Real-time Collision Avoidance for Robot Manipulators by Braking Trajectory Prediction

Jihong Zhu

Real-time Collision Avoidance for Robot Manipulators by Braking Trajectory Prediction

by

Jihong Zhu

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on November 24, 2015

Student number:	4320743	
Project duration:	May 1, 2015 – October 31, 2015	
Thesis committee:	Prof. dr. ir. R. Babuska,	TU Delft
	Dr. -Ing. J. Kober,	TU Delft, supervisor
	N. Mansfeld MSc.	DLR, supervisor
	Dr. -Ing. H. Vallery,	TU Delft
	Dr. ir. J.W. van Wingerden,	TU Delft

TUDelft

Abstract

Industrial robot manipulators are widely used in manufacturing processes. These robots are designed to carry out repetitive tasks. For safety concerns, they are often kept in cages, humans control them remotely via separate interfaces, physical Human-Robot Interaction (pHRI) is forbidden. Shortly all this is going to change radically. Next generation of the robots are expected to interact with humans directly. They will collaborate with people in applications such as assisted industrial manipulation, elder care, rehabilitation etc. To enable interaction between humans and robots, human safety has to be guaranteed.

One of the pioneers in robot safety, Prof. Alessandro De Luca in his keynote speech at the International Conference on Robotics and Automation (ICRA) 2015, introduced the hierarchy of control architecture in pHRI which puts safety as the fundamental feature of a robot that has to work close to human beings (See Figure 1). Without safety, coexistence and collaboration can not take place.

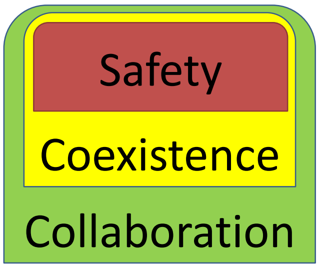


Figure 1: Hierarchy of control architecture, reproduced with reference to [18]

Current state of the art safety strategies include motion planning and collision detection and reaction. At high level control, motion planning is always preferred, as it produces natural movement of the robot. But once the planning fails to find a feasible trajectory to avoid collision, the robot is likely to collide with the object in the environment. Then it is crucial to detect the collision and react to it as soon as possible.

This thesis focus on the augmentation of the current safety layer of the robot control structure by adding on a smart emergency braking layer. The braking is initiated by the prediction of the braking trajectory. The thesis proposed a trajectory prediction framework combined with the braking controllers.

Two different braking controllers are tested in combination with the trajectory prediction scheme. Both results in collision avoidance of the robot in real-time simulation.

One overall collision avoidance scheme has been tested successfully on the DLR Lightweight Robot III (LWR-III). In the test, the robot successfully initiated braking to avoid the collision.

Acknowledgements

I would like to thank my supervisor at DLR, Nico Mansfeld for giving me the opportunity to carry out my master thesis at Institute of Robotics and Mechatronics, DLR. His support during my master thesis is highly appreciated.

I am grateful to have Dr. J. Kober to be my supervisor at TU Delft. I am hugely indebted to him for his guidance and encouragement during my time at DLR.

I want to thank Rico Belder (DLR) for his support in model integration and debugging the RCU, Mehmet Can Özparpucu (DLR) for discussions on linearization and optimal control, also Dr. Simone Baldi (TU Delft) for useful advices on optimal control.

I would like to thank my committee for spending time reading the thesis.

I would like to thank my family and friends for supporting me during my master period.

Delft University of Technology
November 13, 2015

Jihong Zhu

Contents

Acknowledgements **v**

List of Figures **ix**

List of Tables **xi**

1 Introduction **1**

1.1 Motivation 2

1.2 Contribution of the thesis 4

1.3 Outline 5

1.4 Mathematical specification 5

2 Mathematical model for rigid robot manipulators **7**

2.1 Mathematical model 7

2.2 Linearization 7

2.3 Dynamic decoupling 8

3 Braking control laws **11**

3.1 Initial and final conditions. 11

3.2 Joint space analysis. 11

3.3 Decouple space analysis 12

3.3.1 Exsiting braking strategies derived via decoupled space analysis 13

3.3.2 Simultaneous braking 14

3.4 Simulation results 17

4 Braking trajectory prediction **21**

4.1 Problem formulation. 21

4.2 Model analysis 21

4.3 Trajectory prediction in decoupled space 22

4.3.1 Problem statement and model assumptions 22

4.3.2 Prediction for joint feedback control 22

4.3.3 Prediction for simultaneous braking 23

4.3.4 Accuracy enhancement by varying inertia matrix 25

4.4 Prediction validation. 26

5 Integrated collision avoidance scheme **29**

5.1 Model Integration 29

5.2 Safety layer augmentation. 30

5.3 Considered scenario 30

6 Robot implementation **33**

6.1 LWR-III 33

6.1.1 Hardware overview. 33

6.1.2 Control aspects 33

6.2 Experimental steps . . . 34
6.3 Real-time simluation . . . 36
6.3.1 Constant inertia matrix . . . 37
6.3.2 Varying inertia matrix . . . 38
6.4 Robot testing . . . 39

7 Summary and future works 43

7.1 Summary . . . 43
7.1.1 Braking controller . . . 43
7.1.2 Braking trajectory prediction . . . 43
7.1.3 Collision avoidance . . . 44
7.2 Future work . . . 44

Bibliography 45

A Glossary 47

List of Figures

1 Hierarchy of control architecture, reproduced with reference to [18] iii

1.1 YuMi [1] 1
1.2 State of the art of the robots at DLR 2
1.3 The robot moving an knife towards human[10] 3
1.4 Car driving senarios 3
1.5 Collision scenario when no action taken 3
1.6 Collision scenario when braking trajectory is estimated 4
1.7 Fully integrated into the safety control layer of the robot 4

3.1 Angular velocities using joint feedback braking laws 12
3.2 Mass of $m_{i,Q}$ with an external force $\tau_{i,Q}$ 13
3.3 Obtaining a symmetric control region in decoupled space via scaling, reproduced with references to [20] 13
3.4 Example in two dimensional case 15
3.5 Example in two dimensional case where the algorithm fail to find the control input . . 16
3.6 Angular velocities using simultaneous braking law 17
3.7 Input torques - Simultaneous braking law 18
3.8 Oscillation behaviour explained 18
3.9 Input torques - Comparison of original braking, small region method and Algorithm 3 19

4.1 Prediction trajectory using the constant inertia matrix assumption 26
4.2 Prediction trajectory using the varying inertia matrix method 26
4.3 End-effector deviation for 50 test positions - Constant inertia matrix 27
4.4 End-effector deviation for 50 test positions - Accuracy enhancement 27
4.5 Computation time comparison 28

5.1 Integrated collision aviodance scheme 29
5.2 Collision detection and reaction scheme for guaranteeing safety [18] 30
5.3 A scenario where proposed scheme is able to provide additional safety margin 31

6.1 The DLR lightweight robot [17] 34
6.2 The mechatronic joint design of the LWR including actuation, electronics, and sensing [17] 34
6.3 Moving the robot with gravity and Coriolis force compensation 35
6.4 Visualization by Rvis packages 36
6.5 Real time simulation trajectory with the obtstacle 36
6.6 Real time simulation trajectory with the obtstacle 37
6.7 robot motion with collision avoidance (constant inertia matrix) 37
6.8 End-effector trajectory with collision avoidance (constant inertia matrix) 38
6.9 robot motion with collision avoidance (updated inertia matrix) 38
6.10 End-effector trajectory with collision avoidance (updated inertia matrix) 39
6.11 The robot in the left to right motion 40
6.12 Obstacle plane 40
6.13 The robot motion with the collision aviodance, threshold $\epsilon = 0.05$ meter 41
6.14 The end-effector trajectory with the collision aviodance 41

List of Tables

4.1 An analogy between the linear motion and state evolution in (4.6) 23

5.1 A comparison between original and augmented safety layer under the given scenario 31

6.1 Simulation scenario 36
6.2 Robot test scenario 39
6.3 Distance to the obstacle plane with different threshold settings 42

1

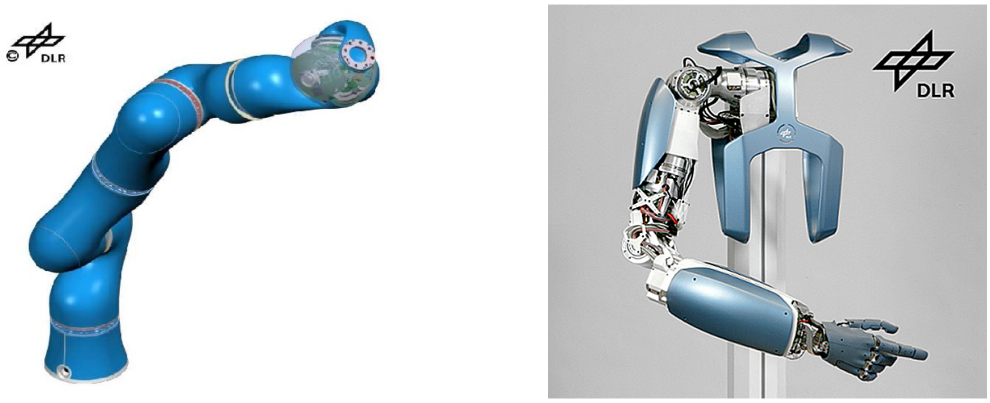
Introduction

In 1920, Rossum's Universal Robots, a play which first depicted a future with artificial humans, introduced the word "robot" into the English language. Since then, robots have become important elements in many science fiction movies and novels. Popular notions often depict a future where humans and robots existing together, sharing the workloads. Currently, although industrial robots are widely used in manufacturing processes, they are often of high mass and torques, operated with high speed and kept in safety cages. Human-robot interaction is forbidden.

However, a paradigm shift has already taken place to design light-weighted, compliant and adaptive robots that are able to interact with humans in an unstructured environment. This trend can be seen already in market predictions, robotics research and designs. In a recent report from Business Intelligence [12], there will be a $1.5 billion market for consumer and business robots by 2019. The consumer-robot market is the fastest growing. The market for consumer and office robots will grow at a compound annual growth rate of 17% between 2014 and 2019, seven times faster than the market for manufacturing robots. A paper from Willow Garage Labs [15] hints also a future that humans and robots are able to work together to handle complex tasks. Just recently, ABB introduced its first collaborative robot YuMi (Figure 1.1) at Hannover Messe on April 13, 2015. At Deutsches Zentrum fur Luft- und Raufahrt (DLR), prominent robot design examples are LWR-III (Figure 1.2a) and Hand arm system (Hasy) (Figure 1.2b).

Figure 1.1: YuMi [1]

The future of collaborative robots is very promising. It can be used for various purposes, such as manufacturing, medical operations, elderly care etc. Especially in countries like Japan and China, the need for robot companions grows rapidly due to the ageing population. The fundamental requirement for these robots is to ensure human safety in the interaction. The primary method of



(a) LWR-III [19]

(b) Hasy [9]

Figure 1.2: State of the art of the robots at DLR

injury avoidance is always to segregate humans and robots from one another [8]. For robot manipulators that are designed to interact with humans, the segregation principle is not applicable. This thesis approaches the safety in pHRI by designing a novel collision avoidance scheme.

1.1. Motivation

In 1938, Isaac Asimov formulated the Three Laws of Robotics which states: ‘A robot may not injure a human being or, through interaction, allow a human being to come to harm’. Human safety has been studied since the very beginning of industrial robotics. But until now, the safety of pHRI is not a solved problem. According to a report from Chinese governmental agency [3], each year approximately 13000 people in China are injured/dead in robot-related accidents. Just recently, a Volkswagen worker was grabbed and killed by a robot in a production plant [6].

Many researchers have worked on safety in pHRI and there are a lot of existing strategies to avoid human injuries. They can be classified into 2 categories:

- Pre-collision strategy: collision avoidance is the primary goal and requires (at least, local) knowledge of the current environment geometry and computationally intensive motion planning techniques [5].

- Post-collision strategy: collision detection and reaction.

The the former eliminate the possibility of collisions while the latter reduce the possibility of human injuries.

Classical pre-collision strategies are based on the Artificial Potential Field (APF) concept proposed by [14]. In APF, the robot is assumed to be a particle that moves under the influence of the force field. The target destination exerts a force that attracts the robot while the obstacle exerts repulsive force. Variation of APF are used in motion planning of robot manipulators [25], [26]. New approaches [16] [28], and [29] use quadratic programming and neural network to solve the planning problem.

On the other hand in post-collision strategies, a collision detection and avoidance scheme is proposed in [11]. The main author himself also volunteered in testing the scheme while the robot was moving an knife towards his arm [1].

The results in [11] are very persuasive in terms of guaranteeing safety in pHRI under collisions. However, collisions are still unavoidable using the scheme. For household applications, people in

[1] A video is available at https://www.youtube.com/watch?v=dMh6cHSG3ng

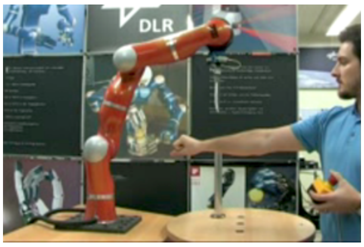


Figure 1.3: The robot moving an knife towards human[10]

general will not like to be hit by the robot, even though it may not cause any harm. In addition, a sudden impact on the human who is executing a normal task, such as holding a cup, might result in unpleasant consequences. Especially in elderly care, collisions may result in severe injuries (even with collision reactions).

In the thesis, a smart emergency braking is proposed to augment the pre-collision strategies. Instead of considering robot collisions, an analogy can be found in car driving scenarios which is more intuitive.

The vehicle is similar to the robot mechanical structure, and human driver act as a controller. When an obstacle appears in the routes of the vehicle, the first action the human considered is to avoid collision by re-plan the routes (Figure 1.4a). However, the car itself has limitation in its velocity and acceleration, changing routes may not be possible. Then naturally the human will initiate brake to avoid collisions or alleviate the impact of the collision (Figure 1.4b). Similarly for the robot, it is necessary to implement a smart emergency braking, which should be initiated when the motion planning fails. Therefore, in this thesis, we are seeking for a smart emergency braking scheme to augment the original safety layer of the robot.

(a) Collision aviodance by trajectory planning

(b) Collision aviodance by braking

Figure 1.4: Car driving senarios

To elaborate, assume a human/object is within the normal operating trajectory of the robot, if no action is taken, the robot will collide with the human/object in its operating speed (Figure 1.5).

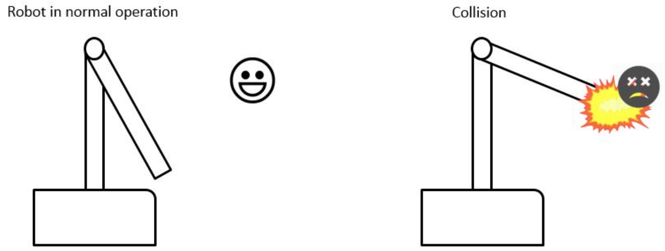


Figure 1.5: Collision scenario when no action taken

With on-line predicted braking trajectory (indicated in Figure 1.6 with red dash plot) and the knowledge of the position of the human/object, the robot is able to know when and/or at which position to initiate braking so that it is able to avoid collision with the human/object (Figure 1.6).

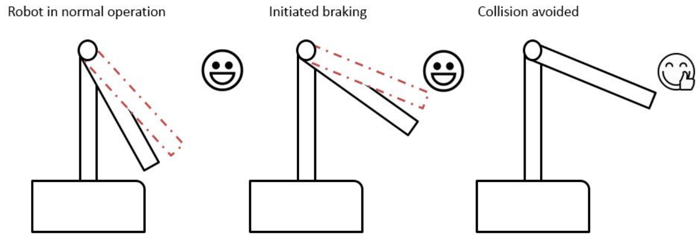


Figure 1.6: Collision scenario when braking trajectory is estimated

The proposed scheme utilizes the prediction of braking trajectory to initiate braking control action and thus avoid collision.

The augmented safety layer is indicated in Figure 1.7. It will provide additional safety margin/redundancy for the robot.

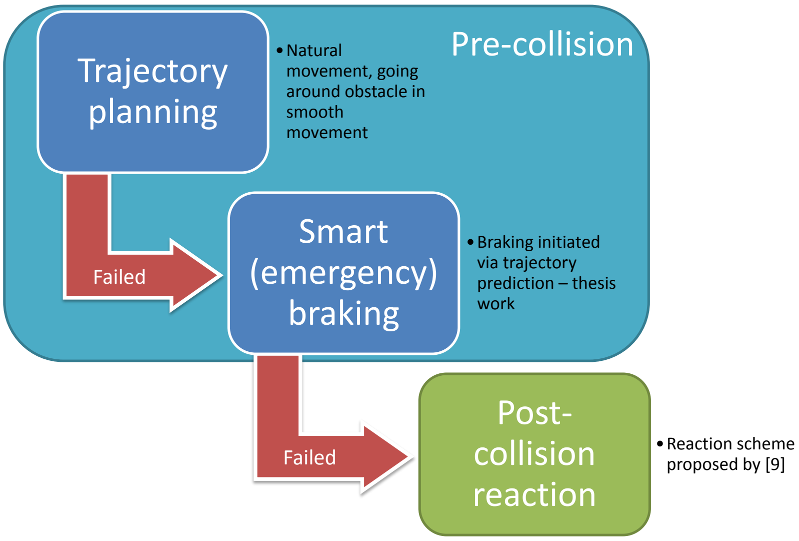

Pre-collision
Trajectory planning
• Natural movement, going around obstacle in smooth movement
Failed
Smart (emergency) braking
• Braking initiated via trajectory prediction – thesis work
Failed
Post-collision reaction
• Reaction scheme proposed by [9]

Figure 1.7: Fully integrated into the safety control layer of the robot

1.2. Contribution of the thesis

The thesis contributes to the pre-collision strategies of robot manipulators by proposing a real-time implementable collision avoidance scheme. The collision is avoided by a braking controller that is activated by a braking trajectory predictor. The predictor provides distances of the end-effector to the object/human in the environment and braking is initiated when the predicted minimum distance to the object/human during braking is less than a certain threshold.

1.3. Outline

The outline of the rest of chapters is explained below:

- Chapter 2: Mathematical model for rigid robot manipulators
  The mathematical model is presented in this chapter for robot manipulators. The model is linearized and decoupled.

- Chapter 3: Braking control laws
  This chapter describes one simple feedback braking control law, and proposed a new braking control law based on decoupled space analysis.

- Chapter 4: Braking trajectory prediction
  This chapter proposes the prediction framework for braking. The simulation is used to validate the prediction accuracy.

- Chapter 5: Integrated collision avoidance scheme
  This chapter introduces the integrated collision avoidance scheme which is the result combination of results in Chapter 3 & 4. One pHRI scenario is considered where the scheme is able to provide additional safety margin.

- Chapter 6: Robot implementation
  The chapter presents the results of collision avoidance scheme in the real-time simulation and robot tests.

- Chapter 7: Summary and future works
  The chapter summarizes the research done in the thesis and provides suggestions for future works.

1.4. Mathematical specification

In the mathematical formulas in the thesis report, non-bold lower case letters such as 'a' represent scalars, lower case bold letters such as '$\boldsymbol{q}$' represent vectors and capital letters such as 'M' represent matrices. Estimated quantities are expressed in '^', such as '$\hat{\theta}$' is the estimated value of θ.

2 Mathematical model for rigid robot manipulators

This chapter gives the mathematical model for rigid robot manipulators. With the gravity, centripetal and Coriolis force compensation available, the non-linear model can be linearized at any (feasible) operating position of the robot. In the last section, a dynamic decoupling is performed on the linearized system. The mathematical framework presented in this chapter serves as a basis for later discussion on braking controller in Chapter 3 and also trajectory prediction design in Chapter 4.

2.1. Mathematical model

A standard assumption in robot dynamics is that manipulators consist of only rigid bodies. In this case, n generalized coordinates are required for a n link robot. The dynamic model of a (n link) rigid joint robot is

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau} + \boldsymbol{\tau}_f \tag{2.1}$$

where the generalized coordinates $\boldsymbol{q} \in \mathbb{R}^n$ can be associated to the positions of the links [4]. The inertia matrix $M(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ is symmetric and positive definite. The centripetal and Coriolis vector is given by $C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \in \mathbb{R}^n$, and $\boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^n$ is the gravity vector. The motor torque vector is $\boldsymbol{\tau} \in \mathbb{R}^n$. The motor friction torque is $\boldsymbol{\tau}_f \in \mathbb{R}^n$. We assume that $\boldsymbol{\tau}_f = \mathbf{0}$. In addition, the motor torques which applied to the robot is always first order filtered, but since the dynamics of the electrical motor is much faster than the mechanical system, in the analysis, ideal motor behaviour is assumed.

The motor torques, and velocities of each joint are usually bounded by

$$|\tau_i| \leq \tau_{\max,i}$$
$$|\dot{q}_i| \leq \dot{q}_{\max,i},$$

where $i = 1,\dots,n$.

2.2. Linearization

Equation (2.1) is a non-linear model, in order to perform linear system analysis, the system needs first to be linearized.

Consider the model in (2.1) with $\boldsymbol{\tau}_f = \mathbf{0}$ and $\tilde{\boldsymbol{\tau}} = \boldsymbol{\tau} - g(\boldsymbol{q}) - C(\boldsymbol{q},\dot{\boldsymbol{q}})$, rewrite (2.1) using $\tilde{\boldsymbol{\tau}}$:

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} = \tilde{\boldsymbol{\tau}} \tag{2.2}$$

In (2.2), gravity and Coriolis force is compensated. Given

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{q} \\ \dot{\boldsymbol{q}} \end{bmatrix}, \; \boldsymbol{u} = \tilde{\boldsymbol{\tau}}$$

Equation (2.2) can be re-written into

$$\dot{\boldsymbol{x}} = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 \\ M^{-1}(\boldsymbol{q}) \end{bmatrix} \boldsymbol{u} \tag{2.3}$$

Given the equilibrium for (2.2):

$$\boldsymbol{x}_{eq} = \begin{bmatrix} \boldsymbol{q}_{eq} \\ 0 \end{bmatrix}, \; \boldsymbol{u}_{eq} = 0 \tag{2.4}$$

Every position with zero angular velocity is an equilibrium for (2.3), therefore the system can be linearized at any given position (assuming no violation of joint limitations).

The first order Taylor expansion of $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$:

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \approx f(\boldsymbol{x}_{eq}, \boldsymbol{u}_{eq}) + \partial_{\boldsymbol{x}} f(\boldsymbol{x}_{eq}, \boldsymbol{u}_{eq})(\boldsymbol{x} - \boldsymbol{x}_{eq}) + \partial_{\boldsymbol{u}} f(\boldsymbol{x}_{eq}, \boldsymbol{u}_{eq})(\boldsymbol{u} - \boldsymbol{u}_{eq}), \tag{2.5}$$

is valid around $(\boldsymbol{x}_{eq}, \boldsymbol{u}_{eq})$.

In (2.3)

$$\dot{x} = f(\boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 \\ M^{-1}(\boldsymbol{q}) \end{bmatrix} \boldsymbol{u}$$

Applying first order Taylor expansion (2.5) on (2.3), the linearized system is then:

$$\dot{\boldsymbol{x}}_{\Delta} = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \boldsymbol{x}_{\Delta} + \begin{bmatrix} 0 \\ \frac{\partial M^{-1}(\boldsymbol{q})}{\partial \boldsymbol{q}} \end{bmatrix} \boldsymbol{x}_{\Delta} + \begin{bmatrix} 0 \\ M^{-1}(\boldsymbol{q}_{eq}) \end{bmatrix} \boldsymbol{u}_{\Delta}, \tag{2.6}$$

where $\boldsymbol{x}_{\Delta} = \boldsymbol{x} - \boldsymbol{x}_{eq}$, and $\boldsymbol{u}_{\Delta} = \boldsymbol{u} - \boldsymbol{u}_{eq}$. The size of ${}^{\partial M^{-1}(\boldsymbol{q})}/_{\partial \boldsymbol{q}} \boldsymbol{x}_{\Delta}$ is very small and is negligible without affecting the original dynamics [21]. Therefore in the rest of the thesis it is omitted to simplify analysis.

At position $\boldsymbol{q}$ with input $u = \tilde{\boldsymbol{\tau}}$, (2.6) yields:

$$M\ddot{\boldsymbol{q}} = \tilde{\boldsymbol{\tau}} \tag{2.7}$$

where $M = M(\boldsymbol{q})$, and $\boldsymbol{q}$ can be any (feasible) operation positions.

2.3. Dynamic decoupling

The linearized system is a n^{th} (n is the number of the links) order Multiple-Input Multiple-Output (MIMO) system. The coupling of the states complicates the analysis. One common approach of decoupling is to perform eigenvalue decomposition to diagonalize M [1]. In decoupled space, n^{th} order MIMO system become n Single-Input Single-Output (SISO) system which can be analysed separately. Therefore in this section the linearized system is decoupled.

The decoupling is achieved by eigenvalue decomposition. Given the $n \times n$ matrix A with n independent eigenvalues, if (2.8) holds, λ is a eigenvalue of A, and $\boldsymbol{x}$ is the corresponding eigenvector.

$$A\boldsymbol{x} = \lambda \boldsymbol{x} \tag{2.8}$$

[1] Not all square matrices are diagonalizable, but in this case, M can always be diagonalized. The details is explained in this section.

Based on (2.8), we have

$$A[\boldsymbol{x}_1\boldsymbol{x}_2\boldsymbol{x}_3\ldots\boldsymbol{x}_n] = \begin{bmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \lambda_n \end{bmatrix} [\boldsymbol{x}_1\boldsymbol{x}_2\boldsymbol{x}_3\ldots\boldsymbol{x}_n]$$

The matrix A can be diagonalized (2.9)

$$\begin{aligned} A &= S\Lambda S^{-1} \\ \text{where } S &= [\boldsymbol{x}_1\boldsymbol{x}_2\boldsymbol{x}_3\ldots\boldsymbol{x}_n] \\ \Lambda &= \begin{bmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \lambda_n \end{bmatrix} \end{aligned} \tag{2.9}$$

If A is symmetric (i.e. $A = A^T$), we can write $A = Q\Lambda Q^{-1} = Q\Lambda Q^T$, where Q is an orthogonal matrix [23]. In addition, eigenvalues of A are real. The diagonal elements of Λ are real numbers [23].

M is the inertia matrix of the robot. It is positive definite and symmetric. A decoupling matrix Q can be applied to (2.7) to diagonalize M where

$$M = QM_QQ^T$$

and M_Q is the diagonal matrix. The decoupling law is

$$\begin{aligned} \boldsymbol{q}_Q &= Q^T\boldsymbol{q} \\ \boldsymbol{\tau}_Q &= Q^T\boldsymbol{\tau} \end{aligned} \tag{2.10}$$

The new dynamics in decoupled space is then

$$M_Q\ddot{\boldsymbol{q}}_Q = \tilde{\boldsymbol{\tau}}_Q \tag{2.11}$$

Since M is positive definite and symmetric, the diagonal elements of M_Q are positive real numbers.

3

Braking control laws

Braking control is an important function for robot manipulators to ensure safety in pHRI. Either in the safety standard for general machinery (IEC 60204-1) or robots (ISO 10218-1), a stopping function is always compulsory. In ISO 10218-1 the requirement is that every robot shall have a protective stop function and an emergency stop function [2].

For robot manipulators, there exists already a few braking control strategies. The braking control laws can either be derived based on joint space analysis [5], [13], and [22] or decoupled space analysis [20], [21]. In the first section, the initial and final condition for braking is presented. Then two braking control laws are proposed: a feedback control law and a novel braking control law derived based on decoupled space analysis. All the control laws in this chapter are derived under the assumption that centripetal, Coriolis and the gravity force of the robot is already compensated.

3.1. Initial and final conditions

A braking action can happen at any instance during the robot operations, and end with the robot being at rest. To represent the braking motion in mathematical terms, the initial and final conditions for braking is presented:

$$\begin{aligned}
&\boldsymbol{q}(0) = \boldsymbol{q}_0, \quad \boldsymbol{q}(t_f) \text{ free, or } \boldsymbol{q}(t_f) = \boldsymbol{q}_d(t_f)\\
&\dot{\boldsymbol{q}}(0) = \dot{\boldsymbol{q}}_0, \quad \dot{\boldsymbol{q}}(t_f) = \boldsymbol{0}\\
&\ddot{\boldsymbol{q}}(0) = \ddot{\boldsymbol{q}}_0, \quad \ddot{\boldsymbol{q}}(t_f) = \boldsymbol{0},
\end{aligned}$$

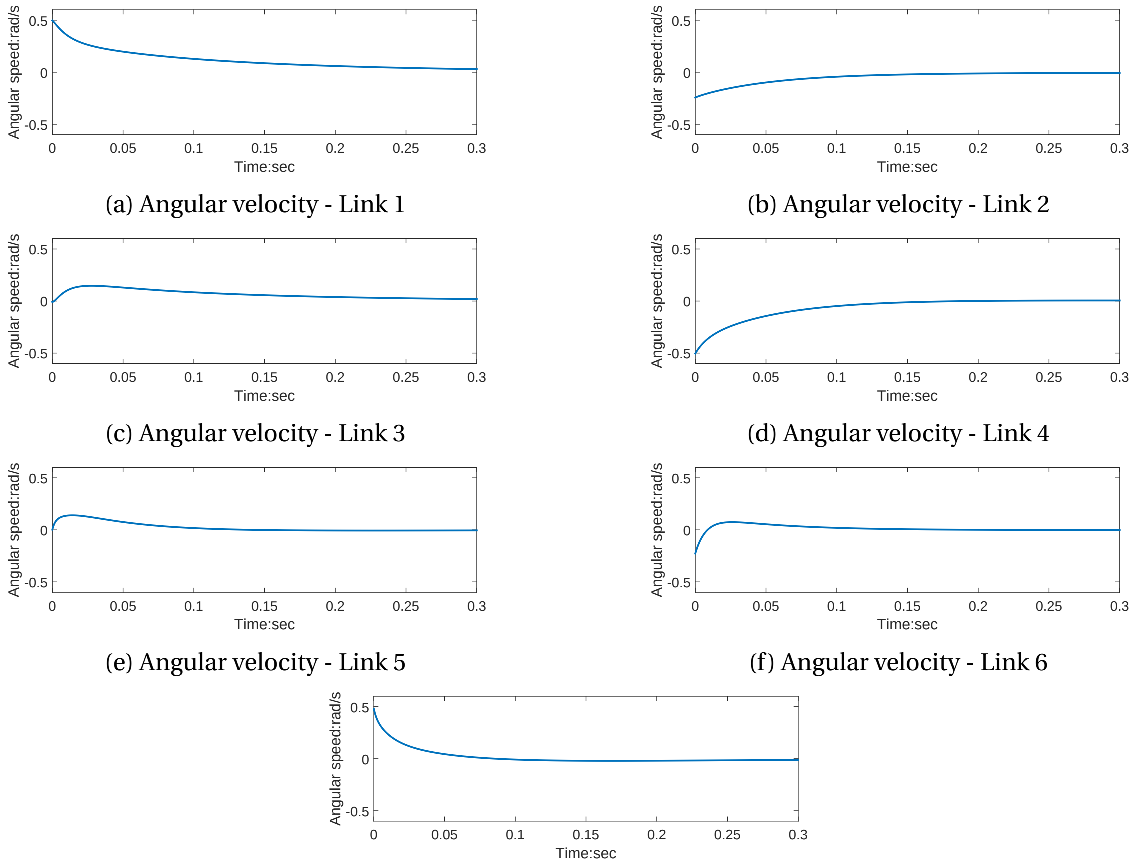
where $\boldsymbol{q}(0)$, $\dot{\boldsymbol{q}}(0)$, and $\ddot{\boldsymbol{q}}(0)$ are the initial link position, velocity, and acceleration. At the final time t_f the link velocity and acceleration are zero, while the final position can be unspecified or a certain desired position.

3.2. Joint space analysis

A simple feedback law can be derived for the robot in joint space. For each joint the feedback law is

$$u_i = \tilde{\tau}_i = -k_{i,d}\dot{q}_i$$

The control law is straightforward, and by tuning of $k_{i,d}$ one can achieve a reasonably fast braking for the robot manipulator. Figure 3.1 shows the evolution of the link velocities during braking simulated with the robot dynamics model. Frictions and motor dynamics is not included in the model. From Figure 3.1, there are some overshoot behavior observed on link 3,5 and 6. Especially in link 3 and 5, the initial velocity is 0. The overshoot behavior may originate from the coupling in the robot dynamics.



(a) Angular velocity - Link 1

(b) Angular velocity - Link 2

(c) Angular velocity - Link 3

(d) Angular velocity - Link 4

(e) Angular velocity - Link 5

(f) Angular velocity - Link 6

(g) Angular velocity - Link 7

Figure 3.1: Angular velocities using joint feedback braking laws

The joint space feedback law allows a smooth reaching of the equilibrium. The controller is easy to implement. However, the tuning of $k_{i,d}$ depends mostly on experiences. In addition, in the feedback law, the actuator limits are not taken into account.

3.3. Decouple space analysis

In practical implementation of the braking controller, actuator limits should always be taken into consideration. In addition for braking, the robot is desired to be stopped as soon as possible. In decoupled space, each state can be analysed individually. However, inputs are coupled via transformation matrix Q, the maximum/minimum values of each control input are no longer independent from each other. Therefore although the dynamic is simple, finding the time optimal control in decoupled space without violation of actuation limits is not a trivial task. Existing method includes [20]. In this section, a novel braking control in decoupled space is proposed. The control is able to stop the robot in near-optimal time.

In (2.11) for individual state:

$$m_{i,Q}\ddot{q}_{i,Q} = \tilde{\tau}_{i,Q}, \tag{3.1}$$

each state derivative needs to be decelerated to 0 (i.e., $\dot{q}_{i,Q} = 0,\ \forall i$). Note that in decoupled space, each state is dependent only on the input $\tau_{i,Q}$. The overall dynamic for individual state is a second order system. The new state in decouple space behaves like a mass of $m_{i,Q}$ with a speed of $\dot{q}_{i,Q}$ and an external force $\tau_{i,Q}$ applied on (See Figure 3.2).

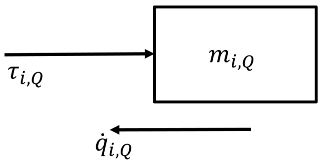


Figure 3.2: Mass of $m_{i,Q}$ with an external force $\tau_{i,Q}$

3.3.1. Exsiting braking strategies derived via decoupled space analysis

To find a region in decoupled space where the maximum and minimum inputs are independent, [20] seek for a hypercube Ω'_Q with independent minima/maxima in decoupled space that lies completely within the original bounds Ω when it is re-transformed to original space (Ω'). The largest possible hypercube Ω' can be found by Algorithm 1 where the original maximum and minimum torque limits in original space are $\boldsymbol{\tau}'_M := [\tau'_{M,1}, \ldots, \tau'_{M,n}]^T$ and $\boldsymbol{\tau}'_m := [\tau'_{m,1}, \ldots, \tau'_{m,n}]^T$. The two dimensional example is explained in Figure 3.3.

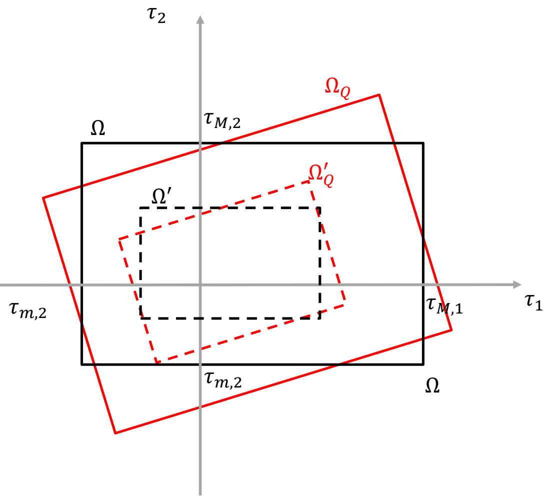


Figure 3.3: Obtaining a symmetric control region in decoupled space via scaling, reproduced with references to [20]

The control bounds in decoupled space now satisfy

$$\tau'_{V,m} \leq \tau'_{V,i} \leq \tau'_{V,M}. \tag{3.2}$$

The control law for braking each modal coordinate is

$$\tau'_{V,i} = \begin{cases} \tau'_{V,m,i} & \text{if } \dot{q}_{V,i} \geq 0 \\ \tau'_{V,M,i} & \text{if } \dot{q}_{V,i} < 0, \end{cases} \tag{3.3}$$

where $i = 1, \ldots, n$. Finally, the control input in original space is given by retransformation via V

$$\boldsymbol{\tau} = V\boldsymbol{\tau}'_V + \boldsymbol{g}(\boldsymbol{q}). \tag{3.4}$$

In terms of braking time, the decoupling based controller (3.4) shows generally a poorer performance than the joint space controller because the motor torques are not fully exploited [20]. In the next section, a new braking scheme is proposed using decoupled space analysis.

Algorithm 1 Calculate control bounds in decoupled space [20]

```
$k \leftarrow 1$
**for** $i \leftarrow 1$ to $2^n$ **do**
    $\boldsymbol{v}_{i,\tilde{V}} \leftarrow V\,\boldsymbol{v}_i$
    $\boldsymbol{d}_i \leftarrow |\boldsymbol{v}_{i,\tilde{V}}| - |\boldsymbol{v}_i|$
    $j \leftarrow \operatorname{argmax} \boldsymbol{d}_i(j)$
    **if** $|\boldsymbol{v}_{i,\tilde{V}}(j)| > |\boldsymbol{v}_i(j)|$ **then**
        $k \leftarrow \min\left\{k, \frac{|\boldsymbol{v}_i(j)|}{|\boldsymbol{v}_{i,\tilde{V}}(j)|}\right\}$
    **end if**
**end for**
$\boldsymbol{\tau}'_{V,M} \leftarrow k\boldsymbol{\tau}'_M$
$\boldsymbol{\tau}'_{V,m} \leftarrow k\boldsymbol{\tau}'_m$
```

3.3.2. Simultaneous braking

For braking, in decoupled space analysis, each state can be seen as a mass with initial velocity, and the torque (seen as force) is applied to reduce the velocity to zero. In the controller design, this simple model is used to derive time optimal control strategy. In joint space, assume that the time to stop the i^{th} joint is t_i. The final time to stop the robot is then:

$$t_{\text{final}} = \max(t_i)$$

At the final time of braking, all joint speeds are decelerated to 0, $\dot{\boldsymbol{q}} = 0$. In decoupled space, correspondingly, $\dot{\boldsymbol{q}}_Q = 0$. The final time expressed in decoupled space is:

$$t_{\text{final}} = \max(t_{i,Q})$$

In decoupled space, all states are controlled independently. Thus to stop the whole robot in minimum time, each state in decoupled space can be controlled such that all of them are stopped simultaneously as indicated in

$$t_{\text{final}} = t_{1,Q} = t_{2,Q} = \ldots = t_{n,Q} \tag{3.5}$$

There is no need to apply a bigger torque on one state stop it sooner since the overall time depends on the slowest descending state. For each state in decoupled space:

$$t_{i,Q} = \frac{m_{i,Q}\dot{q}_{i,Q}}{\tau_{i,Q}} \tag{3.6}$$

Combine (3.5) and (3.6), the relationship (3.7) is obtained:

$$\frac{m_{1,Q}\dot{q}_{1,Q}}{\tau_{1,Q}} = \frac{m_{2,Q}\dot{q}_{2,Q}}{\tau_{2,Q}} = \ldots = \frac{m_{i,Q}\dot{q}_{i,Q}}{\tau_{i,Q}} = \ldots = t_{\text{final}} \tag{3.7}$$

From (3.7) the torque ratio in decoupled space can be obtained:

$$\frac{\tau_{i,Q}}{\tau_{j,Q}} = \frac{m_{i,Q}\dot{q}_{i,Q}}{m_{j,Q}\dot{q}_{j,Q}}, \ \forall i,j \tag{3.8}$$

Then, to ensure the control is within the actuator limits, let's first consider a two actuators case indicated in Figure 3.4.

The control limits in the joint space are given by hyper-rectangle Ω. Ω is transformed into the decoupled space via coordinate transformation matrix Q. The resulting hyperplane is Ω_Q. According to ratio calculated in (3.8), a directional vector can be obtained in decoupled space, indicated by

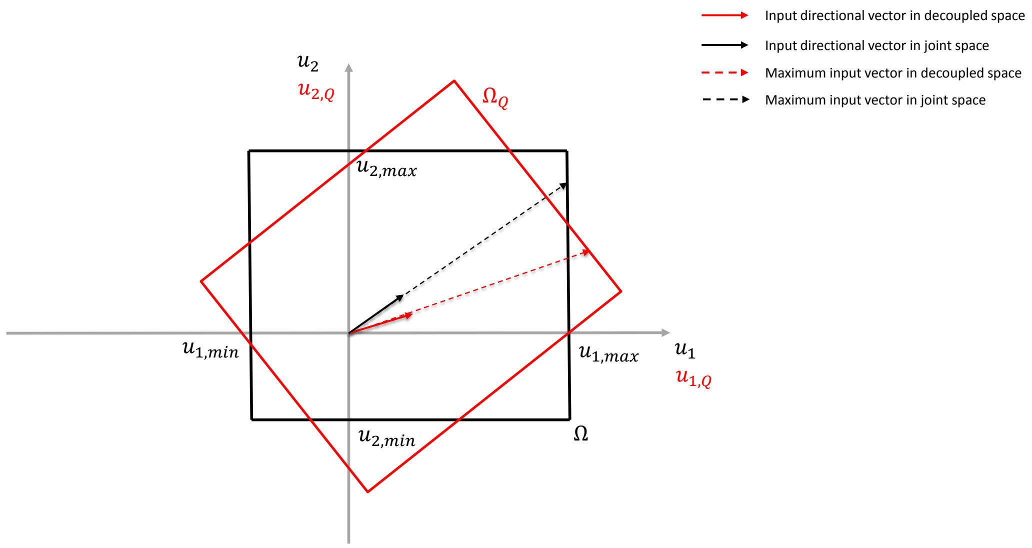

→ Input directional vector in decoupled space
→ Input directional vector in joint space
- - → Maximum input vector in decoupled space
- - → Maximum input vector in joint space
u_2
$u_{2,Q}$
Ω_Q
$u_{2,max}$
$u_{1,min}$
$u_{1,max}$
u_1
$u_{1,Q}$
$u_{2,min}$
Ω

Figure 3.4: Example in two dimensional case

solid red arrow in Figure 3.4. The directional vector can be transformed back to joint space which is indicated by solid black arrow, then scaled to fit within Ω, represented by black dash arrow. In this way, the simultaneous braking control laws with actuator limits in the joint space can be obtained. The algorithm is indicated below, where n is the number of joints.

Algorithm 2 Find the optimal time control based on decoupled space analysis with actuator limits

```
$\boldsymbol{\tau}_{base,Q} \leftarrow 0$
$\boldsymbol{\tau}_{limit} \leftarrow 0$
for $i \leftarrow 1$ to $n$ do
    {find the directional vector in decoupled space}
    $\tau_{base,Q}(i) \leftarrow -\mathrm{sign}(\dot{q}_{i,Q}) m_{i,Q} |\dot{q}_{i,Q}|$
end for
{transform the directional vector into joint space}
$\boldsymbol{\tau}_{base} \leftarrow Q\boldsymbol{\tau}_{base,Q}$
{create hyperrectangle Ω in joint space}
for $i \leftarrow 1$ to $n$ do
    if sign$(\dot{q}_{i,Q}) = 1$ then
        $\tau_{limit} \leftarrow u_{\max}$
    end if
    if sign$(\dot{q}_{i,Q}) = -1$ then
        $\tau_{limit} \leftarrow u_{\min}$
    end if
end for
{find the maximum ratio applied on the directional vector without violation of actuator limits}
$r \leftarrow \min(|\frac{\boldsymbol{\tau}_{base}}{\boldsymbol{\tau}_{limit}}|)$
{time optimal control in joint space}
$\tilde{\boldsymbol{\tau}} \leftarrow r\boldsymbol{\tau}_{base}$
```

Based on ratio calculated in decoupled space, the resulting joint space controller is indicated in (3.9).

$$\boldsymbol{\tau} = r\boldsymbol{\tau}_{\text{base}} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) \tag{3.9}$$

The control is able to stop the robot in a short time. All the joints will be stopped simultaneously. Although from decoupled space analysis, the control is able to stop the robot in optimal time, since the actual dynamics of the robot is non-linear, the stopping time will only be near-optimal. One constraint of this approach is that control limits in joint space should satisfy (3.10).

$$\begin{aligned} u_{\text{i,max}} > 0,\ \forall i \\ u_{\text{i,min}} < 0,\ \forall i \end{aligned} \tag{3.10}$$

Otherwise, the algorithm will fail to find a feasible control. Figure 3.5 shows a scenario where (3.10) is not satisfied ($u_{1,\text{min}} > 0$). In this case the directional vector is not inside the hyper-rectangle Ω. The torque vector that provides the largest deceleration is indicated by light-blue dash line in Figure 3.5, but the actual calculated torques are much smaller. In addition, under the same scenario if Ω was in the left half plane (correspond to ($u_{1,\text{max}} < 0$)), no feasible control can be derived from Algorithm 2.

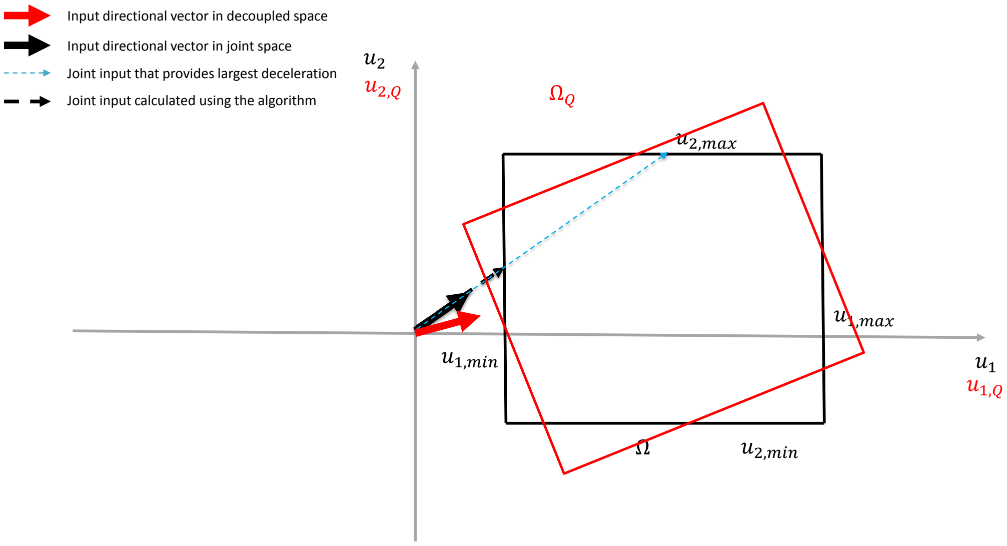

Input directional vector in decoupled space
Input directional vector in joint space
Joint input that provides largest deceleration
Joint input calculated using the algorithm
u_2
$u_{2,Q}$
Ω_Q
$u_{2,max}$
$u_{1,max}$
$u_{1,min}$
u_1
$u_{1,Q}$
Ω
$u_{2,min}$

Figure 3.5: Example in two dimensional case where the algorithm fail to find the control input

For braking control in decoupled space,

$$\begin{aligned} \boldsymbol{u}_{\text{max}} &= \boldsymbol{\tau}_{\text{max}} - C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q}) \\ \boldsymbol{u}_{\text{min}} &= -\boldsymbol{\tau}_{\text{max}} - C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q}) \end{aligned}$$

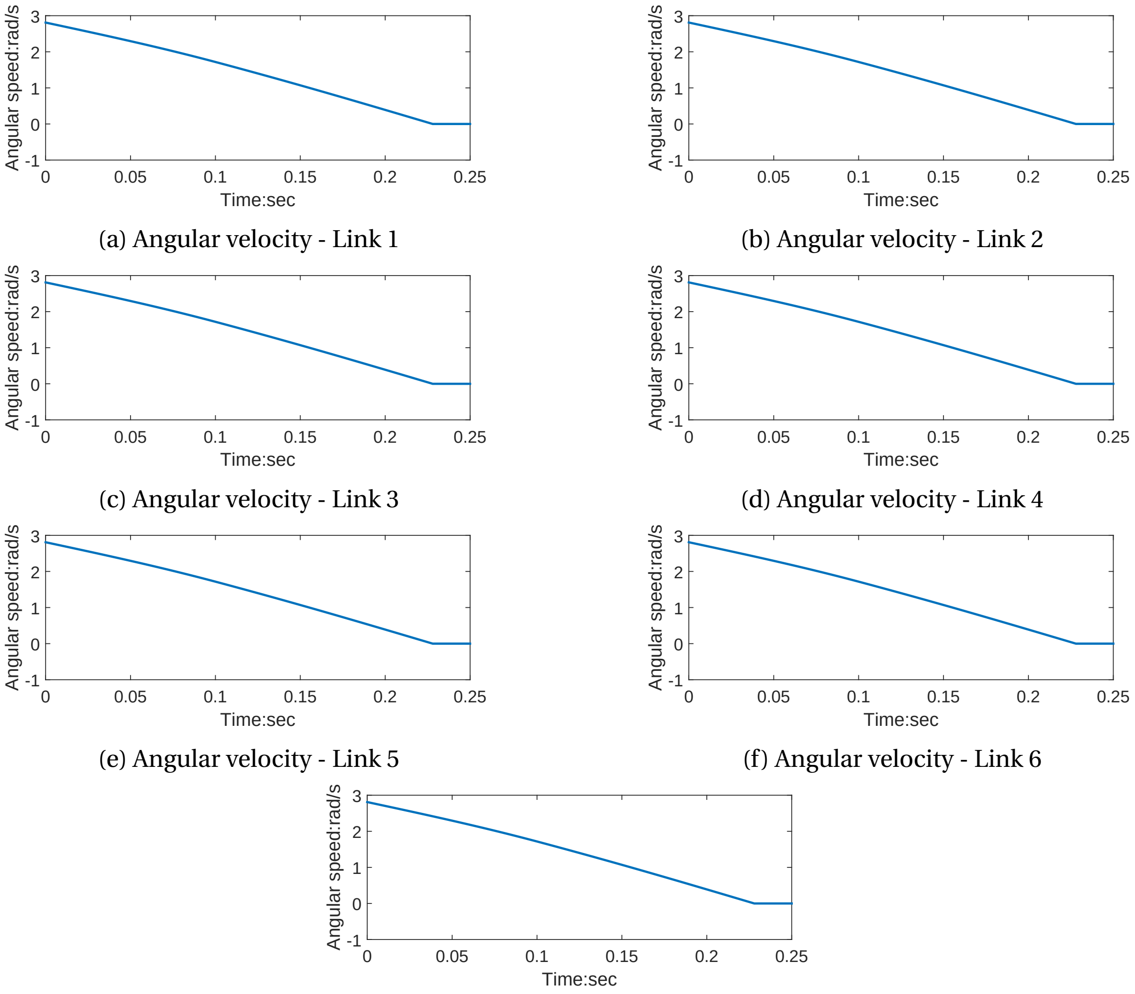
The input torque limits should always be greater than gravity plus centripetal and Coriolis torques:

$$\boldsymbol{\tau}_{\text{max}} > |C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q})|$$

Therefore (3.10) is always satisfied.

3.4. Simulation results

The proposed controller is tested on robot dynamics model. Since the controller in Algorithm 2 is of bang-bang type, to avoid applying maximum torques, the limit $(\boldsymbol{u}_{\max}, \boldsymbol{u}_{\min})$ is scaled half in the simulation. The initial joint positions are selected from a trajectory data of LWR-III, and velocities are assumed to be maximum for all joints.



(a) Angular velocity - Link 1

(b) Angular velocity - Link 2

(c) Angular velocity - Link 3

(d) Angular velocity - Link 4

(e) Angular velocity - Link 5

(f) Angular velocity - Link 6

(g) Angular velocity - Link 7

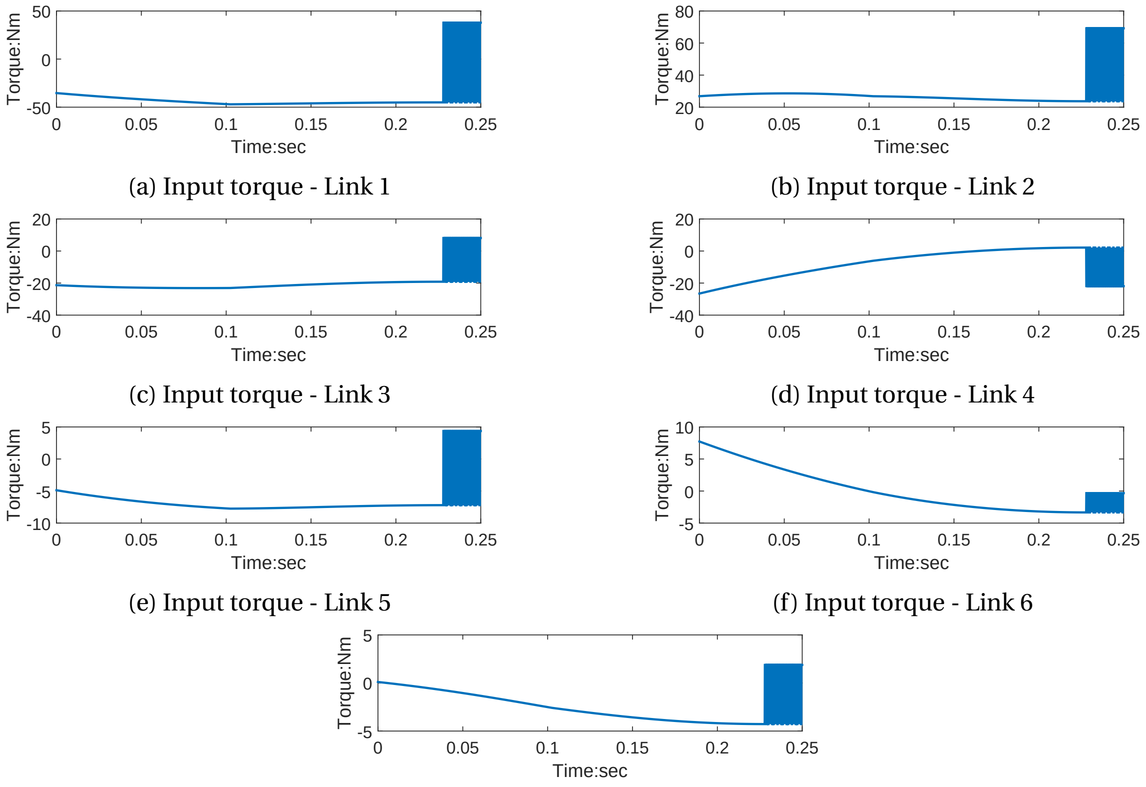
Figure 3.6: Angular velocities using simultaneous braking law

In the simulation, the braking controller is able to stop the robot at maximum velocities in approximately 0.23 second. Angular velocities of the links reach 0 simultaneously, which is expected. But the bang-bang type control results in an oscillatory behaviour at low velocities as shown in Figure 3.7.

The general solution to the problem is to insert a small region and switch to a smooth control law. In [5], the authors proposed a bang-bang type controller for stopping the robot and corresponding switch control to enable a smooth reaching of the equilibrium. The same kind of regime can also be applied on simultaneous braking. The resulting control law is indicated in (3.11).

$$\tau = \left\{ \begin{array}{ll} r\tau_{i,\text{base}} & \text{if } |\dot{q}_i| \geq \epsilon_i \\ r\tau_{i,\text{base}} |\dot{q}_i| / \epsilon_i & \text{if } |\dot{q}_i| < \epsilon_i \end{array} \right\} + c(\boldsymbol{q}, \dot{\boldsymbol{q}}) + g(\boldsymbol{q}), \tag{3.11}$$

with $\epsilon_i > 0$.



(a) Input torque - Link 1

(b) Input torque - Link 2

(c) Input torque - Link 3

(d) Input torque - Link 4

(e) Input torque - Link 5

(f) Input torque - Link 6

(g) Input torque - Link 7

Figure 3.7: Input torques - Simultaneous braking law

An alternative solution can be derived by analyzing the cause of the oscillatory behaviour.

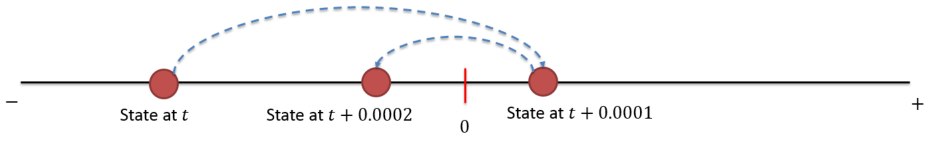


Figure 3.8: Oscillation behaviour explained

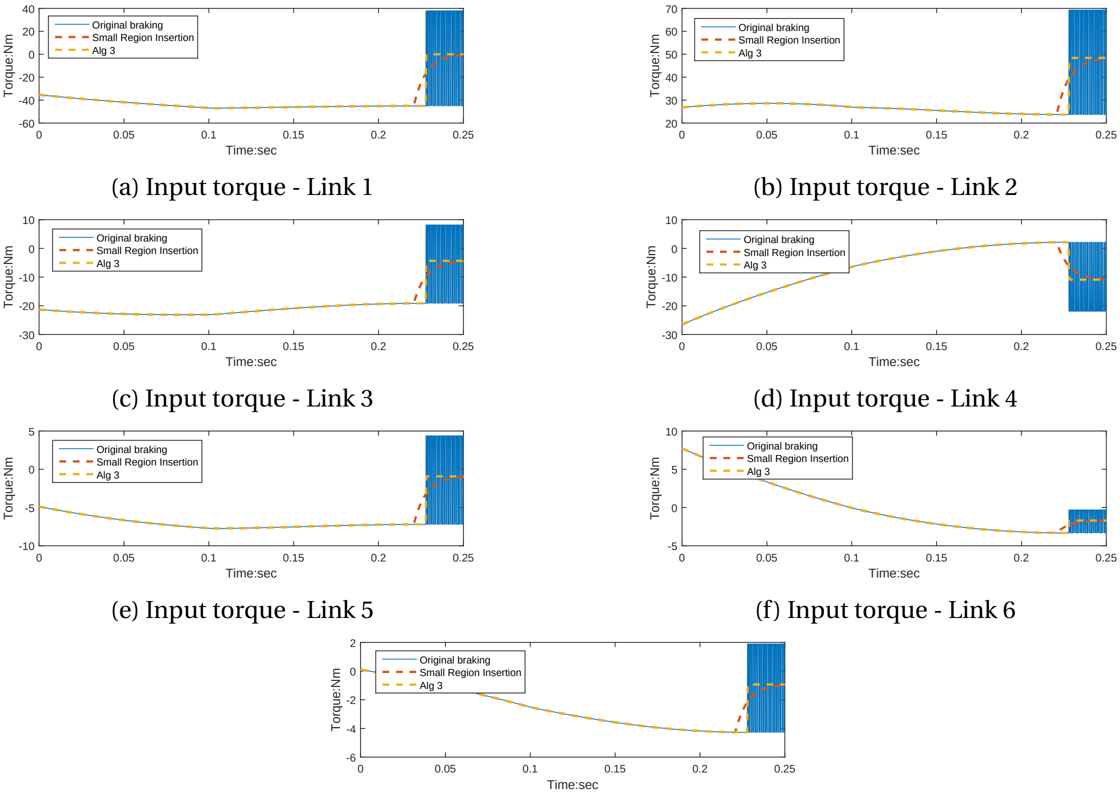
The simulation is in the discrete time with a sampling rate of 0.0001 second, therefore the current control will be applied for 0.0001 second, until the next control input is triggered. Simultaneous braking is a bang-bang type controller. The cause of the oscillation is explained with Figure 3.8.

The state at t is negative. The controller always check for velocities and apply maximum torques available. It might happen that after one control period it does not reside in 0 but becomes positive, then after the next control period it becomes negative again. The state will oscillate around the 0. The problem can be solved by introduce a check for oscillation before applying the torques by Algorithm 3.

Algorithm 3 Smoothing the simultaneous braking

```
$\tilde{\boldsymbol{\tau}}$ = Algorithm 2($\tilde{\boldsymbol{\tau}}_m, \tilde{\boldsymbol{\tau}}_M, \boldsymbol{q}_Q, \dot{\boldsymbol{q}}_Q, M_Q$)
{Transform the control into decoupled space}
$\tilde{\boldsymbol{\tau}}_Q = Q^T \tilde{\boldsymbol{\tau}}$
{Check for the oscillation, then switch control}
for $i \leftarrow 1$ to $n$ do
    {Check for the time that $i^{th}$ state reaches 0}
    $t_{\text{check}} = |\frac{m_{i,Q}\dot{\boldsymbol{q}}_Q}{u_{i,Q}}|$
    if $t_{\text{check}} <=$ sampling rate then
        $\tilde{\boldsymbol{\tau}}_{i,Q} \leftarrow -\frac{m_{i,Q}\dot{\boldsymbol{q}}_Q}{\text{sampling rate}}$
    end if
end for
```

Below shows the input torques of simulation using small region method and Algorithm 3 under the same initial conditions.



(a) Input torque - Link 1

(b) Input torque - Link 2

(c) Input torque - Link 3

(d) Input torque - Link 4

(e) Input torque - Link 5

(f) Input torque - Link 6

(g) Input torque - Link 7

Figure 3.9: Input torques - Comparison of original braking, small region method and Algorithm 3

Both methods can provide smooth torques in the end, but small region method offers smoother transition at low velocities.

4

Braking trajectory prediction

The braking controller is activated by the prediction of the braking trajectory along the operation of the robot. However, the trajectory prediction is not a trivial task. It requires solving a non-linear time variant system, which is often computationally intensive. In this chapter we show that the trajectory can be solved analytically in decoupled space without complex computation. In Chapter 5, an integrated scheme is presented to achieve collision avoidance.

4.1. Problem formulation

Given the robot dynamics (2.1), and the control law $\boldsymbol{\tau} = f(\boldsymbol{q}, \dot{\boldsymbol{q}}, ...)$, one wants to predict the trajectory of the joints from current robot states $\boldsymbol{q}_0, \dot{\boldsymbol{q}}_0$ until the robot reaches its stopping position $\dot{\boldsymbol{q}} = 0, \ddot{\boldsymbol{q}} = 0$.

The joint trajectory can be transformed into the end-effector trajectory in Cartesian space by forward kinematics. An object in the environment can be represented by a set of Cartesian points, denoted by $\mathscr{X}_{\text{obs}}$. The minimum distance from the object to the trajectory, $\boldsymbol{d} = \text{min\_dist}(\boldsymbol{x}_{\text{End}}, \mathscr{X}_{\text{obs}})$, can be realized by either GJK algorithm [7] or the Kinematic Continuous Collision Detection Library [24]. For braking starts at t_0 ends at t_{final}, it is then possible to predict the minimum distance between the end-effector and an object during braking, namely $\hat{d}_{\text{min}} = \min(\boldsymbol{d}), t \in [t_0, t_{\text{final}}]$. A collision is likely to occur if $\hat{d}_{\text{min}} \leq 0$.

4.2. Model analysis

Assume a general control law

$$\boldsymbol{\tau} = \boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ldots)$$

Given robot dynamics model (2.1), the most straight forward way of predicting the trajectory is to try to solve

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ldots) \tag{4.1}$$

To utilize the trajectory prediction, the trajectory needs to be solved on-line with a fast sampling rate (for LWR-III is 0.001 second). Intensive computation at a relatively fast sampling rate is often hardware demanding. In this section, we simplified the model used for prediction, and motivate each step of simplification. A performance evaluation is given in the next section.

One can always rewrite $\boldsymbol{f}$ into two parts:

$$\tilde{\boldsymbol{\tau}} = \boldsymbol{f} - C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q}) \tag{4.2}$$

$$\boldsymbol{\tau}_{c+g} = C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q}) \tag{4.3}$$

Therefore, one can always write (4.1) into (4.4):

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} = \tilde{\boldsymbol{\tau}}, \tag{4.4}$$

where $\tilde{\boldsymbol{\tau}} = \boldsymbol{f} - \boldsymbol{\tau}_{c+g}$.

In Section 2.3, it is shown that (4.4) can be linearized at any (feasible) position of the robot operation. At any given instance, using the position information, (4.4) is linearized:

$$M\ddot{\boldsymbol{q}} = \tilde{\boldsymbol{\tau}} \tag{4.5}$$

The trajectory can be obtained by solving (4.5). In MATLAB, (4.5) needs to be rewritten into a set of first order differential equations, then solved using Ordinary Differential Equations solvers. The coupling of the equations complicates the solution of (4.5).

A dynamic decoupling can be applied to (4.5) as indicated in Section 2.2. After decoupling each state is independently controlled. The trajectory of each state (in decoupled space) can be solved individually.

4.3. Trajectory prediction in decoupled space

In this section, the prediction framework is presented under the assumption that the inertia matrix is constant during braking. Predictions are made for two braking control laws using the framework. At last, an accuracy enhancement is proposed by considering varying inertia matrix in the framework.

4.3.1. Problem statement and model assumptions

Given the initial joint positions $\boldsymbol{q}_0$ and velocities $\dot{\boldsymbol{q}}_0$ at braking, the initial positions and velocities in decoupled space can be obtain using (2.10):

$$\begin{aligned}\boldsymbol{q}_{Q,0} &= Q^T\boldsymbol{q}_0\\ \boldsymbol{\tau}_{Q,0} &= Q^T\boldsymbol{\tau}_0\end{aligned}$$

Individual state dynamic is

$$m_{i,Q}\ddot{q}_{i,Q} = \tilde{\tau}_{i,Q} \tag{4.6}$$

The state trajectory can be obtained by solving (4.6) till $\dot{q}_{i,Q} = 0$.

For braking, the robot is desired to be stopped as soon as possible, joint movements during braking can be assumed a local behaviour around a constant configuration. Therefore one can assume during the braking period, the inertia matrix M stays constant.

4.3.2. Prediction for joint feedback control

For joint space feedback control on a n (rigid) link robot, consider the simplest case where $K_d = -k_d \cdot I$, assuming a constant inertia matrix during braking, the dynamic of the close loop system is

$$M\ddot{\boldsymbol{q}} = K_d\dot{\boldsymbol{q}} \quad \text{with} \quad K_d = \begin{bmatrix} -k_d & 0 & \dots & 0\\ 0 & -k_d & \dots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \dots & -k_d \end{bmatrix} \tag{4.7}$$

In decoupled space,

$$\begin{aligned}\ddot{\boldsymbol{q}} &= Q\ddot{\boldsymbol{q}}_Q\\ \dot{\boldsymbol{q}} &= Q\dot{\boldsymbol{q}}_Q\end{aligned} \tag{4.8}$$

Take (4.8) into (4.7) yields

$$MQ\ddot{\boldsymbol{q}}_Q = K_d Q\dot{\boldsymbol{q}}_Q, \tag{4.9}$$

multiply (4.9) both side with Q^T

$$Q^T MQ\ddot{\boldsymbol{q}}_Q = K_d Q^T Q\dot{\boldsymbol{q}}_Q \tag{4.10}$$

Since

$$Q^T Q = I$$

(4.10) is equivalent to

$$M_Q\ddot{\boldsymbol{q}}_Q = K_d\dot{\boldsymbol{q}}_Q$$

For individual state in decoupled space:

$$m_{i,Q}\ddot{q}_{i,Q} = -k_d\dot{q}_{i,Q} \tag{4.11}$$

Assume $x_i = (\begin{smallmatrix} q_{i,Q} \\ \dot{q}_{i,Q} \end{smallmatrix})$, (4.11) becomes

$$\dot{x}_i = Ax_i, \text{ with } A = \begin{bmatrix} 0 & 1 \\ 0 & \frac{-k_d}{m_{i,Q}} \end{bmatrix} \tag{4.12}$$

It is an autonomous system, assume initial states $x_i(t_0)$, the evolution of the states is then

$$x_i(t) = e^{At}x_i(t_0) \tag{4.13}$$

if no friction is considered, the close loop system reaches equilibrium asymptomatically. The trajectory in decoupled space can be solved by taking the time evolution of (4.13) (See Algorithm 4.).

4.3.3. Prediction for simultaneous braking

Under the assumption, simultaneous braking will result in constant torques during the whole braking period. Before attempt to solve (4.6), an analogy can be made comparing the dynamic of (4.6) and a linear motion of a mass in Table 4.1.

Table 4.1: An analogy between the linear motion and state evolution in (4.6)

	linear motion	state evolution
speed	v	$\dot{q}_{i,Q}$
mass\inertia	m	$m_{i,Q}$
force\torque	F	$\tilde{\tau}_{i,Q}$
acceleration	$a = {}^F/_m$	$a = {}^{\tilde{\tau}_{i,Q}}/_{m_{i,Q}}$

To solve for the distance in a linear motion given initial position d_0 and velocity v_0

$$d = d_0 + v_0 t + \frac{1}{2}\frac{F}{m}t^2$$

Similarly for the state trajectory

$$q_{i,Q} = q_{0i,Q} + \dot{q}_{0i,Q}t_{\text{final}} + \frac{1}{2}\frac{\tilde{\tau}_{i,Q}}{m_{i,Q}}t^2_{\text{final}}, \tag{4.14}$$

where t_{final} is the time when $\dot{q}_{i,Q} = 0$.

Algorithm 4 Trajectory prediction of joint feedback control under constant inertia mass assumption

```
{Assign a stopping velocity for the while loop}
$v_{\text{stop}} \leftarrow \delta$
{Assign a step size for trajectory prediction}
step $\leftarrow t$
{Number of iteration}
$j = 1$
{Dynamics decoupling}
$[M_Q, Q]$ = eigenvalue decomposition($M(\boldsymbol{q})$)
$\hat{\boldsymbol{q}}_Q \leftarrow Q^T \boldsymbol{q}_0$
$\dot{\hat{\boldsymbol{q}}}_Q \leftarrow Q^T \dot{\boldsymbol{q}}_0$
{Solving the trajectory in decoupled space}
while max($|\dot{\hat{\boldsymbol{q}}}_Q|$) > $v_{\text{stop}}$ do
    for $i = 1$ to $n$ do
        {Calculate A matrix}
        $A \leftarrow [0\ 1; 0\ {}^{-k_d}/_{m_{i,Q}}]$
        {Assign initial conditions}
        $x_0 = [q_{i,Q}; \dot{q}_{i,Q}]$
        {State evolutions}
        $x_{\text{new}} = \text{expm}(A \cdot j \cdot t) x_0$    {expm is the matrix exponential}
        $\hat{q}_{i,Q} \leftarrow x_{\text{new}}(1)$
        $\dot{\hat{q}}_{i,Q} \leftarrow x_{\text{new}}(2)$
    end for
end while
{Iteration increment}
$j = j + 1$
{Position and velocity back in original space}
$\hat{\boldsymbol{q}}(j) \leftarrow Q \hat{\boldsymbol{q}}_Q$
$\dot{\hat{\boldsymbol{q}}}(j) \leftarrow Q \dot{\hat{\boldsymbol{q}}}_Q$
{End-effector position in Cartesian space}
$\hat{\boldsymbol{x}}_{\text{End}}(j)$ = forward kinematics($\hat{\boldsymbol{q}}$)($j$)
{Distance to the object}
$\hat{d}(j)$ = mini_dist($\hat{\boldsymbol{x}}_{\text{End}}(j), \mathscr{X}_{\text{obs}}$)
{Minimum distance to the object for entire trajectory}
$\hat{d}_{\text{min}} = \min(\hat{\boldsymbol{d}})$
```

In (4.14), $q_{0i,Q}$, $\dot{q}_{0i,Q}$ and $m_{i,Q}$ are fixed value which relates to current robot configuration and operation status. The braking trajectory depends on $\tilde{\tau}_{i,Q}$ and t_{final}. And these two quantities are related. We referred back to Chapter 2's statement which indicates that proposed simultaneous braking gives an edge to solving the trajectory in (4.14)'s framework. The simultaneous braking is a bang-bang type controller. Under constant compensation torques and inertia matrix assumption, the braking torque $\tilde{\tau}_{i,Q}$ for each state is constant over the braking period.

The constant torque $\tilde{\tau}_{i,Q}$ can be calculated based on Algorithm 2 given initial configuration of the robot. Then for each state in decoupled space:

$$t_{\text{final}} = \frac{m_{i,Q}\dot{q}_{0i,Q}}{\tilde{\tau}_{i,Q}}$$

Take $\tilde{\tau}_{i,Q}$ into (4.14), and solve (4.14) until t_{final}, the trajectory of the state during braking can be obtained. The calculation is done in discrete manner. The method is indicated in Algorithm 5.

Algorithm 5 Trajectory prediction of simultaneous braking under constant inertia mass assumption

```
$n \leftarrow$ number of points on the whole braking trajectory
{Available torque for braking controller in original space}
$\tilde{\boldsymbol{\tau}}_m(\boldsymbol{q}) \leftarrow -\left(\boldsymbol{\tau}_{\max} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q})\right)$
$\tilde{\boldsymbol{\tau}}_M(\boldsymbol{q}) \leftarrow \boldsymbol{\tau}_{\max} - C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q})$
{Dynamics decoupling}
$[M_Q, Q] =$ eigenvalue decomposition$(M(\boldsymbol{q}))$
$\boldsymbol{q}_{0,Q} \leftarrow Q^T \boldsymbol{q}_0$
$\dot{\boldsymbol{q}}_{0,Q} \leftarrow Q^T \dot{\boldsymbol{q}}_0$
{Braking torques calculation}
$\tilde{\boldsymbol{\tau}} =$ Algorithm 2$(\tilde{\boldsymbol{\tau}}_m(\boldsymbol{q}), \tilde{\boldsymbol{\tau}}_M(\boldsymbol{q}), \boldsymbol{q}_{0,Q}, \dot{\boldsymbol{q}}_{0,Q}, M_Q)$
{Braking torques in decoupled space}
$\tilde{\boldsymbol{\tau}}_Q \leftarrow Q^T \tilde{\boldsymbol{\tau}}$
{Final time calculation}
$\hat{t}_{\text{final}} \leftarrow \frac{m_{i,Q}\dot{q}_{0i,Q}}{\tilde{\tau}_{i,Q}}$
{Time vector for braking trajectory}
$t_0 \leftarrow 0$
$t_i \leftarrow \frac{\hat{t}_{\text{final}} \cdot i}{n}, \qquad j = 1, 2, \ldots, n$
{Calculation & analysis of full braking trajectory}
for $j = 1$ to $n$ do
    for $i = 1$ to length$(\boldsymbol{q})$ do
        $\hat{q}_{i,Q}(j) \leftarrow q_{0i,Q} + \dot{q}_{0i,Q} t_j + \frac{1}{2}\frac{\tilde{\tau}_{i,Q}}{m_{i,Q}} t_j^2$
    end for
    {Position back in original space}
    $\hat{\boldsymbol{q}}(j) \leftarrow Q\hat{\boldsymbol{q}}_Q(j)$
    {End-effector position in Cartesian space}
    $\hat{\boldsymbol{x}}_{\text{End}}(j) =$ forward kinematics$(\hat{\boldsymbol{q}}(j))$
    {Distance to the object}
    $\hat{d}(j) =$ mini_dist$(\hat{\boldsymbol{x}}_{\text{End}}(j), \mathscr{X}_{\text{obs}})$
end for
{Minimum distance to the object for entire trajectory}
$\hat{d}_{\min} = \min(\hat{\boldsymbol{d}})$
```

The decoupled framework is able to predict the trajectory during braking with some deviation. One exemplary result is obtained by Algorithm 5 with a braking time of 0.1 second. The comparison of real and predicted trajectory indicated in Figure 4.1. The deviation comes from the simplification of the robot dynamics in the prediction framework.

4.3.4. Accuracy enhancement by varying inertia matrix

Based on previous model, to further increase accuracy, one can update the inertia matrix and compensation torques ($C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q})$) during the prediction. This requires solving (4.12) and (4.14) in a step by step manner, in each iteration, $\hat{\boldsymbol{q}}(j)$ and $\hat{\dot{\boldsymbol{q}}}(j)$ are used to update inertia matrix and compensation torques. These new values then being used in the next iteration. The enhanced method is applied on Algorithm 5. The result of the enhancement under the same braking condition with Figure 4.1 is indicated in Figure 4.2.

With a updating rate of 0.01 second, now the framework is able to predict the trajectory accurately.

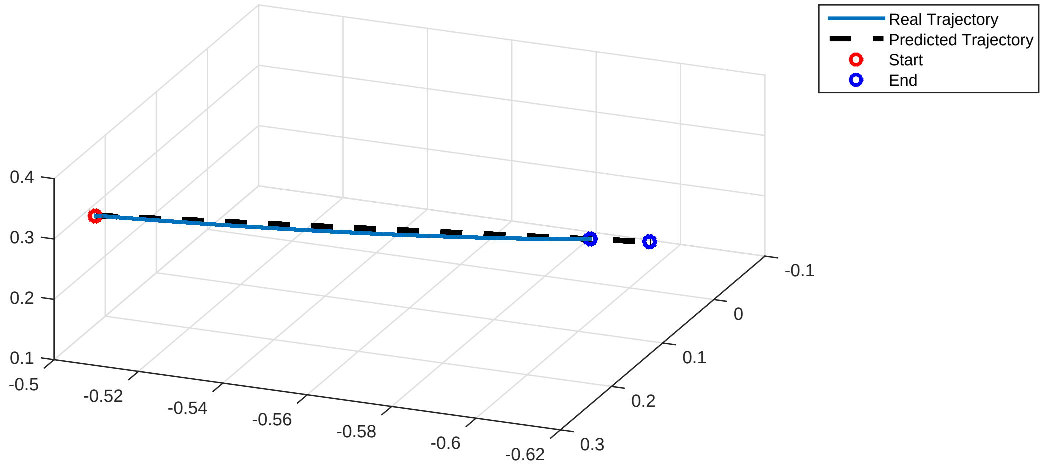


Figure 4.1: Prediction trajectory using the constant inertia matrix assumption

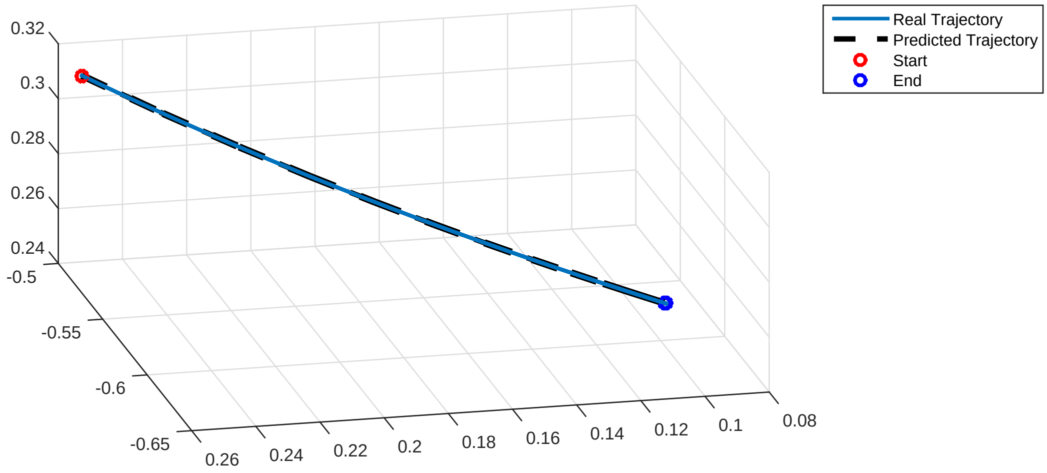


Figure 4.2: Prediction trajectory using the varying inertia matrix method

4.4. Prediction validation

Final position deviation of the end-effector is one of the measures of the prediction accuracy. It is defined in (4.15)

$$d_{\text{final}} = \sqrt{(x_{\text{final}} - \hat{x}_{\text{final}})^2 + (y_{\text{final}} - \hat{y}_{\text{final}})^2 + (z_{\text{final}} - \hat{z}_{\text{final}})^2} \tag{4.15}$$

Fifty joint positions from real motion of the robot were selected. To consider the worst case scenarios, each position is assigned with maximum joint velocity. They are unlikely to happen in practical operations. But for validation purpose, since the assumption is valid only at the vicinity of the constant configuration, higher velocity will usually requires longer time to stop the robot, thus greater changes in the configuration.

Figure 4.3 shows the distribution of $\boldsymbol{d} = [x_{\text{final}} - \hat{x}_{\text{final}}, y_{\text{final}} - \hat{y}_{\text{final}}, z_{\text{final}} - \hat{z}_{\text{final}}]$ in Cartesian coordinate for fifty different initial positions with Algorithm 5. The trajectory is solved by a MATLAB script and then compared with the Simulink simulation.

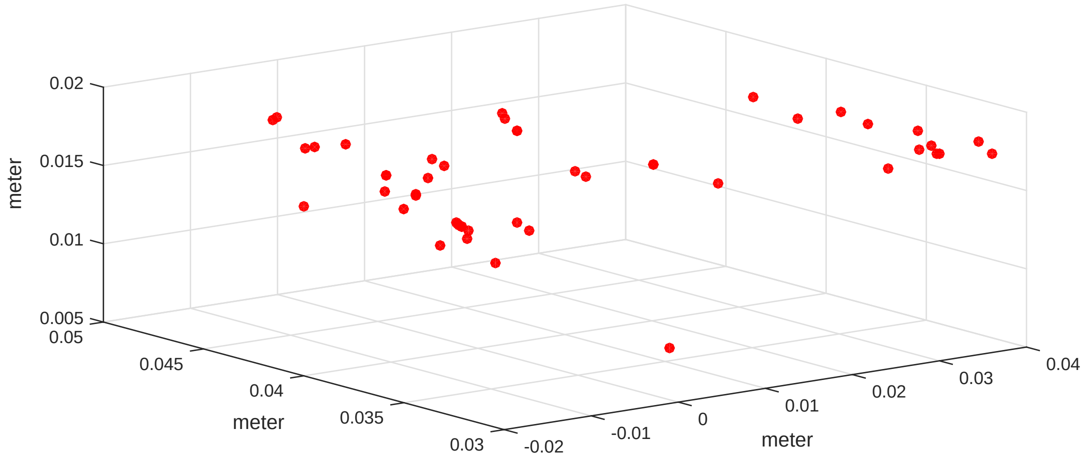


Figure 4.3: End-effector deviation for 50 test positions - Constant inertia matrix

The maximum deviation of the final position is 0.0538 meter.

If the trajectory is solved by the enhancement method with a step of 0.01 second, the resulting deviation is shown in Figure 4.4.

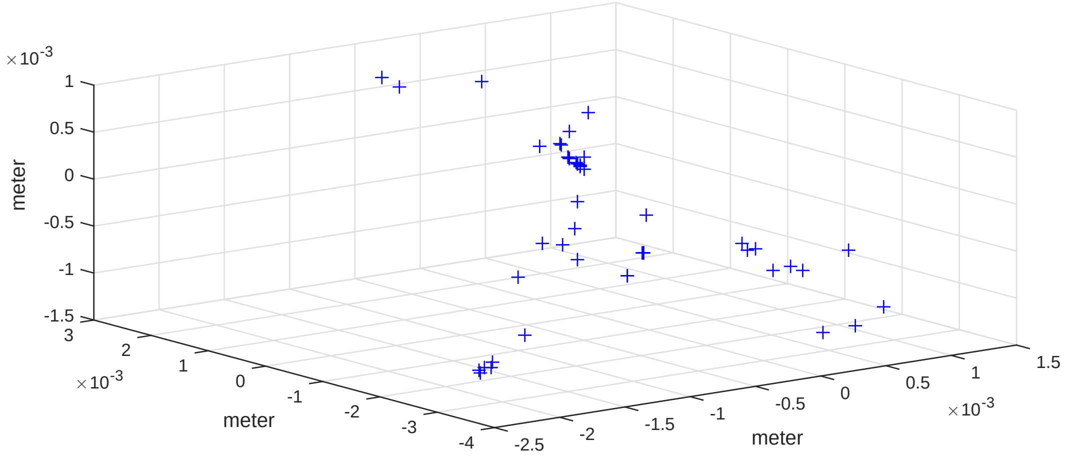


Figure 4.4: End-effector deviation for 50 test positions - Accuracy enhancement

The maximum deviation of the final position is 0.0035 meter. The enhancement method has significantly improved the accuracy of the prediction framework. The computation time of both is evaluated by *tic* and *toc* command in MATLAB. The computation time for fifty data using both methods is indicated in Figure 4.5.

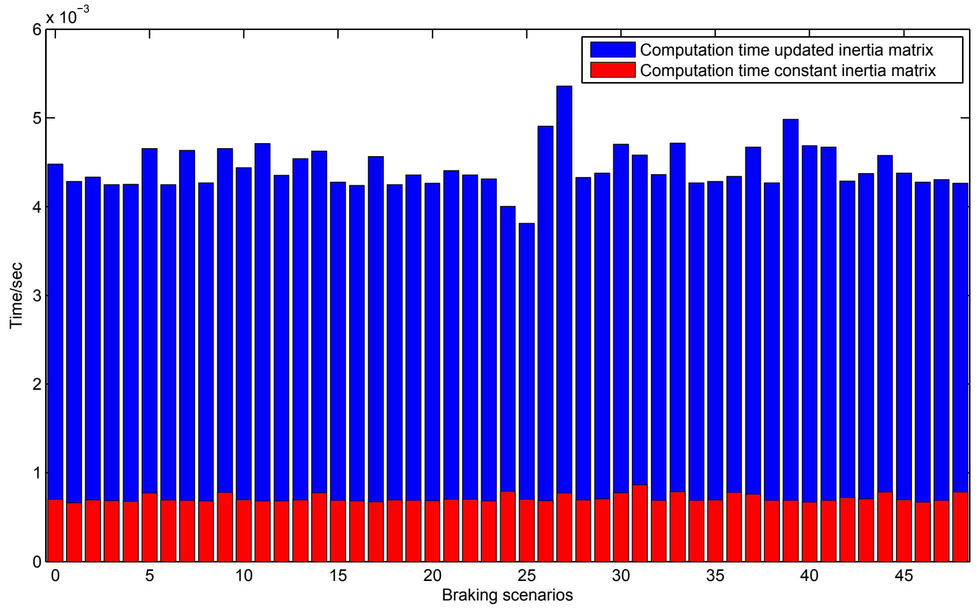


Figure 4.5: Computation time comparison

It is expected that updated inertia matrix method is more computational demanding. Figure 4.5 gives an idea on the differences of both methods in terms of computation time in the MATLAB environment for the fifty data set [1].

[1] Simulink environment is used in practical implementation of the prediction framework with minor changes on the algorithm.

5

Integrated collision avoidance scheme

In this chapter, the overall scheme is presented by combination of the braking controller and the trajectory prediction. Then the scheme is integrated into the safety layer of the robot. At the end of this chapter, a pHRI scenario is presented where the scheme is able to offer additional safety margin for the robot as stated in Chapter 1.

5.1. Model Integration

The overall scheme is indicated in Figure 5.1.

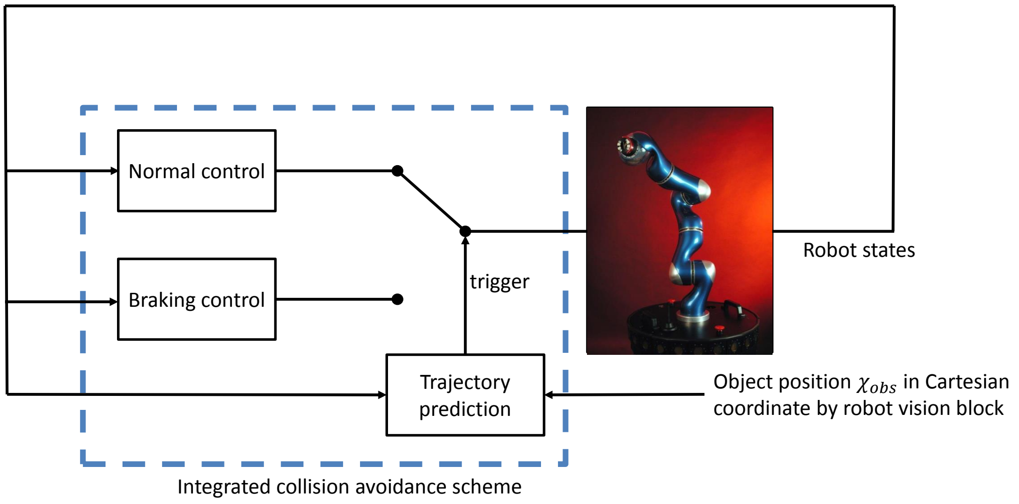

Normal control
Braking control
trigger
Trajectory prediction
Robot states
Object position χ_{obs} in Cartesian coordinate by robot vision block
Integrated collision avoidance scheme

Figure 5.1: Integrated collision aviodance scheme

The trajectory prediction model runs in parallel with the robot normal control, the current robot states are fed into the prediction block. In addition, it requires the support from the robot vision block to feed the position of the object in the Cartesian coordinate, thus to predict the minimum distance $\hat{d}_{\text{min}}$ (of the end-effector) to the object in the Cartesian coordinate.

To trigger the braking controller, a threshold $\epsilon \geq 0$ can be selected:

$$y = \text{step}(\hat{d}_{min} - \epsilon)\begin{cases} 0 \text{ (no action required)} & \text{if } \hat{d}_{min} > \epsilon \\ 1 \text{ (initiate braking)} & \text{else} \end{cases}$$

5.2. Safety layer augmentation

In the keynote speech of Prof. De Luca at ICRA 2015, safety layer is regarded as the fundamental of pHRI [1]. His research is on collision detection and reaction. The scheme is activated when the normal control mode fails as depicted in Figure 5.2.

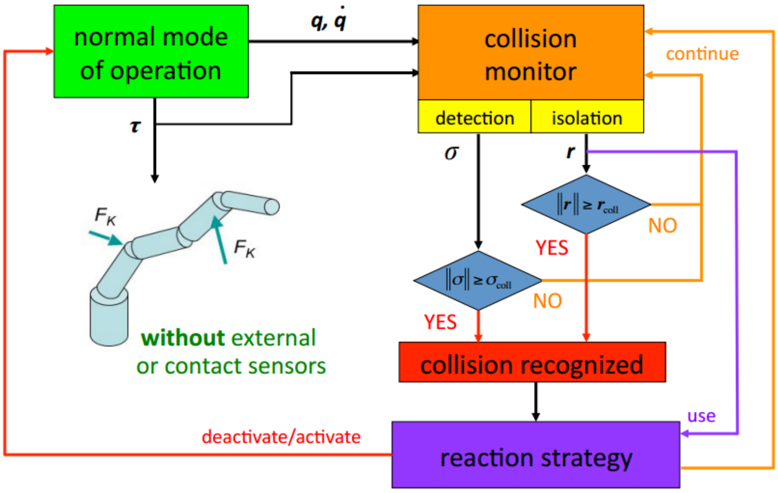


Figure 5.2: Collision detection and reaction scheme for guaranteeing safety [18]

The logic is straight-forward, if the normal mode of operation result in a collision, then the collision monitor will detect it, and then react to it. The proposed scheme is able to augment the original safety layer.

As discuss in Chapter 1, collisions are always not desirable in pHRI (even the collision will not cause harm to the human). The integrated scheme proposed in this thesis is a low level collision avoidance strategy. At higher level, normal trajectory planning is always more natural. But once the planning fails, instead of switching directly to collision detection and reaction, an emergency braking might still able to avoid collisions. The feasibility of initiating the braking depends on the trajectory prediction. If emergency braking is feasible, it is always better to brake then avoid the collision than detect and react to the collision. But if emergency braking is also not feasible, the scheme proposed in [5] can be initiated to react to the collision.

After integration into original safety layer, an additional safety margin is added to the robot. To further illustrate augmentation of the safety layer by the proposed scheme, a common scenario is considered in the last section.

5.3. Considered scenario

Consider a robot is executing a task where a human holding a coffee suddenly walked into the planned trajectory of the robot as depicted in Figure 5.3. It will be a common scenario in most of the workplaces in the future where collaborative robots are used to help humans. In this case collisions will interfere with the human's normal task, the coffee might spill and burn the human's hand, also the cup might fall. These consequences are not desirable in the workplace.

[1] Full video available at https://www.youtube.com/watch?v=DMQbsp-iVzw

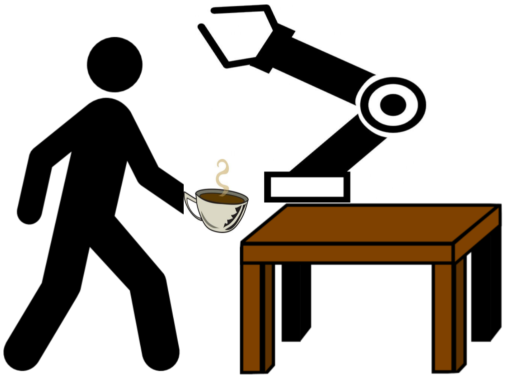


Figure 5.3: A scenario where proposed scheme is able to provide additional safety margin

The results of original and augmented safety layer are presented in Table 5.1

Table 5.1: A comparison between original and augmented safety layer under the given scenario

	Condition	Result	
		Original Safety	Augmented Safety
Trajectory planning	Succeed	Collision avoided	Collision avoided
	Failed	Check for collision detection and avoidance	Check for emergency braking
Emergency braking	Succeed	N/A	Collision avoided
	Failed	N/A	Check for collision detection and avoidance
Collision detection and avoidance	N/A	Collision with reaction	Collision with reaction

For the original safety, the collision can only be avoided if the trajectory planning is able to find a feasible solution. But for the augmented safety, the emergency braking can be used as a backup collision avoidance solution when the trajectory planning fails, thus increase the safety margin of the robot. If the emergency braking also fails, both methods will result in the same collision scenario.

6

Robot implementation

In the previous chapters, the braking controller and trajectory prediction is tested on a robot dynamics model on the Simulink. It is a simplification of the real robot model. In this chapter the integrated scheme is first tested on the real-time Robot Control Unit (RCU) with a full dynamics model. The full dynamics model is much more accurate, it includes also the motor dynamics and some friction terms. If the scheme works on the RCU model, it will have high possibility to work on the real robot. After successfully testing the scheme on the real-time RCU, the RCU can be connected to the LWR-III for the final performance evaluation. The chapter starts by introducing the LWR-III. Then a short explanation is given for the experimental steps. In the end, the real-time and robot testing results are presented.

6.1. LWR-III

The DLR lightweight robots (see Figure 6.1) have been developed for application areas which are fundamentally different from the ones of classical industrial robotics [17]. It has been developed for interaction with humans in unstructured, everyday environment. For mechanical design, a low robot mass is desired to enable mobility and minimize injury risk. For electronic design, the robots are equipped with a high number of sensors, such as joint torque sensors, redundant position sensing, and wrist force-torque sensing. In addition, integration of motor and sensor electronics reduce the number of wires in the arm. This in turn requires a fast and deterministic bus communication for implementation of control algorithms.

6.1.1. Hardware overview

LWR-III has seven degrees of freedom, a load-to-weight ratio of approximately 1:1, a total system weight of less than 15 kilograms for arms with a work space of up to 1.5m, and a high dynamic performance [17].

The robot joints are controlled by the central computer via an optical fiber bus system using the standardized real-time SERCOS protocol. The desired and actual motor positions, the link torques and the link positions are transmitted once every millisecond [17]. The joint torque is applied at 3kHz rate, while the dynamics of the robot and Cartesian control are computed by the central computer at the rate of 1kHz.

The mechatronic joint design is indicated in Figure 6.2

6.1.2. Control aspects

Based on the hardware design, a suitable control scheme is developed to enable the robot operating close to human in unstructured environments. Compare to standard industrial robot control, the

Figure 6.1: The DLR lightweight robot [17]

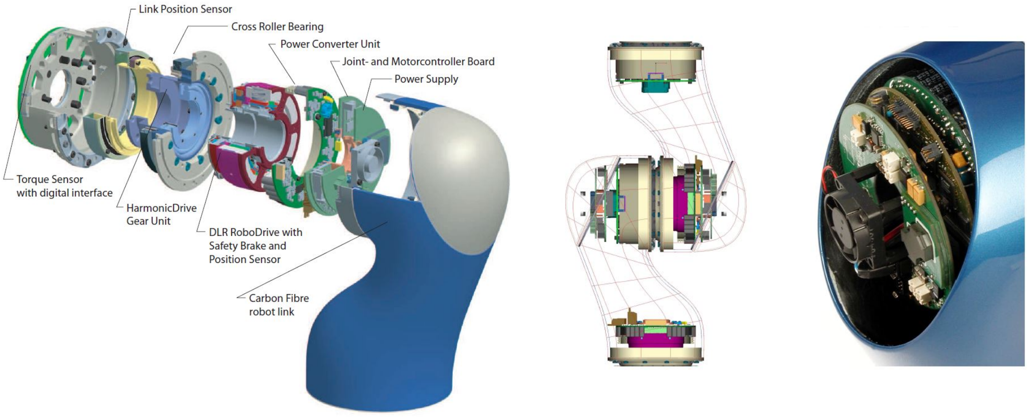

Link Position Sensor
Cross Roller Bearing
Power Converter Unit
Joint- and Motorcontroller Board
Power Supply
Torque Sensor with digital interface
HarmonicDrive Gear Unit
DLR RoboDrive with Safety Brake and Position Sensor
Carbon Fibre robot link

Figure 6.2: The mechatronic joint design of the LWR including actuation, electronics, and sensing [17]

following two aspects are of vital importance:

- Position control has to compensate the effects of the robot elasticity (such as vibrations or the steady state position displacement) to ensure the performance of positioning and trajectory tracking [17].
- The use of joint torque sensors for control.

6.2. Experimental steps

The robot is controlled by a pre-defined python command script on a 32-bit Linux computer.

1. **open_brakes()**
   This command opens the brake on each joint and prepared for the robot motion. It is mandatory to open the brakes before commanding any motion to the robot.
2. **robot()**
   This command connects the robot to the real-time machine to enable testing on the robot.
3. **zero_g()**
   This command activates the gravity and Coriolis force compensation. The compensation re-

sults in the robot floating in the space and easy to be moved by a human. Figure 6.3 shows the moving of the robot under gravity and Coriolis force compensation.

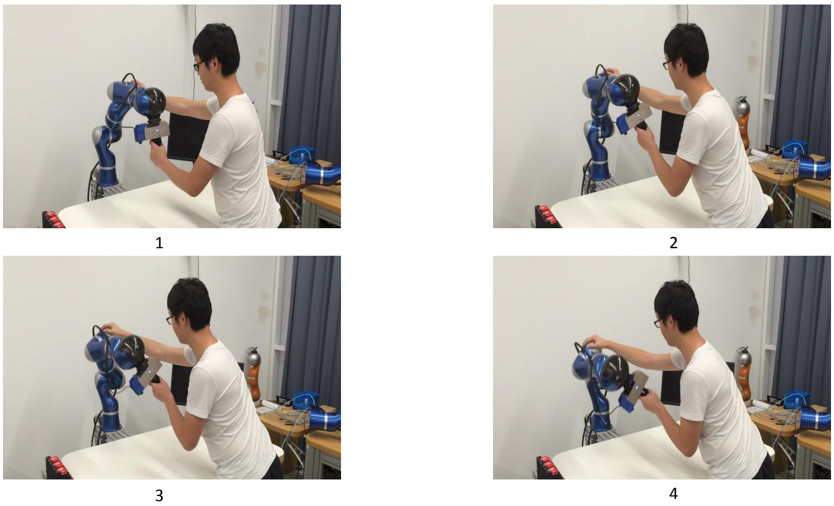

1
2
3
4

Figure 6.3: Moving the robot with gravity and Coriolis force compensation

4. **move_joint(**$[q_1, q_2, \dots, q_7]$**,**t**)**
   This command moves the robot to the desired joint positions which specified by the position vector $[q_1, q_2, \dots, q_7]$ in t second(s).

The collision avoidance scheme needs to be integrated into the RCU in Simulink. After the new RCU is made, the experiment steps are as follows:

1. Compile the Simulink to an executable RCU file for the VxWorks machine. [1]

2. Connect to the VxWorks machine.

3. Execute the RCU file.

4. Start the action bridge to connect the Linux computer with the VxWorks machine.

5. Open the RCU model in Simulink.

6. Connect to the Simulink model by selecting external mode of the model.

7. Control the robot by the pre-definite python script.

The motion of the robot is visualized by the Rvis package in Robot Operation System (ROS) as shown in Figure 6.4.

[1] VxWorks is a real-time operating system (RTOS) developed as proprietary software by Wind River of Alameda, California, US.[27]

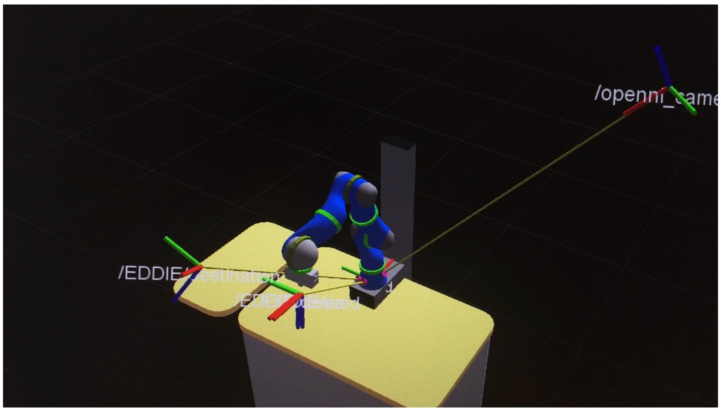


Figure 6.4: Visualization by Rvis packages

Note that there are three coordinate marks in the figure. The one on the far right is the camera position. It is always fixed. The one in the middle tracks the current end-effector position. The last one on the left is the commanded new position for the end-effector.

6.3. Real-time simluation

In real time simulation, for the ease of distance calculation, the obstacle is defined as a sphere centered at $(x_{\text{obs}}, y_{\text{obs}}, z_{\text{obs}})$ in Cartesian coordinate with radius r. Then the minimum distance of the end-effector to the object can be represented in

$$\text{mini\_dist} = \sqrt{(\boldsymbol{x}_{\text{End},x} - x_{\text{obs}})^2 + (\boldsymbol{x}_{\text{End},y} - y_{\text{obs}})^2 + (\boldsymbol{x}_{\text{End},z} - z_{\text{obs}})^2} - r \tag{6.1}$$

The motion of the robot is shown in Figure 6.5.

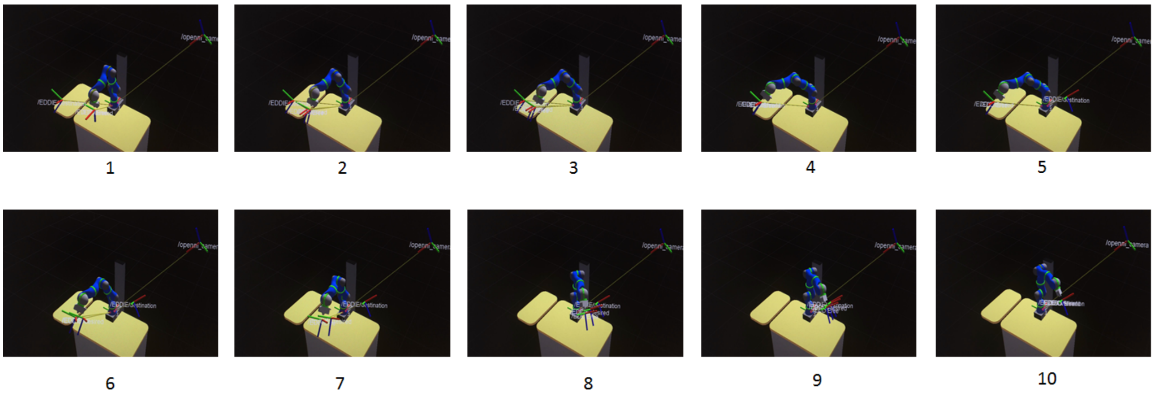


Figure 6.5: Real time simulation trajectory with the obtstacle

The obstacle is selected to be within the trajectory, the full end-effector trajectory is collected from the RCU model. Figure 6.6 shows the trajectory and the obstacle.

The simulation scenarios is indicated in the Table 6.1

Table 6.1: Simulation scenario

Braking controller	Simultaneous braking with Alg 3 for smoothing
Braking distance threshold ϵ	0.05 meter
Center of the obstacle	$[-0.4; 0; 0.7]$
Radius of the obstacle	0.1 meter

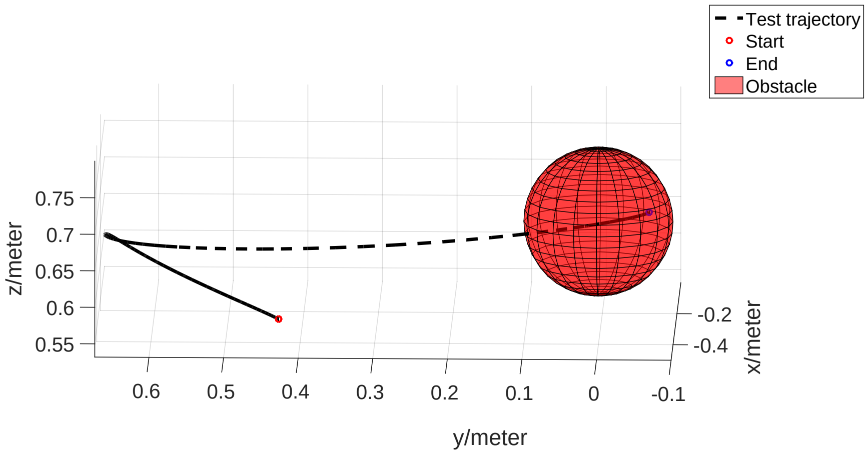


Figure 6.6: Real time simulation trajectory with the obtstacle

6.3.1. Constant inertia matrix

Under constant inertia matrix assumption, with collision avoidance by trajectory prediction, the resulting motion is shown in Figure 6.7

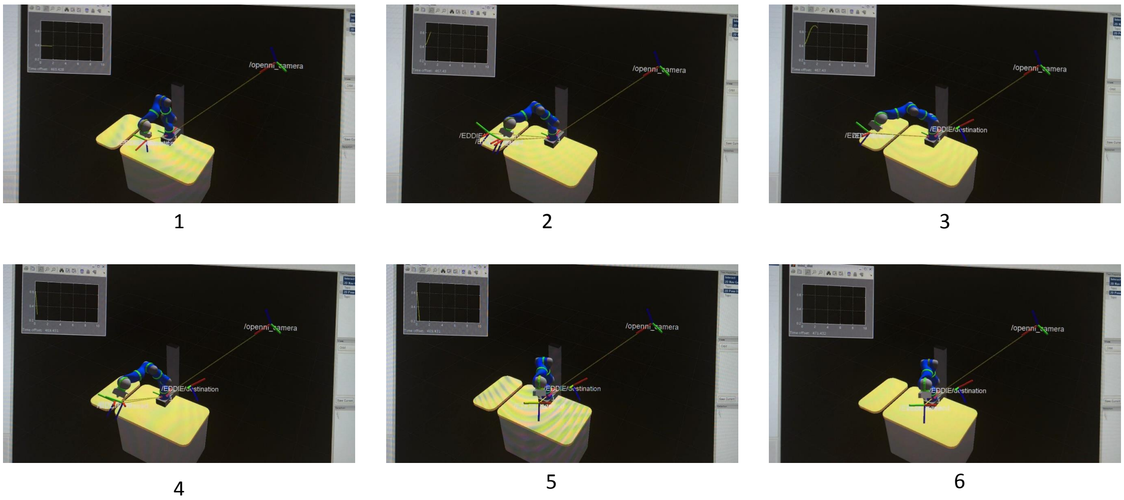


Figure 6.7: robot motion with collision avoidance (constant inertia matrix)

The scope at the up right of Figure 6.7 is the predicted minimum distance to the obstacle. It increases during the first motion of the robot, and then decreases. The robot stopped in the middle of the commanded trajectory. To verify if the robot stopped before the obstacle, the trajectory data can be obtained from the RCU model. Figure 6.8 shows the end-effector trajectory. In Figure 6.8, the end-effector stopped long before the obstacle. The distance to the obstacle is 0.1473 meter while the threshold is 0.05 meter. The mismatch is around 0.1 meter. The constant inertia matrix assumption is a simplification of the robot model thus inevitably introduces prediction error.

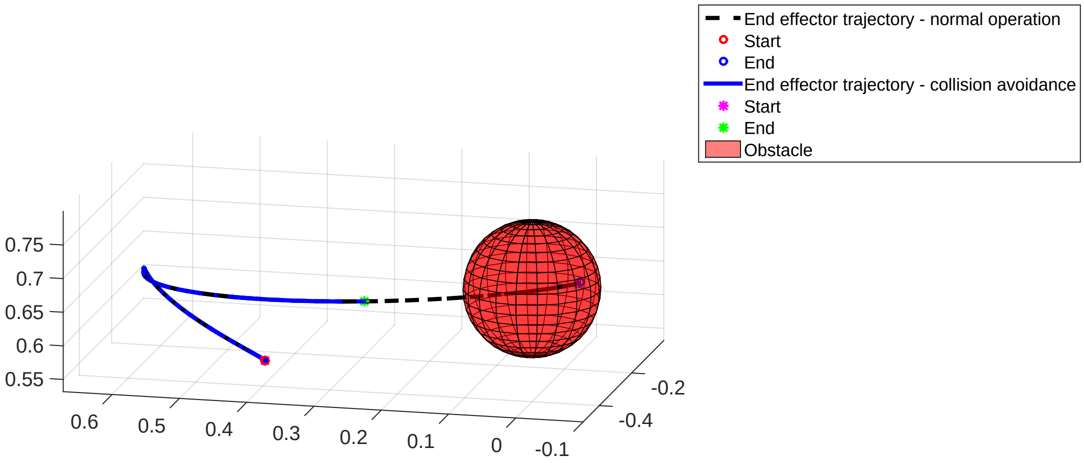


Figure 6.8: End-effector trajectory with collision avoidance (constant inertia matrix)

6.3.2. Varying inertia matrix

Using varying inertia matrix, with collision avoidance by trajectory prediction, the resulting motion is shown in Figure 6.9

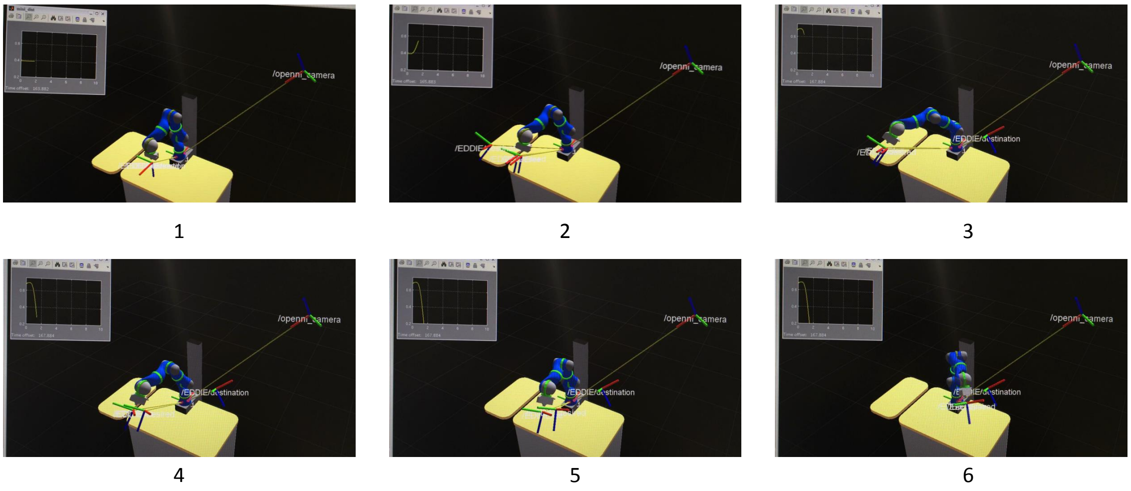


Figure 6.9: robot motion with collision avoidance (updated inertia matrix)

With predicted minimum distance to the obstacle, the robot stopped in the middle of the commanded trajectory. To verify if the robot stopped before the obstacle, the trajectory data can be obtained from the RCU model. Figure 6.10 shows the end-effector trajectory.

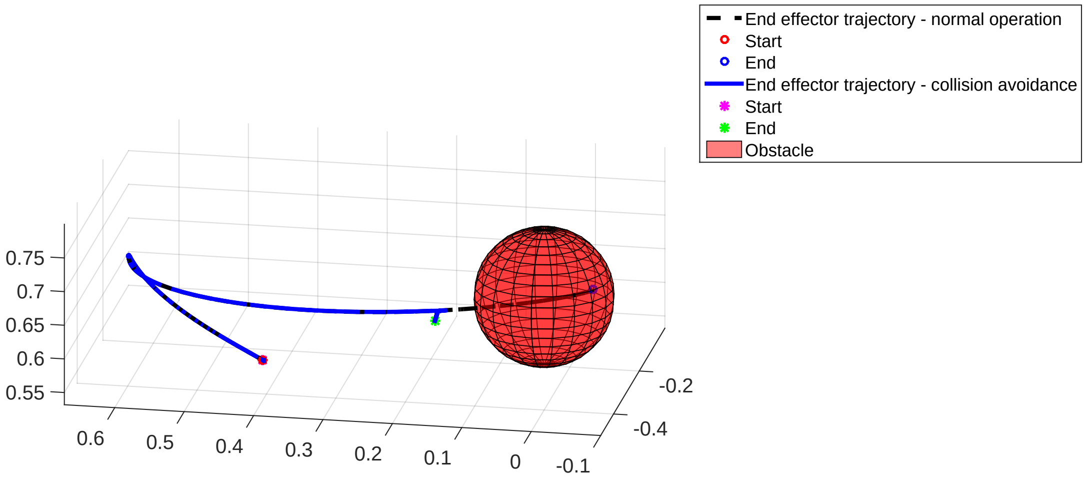


Figure 6.10: End-effector trajectory with collision avoidance (updated inertia matrix)

In Figure 6.10, the end-effector stopped before the obstacle. The distance to the obstacle is 0.0585 meter while the threshold is 0.05 meter. The mismatch is around 0.0085 meter. The updated inertia matrix gives much more accurate prediction in the real-time RCU test.

6.4. Robot testing

In the real robot testing, obstacle is represented by a plane located $y = -0.0013$. The simultaneous braking is tested on the LWR-III several times. The robot stopped but with oscillation in the final position. Therefore unfortunately, it has not been tested in the full collision avoidance scheme. Joint feedback control are used instead.

The testing scenario is showed below:

Table 6.2: Robot test scenario

Braking controller	Joint space feedback
Trajectory prediction	Constant inertia matrix
Braking distance threshold ϵ	0.05/0.1/0.15 meter
Obstacle plane	$y = -0.0013$

A common left to right motion is selected as the intended motion. The resulting motion is indicated in Figure 6.11.

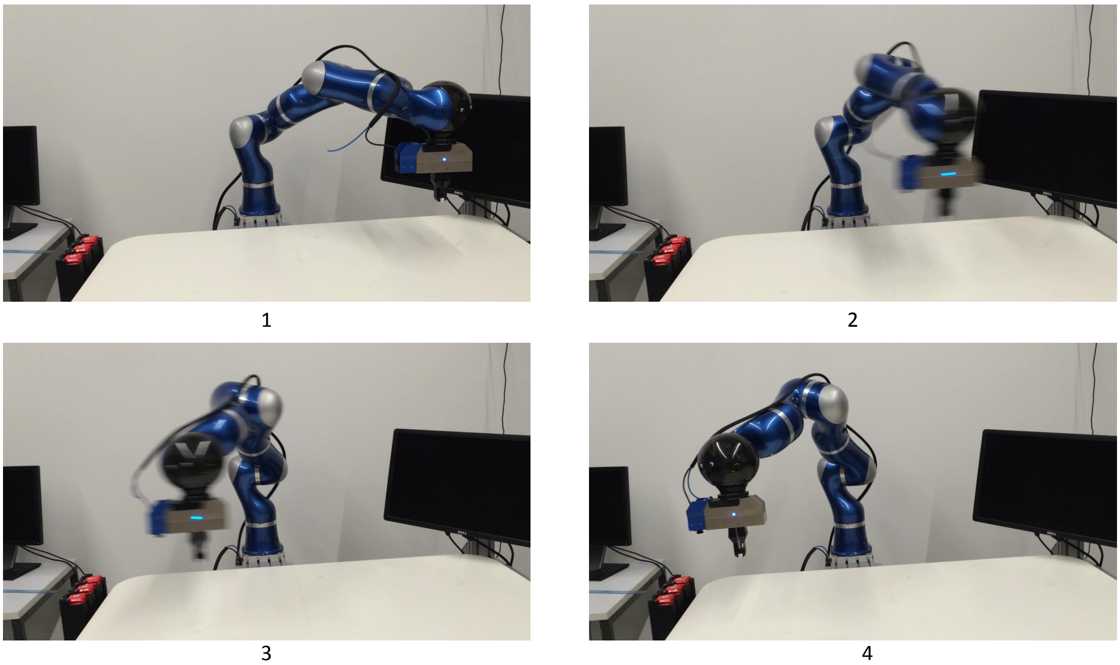
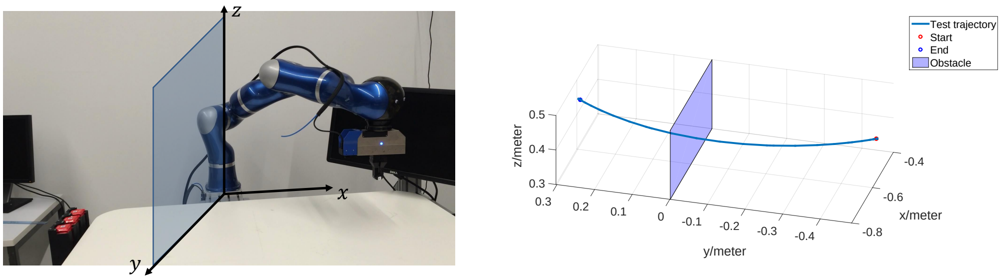


Figure 6.11: The robot in the left to right motion

The plane is place at $y = -0.0013$, which is depicted in Figure 6.12



(a) The robot with the obstacle plane

(b) End-effector trajectory with the obstacle plane

Figure 6.12: Obstacle plane

The resulting motion of robot under threshold $\epsilon = 0.05$ is depicted in Figure 6.13 [2].

[2] The collision avoidance is tested with $\epsilon = 0.05, 0.1, 0.15$. Here only motion under $\epsilon = 0.05$ is showed.

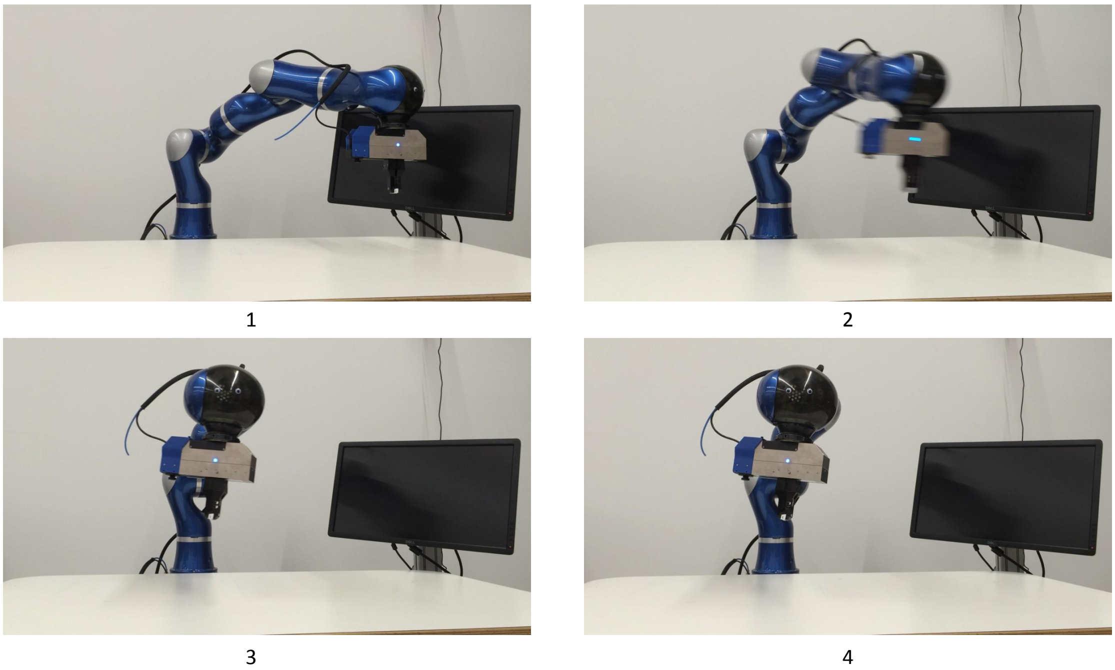


Figure 6.13: The robot motion with the collision aviodance, threshold $\epsilon = 0.05$ meter

With the collision avoidance, the robot stopped in the middle of the commanded trajectory. To verify if the robot stopped before the obstacle plane, the trajectory data can be obtained from the robot. Figure 6.14 shows the end-effector trajectory.

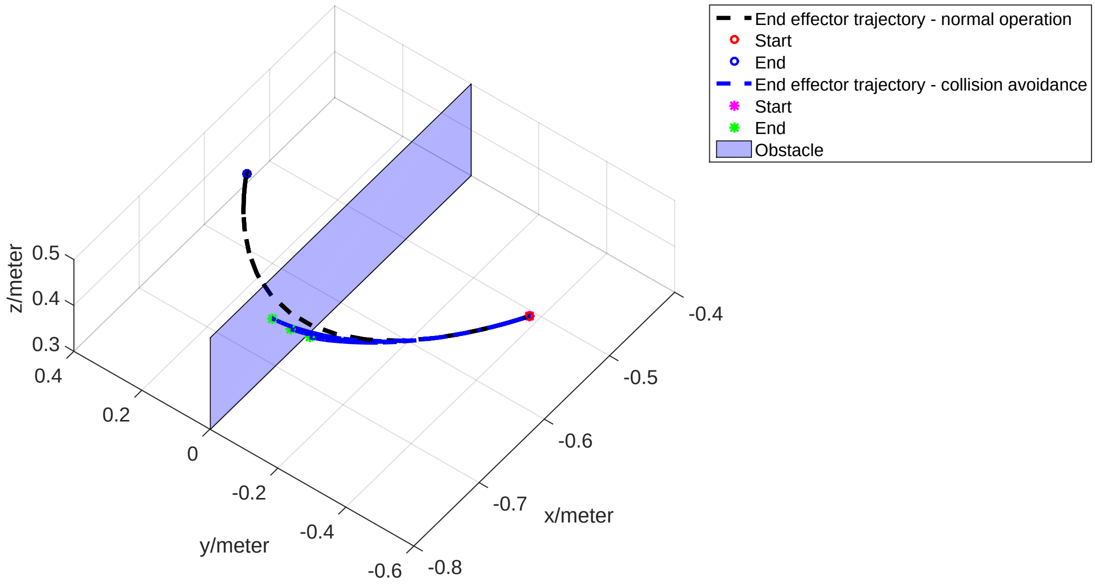


Figure 6.14: The end-effector trajectory with the collision aviodance

The three collision avoidance trajectories in Figure 6.14 correspond to $\epsilon = 0.05, 0.1, 0.15$. The table below shows the distance to the obstacle plane calculated using the data from the robot with different threshold settings.

Table 6.3: Distance to the obstacle plane with different threshold settings

Threshold/meter	Distance to the obstacle plane/meter
$\epsilon = 0.05$	0.0270
$\epsilon = 0.1$	0.0800
$\epsilon = 0.15$	0.1280

The experiment on the robot shows that the collision avoidance scheme has successfully initiated braking and avoided collisions with the obstacle plane. The prediction error is around 0.02 meter.

7

Summary and future works

The thesis proposed a novel collision avoidance scheme by trajectory prediction. The scheme is able to provide extra safety margin to the current state of the art of the robot safety control.

7.1. Summary

The collision avoidance scheme consists of two parts:

- Braking controller
- Braking trajectory prediction

7.1.1. Braking controller

There are many existing control laws to stop the robot. The results presented in [5] are reaction strategies after collisions, some of them requires information from the collision detection block. The braking proposed in [13] is an optimal braking strategy in terms of reducing the impact of collisions. In this thesis the controller needs to fit in the prediction framework to be integrated into the collision avoidance scheme. In this context, two controllers are present in Chapter 3.

The thesis presents first a simple feedback controller for stopping the robot. Then based on decoupled space analysis, a novel simultaneous braking controller is proposed in Section 3.3.2. It is a time optimal control laws derived based on decoupled space analysis considering the input limitations. The resulting controller is of bang-bang type. To solve the oscillation problem of the controller, two smoothing methods are proposed in Section 3.4.

7.1.2. Braking trajectory prediction

Currently, there is no literature on braking trajectory prediction of the robot. The thesis focuses on to proposing a framework that allows real-time implementation of the prediction.

The thesis approaches the trajectory prediction problem by proposing a framework in decoupled space in Chapter 4. Within this framework, the trajectory can be solved analytically. The framework is first used to solve the trajectory for a joint space feedback controller in Section 4.3.2, the analytical solution is derived and the corresponding algorithm is proposed. Then it is also applied on simultaneous braking in Section 4.3.3. The use of a varying inertia matrix to enhance accuracy in the prediction is proposed in Section 4.3.4.

Both methods are able to predict the trajectory. The enhanced prediction is much more accurate but also more computationally demanding.

7.1.3. Collision avoidance

The predicted trajectory is used to calculate the minimum distance to an object in the environment. The minimum distance is then used to determine when to initiate braking to avoid collisions with the object.

The collision avoidance scheme is integrated into the RCU for performance evaluation. In the real-time RCU simulation, the original and enhanced predictions are both used with simultaneous braking. Both successfully initiated braking to avoid collisions. But from the data, enhanced prediction shows much better performance in terms of prediction accuracy.

One collision avoidance scheme is successfully tested on the real robot, and is able to initiate braking to avoid collision with the obstacle.

7.2. Future work

Due to the limitation of time, here some recommendations for further research on the topic:

- *The cause of the oscillation problem*
  In the real-time RCU simulation, the simultaneous braking with the proposed smooth methods is able to eliminate oscillation. But on the real robot, the oscillation still exists. To investigate into the cause of the problem, one can first consider only single and two joints cases instead of seven joints in the robot testing.

- *Further evaluation of the trajectory prediction*
  The prediction is tested on some nominal trajectory, and the position data is selected all from one single motion. The accuracy can be further validated by considering a few more trajectories which cover most of the operational space of the robot.

- *Test the scheme along with the motion planning techniques*
  The scheme is able to provide advantages by avoid collision when trajectory planning fails. Then it is crucial to test it along with the planning and further validate the statement of augmenting the original safety.

Bibliography

[1] http://spectrum.ieee.org/img/yumi_295-1429043944635.jpg.

[2] Robots and robotic devices - safety requirements for industrial robots - part 1: Robots, 2011.

[3] http://zhidao.baidu.com/question/28034427.html, 2012.

[4] Alin Albu-Schäffer, Christian Ott, and Gerd Hirzinger. A passivity based cartesian impedance controller for flexible joint robots-part ii: Full state feedback, impedance design and experiments. In *Proceedings of 2004 IEEE International Conference on Robotics and Automation.*, volume 3, pages 2666–2672. IEEE, 2004.

[5] Alessandro De Luca, Alin Albu-Schäffer, Sami Haddadin, and Gerd Hirzinger. Collision detection and safe reaction with the DLR-III lightweight manipulator arm. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1623–1630. IEEE, 2006.

[6] Peter Dockrill. Volkswagen worker grabbed and killed by robot in german plant. http://www.sciencealert.com/volkswagen-worker-grabbed-and-killed-by-robot-in-german-plant, July 2015.

[7] Elmer G Gilbert, Daniel W Johnson, and S Sathiya Keerthi. A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal of Robotics and Automation*, 4(2):193–203, 1988.

[8] James H Graham. *Safety, reliability and human factors in robotic systems*. John Wiley & Sons, Inc., 1991.

[9] Markus Grebenstein, Alin Albu-Schäffer, Thomas Bahls, Maxime Chalon, Oliver Eiberger, Werner Friedl, Robin Gruber, Sami Haddadin, Ulrich Hagn, Robert Haslinger, et al. The DLR hand arm system. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3175–3182. IEEE, 2011.

[10] Erico Guizzo. Robots with knives: A study of soft-tissue injury in robotics. *IEEE Spectrum*, May 2010.

[11] Sami Haddadin, A Albu-Schäffer, Alessandro De Luca, and Gerd Hirzinger. Collision detection and reaction: A contribution to safe physical human-robot interaction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. IROS 2008.*, pages 3356–3363. IEEE, 2008.

[12] Business Intelligence. The robotics market report: The fast-multiplying opportunities in consumer, industrial, and office robots, May 2015.

[13] Seong-Hee Jeong and Takayuki Takahashi. Optimal braking for impact force reduction using the dynamics of redundant manipulators. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 1898–1903. IEEE, 2006.

[14] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–98, 1986.

[15] Adam Eric Leeper, Kaijen Hsiao, Matei Ciocarlie, Leila Takayama, and David Gossow. Strategies for human-in-the-loop robotic grasping. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, pages 1–8. ACM, 2012.

[16] Howard Li, Simon X Yang, and Mae L Seto. Neural-network-based path planning for a multi-robot system with moving obstacles. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 39(4):410–419, 2009.

[17] Clive Loughlin, A Albu-Schäffer, S Haddadin, Ch Ott, A Stemmer, T Wimböck, and G Hirzinger. The DLR lightweight robot: design and control concepts for robots in human environments. *Industrial Robot: an international journal*, 34(5):376–385, 2007.

[18] Alessandro De Luca. A control architecture for human robot collaboration. 2015. URL `http://www.diag.uniroma1.it/~deluca/ICRA15_Keynote_ADL.pdf`.

[19] Nico Mansfeld. Principle performance requirements, safety analysis, and control for elastic joint robot. Master's thesis, TU Munich, Sept 2012.

[20] Nico Mansfeld and Sami Haddadin. Reaching desired states time-optimally from equilibrium and vice versa for visco-elastic joint robots with limited elastic deflection. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*, pages 3904–3911. IEEE, 2014.

[21] Florian Petit and A Albu-Schäffer. State feedback damping control for a multi dof variable stiffness robot arm. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5561–5567. IEEE, 2011.

[22] Bruno Siciliano and Oussama Khatib. *Springer handbook of robotics*. Springer Science & Business Media, 2008.

[23] Gilbert Strang. Symmetric matrices and positive definiteness. http://ocw.mit.edu/courses/mathematics/18-06sc-linear-algebra-fall-2011/positive-definite-matrices-and-applications/symmetric-matrices-and-positive-definiteness/MIT18_06SCF11_Ses3.1sum.pdf, 2011.

[24] Holger Täubig and Udo Frese. A new library for real-time continuous collision detection. In *Proceedings of ROBOTIK 2012 7th German Conference on Robotics*, pages 1–5. VDE, 2012.

[25] Toshio Tsuji, Yoshiyuki Tanaka, Pietro G Morasso, Vittorio Sanguineti, and Makoto Kaneko. Biomimetic trajectory generation of robots via artificial potential field with time base generator. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 32(4): 426–439, 2002.

[26] Richard Volpe and Pradeep Khosla. Manipulator control with superquadric artificial potential functions: Theory and experiments. *IEEE Transactions on Systems, Man and Cybernetics*, 20(6): 1423–1436, 1990.

[27] Wikipedia. Vxworks. https://en.wikipedia.org/wiki/VxWorks.

[28] Yunong Zhang and Jun Wang. Obstacle avoidance of redundant manipulators using a dual neural network. In *Proceedings of the IEEE International Conference on Robotics and Automation, 2003. ICRA'03*, volume 2, pages 2747–2752. IEEE, 2003.

[29] Yunong Zhang, Zhonghua Li, and Hong-Zhou Tan. Inequality-based manipulator-obstacle avoidance using the LVI-based primal-dual neural network. In *IEEE International Conference on Robotics and Biomimetics, 2006. ROBIO'06.*, pages 1459–1464. IEEE, 2006.

A

Glossary

ICRA	International Conference on Robotics and Automation
APF	Artificial Potential Field
pHRI	physical Human-Robot Interaction
LWR-III	DLR Lightweight Robot III
DLR	Deutsches Zentrum fur Luft- und Raufahrt
Hasy	Hand arm system
RCU	Robot Control Unit
ROS	Robot Operation System
MIMO	Multiple-Input Multiple-Output
SISO	Single-Input Single-Output