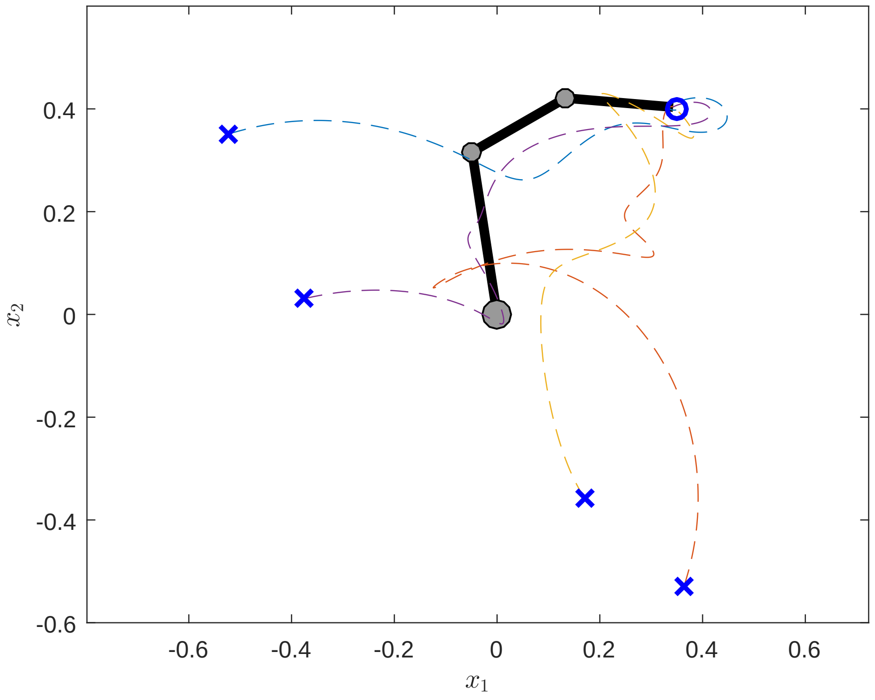


Figure 3.2: Simulation of the transpose Jacobian control for different initial conditions

Remark 4. Although the transpose Jacobian control (3.1) has a term of $N(q)$, this gravity compensation can be estimated accurately using the recursive Newton-Euler method [15]. Many commercial robot manipulators have built-in gravity calibration and compensation algorithms, so we assume that this $N(q)$ term does not break the inverse free principle of the controller.

3.2 Gyroscopic Force Shaping

3.2.1 Transpose Jacobian Control with Gyroscopic Force

We first observe that the Lyapunov function in (3.3) is the total (artificial) energy of the closed-loop system, which is not affected by a gyroscopic force. Consider if a gyroscopic component τ^g is added to the transpose Jacobian control:

$$\tau = N + \frac{\partial}{\partial q}\hat{U}_p - C_d\dot{q} + \tau^g \tag{3.8}$$

Lemma 3.1. *For the closed-loop system with a robot manipulator (2.5) and the control law (3.8) with τ^g being gyroscopic (i.e. $\dot{q}^T\tau^g = 0$), the equilibrium points characterized by $x(q) = \bar{x}$ and $\dot{q} = 0$ are asymptotically stable.*

Proof. The proof is very similar to that of Theorem 3.1. Consider again the total artificial energy as the Lyapunov function candidate:

$$V = T + \hat{U}_p = \frac{1}{2}\dot{q}^T M\dot{q} + \hat{U}_p \tag{3.9}$$

We take its time derivative:

$$\begin{aligned}
\dot{V} &= \dot{q}^T M\ddot{q} + \frac{1}{2}\dot{q}^T\dot{M}\dot{q} - \dot{q}^T\frac{\partial}{\partial q}\hat{U}_p \\
&= \dot{q}^T(\tau - C\dot{q} - N + \frac{1}{2}\dot{M}\dot{q}) - \dot{q}^T\frac{\partial}{\partial q}\hat{U}_p \\
&= \dot{q}^T\Big(N + \frac{\partial}{\partial q}\hat{U}_p - C_d\dot{q} + \tau^g - C\dot{q} - N + \frac{1}{2}\dot{M}\dot{q}\Big) - \dot{q}^T\frac{\partial}{\partial q}\hat{U}_p \\
&= \dot{q}^T\Big(-C_d\dot{q} + (\frac{1}{2}\dot{M} - C)\dot{q} + \tau^g\Big) \\
&= -\dot{q}^T C_d\dot{q} \quad \leq 0
\end{aligned} \tag{3.10}$$

where the last equality used the fact that $\frac{1}{2}\dot{M} - C$ is skew-symmetric and τ^g is gyroscopic. The asymptotic stability is then proved using LaSalle's invariance principle [78] in the same fashion as Theorem 3.1 and is not repeated here. □

Because a major advantage of the transpose Jacobian control is that it is free of inverse kinematics, it defeats the purpose if the additional gyroscopic force term τ^g is formulated in the joint space. Our goal is to formulate the gyroscopic force in the task space as well. When mapping a task space gyroscopic force F_p^g into a joint torque by $\tau_p^g = J_p^T F_p^g$, the following lemma states that the energy-conserving nature is preserved:

Lemma 3.2. *A gyroscopic force F_p^g in the task space corresponds to a gyroscopic torque in the joint space when mapped by J_p^T.*

Proof. Substituting $\dot{x} = J_p\dot{q}$ into $\dot{x}^T F_p^g = 0$, we have:

$$0 = \dot{x}^T F_p^g = (J_p\dot{q})^T F_p^g = \dot{q}^T J_p^T F_p^g = \dot{q}^T \tau_p^g \tag{3.11}$$

Clearly, τ_p^g is gyroscopic in the joint space by Definition 2.2. □

The implication of Lemma 3.2 is that, we can now directly design gyroscopic forces in the task space. The corresponding joint torque mapped from the transpose Jacobian matrix will automatically be guaranteed to also be gyroscopic. The final control law is then

$$\tau = N + \frac{\partial}{\partial q}\hat{U}_p - C_d\dot{q} + J_p^T F_p^g \tag{3.12}$$

We state the convergence of x to the target $\bar{x}$ in a theorem:

Theorem 3.2. *For the closed-loop system with a robot manipulator (2.5) and the control law (3.12) with F_p^g being gyroscopic in the task space(i.e. $\dot{x}^T F_p^g = 0$), the equilibrium points characterized by $x(q) = \bar{x}$ and $\dot{q} = 0$ are asymptotically stable.*

Proof. The proof is straightforward by combining Lemma 3.2 and 3.1. □

3.2.2 Task Space Gyroscopic Force Design

Theorem 3.2 implies that $\lim_{t\to\infty} x(t) = \bar{x}$ in general. Our question now is then how to design the gyroscopic force F_p^g in the task space so the EEF path following objective can be achieved. We proposed 2 different formulations in [64]. The first method directly uses the property of the desired path and results in a gyroscopic force linear in velocity. This approach is subject to limitations on the admissible paths and is in general more difficult to formulate. In this thesis, we focus on the desired velocity field approach which results in a gyroscopic force quadratic in velocity.

Given the dimension m of the task space ($m = 2$ for planar movements and $m = 3$ for movements in 3D). Assuming that a pre-defined smooth vector field $\bar{v} : \mathbb{R}^m \to \mathbb{R}^m, m \in \{2, 3\}$ is available as the desired velocity at each point in the task space, as illustrated in Figure 3.3. In this case, the task space gyroscopic force depicted by the arrow with white triangle head can be constructed as

$$\begin{aligned} F_p^g\left(\bar{v}, x, \dot{x}\right) &= k_p^g \dot{x} \times \left(\bar{v} \times \dot{x}\right) \\ &= k_p^g \underbrace{\left(\dot{x}\bar{v}^T - \bar{v}\dot{x}^T\right)}_{\text{skew-symmetric}} \dot{x} \end{aligned} \tag{3.13}$$

where the second equality is from the triple product identities, and $k_p^g > 0$ is the gain for the gyroscopic force. It is easy to verify from the cross-product and the skew-symmetry that $\dot{x}^T F_p^g = 0$.

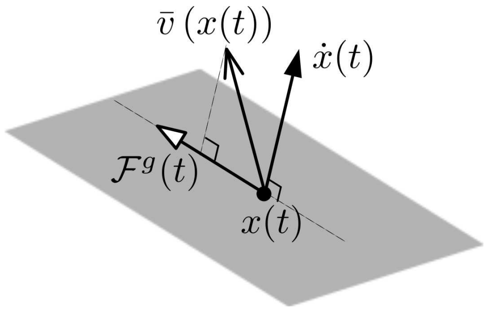


Figure 3.3: Construction of F_p^g quadratic in velocity

As illustrated by Figure 3.3, this formulation of the gyroscopic force makes F_p^g in (3.13) pull $\dot{x}$ toward $\bar{v}$ at all times. As a result of the cross product, it will pull "harder" when the angle between $\dot{x}$ and $\bar{v}$ is greater. Also, the magnitude of F_p^g is quadratically proportional to that of $\dot{x}$, thereby increasing the gyroscopic effect quadratically as the EEF speed increases. In some sense, this form of gyroscopic force is also natural when compared to that linear in velocity and those of higher orders, because the Coriolis and centrifugal force $C\dot{q}$ that exists in the internal dynamics of the manipulator is also quadratic in velocity. One can consider this approach as gyroscopic force shaping because we modified the internal gyroscopic force from $\frac{1}{2}(\dot{M} - 2C)\dot{q}$ to $\frac{1}{2}(\dot{M} - 2C)\dot{q} + J_p^T F_p^g$.

The question now is how to formulate a desired velocity field $\bar{v}(x)$ for our control purposes. We first look at the simple example of reaching motion in 2D along the negative

x_1 axis, with the desired path being a straight line. The velocity field in Figure 3.4 can be used; it is described by

$$\bar{\mathcal{V}}(x) = \frac{1}{\sqrt{1+\rho^2 x_2^2}} \begin{bmatrix} -1 \\ -\rho x_2 \end{bmatrix} \tag{3.14}$$

where the parameter $\rho > 0$ determines the regularity of the vector field around the path. Large values of ρ essentially makes the desired velocity field discontinuous: along the path, the velocity field points from $x(t_0)$ to $\bar{x}$; at locations away from the desired path, the velocity field will point perpendicularly towards the path. Note that this idea is analogous to sliding mode control with the desired path acting as the sliding surface.

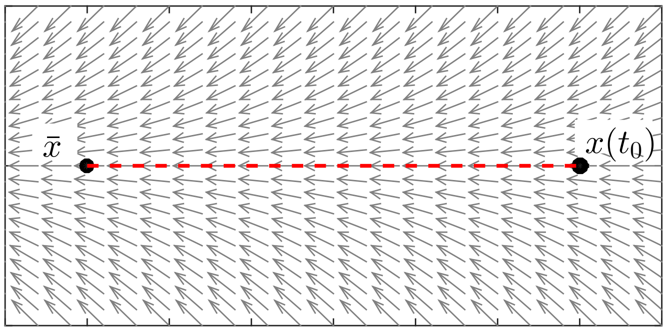


Figure 3.4: Velocity field shape for rectilinear path

Though Figure 3.4 is an example for a specific set of $\bar{x}$ and $x(t_0)$, one can apply simple translations and rotations to suit it for different initial/target positions . Given arbitrary $x(t_0)$ and $\bar{x}$, a new desired velocity field $\bar{v}(x)$ can be constructed from $\bar{\mathcal{V}}(x)$ as

$$\bar{v}(x) = R^T(\theta)\bar{\mathcal{V}}\left(R(\theta)\frac{x(t)-\bar{x}}{\|x(t_0)-\bar{x}\|}\right) \tag{3.15}$$

where $R(\theta)$ is a rotation matrix with θ being the angle between the positive x_1 axis and the vector $x(t_0) - \bar{x}$. If the reaching motion occurs in 3D, one can define the base vector field as:

$$\bar{\mathcal{V}}(x) = \frac{1}{\sqrt{1+\rho^2 x_2^2+\rho^2 x_3^2}} \begin{bmatrix} -1 \\ -\rho x_2 \\ -\rho x_3 \end{bmatrix} \tag{3.16}$$

and then apply transformations similar to that in (3.15) for different initial and target conditions.

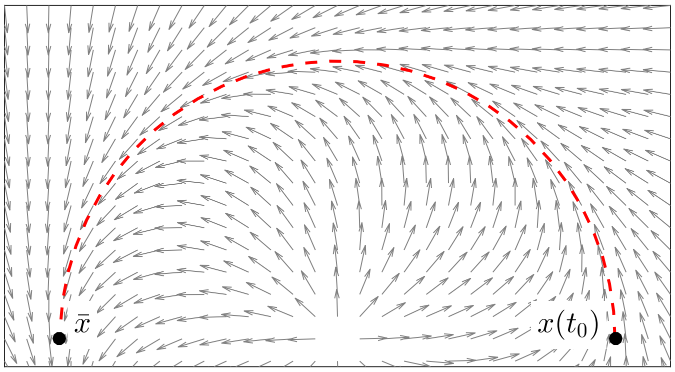


Figure 3.5: Velocity field shape for circular path

Figure 3.5 is an example velocity field that encodes a circular path. Assuming that the center of the curve is located at $x = 0$, we design the base velocity field as

$$\bar{\mathcal{V}}(x) = \begin{bmatrix} \bar{\mathcal{V}}_1(x) \\ \bar{\mathcal{V}}_2(x) \end{bmatrix} = \begin{bmatrix} -\beta x_2 - \alpha x_1(1 - \frac{r}{\|x\|)}) \\ \beta x_1 - \alpha x_2(1 - \frac{r}{\|x\|)}) \end{bmatrix}, \quad \forall \|x\| \neq 0 \tag{3.17}$$

where $r > 0$ is the radius of the curve, and α and β define the direction and shape of the velocity field. We assign $\bar{\mathcal{V}} = 0$ at the center of the circle. Again, one can apply affine transformation to (3.17) for applications where the center of the circle is not at the origin or to stretch the velocity field to get elliptical or spline paths.

In summary, we have synthesized potential energy shaping with gyroscopic force shaping for path following. The control law is

$$\tau = N + \frac{\partial}{\partial q}\hat{U}_p - C_d\dot{q} + k_p^g J_p^T \left(\dot{x}\bar{v}^T - \bar{v}\dot{x}^T\right)\dot{x} \tag{3.18}$$

where $\hat{U}_p$ is the APE and $\bar{v}$ is the given velocity field for the desired path. This control law enjoys the property of simplicity and is free of inverse kinematics. A drawback of this approach compared to those discussed in Section 1.2.4 is that, it does not convert the desired path into an invariant set for the closed-loop system. However, because of the energy-conserving nature of the gyroscopic force, one can often simply increase the gain k_p^g to achieve better path result. A convergence analysis of the path control by gyroscopic force is presented in Appendix A.

3.3 Experimental Results using a 2-DoF Manipulator

We verified the controller in (3.18) with a 2-DoF planar manipulator in the Advanced Robotics Laboratory at the University of Waterloo, illustrated in Figure 3.6. The manipulator is mounted horizontally on a steel table so the effect of gravity can be neglected. Table 3.1 summarizes the physical parameters of the experimental setup. The joints of the manipulator are each powered by a direct drive motor equipped with a 16-*bit* encoder from *Yaskawa, Inc.* The motors are capable of supplying up to $30Nm$ and $6Nm$ respectively at joint 1 and joint 2. A *National Instruments* cRIO-9030 real-time controller is used to control the motors at a frequency of $400Hz$.

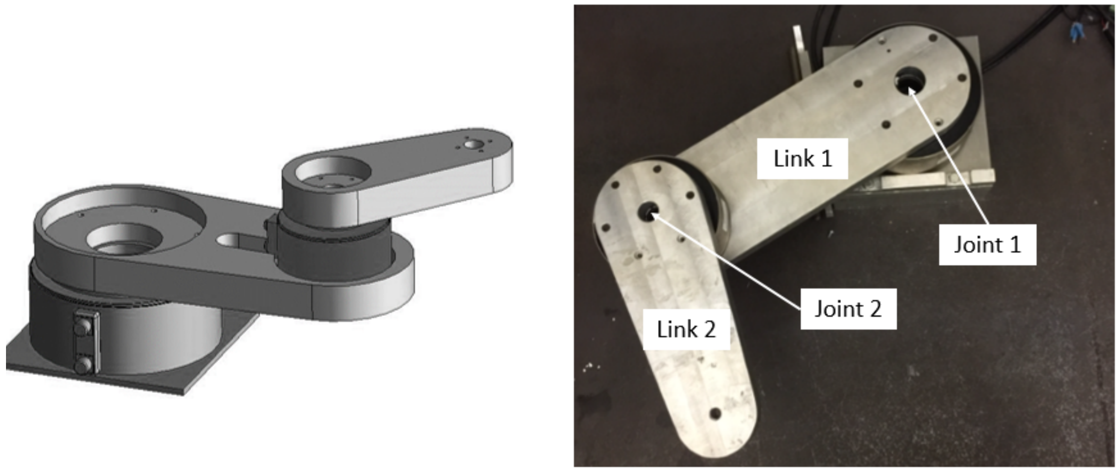


Figure 3.6: Experimental setup of the 2-DoF manipulator.

In a few first attempts to use the transpose Jacobian controller with quadratic form of APE (3.2), noticeable steady state errors on the order of $5cm$ were observed; in other words, because of friction, the EEF never reached the target but instead stopped somewhere close to it. We opted to use the alternative form of APE in (3.6) to mitigate this issue. A good reaching accuracy (within $1.5cm$ from the target) was achieved by setting the following gain values:

$$K_p = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}, C_d = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}, \lambda = 5$$

A series of experiments were then conducted with the same base controller with various parameters for the gyroscopic force term. The manipulator starts with an initial

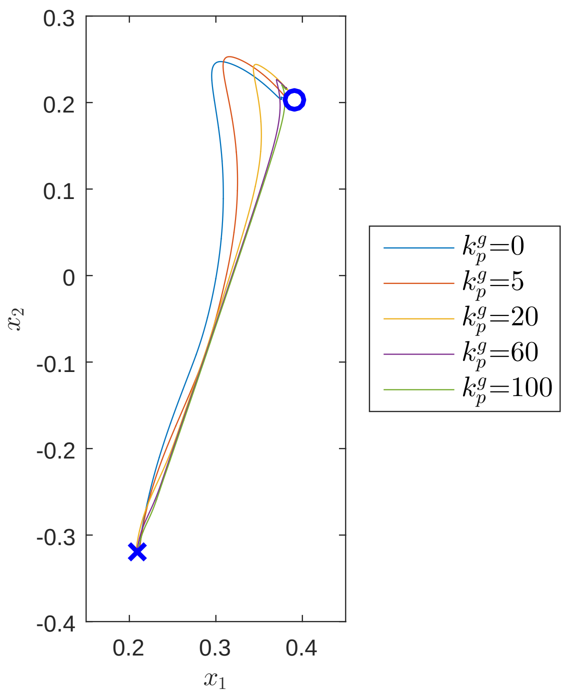
Table 3.1: Parameters of the 2-DoF manipulator.

Parameter	**Symbol**	**Value**	**Unit**
Link 1 length	l_1	0.32	m
Link 2 length	l_2	0.21	m
Link 1 mass	m_1	9.244	kg
Link 2 mass	m_2	3.529	kg
Link 1 center of mass coordinate from joint 1	r_1	0.16	m
Link 2 center of mass coordinate from joint 2	r_2	0.046	m
Link 1 moment of inertia at center of mass	I_1	0.2097	kgm^2
Link 2 moment of inertia at center of mass	I_2	0.0206	kgm^2

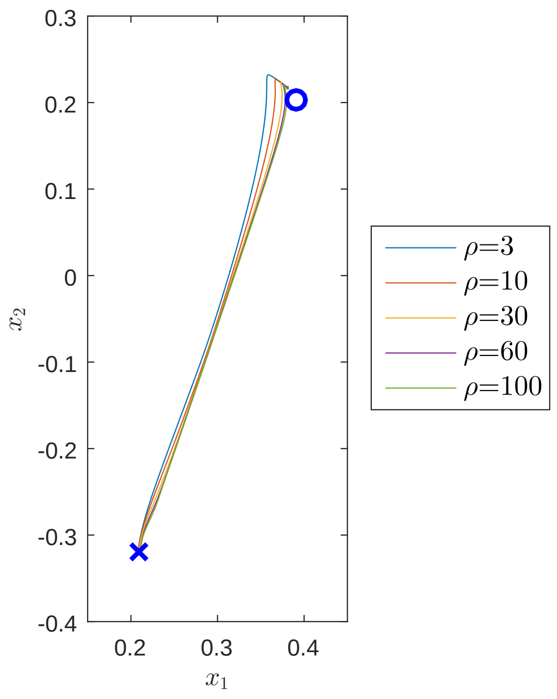
configuration of $q(t_0) = \text{col}(-\frac{\pi}{2}, \frac{\pi}{2})$ which corresponds to an EEF position of $x(t_0) = \text{col}(0.21, -0.32)m$. The desired path for is a rectilinear path from $x(t_0)$ to $\bar{x} = \text{col}(0.39, 0.204)m$.

First, by fixing the velocity field shape in (3.14) with $\rho = 100$, the effect of different gyroscopic force gain k_p^g is shown in Figure 3.7(a). Under the base transpose Jacobian control with no gyroscopic force ($k_p^g = 0$), the EEF deviates from the desired path, especially when getting close to $\bar{x}$. This deviation is significantly reduced when gyroscopic force is added, with better performances under larger values of k_p^g. When $k_p^g > 60$ is used, the resulting path essentially overlaps with the straight line, except for a small region near $\bar{x}$. This is because the gyroscopic force term in (3.13) is quadratic in the EEF velocity, which is low at the beginning and the end of the movement period.

We then analyzed the effect of the velocity field on the path profile by varying ρ while keeping the gyroscopic force gain constant at $k_p^g = 100$. Figure 3.7(b) shows the result when ρ varies from 3 to 100. Larger ρ helps straighten the resulting path, but the effect is small when compared to k_p^g. The difference in path profile is almost unnoticeable between $\rho = 60$ and $\rho = 100$.



(a)



(b)

Figure 3.7: Experimental results for rectilinear path with using the 2-DoF manipulator. **(a)**: Effect of k_p^g on the path profile. The case with $k_p^g = 0$ is transpose Jacobian control with no gyroscopic force. **(b)**: Effect of ρ on the path profile.

3.4 Simulation Results using a 3-DoF Manipulator

Simulations with a 3-DoF planar manipulator were conducted to test the effectiveness of the proposed controller in a redundant scenario. The first 2 links are set to be the same as the 2-DoF manipulator in Table 3.1, and the last link is assumed to be the same as the second link. The equations of motion describing the dynamics of the system (2.4) are derived in *Matlab*. Appendix B lists the codes that generate the mass matrix and the Coriolis and centrifugal forces.

We first verify the control law for rectilinear paths. Recall that, in Figure 3.2, the results of using only the transpose Jacobian control are plotted, indicating the uneven performance on path profiles. The simulations in Figure 3.2 uses a quadratic APE with the same control gains:

$$K_p = \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix}, \quad C_d = \begin{bmatrix} 0.7 & 0 & 0 \\ 0 & 0.7 & 0 \\ 0 & 0 & 0.7 \end{bmatrix}$$

We then added gyroscopic force (3.13) with desired velocity field in (3.14). Setting $\rho = 100$ and $k_p^g = 100$, the results are plotted in Figure 3.8. Some overshoot is present as the EEF moves past the target in some cases; this can be dealt with by tuning the transpose Jacobian control.

Similar to the experimental results for the 2-DoF manipulator, the EEF paths were significantly straightened by the gyroscopic force. The effect is not the same for all trials though, as some deviations can be observed for the blue and purple paths but not the other 2 paths, except in a small neighborhood near $\bar{x}$. This could be a result of lower velocities from the APE since their initial EEF positions are closer to the target. One can increase the gain k_p^g to mitigate this, but the paths presented in Figure 3.8 are already much straighter than those in Figure 3.2 and are satisfactory for the purpose of reproducing the paths of human reaching motion. Another contributing factor is that the inertia effect is not uniform during the movement, so gyroscopic forces of the same magnitude will have varying effectiveness along the trajectory. One can cope with the inertia effect by incorporating properties of the inertia ellipsoid [80] if the mass matrix is known, as introduced in [64]. Such approach is not presented here because this thesis aims for an inverse-free control law.

It was also noticed that this formulation of gyroscopic forces in general leads to shorter time needed to reach the target. An intuitive explanation for this is that, when starting with the same amount of energy defined by the APE, manipulators with gyroscopic force control follows a linear path from $x(t_0)$ to $\bar{x}$ which is shorter than any other choice of path, and hence does not "waste" any time by traversing to other regions in the task space. However, because of the varying inertia effects, this difference can hardly be quantified systematically.

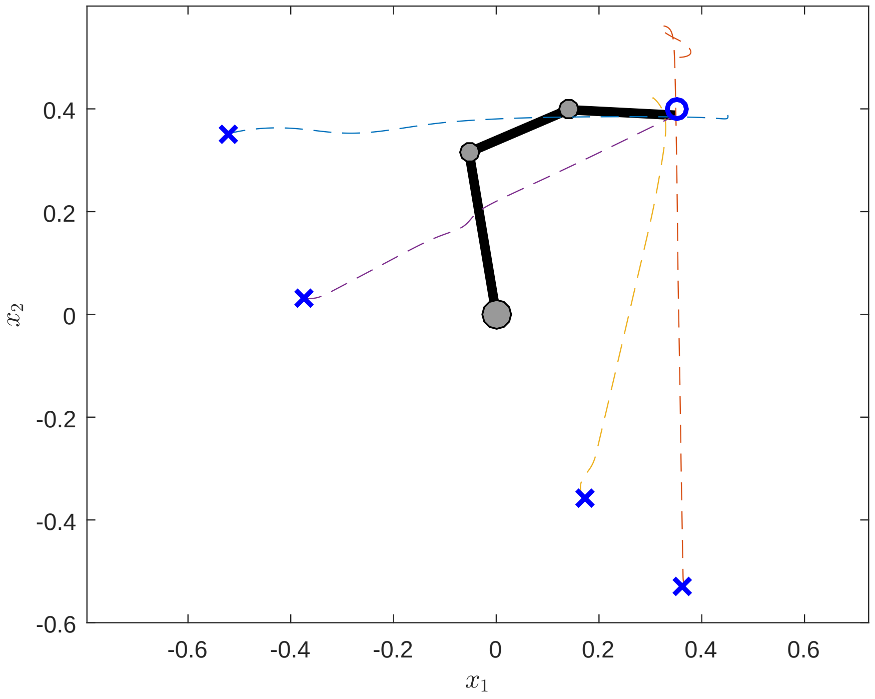
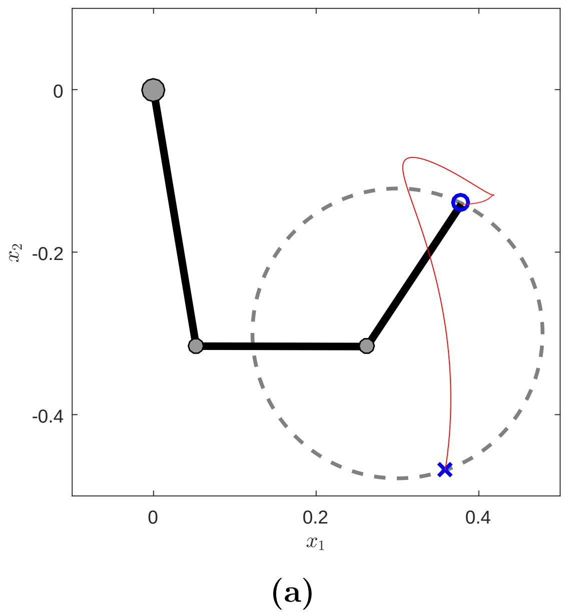


Figure 3.8: Simulation of the proposed control law compared to the transpose Jacobian control in Figure 3.2



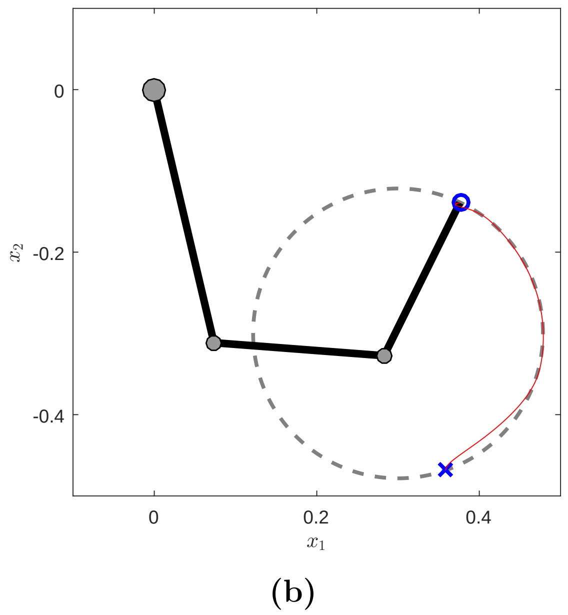


Figure 3.9: Simulation results for a curved path with using the 3-DoF manipulator. **(a)**: Transpose Jacobian control. **(b)**: Proposed controller (3.18).

We simulated the controller for circular path following using the same gains. The manipulator starts at a configuration of $q = \text{col}(-\frac{\pi}{2}, \frac{\pi}{4}, \frac{\pi}{4})$ and intends to reach $\bar{x} = \text{col}(0.38, -0.14)m$ by tracing a curve with radius of $0.18m$. We employed the velocity field in (3.17) with $\alpha = 100$ and $\beta = 5$. Th result is plotted in Figure 3.9(b) compared to the transpose Jacobian control in Figure 3.9(a). Despite the good following performance overall, noticeable deviation is present at the beginning of the motion. The maximum contour error is $1.2cm$, or 6.7% of the curve radius. This is because in order to trace the curve, the gyroscopic force will need to "fight against" the force due to APE, which points directly from $x(t_0)$ to $\bar{x}$, but because the EEF velocity is low at the initial stage of the movement, the effect of the gyroscopic force is not significant enough.

3.5 Experimental Results using a 4-DoF WAM Robot

Experiments on the proposed control algorithm were carried out using a 4-DoF WAM robot from *Barrett Technologies, Inc*, shown in Figure 3.10. This anthropomorphic arm mimics the human shoulder (3-DoF) and elbow (1-DoF) and is redundant for reaching tasks in 3D. The joints are tendon-driven, powered by brushless motors that allow low friction and high backdrivability. The parameters and CAD models of the robot is available on [81]. In torque mode, the control loop frequency can reach up to $1kHz$.

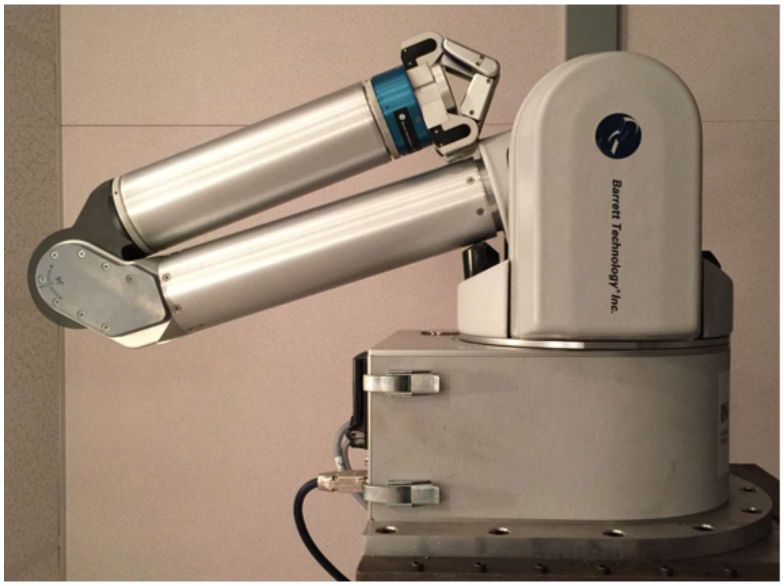


Figure 3.10: Experimental setup of the 4-DoF WAM robot

In the experiments, the WAM starts with an initial configuration of $q(t_0) = \text{col}(0, 0.52\pi, 0, 0.72\pi)$ which corresponds to initial EEF position at $x(t_0) = \text{col}(-0.358, 0.019, 0.288)m$. The target position is first set to $\bar{x} = \text{col}(0.2, -0.5, 0.5)m$ that is $63cm$ away from $x(t_0)$. Similar to the experiments with the 2-DoF manipulator, transpose Jacobian control with quadratic APE (3.2) resulted in significant steady state errors. Despite the low friction in the WAM compared to other robots, the EEF stopped up to $10cm$ from the target, unless if an aggressive gain is used. Again, we used the alternative form of APE in (3.6) instead to improve the reaching accuracy. Additionally, we noticed that the steady state error is largely contributed by the Coulomb friction at the first shoulder joint that carries most of the arm mass, so first damping term is set to be very small. The steady state error is lowered to within $3cm$ under the following gains:

$$K_p = \begin{bmatrix} 15 & 0 & 0 \\ 0 & 15 & 0 \\ 0 & 0 & 15 \end{bmatrix}, C_d = \begin{bmatrix} 0.3 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}, \lambda = 40$$

After having satisfactory result with the transpose Jacobian control, we the injected the artificial gyroscopic force (3.13) with the velocity field for straight line path in 3D (3.16). The gyroscopic gain k_p^g is varied while keeping the velocity field shape constant with $\rho = 50$. Figure 3.11 plots the path of the EEF.

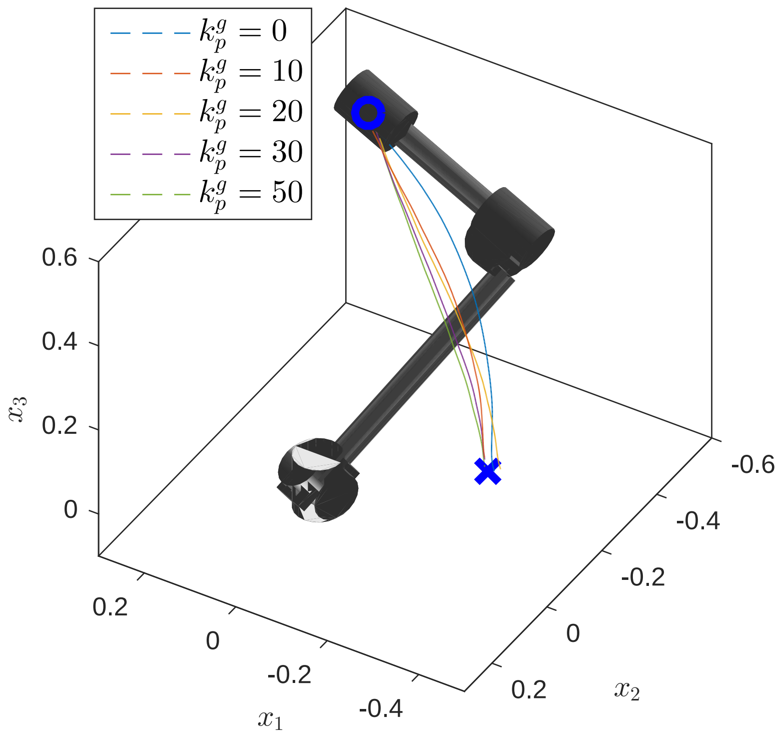


Figure 3.11: Experimental results for a rectilinear path using the 4-DoF WAM robot for $\bar{x} = \text{col}(0.2, -0.5, 0.5)m$

Just like for the 2-DoF manipulator, the results proved the effectiveness of the gyroscopic forces for path following. Larger gyroscopic gains lead to straighter path profiles. We then experimented the controller for a different target at $\bar{x} = \text{col}(0, 0.5, 0.5)m$, and the results are illustrated in Figure 3.12. Gyroscopic force with gain $k_p^g = 50$ works well for both cases. In a more general setting, we suggest making k_p^g greater for shorter paths to account for slower speeds due to the smaller APE, because the gyroscopic force (3.13) is quadratic in velocity.

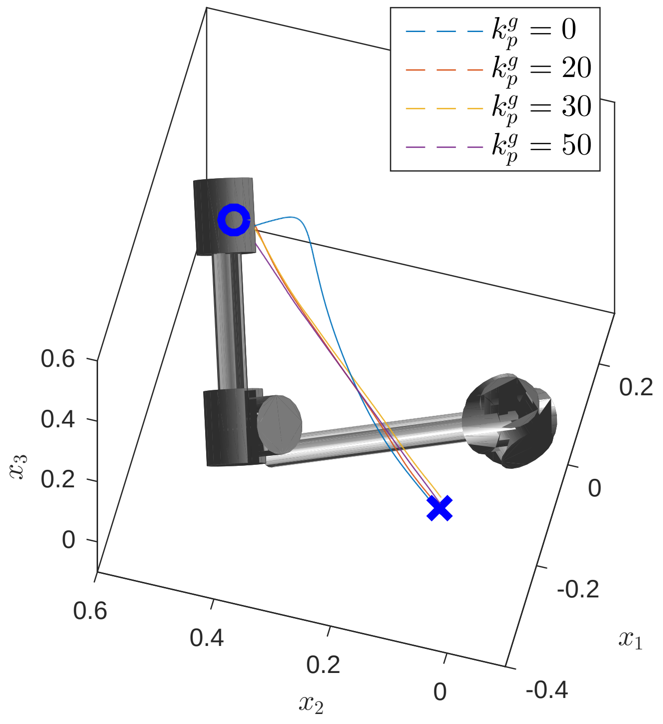


Figure 3.12: Experimental results for a rectilinear path using the 4-DoF WAM robot for $\bar{x} = \text{col}(0, 0.5, 0.5)m$

Remark 5. As opposed to gain tuning in PID controls where large gains may cause unsafe fast motions or even instability, we observed that using large k_p^g does not lead to these problems when experimenting with the WAM robot. This is because of the energy-conserving nature of the gyroscopic force. Theoretically, gyroscopic forces will not alter the passivity of the system despite the magnitude of k_p^g. When experimenting with the 2-DoF manipulator, however, we observed an unstable chattering behavior of the robot with $k_p^g = 200$. Compared with the WAM experiment, we believe this was because the control frequency was lower ($400Hz$ compared to $1kHz$) and the joint velocity was not filtered. Indeed, issues such as time delay, noise in velocity feedback, and input disturbances will break the energy-conserving property of the gyroscopic force. Therefore, we still advise exercising caution before setting k_p^g to very large, especially when these imperfect conditions are a concern.

Chapter 4

Combined Position and Orientation Control of the End Effector

In this chapter, the objective is to control both the position and orientation of the EEF along the path. We will begin with the simpler case of planar motions, where the task space can be represented as a subspace of $\mathbb{R}^3$, and the same control strategies proposed in Chapter 3 can be used. Starting from Section 4.2, we focus on motions in 3D where the EEF pose is described by $SE(3)$ matrices.

4.1 Position and Orientation Control for Planar Motion

If the task space of the manipulator is constrained on a plane, as in the case of the 2-DoF and 3-DoF manipulators presented in Chapter 3, its pose can be described as elements of the special Euclidean group $SE(2)$:

$$g(q) = \begin{bmatrix} R(q) & x(q) \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} cos\big(\theta(q)\big) & sin\big(\theta(q)\big) & p_1(q) \\ -sin\big(\theta(q)\big) & cos\big(\theta(q)\big) & p_2(q) \\ 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

which is a simpler case of the 3D rigid body kinematics described in (2.1). Then we can formulate the task space without loss of generality in $\mathbb{R}^3$ as

$$h(q) = \begin{bmatrix} x_1(q) \\ x_2(q) \\ \theta(q) \end{bmatrix} \tag{4.2}$$

As an example, the forward kinematics of the planar 3-DoF manipulator, illustrated in Figure 4.1, can be represented by:

$$h(q) = \begin{bmatrix} x_1(q) \\ x_2(q) \\ \theta(q) \end{bmatrix} = \begin{bmatrix} l_1 \cos(q_1) + l_2 \cos(q_1 + q_2) + l_3 \cos(q_1 + q_2 + q_3) \\ l_1 \sin(q_1) + l_2 \sin(q_1 + q_2) + l_3 \sin(q_1 + q_2 + q_3) \\ q_1 + q_2 + q_3 \end{bmatrix} \tag{4.3}$$

where θ is the angle between the inertia frame and the EEF body frame. One can then

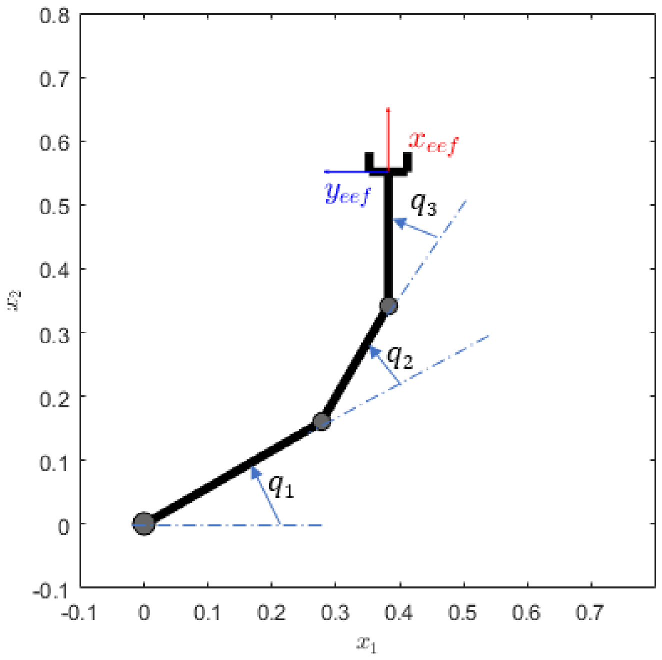


Figure 4.1: Schematic of the 3-DoF manipulator kinematics

derive the differential kinematics as:

$$\dot{h}(q) = \frac{\partial h}{\partial q}\dot{q} = \begin{bmatrix} \dot{x}(q) \\ \dot{\theta}(q) \end{bmatrix} = \begin{bmatrix} & J_p(q) & \\ 1 & 1 & 1 \end{bmatrix} \dot{q} = J(q)\dot{q} \tag{4.4}$$

where $J_p(q) = \frac{\partial x}{\partial q}$ is the Jacobian mapping for position as discussed in the previous chapters.

Having this parametrization in $\mathbb{R}^3$, we may treat θ as x_3, so the formulation of gyroscopic force for position control in 3D presented in Section 3.2.2 can be directly applied here. The control law is then very similar to (3.18):

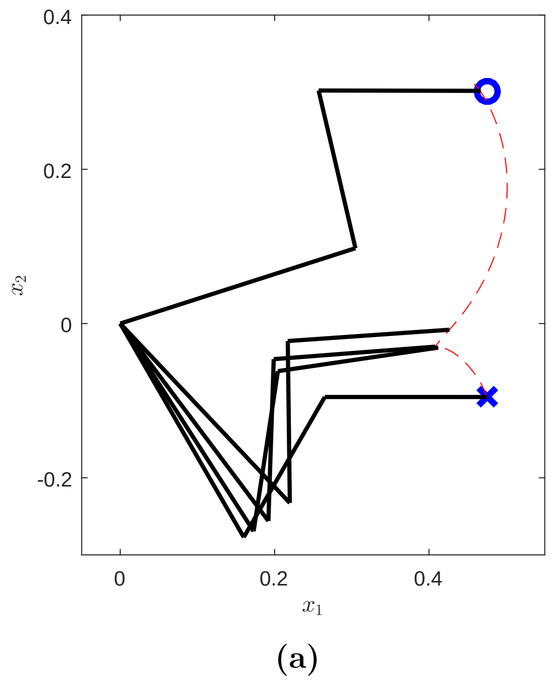
$$\tau = N + \frac{\partial}{\partial q}\hat{U} - C_d\dot{q} + k^g J^T \left(\dot{h}\bar{v}^T - \bar{v}\dot{h}^T\right)\dot{h} \tag{4.5}$$

Consider an example with the objective of having the EEF trace a straight line while keeping its orientation normal to the line; tasks such as wiping a surface can be represented this way. We first design the APE as $\hat{U} = \frac{1}{2}(\bar{h} - h)^T K(\bar{h} - h)$, where $\bar{h}$ represents the target pose in task space. The velocity field in (3.16) can be used to encode this task. In this case, this velocity field means that, only movements long the straight line is desired; any deviation from the line or change in orientation θ should be avoided.

Figure 4.2 compares the transpose Jacobian control (3.1) and the proposed control law (4.5) using a simulation with the 3-DoF manipulator. Starting from an initial configuration of $q = \text{col}(-\frac{\pi}{3}, \frac{2\pi}{3}, -\frac{\pi}{3})$ or EEF pose of $h(t_0) = \text{col}(0.475m, -0.095m, 0)$, the target pose is assigned as $\bar{h} = \text{col}(0.0475m, 0.3m, 0)$. The control gains are:

$$K = \begin{bmatrix} 8 & 0 & 0 \\ 0 & 8 & 0 \\ 0 & 0 & 8 \end{bmatrix}, C_d = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.6 & 0 \\ 0 & 0 & 0.6 \end{bmatrix}, \rho = 100, k^g = 200$$

Evidently, the addition of the gyroscopic force enables the EEF to follow the desired poses during motion. The EEF stayed within $1.2cm$ from the desired path, and the largest orientation error was 0.2 degree. Despite the large gain k^g used, the maximum joint torque needed by the controller only increased to $2.84Nm$ from $2.25Nm$ in the transpose Jacobian control.



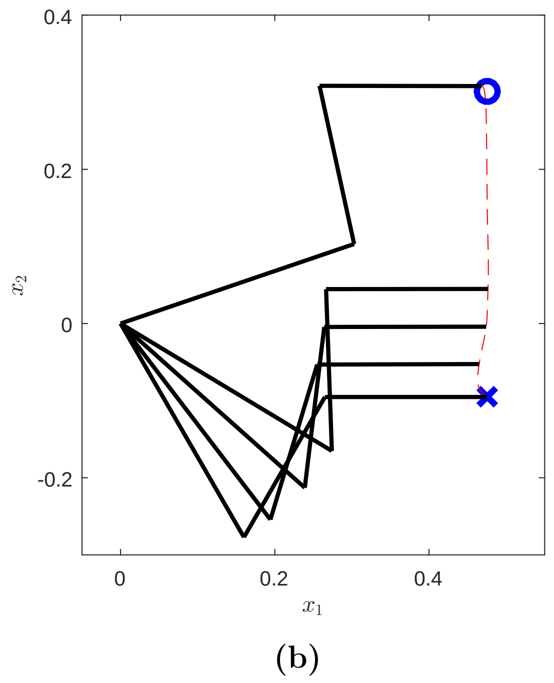


Figure 4.2: Simulation results for position and orientation control on $SE(2)$ using the 3-DoF manipulator. **(a)**: Transpose Jacobian control. **(b)**: Proposed controller.

4.2 Transpose Jacobian Control for Orientation in 3D

Now we consider the more general case with motions in 3D, where the EEF orientation can be described by elements of the special orthogonal group $SO(3)$ (i.e. rotation matrices) (2.1). Similar to position control, one can design an APE for the orientation and implement transpose Jacobian control. Unlike position which is almost ubiquitously expressed as variables in the Cartesian space, orientation has different parameterizations. One common approach is to use Euler angles in $\mathbb{R}^3$ as an minimal representation of $SO(3)$. However, it is well known that this representation suffers from singularity problems [82]. To be more specific, when the orientation travels past a singular condition, the Euler angle representation loses the ability to describe change in orientation in certain directions. The time derivative of one or more of the Euler angles will approach infinity, leading to a discontinuous orientation parametrization which causes an undesirable sudden change in the input signal that may affect the stability of the system.

Alternatively, people have successfully implemented the transpose Jacobian control for the orientation of a spherical wrist using unit quaternions [83]. However, this redundant representation brings the issue of non-unique mapping from $SO(3)$. In addition, the formulation of gyroscopic force proposed in this thesis is velocity dependent, but the time derivative of unit quaternions does not have an intuitive meaning. Therefore, we propose the direct use of the rotation matrices in $SO(3)$ similar to the works in [84] and [52]. First, we express the orientation $R(q)$ and the desired orientation $\bar{R}$ each in terms of the 3 orthogonal unit vectors. Namely,

$$R(q) = [n(q), s(q), a(q)]; \qquad \bar{R} = [\bar{n}, \bar{s}, \bar{a}] \tag{4.6}$$

Define an orientation error from R to $\bar{R}$ as

$$e_o = \frac{1}{2}(n \times \bar{n} + s \times \bar{s} + a \times \bar{a}). \tag{4.7}$$

As a matter of fact, if we define a relative rotation $R_{err} = \bar{R}R^{-1} = \bar{R}R^T$ and parametrize it as a single rotation along the axis r by a magnitude of ϕ, one can show that $e_o = r\sin(\phi)$. The authors of [52] implemented the transpose Jacobian control for the EEF orientation by the following control law:

$$\tau_o = N(q) + \gamma J_o^T(q)e_o - C_d\dot{q}, \qquad \gamma > 0 \tag{4.8}$$

where $J_o(q)$ is the Jacobian that maps the joint velocity to the angular velocity of the EEF with respect to the inertial frame: $\omega = J_o\dot{q}$. Though it was shown in simulation that $R(q)$ converges asymptotically to $\bar{R}$, the authors of [84] and [52] did not provide much discussion on the APE behind this transpose Jacobian control law (4.8). We formulated the orientation APE in the theorem below.

Theorem 4.1. *The control law in* (4.8) *is the transpose Jacobian control with the orientation APE of*

$$\hat{U}_o(R(q)) = \frac{1}{2}Tr(K_o(I_3 - \bar{R}^T R(q))), \qquad K_o = \gamma I_3 \tag{4.9}$$

Proof. To prove the equivalence of the control law (4.8) and the APE (4.9), we need to show that

$$\frac{\partial \hat{U}_o}{\partial q} = \gamma J_o^T(q)e_o$$

First, rewrite (4.9) using the 3-vector notation in (4.6)

$$\begin{aligned}\hat{U}_o &= \frac{\gamma}{2}\mathrm{Tr}(I_3 - \bar{R}^T R) \\ &= \frac{\gamma}{2}(3 - \bar{n}^T n - \bar{s}^T s - \bar{a}^T a)\end{aligned}$$

Then the gradient of $\bar{U}_o$ is

$$\frac{\partial \hat{U}_o}{\partial q} = -\frac{\gamma}{2}(\frac{\partial n}{\partial q}^T \bar{n} + \frac{\partial s}{\partial q}^T \bar{s} + \frac{\partial a}{\partial q}^T \bar{a}) \tag{4.10}$$

We know that [14]

$$[\dot{n}, \dot{s}, \dot{a}] = \dot{R} = \widehat{\omega} R$$

where the wedge operator $\widehat{\bullet}$ is defined in (2.3), and $\omega \in \mathbb{R}^3$ represents the instantaneous angular velocity with respect to each axis of the inertia frame. For a moment, let us just look at the first axis n:

$$\dot{n} = \widehat{\omega} n = \omega \times n = -n \times \omega = -\widehat{n}\omega = -\widehat{n} J_o \dot{q}$$

Since $\dot{n} = \frac{\partial n}{\partial q}\dot{q}$, we conclude that $\frac{\partial n}{\partial q} = -\widehat{n} J_o$. Following the same steps, one can conclude that $\frac{\partial s}{\partial q} = -\widehat{s} J_o$ and $\frac{\partial a}{\partial q} = -\widehat{a} J_o$. Plug this information into (4.10):

$$\begin{aligned}\frac{\partial \hat{U}_o}{\partial q} &= -\frac{\gamma}{2}(\frac{\partial n}{\partial q}^T \bar{n} + \frac{\partial s}{\partial q}^T \bar{s} + \frac{\partial a}{\partial q}^T \bar{a}) \\ &= \frac{\gamma}{2}(J_o^T \widehat{n}^T \bar{n} + J_o^T \widehat{s}^T \bar{s} + J_o^T \widehat{a}^T \bar{a}) \\ &= \frac{\gamma}{2} J_o^T (\widehat{n}^T \bar{n} + \widehat{s}^T \bar{s} + \widehat{a}^T \bar{a}) \\ &= \frac{\gamma}{2} J_o^T (-n \times \bar{n} - s \times \bar{s} - a \times \bar{a}) \\ &= \frac{\gamma}{2} J_o^T (\bar{n} \times n + \bar{s} \times s + \bar{a} \times a) \\ &= \gamma J_o^T e_o\end{aligned}$$

□

The interested readers are referred to [85] for a detailed discussion on the properties of this APE. We assume that this gives $\lim_{t\to\infty} R(t) = \bar{R}$ without loss of generality. Although we have just proved that the effect of the control input (4.8) is to create an APE in the form of (4.9), (4.8) is advantageous in the sense that it can be applied to tasks where the orientation is not fully defined. For instance, in applications such as needle insertion, the objective is to align the orientation of the needle with that of the hole, and self-rotation about that axis is not important. One can then apply control (4.8) by using a and $\bar{a}$ to represent the axis of the needle and the hole, and disregard the terms involving the other axes. By doing so, the control will guarantee that $\lim_{t\to\infty} a(t) = \bar{a}$ while allowing the needle to self-rotate freely.

4.3 Transpose Jacobian Control for Position and Orientation in 3D

Denote the desired EEF pose by $\bar{g} \in SE(3)$. For a manipulator with DoF> 6, one can combine the APE for position $\hat{U}_p$ and the APE for orientation $\hat{U}_o$ to generate the transpose Jacobian control law:

$$\tau = N(q) + \frac{\partial \hat{U}_p}{\partial q} + \frac{\partial \hat{U}_o}{\partial q} - C_d \dot{q} \tag{4.11}$$

Theorem 4.2. *For the closed-loop system with a robot manipulator (2.5) and the transpose Jacobian control law (4.11) with properly designed APEs $\hat{U}_p$ and $\hat{U}_o$, the equilibrium points characterized by $x(q) = \bar{x}$, $R(q) = \bar{R}$ and $\dot{q} = 0$ are asymptotically stable.*

Proof. Consider the Lyapunov function candidate:

$$V = T(q, \dot{q}) + \hat{U}_p(x(q)) + \hat{U}_o(R(q))$$

The rest of the proof is the same as Theorem 3.1 and is not repeated here. □

Remark 6. Using the same arguments as in Remark 2, Theorem 4.2 means that in general the control (4.11) renders $\lim_{t\to\infty} g(t) = \bar{g}$.

We verified the controller via simulation on a 7-DoF WAM robot, whose parameters are available online at [81]. Due to the complexity of the robot, rather than deriving the closed-form equations of motion, we conducted the simulation numerically using the *Matlab Robotic Toolbox* by Corke [86]. The robot starts with an initial configuration of

$q(t_0) = \mathrm{col}(-3.11, 2.28, -1.44, 2.18, -0.91, 1.27, 2.10)$, which corresponds to the EEF pose of

$$g(t_0) = \begin{bmatrix} R(t_0) & x(t_0) \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -0.30m \\ 0 & 1 & 0 & 0.30m \\ 0 & 0 & 1 & -0.30m \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

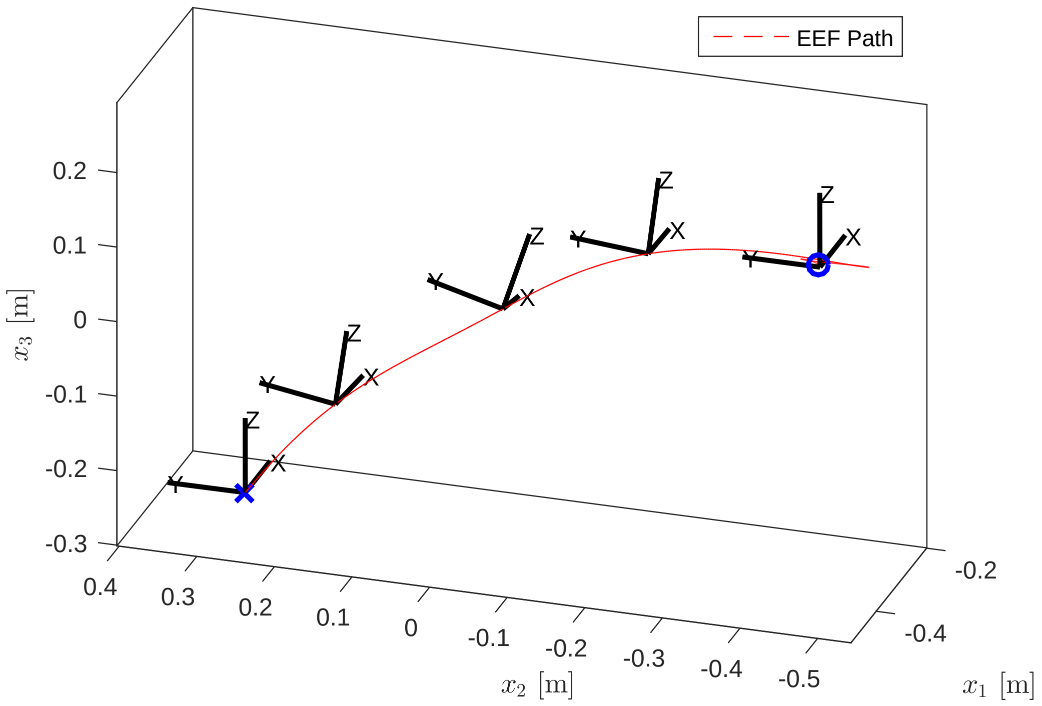


Figure 4.3: Simulation result of transpose Jacobian control (4.11) on $SE(3)$ using a 7-DoF WAM robot. Both the position and orientation of the EEF converges to the desired configuration.

The target EEF pose is

$$\bar{g} = \begin{bmatrix} \bar{R} & \bar{x} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -0.50m \\ 0 & 1 & 0 & -0.50m \\ 0 & 0 & 1 & 0.20m \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

We employed the position APE quadratic in position and the orientation APE in (4.9). The controller gains are tuned to

$$K_p = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}, \quad K_o = I_3, \quad C_d = \mathrm{diag}(2.5, 2.5, 0.5, 0.5, 0.1, 0.1, 0.1)$$

Figure 4.3 plots the result of the EEF pose throughout the trajectory. Obviously $g(q)$ converges to $\bar{g}$. However, note that although in this case $R(t_0) = \bar{R}$, the EEF orientation is not well-maintained during the movement. The maximum orientation APE (which represents the magnitude of orientation error) is 0.073. This corresponds to a relative rotation of roll-pitch-yaw of 15.8^o, 15.0^o, and 1.7^o from $\bar{R}$.

In many applications, it is not necessary to control the full orientation of the EEF: in needle insertion, the goal is to guarantee that the axis of the needle aligns with the hole; when carrying a cup of water, the goal is to ensure the cup does not tilt to prevent spilling. Rotations about that axis, however, is for the most part irrelevant in these problems. Without loss of generality, we assume that the axis of interest is $a(q)$, or the body z-axis at the EEF. One can simply apply a constant rotation if the axis of interest is different. Compared to (4.11) which used the gradient of the orientation APE, we can relax the conditions on the other 2 axes $n(q)$ and $s(q)$, and the control law comes down to:

$$\tau = N(q) + \frac{\partial \hat{U}_p}{\partial q} + \gamma J_o^T (\bar{a} \times a) - C_d \dot{q} \tag{4.12}$$

Simulation study was performed on control (4.12) with the same conditions in Figure 4.3. The difference here is that, instead of having a desired orientation matrix $\bar{R}$, we only need $\bar{a} = \mathrm{col}(0, 0, 1)$, and the gain becomes $\gamma = 1$ instead of $K_o = I_3$. The results are depicted in Figure 4.4.

Compared with Figure 4.3, only the body z-axis of the EEF converged to the desired vertically upward orientation. Again, the initial condition is one with $a(t_0) = \bar{a}$, but the body z-axis does not remain close to this configuration throughout the movement. The maximum tilt in this simulation was 20.9^o, and if the EEF were to carry a nearly full cup of water, it is almost guaranteed to spill. One can limit this orientation error by increasing K_o, but because both the position and orientation APEs are in fact functions of the joint angles, this gain change will affect the position path profile. This coupling between the APEs makes the transpose Jacobian control more difficult to tune compared to position-only problems. As a result, we seek for help from gyroscopic forces to maintain the orientation rather than tuning the transpose Jacobian control.

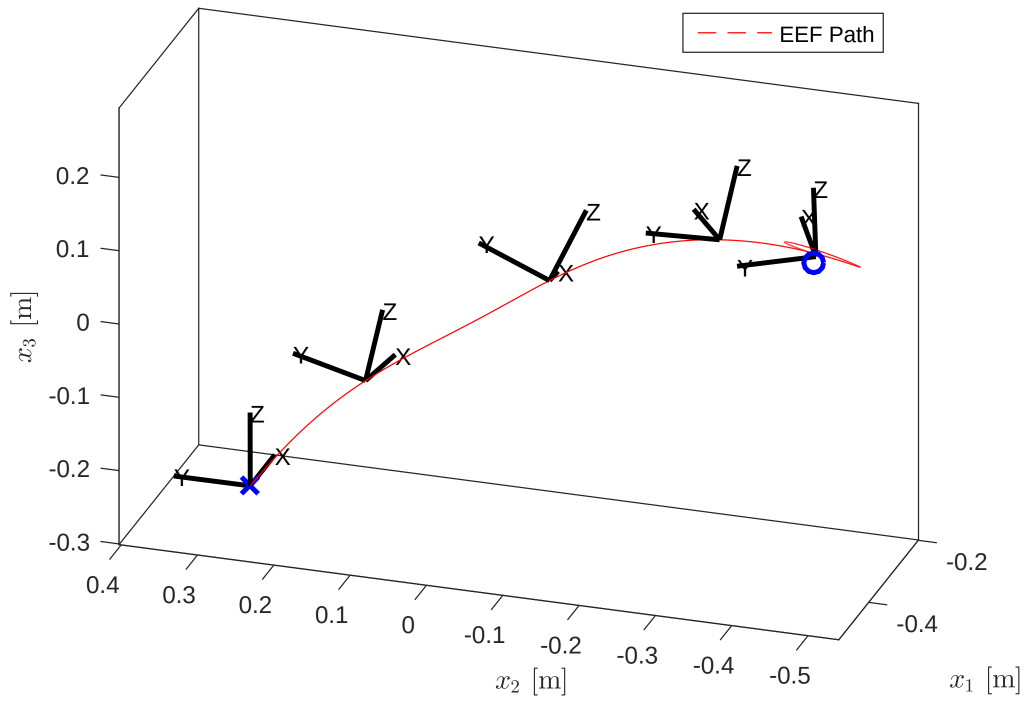


Figure 4.4: Simulation result of transpose Jacobian control (4.12) using a 7-DoF WAM robot. In addition to the EEF position, only the body z-axis is controlled.

4.4 Gyroscopic Force Shaping for Orientation Control

Similar to the previous chapter, we propose the direct formulation of gyroscopic forces in the task space to control the EEF pose during the motion. The final control law is:

$$\tau = N + \frac{\partial}{\partial q}\hat{U}_p + \frac{\partial}{\partial q}\hat{U}_o - C_d\dot{q} + J_p^T F_p^g + J_o^T F_o^g \tag{4.13}$$

We state the convergence of g to the target $\bar{g}$ in a theorem:

Theorem 4.3. *For the closed-loop system with a robot manipulator (2.5) and the control law (4.13) with F_p^g and F_o^g being gyroscopic in the task space(i.e. $\dot{x}^T F_p^g = \omega^T F_o^g = 0$), the equilibrium points characterized by $g(q) = \bar{g}$ and $\dot{q} = 0$ are asymptotically stable.*

Proof. The proof is the same as for Theorem 3.2 □

The triple product formulation of gyroscopic force in (3.13) is also used here for orientation control:

$$\begin{aligned} F_o^g(\bar{\omega}, \omega) &= k_o^g \omega \times (\bar{\omega} \times \omega) \\ &= k_o^g \underbrace{\left(\omega\bar{\omega}^T - \bar{\omega}\omega^T\right)}_{\text{skew-symmetric}} \omega \end{aligned} \tag{4.14}$$

where $\bar{\omega}$ refers to the desired angular velocity of the EEF. The remaining question then is how to design $\bar{\omega}$. Let us consider the case where the goal is to maintain an orientation $\bar{R}$. We define

$$dR := \bar{R}R^T(q)$$

which is the relative rotation between the $R(q)$ and $\bar{R}$ expressed in the inertial frame because $(dR)R = \bar{R}R^T(q)R(q) = \bar{R}$. Assuming that during motion, $R(q)$ is sufficiently close to $\bar{R}$ (this is a reasonable assumption especially in the case where we try to maintain an initial orientation), then dR can be understood as a small orientation change, and hence $dR - I_3$ will be a skew-symmetric matrix similar to $\dot{R}$. We define the desired angular velocity as

$$\bar{\omega} = (dR - I_3)^\vee = \left(\bar{R}R^T - I_3\right)^\vee \tag{4.15}$$

because $\bar{\omega}$ resides in the same space as ω and indicates a desired incremental motion. In reality, $dR - I_3$ is not a perfectly skew-symmetric matrix as

$$dR = \begin{bmatrix} dR_{1,1} & dR_{1,2} & dR_{1,3} \\ dR_{2,1} & dR_{2,2} & dR_{2,3} \\ dR_{3,1} & dR_{3,2} & dR_{3,3} \end{bmatrix}$$

with $dR_{1,1} \approx dR_{2,2} \approx dR_{3,3} \approx 1$, $dR_{3,2} \approx -dR_{2,3}$, $dR_{1,3} \approx -dR_{3,1}$, and $dR_{2,1} \approx -dR_{1,2}$. In this case, we approximate the vee operator by $\bullet^\vee$

$$\bar{\omega} = (dR - I_3)^\vee = \frac{1}{2}\begin{bmatrix} dR_{3,2} - dR_{2,3} \\ dR_{1,3} - dR_{3,1} \\ dR_{2,1} - dR_{1,2} \end{bmatrix}$$

Table 4.1: Summary of the proposed control scheme

Overall Control Law	$\tau = N + \frac{\partial}{\partial q}\hat{U}_p + \frac{\partial}{\partial q}\hat{U}_o - C_d\dot{q} + J_p^T F_p^g + J_o^T F_o^g$ (4.16)
Position APE - Quadratic	$\hat{U}_p(x) = \frac{1}{2}(\bar{x} - x)^T K_p (\bar{x} - x)$ (4.17)
Position APE - Alternative	$\hat{U}_p(x) = \sum_{i=1}^{3} \frac{k_i}{\lambda_i} \ln[\cosh(\lambda(\bar{x}_i - x_i))]$ (4.18)
Gyroscopic Force - Position	$F_p^g = k_p^g \dot{x} \times (\bar{v} \times \dot{x})$ (4.19)
Desired Translational Velocity	Equations (3.14), (3.15), (3.16), (3.17)
Orientation APE - Full	$\hat{U}_o(R) = \frac{1}{2}\mathrm{Tr}(K_o(I_3 - \bar{R}^T R))$ (4.20)
Gyroscopic Force - Orientation	$F_o^g = k_o^g \omega \times (\bar{\omega} \times \omega)$ (4.21)
Desired Angular Velocity - Full	$\bar{\omega} = (\bar{R}R^T - I_3)^\vee$ (4.22)
Orientation APE - Single Axis	$\hat{U}_o(a) = \frac{\gamma}{2}(1 - \bar{a}^T a)$ (4.23)
Desired Angular Velocity - Single Axis	$\bar{\omega} = a \times \bar{a}$ (4.24)

If only the orientation of one axis is of interest, the desired angular velocity in (4.15) cannot be defined because $\bar{R}$ is not given. We can borrow the idea of orientation error from (4.7) to define the desired angular velocity:

$$\bar{\omega} = a \times \bar{a} \tag{4.25}$$

In fact, this formulation of $\bar{\omega}$ equals to $-2e_o$ of the error term in (4.7) without the other 2 axes. Physically, this indicates that the desired angular velocity is one that points opposite to the error and is a suitable choice in order to maintain the orientation of the body z-axis.

Table 4.1 summarizes the proposed control law, including formulations APE, task space gyroscopic forces, and desired velocity field for both position and orientation control.

4.5 Simulation Results of the Proposed Controller

The proposed controller (4.16) is tested via simulation using the 7-DoF WAM robot. First, we consider controlling the full pose by using the same APE as in Figure 4.3. The desired translational velocity for rectilinear path (3.16) was employed along with the desired angular velocity (4.22) to generate the gyroscopic forces (4.19) (4.21). The gyroscopic force gains are tuned as

$$k_p^g = 30, \quad k_o^g = 10$$

Figure 4.5 illustrates the resulting motion. The maximum orientation APE occurred was 0.0072, roughly 10 times smaller than Figure 4.3 without gyroscopic force. This corresponds to a relative rotation of roll-pitch-yaw of only 3.34^o, -0.80^o, and 5.96^o from $\bar{R}$. In addition, the maximum deviation from the straight-line was $2.63cm$ which is very small considering that the total path length was $96.7cm$.

Using the same gains, we verified the proposed strategy (4.16) for single-axis orientation control using orientation APE (4.23) and desired angular velocity in (4.24). The motion profile is plotted in Figure 4.6.

We can draw the same conclusion as the previous example. In contrast with Figure 4.4 which used only the transpose Jacobian control, the addition of gyroscopic force not only straightens the path but also ensures that the body z-axis is kept upright. The maximum tilt of the body z-axis was 1.9^o, compare to 20.9^o when no gyroscopic force was used. The largest deviation from the straight-line path was approximately $2.04cm$.

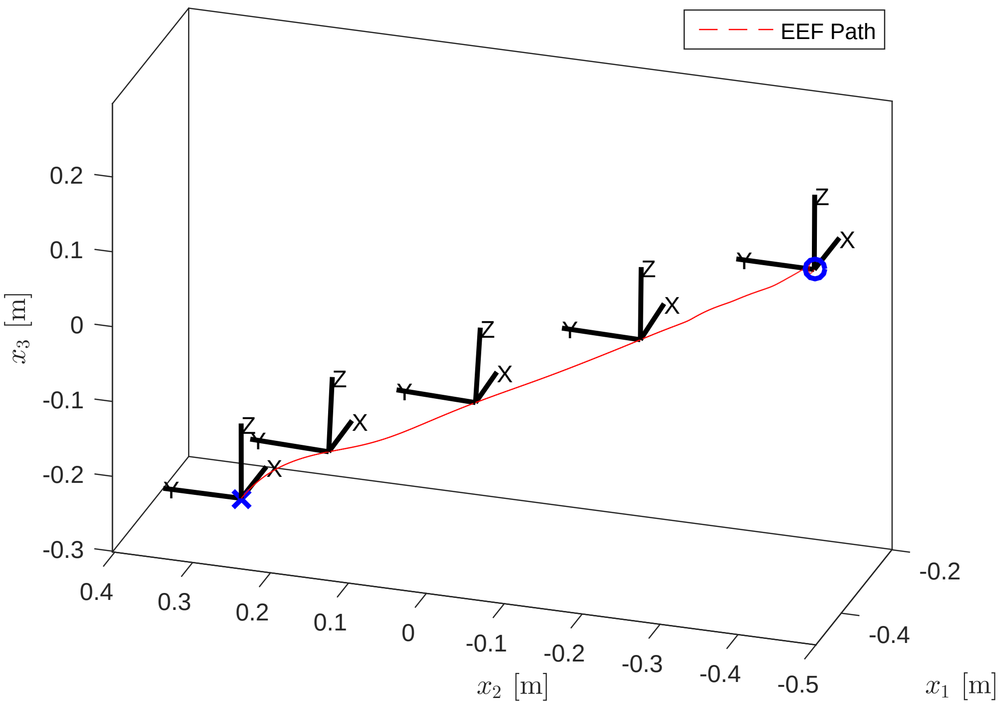


Figure 4.5: Simulation result of proposed controller (4.16) on $SE(3)$ using a 7-DoF WAM robot. Compared to Figure 4.3, the resulting path closely follows a straight line, and the desired orientation is well-maintained during the movement.

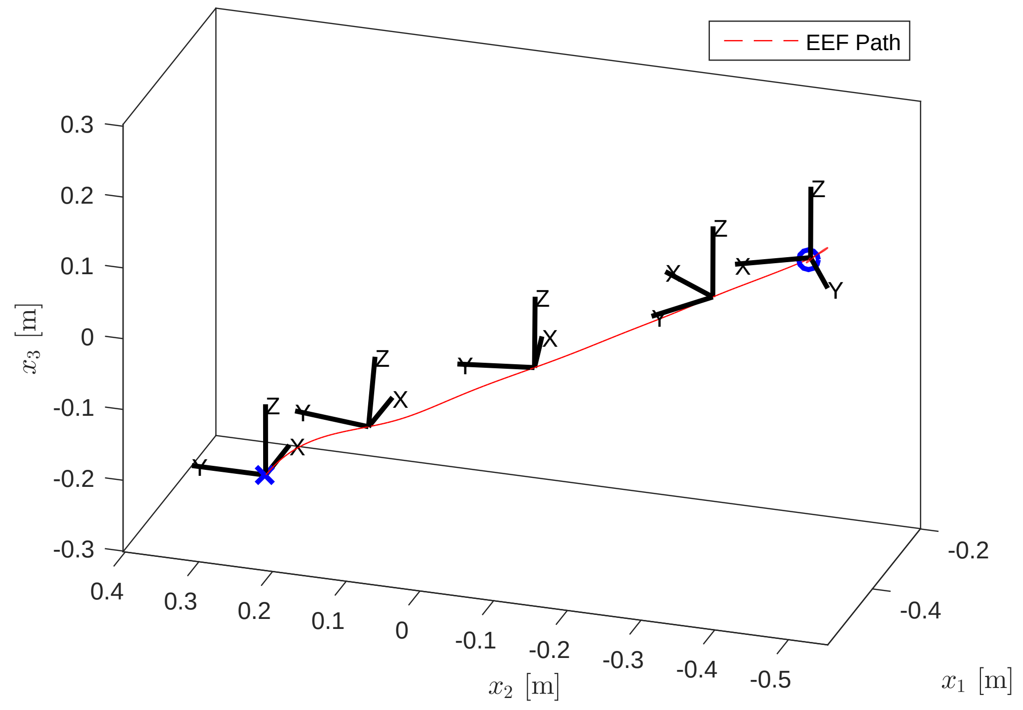


Figure 4.6: Simulation result of proposed controller (4.16) using a 7-DoF WAM robot. Compared to Figure 4.4 the resulting path closely follows a straight line, and the orientation of the body z-axis is well-maintained.

Chapter 5

Conclusions and Future Work

In this thesis, we consider the path following problem for robot manipulators and propose a control scheme that combines potential energy shaping and injection of artificial gyroscopic forces. The proposed strategy uses the transpose Jacobian control as the base controller to guarantee the convergence of the EEF pose, and adds gyroscopic forces as an auxiliary term to steer the velocity during motion. A novel approach using desired velocities to formulate gyroscopic forces directly in the task space is presented. The final control law (4.16) is free of inverse kinematics and inverse dynamics. Online adjustments of the target pose are simple to perform if the manipulator is to operate in a changing environment.

We start with the problem of controlling only the EEF position during movement in Chapter 3, incorporating examples of rectilinear and circular paths. Chapter 4 extends the work to the control of both position and orientation, thereby expanding the potential applications of this work. Although the proposed controller cannot ensure the invariance of the desired path like other existing methods do, we verify its effectiveness in meeting the path following objective by simulation and experiments. A convergence analysis is also provided in Appendix A, suggesting the robustness of the gyroscopic force for steering the direction of motion. Indeed, the gyroscopic force works by not doing any work.

Due to lack of availability of suitable hardware, experiments on the combined position and orientation control were not conducted. It will be interesting to see the performance of the orientation APE (4.9) in the presence of joint friction. If significant steady state error is present, an alternative APE design similar to (3.6) should be explored.

Arguably, one would usually expect better path following performance at the beginning and end of the movement when performing tasks. The proposed control strategy is limited in the sense that because the gyroscopic forces are formulated as quadratic in velocity, the path following accuracy is often poorer near the initial and target pose. Modifications to the proposed strategy such as using a variable gyroscopic gain need to be considered in order to account for this problem.

The simulations and experiments presented in this thesis are simple examples such as curved paths or straight-line paths while maintaining the orientation. For complicated paths with self-intersections, one can develop a high-level path planning algorithm that breaks down the task into several primal shapes and then apply the proposed controller with suitable desired velocity field for each segment. By doing so, this work can be extended to more sophisticated examples such as drawing and writing with robot manipulators.

References

[1] V. Cichella and E. Xargay, "Geometric 3d path-following control for a fixed-wing uav on so(3)," in *2011 AIAA Guidance, Navigation, and Control Conference*, pp. 3578–3592, Aug 2011.

[2] R. Gill, *Robust Spline Path Followig for Redundant Mechanical Systems.* 2015.

[3] F. Campos and J. Calado, "Approaches to human arm movement controla review," *Annual Reviews in Control*, vol. 33, no. 1, pp. 69 – 77, 2009.

[4] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators.* London, UK: Springer-Verlag, 2000.

[5] Y. Koren, *Robotics for Engineers.* New York, NY: McGraw-Hill, 1985.

[6] S. H. Suh, I. K. Woo, and S. K. Noh, "Development of an automatic trajectory planning system (atps) for spray painting robots," in *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, pp. 1948–1955 vol.3, Apr 1991.

[7] J. Luh, "An anatomy of industrial robots and their controls," *IEEE Transactions on Automatic Control*, vol. 28, pp. 133–153, Feb 1983.

[8] G. S. Chirikjian and J. W. Burdick, "Optimal feedback control as a theory of motor coordination," *Nature Neuroscience*, vol. 5, no. 11, pp. 1226–1235, 2002.

[9] E. Saltzman and J. A. S. Kelso, "Skilled actions: A task-dynamic approach," *Psychological Review*, vol. 94, no. 1, pp. 84–106, 1987.

[10] D. M. Wolpert, Z. Ghahramani, and M. I. Jordan, "Are arm trajectories planned in kinematic or dynamic coordinates? an adaptation study," *Experimental Brain Research*, vol. 103, no. 3, pp. 460–470, 1995.

[11] J. Kelso, D. Southard, and D. Goodman, "On the nature of human interlimb coordination," *Science*, vol. 203, no. 4384, pp. 1029–1031, 1979.

[12] M. Takegaki and S. Arimoto, "A new feedback method for dynamic control of manipulators," *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 103, no. 2, pp. 119–125, 1981.

[13] J. Han, "From pid to active disturbance rejection control," *IEEE Transactions on Industrial Electronics*, vol. 56, pp. 900–906, March 2009.

[14] R. M. Murray and S. S. Sastry, *A mathematical introduction to robotic manipulation.* CRC press, 1994.

[15] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control.* Hoboken, NJ: John Wiley & Sons, Inc, 2006.

[16] L. Sciavicco and B. Siciliano, "Modelling and control of robot manipulators," 2000.

[17] R. Kelly, V. Santibez Davila, and J. A. Lora Perez. London: Springer-Verlag, 2005.

[18] J. S. de la Cruz, D. Kulic, and W. Owen, "Learning inverse dynamics for redundant manipulator control," in *2010 International Conference on Autonomous and Intelligent Systems, AIS 2010*, pp. 1–6, June 2010.

[19] D. Nguyen-Tuong and J. Peters, "Using model knowledge for learning inverse dynamics," in *2010 IEEE International Conference on Robotics and Automation*, pp. 2677–2682, May 2010.

[20] D. Nguyen-Tuong and J. Peters, "Local gaussian process regression for real-time model-based robot control," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 380–385, Sept 2008.

[21] Y. Ye, Z. Yue, and B. Gu, "Adrc control of a 6-dof parallel manipulator for telescope secondary mirror," *Journal of Instrumentation*, vol. 12, no. 03, p. T03006, 2017.

[22] M. Przybya, M. Kordasz, R. Madoski, P. Herman, and P. Sauer, "Active disturbance rejection control of a 2dof manipulator with significant modeling uncertainty," *Bulletin of the Polish Academy of Sciences: Technical Sciences*, vol. 60, no. 3, pp. 509–520, 2012.

[23] D. Tolani, A. Goswami, and N. I. Badler, “Real-time inverse kinematics techniques for anthropomorphic limbs,” *Graphical Models*, vol. 62, no. 5, pp. 353 – 388, 2000.

[24] J. Q. Gan, E. Oyama, E. M. Rosales, and H. Hu, “A complete analytical solution to the inverse kinematics of the pioneer 2 robotic arm,” *Robotica*, vol. 23, pp. 123–129, Jan. 2005.

[25] M. Pfurner, “Closed form inverse kinematics solution for a redundant anthropomorphic robot arm,” *Computer Aided Geometric Design*, vol. 47, pp. 163 – 171, 2016. SI: New Developments Geometry.

[26] A. D’Souza, S. Vijayakumar, and S. Schaal, “Learning inverse kinematics,” in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*, vol. 1, pp. 298–303 vol.1, 2001.

[27] S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert, *Learning Movement Primitives*, pp. 561–572. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005.

[28] S. Schaal, A. Ijspeert, and A. Billard, “Computational approaches to motor learning by imitation,” *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, vol. 358, no. 1431, pp. 537–547, 2003.

[29] S. R. Buss and J.-S. Kim, “Selectively damped least squares for inverse kinematics,” *Journal of Graphics Tools*, vol. 10, no. 3, pp. 37–49, 2005.

[30] S. R. Buss, “Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods,” *IEEE Journal of Robotics and Automation*, vol. 16, no. 17, pp. 1–19, 2004.

[31] A. Aristidou and J. Lasenby, “Fabrik: A fast, iterative solver for the inverse kinematics problem,” *Graphical Models*, vol. 73, no. 5, pp. 243 – 260, 2011.

[32] J. Rosen, J. C. Perry, N. Manning, S. Burns, and B. Hannaford, “The human arm kinematics and dynamics during daily activities - toward a 7 dof upper limb powered exoskeleton,” in *ICAR ’05. Proceedings., 12th International Conference on Advanced Robotics, 2005.*, pp. 532–539, July 2005.

[33] R. R. Ma and A. M. Dollar, “On dexterity and dexterous manipulation,” in *2011 15th International Conference on Advanced Robotics (ICAR)*, pp. 1–7, June 2011.

[34] N. A. Bernstein. Moscow: Medgiz, 1947.

[35] N. A. Bernstein. Oxford: Pergamon Press, 1967.

[36] J. Hollerbach and K. Suh, "Redundancy resolution of manipulators through torque optimization," *IEEE Journal on Robotics and Automation*, vol. 3, pp. 308–316, August 1987.

[37] S. Chiaverini, "Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators," *IEEE Transactions on Robotics and Automation*, vol. 13, pp. 398–410, June 1997.

[38] T. Flash and N. Hogan, "The coordination of arm movements: an experimentally confirmed mathematical model," *Journal of Neuroscience*, vol. 5, pp. 1688–1703, July 1985.

[39] Y. Uno, M. Kawato, and R. Suzuki, "Formation and control of optimal trajectory in human multijoint arm movement. minimum torque-change model," *Biological Cybernetics*, vol. 61, no. 2, pp. 89–101, 1989.

[40] C. M. Harris and D. M. Wolpert, "Signal-dependent noise determines motor planning," *Nature*, vol. 394, pp. 780–784, August 1998.

[41] N. A. Bernstein, "Studies on the biomechanics of hitting using optical recording," *Annals of the Central Institute of Labor*, vol. 1, pp. 19–79, 1923.

[42] M. Latash, "There is no motor redundancy in human movements. there is motor abundance," *Motor Control*, vol. 4, no. 3, pp. 259–261, 2000.

[43] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[44] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal of Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[45] O. Khatib, "Advanced robotic manipulation," 2004.

[46] D. E. Chang, "The method of controlled lagrangians: Energy plus force shaping," *SIAM Journal on Control and Optimization*, vol. 48, no. 8, pp. 4821–4845, 2010.

[47] D. E. Chang, “On the method of interconnection and damping assignment passivity-based control for the stabilization of mechanical systems,” *Regular and Chaotic Dynamics*, vol. 19, no. 5, pp. 556–575, 2014.

[48] J. K. Salisbury, “Active stiffness control of a manipulator in cartesian coordinates,” in *1980 19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, pp. 95–100, Dec 1980.

[49] J. Fiala and A. J. Wavering, “Experimental evaluation of cartesian stiffness control on a seven degree-of-freedom robot arm,” *Journal of Intelligent and Robotic Systems*, vol. 5, no. 1, pp. 5–24, 1992.

[50] R. Kelly, R. Carelli, O. Nasisi, B. Kuchen, and F. Reyes, “Stable visual servoing of camera-in-hand robotic systems,” *IEEE/ASME Transactions on Mechatronics*, vol. 5, pp. 39–48, Mar 2000.

[51] C. Liu, C. C. Cheah, and J.-J. E. Slotine, “Adaptive jacobian tracking control of rigid-link electrically driven robots based on visual task-space information,” *Automatica*, vol. 42, no. 9, pp. 1491 – 1501, 2006.

[52] Y. Masutani, F. Myazaki, and S. Arimoto, “Modeling and sensory feedback control for space manipulators,” in *NASA Conference on Space Telerobotics*, 1989.

[53] S. Arimoto, M. Sekimoto, H. Hashiguchi, and R. Ozawa, “Natural resolution of ill-posedness of inverse kinematics for redundant robots: a challenge to bernstein’s degrees-of-freedom problem,” *Advanced Robotics*, vol. 19, no. 4, pp. 401–434, 2005.

[54] S. Arimoto, “A natural resolution of bernstein’s degrees-of-freedom problem in case of multi-joint reaching,” in *2004 IEEE International Conference on Robotics and Biomimetics*, pp. 88–95, Aug 2004.

[55] S. Arimoto, H. Hashiguchi, M. Sekimoto, and R. Ozawa, “Generation of natural motions for redundant multi-joint systems: A differential-geometric approach based upon the principle of least actions,” *Journal of Robotic Systems*, vol. 22, no. 11, pp. 583–605, 2005.

[56] T. Faulwasser, T. Weber, P. Zometa, and R. Findeisen, “Implementation of nonlinear model predictive path-following control for an industrial robot,” *IEEE Transactions on Control Systems Technology*, vol. 25, pp. 1505–1511, July 2017.

[57] C. Nielsen and M. Maggiore, “On local transverse feedback linearization,” *SIAM Journal on Control and Optimization*, vol. 47, no. 5, pp. 2227–2250, 2008.

[58] C. Nielsen, C. Fulford, and M. Maggiore, “Path following using transverse feedback linearization: Application to a maglev positioning system,” *Automatica*, vol. 46, no. 3, pp. 585 – 590, 2010.

[59] A. Hladio, C. Nielsen, and D. Wang, “Path following for a class of mechanical systems,” *IEEE Transactions on Control Systems Technology*, vol. 21, pp. 2380–2390, Nov 2013.

[60] R. J. Gill, D. Kulic, and C. Nielsen, “Spline path following for redundant mechanical systems,” *CoRR*, vol. abs/1504.06917, 2015.

[61] R. Gill, D. Kulic, and C. Nielsen, “Robust path following for robot manipulators,” in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3412–3418, Nov 2013.

[62] P. Y. Li and R. Horowitz, “Passive velocity field control of mechanical manipulators,” in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 3, pp. 2764–2770 vol.3, May 1995.

[63] P. Y. Li and R. Horowitz, “Passive velocity field control of mechanical manipulators,” *IEEE Transactions on Robotics and Automation*, vol. 15, pp. 751–763, Aug 1999.

[64] N. Wei and S. Jeon, “Gyroscopic forces for mechanical manipulators,” in *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 935–940, July 2016.

[65] N. Wei and S. Jeon, “On the gyroscopic force in mechanical manipulators and its artificial shaping for taskspace movement coordination,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3030–3035, Oct 2016.

[66] W. Thomson and P. G. Tait, *Treatise on Natural Philosophy.* Cambridge University Press, 1879.

[67] D. R. Merkin, *Introduction to the Theory of Stability.* Springer-Verlag New York, Inc., 1996.

[68] D. E. Chang, S. C. Shadden, J. E. Marsden, and R. Olfati-Saber, “Collision avoidance for multiple agent systems,” in *IEEE Conference on Decision and Control*, vol. 1, (Maui HI), pp. 539–543, 2003.

[69] H. Min, F. Sun, and F. Niu, "Decentralized uav formation tracking flight control using gyroscopic force," in *2009 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, pp. 91–96, May 2009.

[70] C. Woolsey, C. K. Reddy, A. M. Bloch, D. E. Chang, N. E. Leonard, and J. E. Marsden, "Controlled lagrangian systems with gyroscopic forcing and dissipation," *European Journal of Control*, vol. 10, no. 5, pp. 478 – 496, 2004.

[71] J. S. Dai and J. R. Jones, "Interrelationship between screw systems and corresponding reciprocal systems and applications," *Mechanism and Machine Theory*, vol. 36, no. 5, pp. 633 – 651, 2001.

[72] J. Zhao, B. Li, X. Yang, and H. Yu, "Geometrical method to determine the reciprocal screws and applications to parallel manipulators," *Robotica*, vol. 27, pp. 929–940, Oct 2009.

[73] J. S. Dai, D. Li, Q. Zhang, and G. Jin, "Mobility analysis of a complex structured ball based on mechanism decomposition and equivalent screw system analysis," *Mechanism and Machine Theory*, vol. 39, no. 4, pp. 445 – 458, 2004.

[74] R. A. Scheidt, M. A. Conditt, E. L. Secco, and F. A. Mussa-Ivaldi, "Interaction of visual and proprioceptive feedback during adaptation of human reaching movements," *Journal of Neurophysiology*, vol. 93, no. 6, pp. 3200–3213, 2005.

[75] G. C. Sing, W. M. Joiner, T. Nanayakkara, J. B. Brayanov, and M. A. Smith, "Primitives for motor adaptation reflect correlated neural tuning to position and velocity," *Neuron*, vol. 64, no. 4, pp. 575 – 589, 2009.

[76] R. A. Scheidt and T. Stoeckmann, "Reach adaptation and final position control amid environmental uncertainty after stroke," *Journal of Neurophysiology*, vol. 97, no. 4, pp. 2824–2836, 2007.

[77] R. Kelly and A. Coello, "Analysis and experimentation of transpose jacobian-based cartesian regulators," *Robotica*, vol. 17, no. 3, pp. 303 – 312, 1999.

[78] J. L. Salle and S. Lefschetz, *Stability by Liapunov's Direct Method with Applications*. Academic Press, 1961.

[79] A. Sard, "The measure of the critical values of differentiable maps," *Bulletin of the American Mathematical Society*, vol. 48, no. 12, pp. 883–890, 1942.

[80] H. Asada, "Dynamic analysis and design of robot manipulators using inertia ellipsoids," in *Proceedings. 1984 IEEE International Conference on Robotics and Automation*, vol. 1, pp. 94–102, Mar 1984.

[81] Barrett, "WAM Arm CAD Models." http://www.barrett.com/products-arm-models.htm. [Online; accessed 06-Nov-2015].

[82] N. A. Chaturvedi, A. K. Sanyal, and N. H. McClamroch, "Rigid-body attitude control," *IEEE Control Systems*, vol. 31, pp. 30–51, June 2011.

[83] R. Kelly and E. Garca, "On transpose jacobian-based regulators using unit quaternions: An energy shaping approach," in *15th Triennial World Congress of the International Federation of Automatic Control*, (Barcelona, Spain), pp. 377–382, July 2002.

[84] J. Luh, M. Walker, and R. Paul, "Resolved-acceleration control of mechanical manipulators," *IEEE Transactions on Automatic Control*, vol. 25, pp. 468–474, Jun 1980.

[85] F. Bullo and A. D. Lewis, *Geometric Control of Mechanical Systems*. New York: Springer-Verlag, 2005.

[86] P. Corke, *Robotics, Vision and Control*. Springer, 2017.

APPENDICES

Appendix A

Convergence Analysis of Gyroscopic Force Control

Compared to other path following control schemes discussed in Section 1.2.4, a disadvantage of the proposed control law (3.18) or (4.16) is that it does not convert the desired path into an attractive and invariant set. However, the simulation and experimental results shown in this thesis indicate that the gyroscopic force is effective to render the EEF pose close to the desired path. The aim of this appendix is to investigate the convergence of path in gyroscopic force control. Due to the level of difficulty involved in directly analyzing the proposed controller, we present the analysis for a simpler scenario in which a constant desired joint velocity is given with a gyroscopic force control without energy shaping.

Consider an n-DoF manipulator with equations of motion (2.5) and non-zero initial velocity $\dot{q}(t_0)$ (hence non-zero kinetic energy). Given a constant desired joint velocity $\bar{\dot{q}}$, consider the following control law:

$$\tau = k_g(\bar{\dot{q}}\dot{q}^T - \dot{q}\bar{\dot{q}}^T)\dot{q} + N(q) \tag{A.1}$$

Note that the first term is gyroscopic in the same form as (3.13) and (4.14), except that it is formulated directly in joint space. Gravity compensation $N(q)$ is applied similar to the transpose Jacobian control. Consider the following Lyapunov function candidate:

$$V = \frac{1}{2}(\dot{q} - \bar{\dot{q}})^T M(\dot{q} - \bar{\dot{q}})$$

The time derivative of V of the closed-loop system is

$$\begin{aligned}
\dot{V} &= (\dot{q} - \dot{\bar{q}})^T M \ddot{q} + \frac{1}{2}(\dot{q} - \dot{\bar{q}})^T \dot{M}(\dot{q} - \dot{\bar{q}}) \\
&= (\dot{q} - \dot{\bar{q}})^T \big\{\tau - C\dot{q} - N + \frac{1}{2}\dot{M}(\dot{q} - \dot{\bar{q}})\big\} \\
&= (\dot{q} - \dot{\bar{q}})^T \big\{k_g(\dot{\bar{q}}\dot{q}^T - \dot{q}\dot{\bar{q}}^T)\dot{q} - C\dot{q} - \frac{1}{2}\dot{M}(\dot{\bar{q}} - \dot{q})\big\} \\
&= (\dot{q} - \dot{\bar{q}})^T \big\{k_g(\dot{\bar{q}}\dot{q}^T - \dot{q}\dot{\bar{q}}^T)\dot{q} + C(\dot{\bar{q}} - \dot{q}) - \frac{1}{2}\dot{M}(\dot{\bar{q}} - \dot{q}) - C\dot{\bar{q}}\big\} \qquad \text{(A.2)} \\
&= k_g(\dot{q} - \dot{\bar{q}})^T(\dot{\bar{q}}\dot{q}^T - \dot{q}\dot{\bar{q}}^T)\dot{q} - (\dot{\bar{q}} - \dot{q})^T \underbrace{(C - \frac{1}{2}\dot{M})}_{\text{skew-symmetric}} (\dot{\bar{q}} - \dot{q}) - (\dot{q} - \dot{\bar{q}})C\dot{\bar{q}} \\
&= k_g(\dot{q} - \dot{\bar{q}})^T(\dot{\bar{q}}\dot{q}^T - \dot{q}\dot{\bar{q}}^T)\dot{q} - (\dot{q} - \dot{\bar{q}})C\dot{\bar{q}} \\
&= -k_g\big(\|\dot{\bar{q}}\|^2\|\dot{q}\|^2 - (\dot{\bar{q}}^T\dot{q})^2\big) - (\dot{q} - \dot{\bar{q}})C\dot{\bar{q}}
\end{aligned}$$

Note that the first term, $-k_g\big(\|\dot{\bar{q}}\|^2\|\dot{q}\|^2 - (\dot{\bar{q}}^T\dot{q})^2\big) \le 0$ by Cauchy-Schwarz inequality. One can add $C(q, \dot{q})\dot{\bar{q}}$ to cancel out the last term in (A.2) and render $\dot{q} = \dot{\bar{q}}$ asymptotically stable.

The control law (A.1) by itself does not guarantee the convergence of joint velocity of the closed-loop system. Part of the reason is that, without gravity, the controller is just a gyroscopic force, and by the passivity property, the kinetic energy of the closed-loop system $T(q, \dot{q}) = \dot{q}^T M(q)\dot{q}$ stays constant. On the other hand, the desired velocity $\dot{\bar{q}}$ cannot satisfy $\dot{\bar{q}}^T M(q)\dot{\bar{q}} = T(q, \dot{q})$. As a result, proving $\lim_{t\to\infty} V(t) = 0$ is impossible.

Although we do not have $\lim_{t\to\infty}\dot{q}(t) = \dot{\bar{q}}$, simulation results show that the direction of $\dot{q}$ stays close to the direction of $\dot{\bar{q}}$. Because path following performance is determined by the direction of $\dot{q}$ instead of its magnitude, we define an error term as the angle between $\dot{q}$ and $\dot{\bar{q}}$:

$$\theta = cos^{-1}\left(\frac{\dot{q}^T\dot{\bar{q}}}{\|\dot{q}\|\|\dot{\bar{q}}\|}\right)$$

Substitute θ into (A.2):

$$
\begin{aligned}
\dot{V} &= -k_g\left(\|\bar{q}\|^2\|\dot{q}\|^2 - (\bar{q}^T\dot{q})^2\right) - (\dot{q}-\bar{q})C\bar{q}\\
&= -k_g\|\bar{q}\|^2\|\dot{q}\|^2(1-cos^2\theta) - (\dot{q}-\bar{q})C\bar{q}\\
&= -k_g\|\bar{q}\|^2\|\dot{q}\|^2 sin^2\theta - (\dot{q}-\bar{q})C\bar{q}\\
&= -k_g\|\bar{q}\|^2\|\dot{q}\|^2 sin^2\theta + \bar{q}C\bar{q} - \dot{q}C\bar{q}\\
&\le -k_g\|\bar{q}\|^2\|\dot{q}\|^2 sin^2\theta + \eta\|\bar{q}\|^2\|\dot{q}\| + \eta\|\dot{q}\|^2\|\bar{q}\|\\
&= \|\bar{q}\|\|\dot{q}\|\left(\eta(\|\dot{q}\|+\|\bar{q}\|) - k_g\|\|\bar{q}\|\|\dot{q}\|sin^2\theta\right)
\end{aligned}
\tag{A.3}
$$

where the second last inequality comes from the fact that $\|C\|$ is bounded by some $\eta\|\dot{q}\|$. Evidently, we can see from (A.3) that $\dot{V} \le 0$ given

$$
sin^2\theta > \frac{\eta(\|\bar{q}\|+\|\dot{q}\|)}{k_g\|\bar{q}\|\|\dot{q}\|} \tag{A.4}
$$

Of course, one can satisfy this condition by having a large k_g. However, as θ approaches 0, it requires k_g to be increased to infinity. We seek instead the conditions on θ for the Lyapunov function to be decreasing given a fixed k_g. Denote $\xi = max\{\frac{\eta(\|\bar{q}\|+\|\dot{q}\|)}{k_g\|\bar{q}\|\|\dot{q}\|}\}$. Note that because of the conservation of kinetic energy and boundedness on the mass matrix, $\|\dot{q}\|$ is bounded by 2 finite positive values, and hence a finite ξ exists.

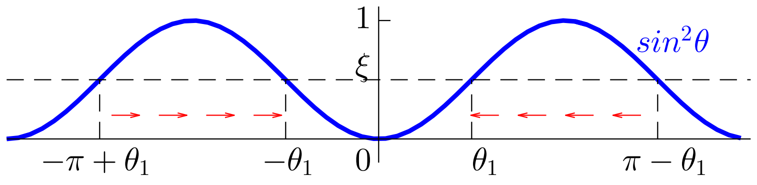


Figure A.1: Illustration of the convergence of a gyroscopic-only controlled system

Assuming k_g is large enough so that $\xi < 1$, and a solution to $sin^2\theta = \xi$ exists. Denote $\theta_1 = sin^{-1}(\sqrt{\xi})$. As illustrated in Figure A.1, for $\theta \in (\theta_1, \pi-\theta_1)$ or $\theta \in (-\pi+\theta_1, -\theta_1)$, (A.4) is satisfied and we have $\dot{V} < 0$. Loosely speaking, $\dot{V} < 0$ means that $\dot{q}$ approaches $\bar{q}$ and $|\theta|$ is decreasing. Therefore, the set $\theta \in [-\theta_1, \theta_1]$ can be interpreted as an invariant set; once the gyroscopic force steers the joint velocities into this set, the error in the direction of motion will oscillate within $\pm\theta_1$. This is illustrated as well in Figure A.1. Cases where the joint velocity is in opposite direction of the desired one (or $\theta = \pm\pi$) are unstable; though the gyroscopic force is 0 in these cases, it will act to pull the joint velocity towards the correct direction as soon as little deviation occurs.

Even with small values of k_g, simulation studies indicate that θ is rarely trapped in $(\pi - \theta_1, \pi)$ or $(-\pi, -\pi + \theta_1)$. Intuitively, with a very small k_g, the system will be dominated by the Coriolis and centrifugal forces, and the direction of the joint velocity will be varying in a periodic fashion. Then there is a high probability that θ enters $(\theta_1, \pi - \theta_1)$ or $(-\pi + \theta_1, -\theta_1)$, and then it follows that it will stay within $\pm\theta_1$ thereafter.

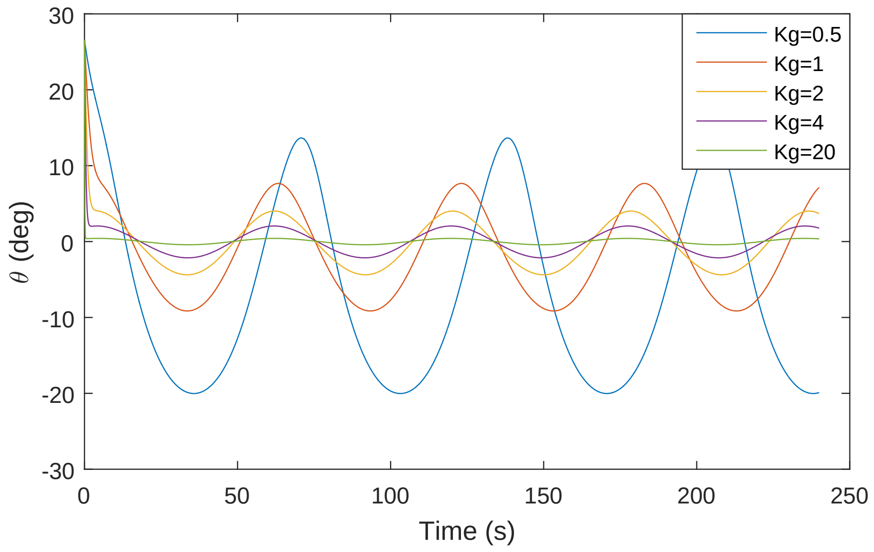


Figure A.2: Simulation result of the angle θ between joint velocity and the desired joint velocity with different control gains k_g using the 2-DoF manipulator.

Table A.1: Oscillation of θ under controller (A.1) in simulation after $20s$

k_g	$\theta_{osc.min}$(deg)	$\theta_{osc.max}$(deg)
0.5	-20.02	13.65
1	-9.14	7.66
2	-4.38	4.02
4	-2.14	2.05
20	-0.42	0.42

Simulation results of the direction error θ under different gyroscopic control gain k_g are shown in Figure A.2. It confirms our conclusion that after the initial convergence stage, θ will oscillate within a bound, which becomes smaller with increasing k_g. The oscillation bounds of θ after $20s$ are summarized in Table A.1. The size of this invariant set is in general inversely proportional to k_g. Additionally, large k_g allows faster convergence of direction of motion in the initial stage, and it seems to also have the effect of ensuring that the oscillation mean of θ stays close to 0.

In summary, this appendix presents the robustness of gyroscopic forces used to steer the direction of joint velocities. Although it is not a direct analysis of the proposed control laws (3.18) and (4.16), it sheds some light on the performance of gyroscopic forces. By making assumptions on the manipulability of the robot and smoothness of the desired velocity field, one may generalize this analysis to velocity control or path following in the task space.

Appendix B

Equation of Motion for the 3-DoF Manipulator

The equation of motion for the 3-DoF Manipulator is derived using the *MatlabSymbolicToolbox*. Because of the length of the equations, we will not directly list the equations here. Instead, this appendix attaches the *Matlab* code used to generate the mass matrix and the Coriolis and centrifugal matrix.

Notation: naming of variables is consistent with those in Table 3.1 except for the addition of link 3. The joint angles are denoted by "q" and "qdot" in the *Matlab* code.

B.1 Mass Matrix

```
M = [I1 + I2 + I3 + L1^2*m2 + L1^2*m3 + L2^2*m3 + m2*r2^2 + m3*r3
    ^2 + r1^2*m1 + 2*L1*sin(q1)*sin(q1 + q2)*m2*r2 + 2*L1*cos(q1)*
    cos(q1 + q2 + q3)*m3*r3 + 2*L1*cos(q1)*cos(q1 + q2)*m2*r2 + 2*
    L2*sin(q1 + q2 + q3)*sin(q1 + q2)*m3*r3 + 2*L2*cos(q1 + q2 +
    q3)*cos(q1 + q2)*m3*r3 + 2*L1*L2*sin(q1)*sin(q1 + q2)*m3 + 2*
    L1*L2*cos(q1)*cos(q1 + q2)*m3 + 2*L1*sin(q1)*sin(q1 + q2 + q3)
    *m3*r3, ...
      I2 + I3 + L2^2*m3 + m2*r2^2 + m3*r3^2 + L1*cos(q1)*cos(q1 +
          q2)*m2*r2 + L1*L2*sin(q1)*sin(q1 + q2)*m3 + L1*L2*cos(q1)*
          cos(q1 + q2)*m3 + L1*cos(q1)*cos(q1 + q2 + q3)*m3*r3 + 2*
    L2*sin(q1 + q2 + q3)*sin(q1 + q2)*m3*r3 + 2*L2*cos(q1 + q2
     + q3)*cos(q1 + q2)*m3*r3 + L1*sin(q1)*sin(q1 + q2 + q3)*
    m3*r3 + L1*sin(q1)*sin(q1 + q2)*m2*r2, ...
  I3 + m3*r3^2 + L1*cos(q1)*cos(q1 + q2 + q3)*m3*r3 + L2*sin(q1
     + q2 + q3)*sin(q1 + q2)*m3*r3 + L2*cos(q1 + q2 + q3)*cos(
    q1 + q2)*m3*r3 + L1*sin(q1)*sin(q1 + q2 + q3)*m3*r3;
  I2 + I3 + L2^2*m3 + m2*r2^2 + m3*r3^2 + L1*cos(q1)*cos(q1 +
    q2)*m2*r2 + L1*L2*sin(q1)*sin(q1 + q2)*m3 + L1*L2*cos(q1)*
    cos(q1 + q2)*m3 + L1*cos(q1)*cos(q1 + q2 + q3)*m3*r3 + 2*
    L2*sin(q1 + q2 + q3)*sin(q1 + q2)*m3*r3 + 2*L2*cos(q1 + q2
     + q3)*cos(q1 + q2)*m3*r3 + L1*sin(q1)*sin(q1 + q2 + q3)*
    m3*r3 + L1*sin(q1)*sin(q1 + q2)*m2*r2, ...
  I2 + I3 + 2*L2*sin(q1 + q2 + q3)*sin(q1 + q2)*m3*r3 + 2*L2*
    cos(q1 + q2 + q3)*cos(q1 + q2)*m3*r3 + L2^2*m3 + m2*r2^2 +
     m3*r3^2, ...
  I3 + L2*sin(q1 + q2 + q3)*sin(q1 + q2)*m3*r3 + L2*cos(q1 + q2
     + q3)*cos(q1 + q2)*m3*r3 + m3*r3^2;
  I3 + m3*r3^2 + L1*cos(q1)*cos(q1 + q2 + q3)*m3*r3 + L2*sin(q1
     + q2 + q3)*sin(q1 + q2)*m3*r3 + L2*cos(q1 + q2 + q3)*cos(
    q1 + q2)*m3*r3 + L1*sin(q1)*sin(q1 + q2 + q3)*m3*r3, ...
  I3 + L2*sin(q1 + q2 + q3)*sin(q1 + q2)*m3*r3 + L2*cos(q1 + q2
     + q3)*cos(q1 + q2)*m3*r3 + m3*r3^2, ...
  m3*r3^2 + I3];
```

B.2 Coriolis and Centrifugal Matrix

```
C = [ - qdot2*L1*L2*cos(q1)*sin(q1 + q2)*m3 + qdot2*L1*L2*sin(q1)
    *cos(q1 + q2)*m3 + m3*r3*qdot2*L1*sin(q1)*cos(q1 + q2 + q3) +
   m3*r3*qdot3*L1*sin(q1)*cos(q1 + q2 + q3) - qdot2*L1*cos(q1)*
   sin(q1 + q2)*m2*r2 - m3*r3*qdot2*L1*cos(q1)*sin(q1 + q2 + q3)
   - m3*r3*qdot3*L1*cos(q1)*sin(q1 + q2 + q3) + qdot2*L1*sin(q1)*
   cos(q1 + q2)*m2*r2 + L2*m3*r3*qdot3*cos(q1 + q2 + q3)*sin(q1 +
    q2) - L2*m3*r3*qdot3*sin(q1 + q2 + q3)*cos(q1 + q2), ...
 - qdot1*L1*L2*cos(q1)*sin(q1 + q2)*m3 - qdot2*L1*L2*cos(q1)*sin(
    q1 + q2)*m3 + qdot1*L1*L2*sin(q1)*cos(q1 + q2)*m3 + qdot2*L1*
    L2*sin(q1)*cos(q1 + q2)*m3 + m3*r3*qdot1*L1*sin(q1)*cos(q1 +
    q2 + q3) + m3*r3*qdot2*L1*sin(q1)*cos(q1 + q2 + q3) + m3*r3*
    qdot3*L1*sin(q1)*cos(q1 + q2 + q3) - qdot1*L1*cos(q1)*sin(q1
    + q2)*m2*r2 - qdot2*L1*cos(q1)*sin(q1 + q2)*m2*r2 - m3*r3*
    qdot1*L1*cos(q1)*sin(q1 + q2 + q3) - m3*r3*qdot2*L1*cos(q1)*
    sin(q1 + q2 + q3) - m3*r3*qdot3*L1*cos(q1)*sin(q1 + q2 + q3)
    + qdot1*L1*sin(q1)*cos(q1 + q2)*m2*r2 + qdot2*L1*sin(q1)*cos(
    q1 + q2)*m2*r2 + L2*m3*r3*qdot3*cos(q1 + q2 + q3)*sin(q1 + q2
    ) - L2*m3*r3*qdot3*sin(q1 + q2 + q3)*cos(q1 + q2), ...
m3*r3*(qdot1 + qdot2 + qdot3)*(L1*sin(q1)*cos(q1 + q2 + q3) - L1*
   cos(q1)*sin(q1 + q2 + q3) + L2*cos(q1 + q2 + q3)*sin(q1 + q2)
   - L2*sin(q1 + q2 + q3)*cos(q1 + q2));
qdot1*L1*L2*cos(q1)*sin(q1 + q2)*m3 - qdot1*L1*L2*sin(q1)*cos(q1
   + q2)*m3 - m3*r3*qdot1*L1*sin(q1)*cos(q1 + q2 + q3) + qdot1*L1
   *cos(q1)*sin(q1 + q2)*m2*r2 + m3*r3*qdot1*L1*cos(q1)*sin(q1 +
   q2 + q3) - qdot1*L1*sin(q1)*cos(q1 + q2)*m2*r2 + L2*m3*r3*
   qdot3*cos(q1 + q2 + q3)*sin(q1 + q2) - L2*m3*r3*qdot3*sin(q1 +
    q2 + q3)*cos(q1 + q2), ...
qdot3*L2*m3*r3*(cos(q1 + q2 + q3)*sin(q1 + q2) - sin(q1 + q2 + q3
   )*cos(q1 + q2)), ...
L2*m3*r3*(qdot1 + qdot2 + qdot3)*(cos(q1 + q2 + q3)*sin(q1 + q2)
   - sin(q1 + q2 + q3)*cos(q1 + q2));
( - qdot1*L1*sin(q1)*cos(q1 + q2 + q3) + qdot1*L1*cos(q1)*sin(q1
   + q2 + q3) - qdot1*L2*cos(q1 + q2 + q3)*sin(q1 + q2) - qdot2*
   L2*cos(q1 + q2 + q3)*sin(q1 + q2) + qdot1*L2*sin(q1 + q2 + q3)
   *cos(q1 + q2) + qdot2*L2*sin(q1 + q2 + q3)*cos(q1 + q2))*m3*r3
   , ...
 - L2*m3*r3*(qdot1 + qdot2)*(cos(q1 + q2 + q3)*sin(q1 + q2) - sin
    (q1 + q2 + q3)*cos(q1 + q2)), ...
0];
```