


# Dynamic Modeling and Simulation of a Multi-fingered Robot Hand

by

Joseph Chun-kong Chan



A dissertation submitted in partial  
fulfillment of the requirements for the degree of  
Master of Philosophy

in the

Division of Mechanical and Automation Engineering


of

The Chinese University of Hong Kong  
Shatin, N.T.,  
Hong Kong, China

August 1998

Dissertation committee in charge:  
Professor Yun-hui Liu, Chair  
Professor Yang-sheng Xu  
Professor Kin-chuen Hui





**Dynamic Modeling and Simulation of a Multi-fingered  
Robot Hand**

© Copyright

by

The Chinese University of Hong Kong  
August 1998

# Abstract

## Dynamic Modeling and Simulation of a Multi-fingered Robot Hand

by

Joseph Chun-kong Chan

A dynamic simulation system for dextrous manipulations can facilitate the applications of multi-fingered robot hands. For examples, it can assist the *programming by demonstration* system to transfer human manipulation skills to robot hand. It can also be used to evaluate different control algorithms before they are applied to real systems. Dextrous manipulation using multi-fingered robot hands involves various and rapid changes in contact constraints and grasping configurations. If different situations are handled case by case in simulation, these will lead to the problem of combinatorial explosion. There is no existing dynamic simulator that can sophisticatedly and efficiently simulate dextrous manipulation using multi-fingered robot hands.

In this dissertation, we derive the kinematics and dynamics of a multi-fingered robot hand manipulating an object, and develop a dynamic simulation system for dextrous manipulations based on them. The dynamic modeling problem is divided into two parts: *contact modeling* and *collision modeling*. Firstly, a unified method is developed to formulate the dynamics under different motions including free motion, sticking contact, rolling, and sliding. Transitions between motions are handled by a new transition model. Secondly, based on the Mirtich's impulse-based simulation paradigm, an improved collision model is developed to determine the instantaneous change of velocity caused by collisions. This improved method can simulate the multiple collisions and mixed contact-collision cases. By combining these methods, we can effectively cope with rapid changes in grasping configurations and kinematic constraints. The validity of the dynamic modeling is verified by the development of a dynamic simulation system for the five-fingered robot hand system developed at this department. It is the first kind of dynamic simulation system for the multi-fingered robot hand which is capable of sophisticate simulation of various situations occurred in robot hand systems. The development of the system provides human operator an interactive way to simulate the dextrous manipulation. The simulation results confirm the validity of the dynamic models, and the accuracy and the efficiency of the developed dynamic simulator.



# 摘要

## 多指靈巧手的動態建模與計算機模擬

陳鎮光

多指靈巧手的計算機動態模擬系統能促進其實際應用。例如它可以輔助「範例編程系統」把人類的操縱技術轉移到多指靈巧手上；對於一些新設計而未曾應用的控制方案，它可幫助評價其性能。靈巧操縱的動作包含多種及經常改變的接觸點約束條件和抓握形態，如每種情況都獨立處理，將會引致組合性之爆炸的問題。現存的動態模擬器並沒有一個能有效地模擬多指靈巧手的靈巧操縱動作。

在這篇論文中，我們推算出多指靈巧手操縱模型的靜態和動態方程，並在此基礎上開發了一套多指靈巧手的動態模擬系統。動態模型的建立分兩個部份：接觸點接觸模型和碰撞接觸模型。首先，我們發展了一種統一方法來建立在不同接觸運動如自由運動、黏性點接觸運動、滾動和滑動下的動態模型。不同接觸運動的轉換是透過一個新的接觸點變遷模型來描述。另一方面，我們改良了莫迪（Mirtich）的衝擊模擬方法，發展了一個新的碰撞模型去計算因碰撞而引致的速度突變。這一模型能夠模擬多點碰撞以及混合觸點接觸和碰撞的情況。綜合以上的方法，我們能有效地處理那些涉及不同的接觸點約束條件和抓握形態的操作。根據我系所有的五指靈巧手，我們研發了一套多指靈巧手操縱的動態模擬系統。這是第一套能模擬不同的靈巧操縱動作，並且能提供操控員一交互式的方法來進行靈巧操縱模擬的系統。透過模擬系統的模擬結果，我們証明了所建立的動態模型的正確性和肯定了模擬系統的效率 and 精確度。

My Lord  
and  
my family

# Acknowledgments

For the two years life in the Department of Mechanical and Automation Engineering, lots of memories and faces in my mind that share my joy and support me to face the challenges and the darkness.

Prof. Yun-hui Liu, my thesis supervisor, is the one who walks most closely with me. His guidance, care, encouragement, and patience allow me to grow in a tough but rewardful research life. Prof. Yeung Yam, Kin-chuen Hui, Yang-sheng Xu, and Ronald Chung give me their precious advice and supports no matter in the pre-graduate, graduate, and the post-graduate studies.

I would like to thank all the friends and the staffs in this department. They make me feel living in a family. Mr Wai-keung Fung, Pak-chio Lam, Mun-chung Ma, Andrew Arengo, Patrick Ho, Stephen Yang, Wai-leung Yeung, Winston Sun, and Kai-hung Tang always accompany me to have lunch and tea break. More importantly, through numerous discussions with them, their thoughts and working attitudes have sharpen my mind and made me to be serious in doing research. For all other technical staffs, computing staffs and administrative staffs, thank for their helps whenever and wherever I get trouble. I have learnt a lot from their maturity and experience through the daily chatting.

I would like to thank Jack Ng, Grace Hong, Josephine Chan, Joyce Chan, Karen Wun, Anita Lam, and other brothers and sisters in the Post-Graduate Ministry. They share, witness the good news, and study God's words together with me. Also, I would like to thank all the staffs from the Campus Crusade for Christ. They set the examples of living by faith in which I can learnt. Football Fu, Ron Chow, Samuel Kwok, Sandy Tsi, Yonne Lau, Nicky Lee, Benny Wong, Jackyle Law, Matthew Chu, and Tony Ho are those who remind me not to store treasure on the ground but in the heaven. From their tears and prayers, they are the one who know my weakness but still encourage me in the love of the Lord.

I wish to thank Suk-che Ho who is the gift from God. Thank for walking with me

side by side, listening my complaints, accepting who I am, and allowing me to learn how to take care of the others.

I would like to pay my sincere gratitude to my sister, Shui-han, my brother, Chun-hay, and my parents. Over six years time spent in CUHK, their love, support, and patience encourage me to try, to face difficulties in failure, and to explore the outside world. What they have done and what they have taught always in my heart that always motivate me to stand up again and again.

Finally, I would like to thank my God. He saves me and sends his witnesses around me. He guides me and teaches me what the truth, love and faith are. He promises me to seek first his kingdom and his righteousness, and then provides me whatever I need, and he really keeps his promises.



# Contents

<b>Abstract</b>	<b>i</b>
<b>Acknowledgments</b>	<b>iv</b>
<b>List of Figures</b>	<b>xi</b>
<b>List of Tables</b>	<b>xii</b>
<b>List of Algorithms</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Related Work . . . . .	5
1.3 Contributions . . . . .	7
1.4 Organization of the Thesis . . . . .	9
<b>2 Contact Modeling: Kinematics</b>	<b>11</b>
2.1 Introduction . . . . .	11
2.2 Contact Kinematics between Two Rigid Bodies . . . . .	14
2.2.1 Contact Modes . . . . .	14
2.2.2 Montana’s Contact Equations . . . . .	15
2.3 Finger Kinematics . . . . .	18
2.3.1 Finger Forward Kinematics . . . . .	19
2.3.2 Finger Jacobian . . . . .	21
2.4 Grasp Kinematics between a Finger and an Object . . . . .	21
2.4.1 Velocity Transformation between Different Coordinate Frames . . . . .	22
2.4.2 Grasp Kinematics for the $i$ th Contact . . . . .	23
2.4.3 Different Fingertip Models and Different Contact Modes . . . . .	25
2.5 Velocity Constraints of the Entire System . . . . .	28

2.6	Summary . . . . .	29
<b>3</b>	<b>Contact Modeling: Dynamics</b>	<b>31</b>
3.1	Introduction . . . . .	31
3.2	Multi-fingered Robot Hand Dynamics . . . . .	33
3.3	Object Dynamics . . . . .	35
3.4	Constrained System Dynamics . . . . .	37
3.5	Summary . . . . .	39
<b>4</b>	<b>Collision Modeling</b>	<b>40</b>
4.1	Introduction . . . . .	40
4.2	Assumptions of Collision . . . . .	42
4.3	Collision Point Velocities . . . . .	43
4.3.1	Collision Point Velocity of the $i$ th Finger . . . . .	43
4.3.2	Collision Point Velocity of the Object . . . . .	46
4.3.3	Relative Collision Point Velocity . . . . .	47
4.4	Equations of Collision . . . . .	47
4.4.1	Sliding Mode Collision . . . . .	48
4.4.2	Sticking Mode Collision . . . . .	49
4.5	Summary . . . . .	51
<b>5</b>	<b>Dynamic Simulation</b>	<b>53</b>
5.1	Introduction . . . . .	53
5.2	Architecture of the Dynamic Simulation System . . . . .	54
5.2.1	Input Devices . . . . .	54
5.2.2	Dynamic Simulator . . . . .	58
5.2.3	Virtual Environment . . . . .	60
5.3	Methodologies and Program Flow of the Dynamic Simulator . . . . .	60
5.3.1	Interference Detection . . . . .	61
5.3.2	Constraint-based Simulation . . . . .	63
5.3.3	Impulse-based Simulation . . . . .	66
5.4	Summary . . . . .	69
<b>6</b>	<b>Simulation Results</b>	<b>71</b>
6.1	Introduction . . . . .	71
6.2	Change of Grasping Configurations . . . . .	71
6.3	Rolling Contact . . . . .	76



6.4	Sliding Contact . . . . .	76
6.5	Collisions . . . . .	85
6.6	Dextrous Manipulation Motions . . . . .	93
6.7	Summary . . . . .	94
<b>7</b>	<b>Conclusions</b>	<b>99</b>
7.1	Summary of Contributions . . . . .	99
7.2	Future Work . . . . .	100
7.2.1	Improvement of Current System . . . . .	100
7.2.2	Applications . . . . .	101
<b>A</b>	<b>Montana's Contact Equations for Finger-object Contact</b>	<b>103</b>
A.1	Local Coordinates Charts . . . . .	103
A.2	Curvature, Torsion and Metric Tensors . . . . .	104
A.3	Montana's Contact Equations . . . . .	106
<b>B</b>	<b>Finger Dynamics</b>	<b>108</b>
B.1	Forward Kinematics of a Robot Finger . . . . .	108
B.1.1	Link-coordinate Transformation . . . . .	109
B.1.2	Forward Kinematics . . . . .	109
B.2	Dynamic Equation of a Robot Finger . . . . .	110
B.2.1	Kinetic and Potential Energy . . . . .	110
B.2.2	Lagrange's Equation . . . . .	111
<b>C</b>	<b>Simulation Configurations</b>	<b>113</b>
C.1	Geometric models . . . . .	113
C.2	Physical Parameters . . . . .	113
C.3	Simulation Parameters . . . . .	116
	<b>Bibliography</b>	<b>124</b>

# List of Figures

1.1	Error arisen in peg-in-hole problem when using traditional end-effector. . . . .	2
1.2	A five-fingered robot hand system developed at the CUHK. . . . .	4
2.1	Inter-penetrations between the virtual fingers of the multi-fingered robot hand and the virtual object in computer simulation. . . . .	12
2.2	Relative contact velocity defined in two contacting rigid bodies. . . . .	16
2.3	Coordinate frames assignment for the Montana's contact equations. . . . .	17
2.4	Input-output relationship of the Montana's contact equations. . . . .	18
2.5	Coordinate frame assignment of the modified Denavit-Hartenberg method. . . . .	20
2.6	Velocities transformation in different coordinate frames. . . . .	23
2.7	Coordinate frames assignment for the contacting $i$ th finger and object. . . . .	24
4.1	Change of the normal component of collision impulse $P$ in the compression phase and restitution phase of a collision. . . . .	43
4.2	Coordinate frame assignment for the colliding $i$ th finger and the object. . . . .	44
5.1	Interactive control of the dynamic simulation system. . . . .	54
5.2	Architecture of the dynamic simulation system for the five-fingered robot hand. . . . .	55
5.3	<i>CyberGlove<sup>TM</sup></i> used in the dynamic simulator as the input device. . . . .	55
5.4	Screen shot of the graphical control panel of the dynamic simulator. . . . .	59
5.5	The "growth" rectangular object and the object in hemisphere shape used in the interference detection of the five-fingered robot hand dynamic simulation system. . . . .	62
5.6	Contact transition model used in the constraint-based simulation. . . . .	64
6.1	Example 1: Change in grasping configuration from a four-fingered grasp to a three-fingered grasp. . . . .	73

6.2	Example 2: Change in grasping configuration from a five-fingered grasp to a three-fingered grasp. . . . .	74
6.3	Change of joint angles of the 3rd and the 4th finger in the process of changing grasping configuration from the five-fingered grasp to the three-fingered grasp. . . . .	75
6.4	Change of object configuration in the process of changing grasping configuration from the five-fingered grasp to the three-fingered grasp. . . . .	75
6.5	Example 3: Rolling between a finger and an object. . . . .	77
6.6	Change of joint angles of the finger in the process of rolling. . . . .	78
6.7	Change of object configuration in the process of rolling. . . . .	78
6.8	Change of constraint force $f_c$ arisen in the process of rolling. . . . .	79
6.9	Change of local contact parameters $[u_{li} \ u_{oi} \ \psi]^T$ in the process of rolling. . . . .	80
6.10	Example 4: Sliding between a finger and an object. . . . .	81
6.11	Change of joint angles of the finger in the process of sliding. . . . .	82
6.12	Change of object configuration in the process of sliding. . . . .	82
6.13	Change of constraint force $f_c$ arisen in the process of sliding. . . . .	83
6.14	Change of local contact parameters $[u_{li} \ u_{oi} \ \psi]^T$ in the process of sliding. . . . .	84
6.15	Example 5: Collision between a finger and an object with mass $m_o = 0.01kg$ . . . . .	86
6.16	Example 6: Collision between a finger and an object with mass $m_o = 0.1kg$ . . . . .	87
6.17	Example 7: Collision between a finger and an object with mass $m_o = 0.5kg$ . . . . .	88
6.18	Change of joint velocities of the finger in the process of collision with different object mass. . . . .	89
6.19	Change of object translational velocity in the process of collision with different object mass. . . . .	90
6.20	Change of object differential ZYZ Euler angles in the process of collision with different object mass. . . . .	91
6.21	Example 8: Dextrous manipulation with mixed contacts and collisions. . . . .	92
6.22	Change of joint velocities of the finger in the process of mixed contacts and collisions. . . . .	93
6.23	Example 9: Simple dextrous manipulation. . . . .	95
6.24	Change of object configuration in the process of a simple dextrous manipulation shown in Example 9. . . . .	96
6.25	Example 10: Simple dextrous manipulation. . . . .	97
6.26	Change of object configuration in the process of a simple dextrous manipulation shown in Example 10. . . . .	98

B.1	Coordinate frames assignment of a finger of the five-fingered robot hand.	108
C.1	Specification of the geometric model of a finger. . . . .	114
C.2	Three dimensional solid model of the five-fingered robot hand constructed by AutoCAD. . . . .	115

# List of Tables

2.1	Finger parameters. . . . .	20
5.1	14 sensory data used in the simulation system. . . . .	57
5.2	Mapping between data glove and five-fingered robot hand. . . . .	58
6.1	Conditions used in the simulation results demonstration. . . . .	72



# List of Algorithms

5.1	Program flow of the dynamic simulator. . . . .	61
5.2	Interference detection of the dynamic simulator. . . . .	63
5.3	Constraint-based simulation of the dynamic simulator. . . . .	63
5.4	Impulse-based simulation of the dynamic simulator. . . . .	66
5.5	Sliding mode collision integration. . . . .	67
5.6	Sticking mode collision integration. . . . .	68



# Chapter 1

## Introduction

### 1.1 Motivation

Robotics is a multi-disciplinary subject involving mechanical engineering, electronic and electrical engineering, computer science, and mathematics. This subject has been studied over several decades. Nowadays, robots are not only examined in laboratories but also have wide applications in industry. John Craig in his robotics textbook (Craig, 1989, page 1) claims that robotic technologies are leading industrial automation through another transition and the scope of which is still unknown.

For a typical industrial robotic system, it consists of a controller, sensing devices, a manipulator and an end-effector. Controller is the cognition device of the system. By using feed-forward and feedback control, it can execute commands issued by operator with the errors within a certain tolerance. For an autonomous system, additional task planner and motion planner are added on top of the controller to execute high-level commands. They work as the brain of the system to make decision on what jobs must be done and how tasks can be accomplished. Sensing device is the perception module of the system, collecting relevant and essential information necessary for the controller. Finally, physical operations such as part insertion and arc welding are executed by manipulator and end-effector which distinguish a robotic system from a solely intelligent system.

When we mention end-effector, people may think it may be a gripper, pincers, or tongs, which are the common end-effectors that we can observe in industry or in movie. These specific tools work well in a pre-defined or known environment. However, when we want to extend the application of robotics to more autonomous or general tasks, there are several limitations of these traditional end-effectors (Murray et al., 1994; Lee

et al., 1994):

- **Limited number of possible grasp:** For a special designed end-effector, it may be good to do some specific jobs but unable to do another. For example, a gripper can hold a rectangular object firmly but not a prismatic object. In this case, we may need different end-effectors to grip different shape of objects.
- **Lack of dexterity:** A gripper holds a rectangular object securely but the secure grasping reduces the mobility of the object. In other words, it is unable to manipulate the gripped object.
- **Inefficiency in motion:** In many cases, large motion of manipulator is required for even small movement of end-effector. For example, if we are required to insert a part into a hole but a small error is occurred during the operation (shown in Figure 1.1), we may need to repeat the job once again in order to complete it (Muñoz et al., 1995). To completely repeat the motion is ineffective to correct the small error accumulated in operation.

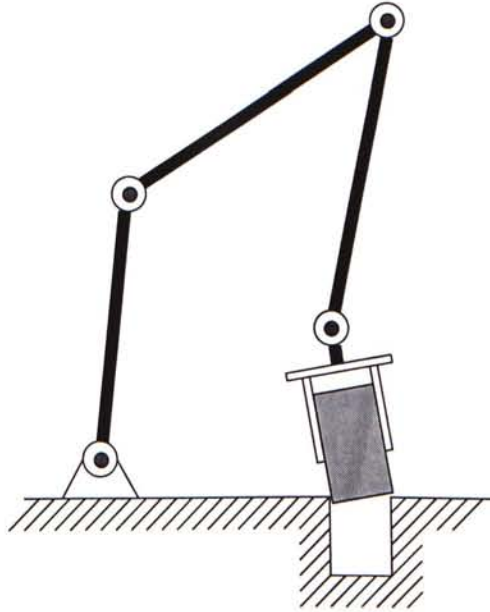


Figure 1.1: Error arisen in peg-in-hole problem when using traditional end-effector.

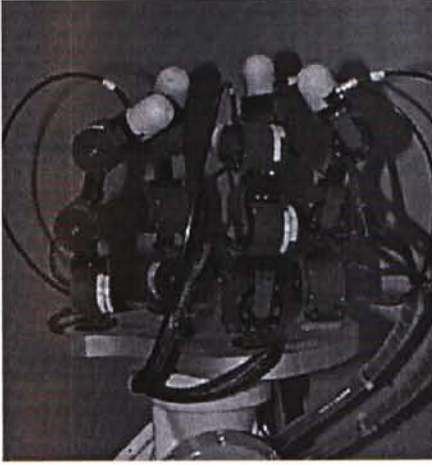
To overcome these drawbacks, people began to design more sophisticated end-effectors and the development of dextrous multi-fingered robot hands has been extensively studied since 1980's.



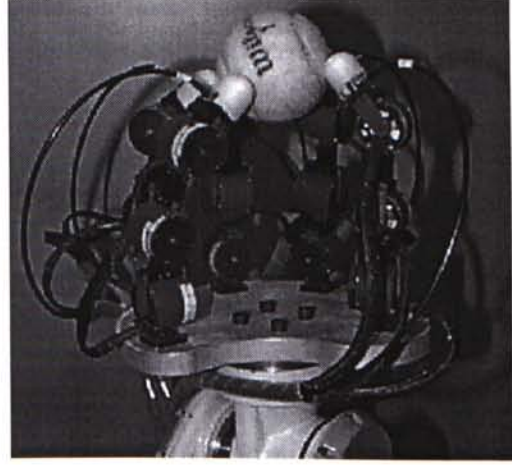
High dexterity and superior ability of fine manipulation offer multi-fingered robot hand extensive potential applications such as in precision industries and robotic aided surgery. Most of the research work are concentrated in mechanical design (Jacobsen et al., 1986; Ali et al., 1993; Umetsu and Oniki, 1993), kinematics (Salisbury and Roth, 1983; Montana, 1995), dynamics and control (Cole et al., 1989, 1992), and stable grasping (Li et al., 1989; Shimoga, 1996). The maturity of the development of these low-level components facilitates some simple applications in the real world. In (Nelson et al., 1995), the Utah/MIT multi-fingered robot is capable of performing assembly work and of screwing a light bulb. However, planning of dextrous manipulation is still one of the major obstacles in more advanced applications.

Manipulation of multi-fingered robot hands is subject to various geometric and mechanical constraints. For example, contact between finger and the part being manipulated is a kind of unilateral constraint. This constraint prevents inter-penetration of physical objects. Rolling contact occurred during dextrous manipulation produces the nonholonomic constraint. It is a kind of velocity constraint, which cannot be integrated into position constraint. Planning of a system with nonholonomic constraints is still the active research area in these years (Li and Canny, 1993). Furthermore, high degree of freedom makes the planning problem more complicated. For the five-fingered robot hand system developed at the Chinese University of Hong Kong (CUHK) shown in Figure 1.2, there are three joints for each finger. If we consider the planning problem of this multi-fingered robot hand, fifteen degrees of freedom plus six degrees of freedom of the manipulated object yield a high degree of freedom problem which is very difficult to solve.

In the research proposal of Liu (1995), the *Dextrous Manipulation Teaching System* DMTS has been proposed to do the planning of a multi-fingered robot hand. The main idea is to develop a human friendly teaching interface to teach the human dextrous manipulation skill to the robot hand system. While the operator is performing the task, the system observes the grasping and the manipulating motions, and segment them into a set of primitive control motions. In such a way, the human operator action acts as the guide for the real multi-fingered robot hand execution, which greatly reduces the planning complexity. This *robot programming by demonstration* method has been developed in assembly task and grasping (Kang, 1994). However, it is difficult to apply these algorithms in teaching dextrous manipulation. Teaching dextrous manipulation requires depth understanding of mechanical interactions between the fingers and the manipulated object such as prolonged contact and impulsive collision. The previous developed systems only consider the geometric understanding of human motion. More-



(a) There are 5 fingers and each finger has 3 joints. All joints are actuated by direct drive motors and 6 dimensional force sensors are equipped at all fingertips.



(b) A snapshot of the hand manipulating a tennis ball.

Figure 1.2: A five-fingered robot hand system developed at the CUHK.

over, the DMTS is capable of learning a better dextrous motion taught by different operators or the same operator at different times based on some performance evaluation modules in the system. Capability of learning provides a superior feature of this DMTS over existing ones.

In the master thesis of Lam (1998), a pure kinematic simulator has been developed. It computes the motions between the human fingers and the manipulated object from the manipulation demonstrated by an operator. The simulated motion determined by this method can be ensured to be satisfied by all the kinematic constraints. This simulator is good to provide us a basic understanding between the interactions of dextrous manipulation and kinematic constraints. However, the dynamics of the hand and the object are not considered in this system. In other words, we cannot know the dynamic behavior of the system and this is the major limitation of a kinematic simulator. In order to teach dextrous manipulation using the DMTS interface, an accurate dynamic simulator is the core and the indispensable module. Consideration of full kinematic and dynamic models, and the contact dynamics such as sticking, rolling, and sliding contacts not only simulates the physically accurate grasping and manipulating motions, but also helps us to examine dynamic behaviors of the multi-fingered robot hand. The primary objectives of this dissertation are:

- to model the kinematics and the dynamics of a multi-fingered robot hand sys-



tem performing various dextrous manipulation task, which involves free motion, sticking contact, rolling contact, sliding contact as well as collision between the fingers and the manipulated object

- to construct a dynamic simulator that simulates dextrous manipulation

The constructed dynamic simulator for the multi-fingered robot hand system does not limit to apply to the DMTS. It is also precious to act as the general platform to evaluate different control algorithms for the multi-fingered robot hand system, multiple manipulators cooperation system, and the multi-legged walking robot since they are also classified as the closed chain multi-body mechanism. Safety, cost-effectiveness, and high efficiency are the advantages of using simulation system to perform preliminary control problem study before doing experiments with the real robots. Different situations can be simulated by just tuning the simulation parameters such as the total degree of freedom of the mechanism or the frictional coefficient between interactions. Existing dynamic simulators work under many assumptions. For example, in (Cole et al., 1989), the authors study control problem of dextrous manipulator with sticking or rolling contacts, assuming no sliding contacts nor collision occurs. Our dynamic simulator is suitable for all kind of dextrous motions.

## 1.2 Related Work

Dynamic simulation of hand motions has been studied in computer science and engineering. For the work of Sanso and Thalmann (1994) and Ip and Chan (1997), a realistic animation of human hand is aimed to develop. Sanso and Thalmann utilize the geometric modeling of human hand and the kinematic control to realistically simulate human grasping motions. Ip and Chan add the dynamic effects into their consideration in animating human hand motion. Both studies limit their work in hand motion simulation but do not consider any grasping or manipulation. In (Kasinski et al., 1991), the authors proposed a computer graphic system for interactive grasp planning. The objective is similar to the one used in DMTS but the authors only consider the geometrical modeling problems in simulation and provide a guideline for construction and applications. They have not constructed any system based on their ideas. Bergamasco et al. (1994) proposed another approach for simulating grasping motions in virtual reality. They firstly determine the contact area between the interacting virtual hand and virtual object. The contact force is then estimated by exploiting inter-penetration between the contacted surfaces.

Since the work mentioned above only consider the unconstrained system (hand without grasped object) or just use simple kinematic and dynamic models to model the hand and the grasped object, they cannot produce accurate simulations. When the development of virtual reality gained its widely attention for the possibly extensive applications in practice, physics-based simulation becomes an active research area. People not only consider an accurate geometric modeling to produce the graphical output of simulation but also pay attention to whether the system is physically realistic. In (Goyal et al., 1994a,b; Keller et al., 1995), dynamic behaviors of the unconstrained rigid bodies are simulated incorporating with physical laws such as Newton-Euler equations and friction law. Rigid bodies which are in contact each other are viewed as those which are connected by simple mechanical components such as spring and damper. Contact force is determined by the deformation of these components. On the other hand, Cremer has developed another two general-purpose dynamic simulators. *Newton* described in (Cremer, 1989; Cremer and Stewart, 1989) is able to allow constraints to be added or deleted while the simulation is running. *Isaac* in (Cremer and Vaněček, 1994) is an extension of *Newton* and it was designed specifically for running in virtual environment. Recently, Mirtich (1996) uses an alternative method to develop his general-purpose dynamic simulator. He uses impulse to model all interactions occurred in simulation. These interactions include both contacts and collision. His modeling approach and developed simulator are efficient to simulate situation where the interactions are rapidly changing. In the field of mechanical engineering, Haug et al. (1986) developed the first dynamic simulator specially for multi-body mechanism. It supports analysis of mechanisms in a standard paradigm: formulate motion equations and kinematic constraints, and then numerically integrate them over time. In (Lee et al., 1994), the authors developed a dynamic simulation package for interactive robotic simulation. The structure of their system sets a standard framework in constructing dynamic simulator for robotic simulation.

Besides the general-purpose dynamic simulators, some other specific-purpose dynamic simulators have been developed in different fields. Fujimoto and Kawamura (1995), and Wendlandt (1997) developed dynamic simulation systems for the biped walking robot. Fujimoto and Kawamura combine the general manipulator simulation method and contact simulation of rigid body mechanics to simulate the walking motion of a biped robot. All interaction forces between the foot and the ground are modeled as the impact forces. They are determined by solving a quadratic programming problem with constraints. Wendlandt's simulation is developed on top of Mirtich's dynamic simulator. A simple contact model was proposed to simulate different contact situa-



tions such as sliding and sticking contacts. On the other hand, McMillan (1995) in his Ph.D work simulated the underwater robotic vehicle systems. Efficiency of simulation is emphasized in his development and the dynamic modeling is done by using recursive approaches. Gillespie (1996) models another type of complex mechanical mechanism: playing piano in a computer system with haptic display. This type of motion involves rapid changes in kinematic constraints. He reformulates the dynamics using the Kane's formulation method (Kane and Levinson, 1985) to handle these changes since this independent coordinate formulation method can help to reduce the number of constraints. It embeds both holonomic and nonholonomic constraints in the equations of motion.

For the dynamic simulation of multi-fingered robot hand system, Hashimoto et al. (1994), and Kunii and Hashimoto (1996) developed a *dynamic force simulator* in a virtual environment. They simulate the dynamics of the object, contact model and characteristics of friction between the human hand and the interacting object. Since their input devices can input the force measured from the human hand, they do not consider the dynamics of robot hand. Reznik and Laugier (1996) constructed a similar dynamic simulator for multi-fingered robot hand system. Rather than using rigid fingertip model in simulation, they use a deformable model to simulate the interactions between the fingertips and the object. The contact forces are computed by using a simplified finite element analysis method. These multi-fingered robot hand dynamic simulators work well to simulate grasping motion. However, since they only use the sticking contact model, no sliding or rolling contact is allowed in simulation. Limiting to use sticking contact model prohibits us to simulate manipulation motions using multi-fingered robot hand. Because the manipulation motions may involve rapid change of contact constraints. Mirza et al. and Baiardi et al. have considered different contact models in dextrous manipulation. Using compliant contact model in (Mirza et al., 1993) can model friction, rolling, slipping, and wedging effects, as well as the changing topological structures of grasp dynamics but their simulator is only a two dimensional simulation system. Also, stability cannot be ensured in numerical integration when parameters are tuned to mimic the rigid body model. Baiardi et al. (1993) derives all the dynamics to model different contact phenomena but their validities have not been proved as no dynamic simulation system has been developed based on these equations.

### 1.3 Contributions

The objective of this dissertation is to develop the kinematic and dynamic models that combine free motion, collision, and contact motions for a multi-fingered robot hand

system. Moreover, by applying the derived equations, a dynamic simulation system is developed for a five-fingered robot hand system. It is constructed based on the full kinematics and dynamics of the robot hand. The developed multi-fingered robot hand dynamic simulation system can help us to examine the dynamic behaviors of dextrous grasping and manipulation. Also, it can act as a platform to evaluate different control algorithm designed for the multi-fingered robot hand. In the development of the dynamic model, the major difficulties we encountered are:

- Rapid changes in contact constraints and grasp configurations are common in dextrous manipulation but there is no unified framework to model the dynamics under different contact situations.
- For the sudden change of velocity of the system caused by collision, there is no concise three-dimensional impulsive collision with friction model.

The main contributions can be divided into two parts and they are:

- Theoretical contributions:
  1. **Unified formulation of contact constraints:** Different contact constraints occurred in dextrous manipulation using multi-fingered robot hand are integrated into a same framework. The unified model includes free motion, sticking contact, rolling contact and sliding contact.
  2. **New contact transition model:** A new contact transition model is proposed to handle the transition of different contact modes.
  3. **Improved impulsive collision model:** A simple but accurate impulsive collision model is developed to handle the three dimensional impact with friction between the fingers and the manipulated object.
- System development:
  1. **New sophisticated dynamic simulation system:** A dynamic simulator of a multi-fingered robot hand with full kinematics and dynamics is developed. To the best of the author's knowledge, this dynamic simulation system is the first of its kind simulator for the multi-fingered robot hand in the world. It allows various grasp configuration of the robot hand and different contact modes in dextrous manipulation motion.
  2. **Systems integration:** The development of the dynamic simulator in virtual environment incorporates the knowledge of virtual reality, multi-body



dynamics, contact modeling, physics-based simulation and human control. The system integration method used in this dissertation can be applied in many other similar systems.

## 1.4 Organization of the Thesis

This dissertation presents the dynamic modeling and simulation of a multi-fingered robot hand in the following chapters:

- **Chapter 2 Contact Modeling: Kinematics:** Various methods for modeling *contact* interaction are reviewed and the constraint-based formulation method is applied. The contact kinematics, finger kinematics, and grasp kinematics of the multi-fingered robot hand system are derived in order to obtain the velocity constraints of the entire system. Different kinds of contact constraints are incorporated in the formulation for the purpose to allow various grasping configurations and various contact modes.
- **Chapter 3 Contact Modeling: Dynamics:** This chapter is to derive the dynamic equations of the fingers and the manipulated object under the velocity constraint derived in Chapter 2. They are essential to the study of the dynamic behaviors of the multi-fingered robot hand system manipulating an object.
- **Chapter 4 Collision Modeling:** A first order differential equation is derived to describe the collision process. Because in different collision conditions, the integration of the differential collision equation are different. Both sliding mode and sticking mode collision integration with details are discussed. There are some special treatments for the system with mixed contact and collision interactions in using our collision model. It will be discussed at the end of the chapter.
- **Chapter 5 Dynamic Simulation:** The overview architecture of the dynamic simulation system of the five-fingered robot hand system is discussed and various components of the system are presented. Since the dynamic simulator is the core of the dynamic simulator, each of its components is further discussed and the implementation issues are also discussed.
- **Chapter 6 Simulation Results:** Dynamic simulation examples and results are presented in this chapter. The examples include the change of grasping configurations, different types of contacts, impulsive collision, and simple dextrous manipulation motions.

- **Chapter 7 Conclusions:** This chapter summaries the work and also discusses the future development of the system.

In a number of appendices, we enclose some reference materials including

- **Appendix A Montana's Contact Equations for Finger-object Contact**
- **Appendix B Finger Dynamics**
- **Appendix C Simulation Configurations**

## Chapter 2

# Contact Modeling: Kinematics

### 2.1 Introduction

In real world, when we grasp or manipulate a rigid object, our fingers never penetrate into the object. However, it is not the case in computer world. Consider Figure 2.1, when we want to use the hand to grasp a virtual rigid object in a computer simulated environment, the fingers will enter into the object if nothing is done to prevent these inter-penetrations from happening. For computer animation systems without considering dynamic effects, this kind of situation can be easily avoided by monitoring kinematic relationships between objects. For example, the grasping simulator developed by (Iwata, 1990) assumes that when the virtual hand is in contact with the virtual object, the virtual object is considered captured and graphically connected with the virtual hand. This method provides a visual effect that the virtual hand just touches the virtual object. However, for dynamic simulation systems, contact force between the contacted objects must be determined in order to simulate the dynamic effects. It is computed based on the dynamic properties of the simulated system to prevent the unrealistic inter-penetrations occurred in simulation.

The most widely used method in computer graphics community (Moore and Wilhelms, 1988; Cadoz et al., 1993; Luciani et al., 1991) to determine the contact constraint force is the *penalty method*. Contact presented during simulation is modeled by a spring or a spring-damper coupler connecting the bodies in contact. The interaction force is computed based on the depth and the speed of penetration. To mimic this unilateral contact constraint, the spring-damper pair is removed when a change from compressive to tensile interaction force is detected. In (Kraus and Kumar, 1997; Kraus et al., 1997), the authors use an improved model to simulate frictional contact motion. Their compli-



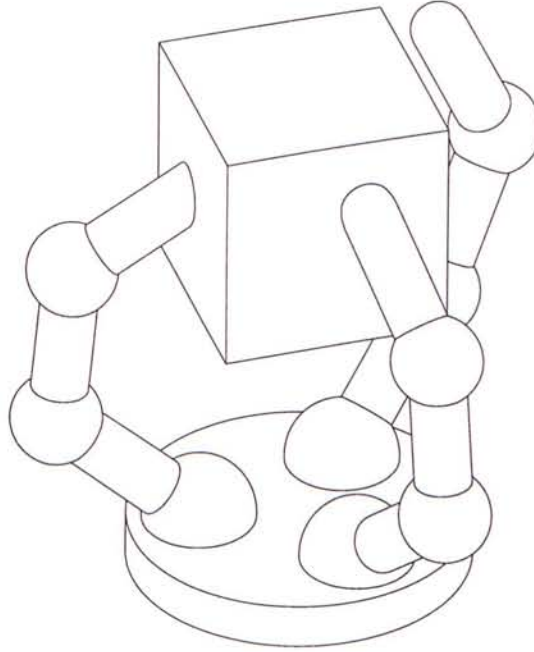


Figure 2.1: Inter-penetrations between the virtual fingers of the multi-fingered robot hand and the virtual object in computer simulation.

ant model uses a pair of spring-damper couplers to model the normal and the tangential contact interactions. The spring-damper couplers allow small local deformation at the contact point, which incorporate mechanisms for energy storage and for energy dissipation. These penalty methods are easy to implement and efficient in simulation. Their major disadvantage is to yield stiff equation<sup>1</sup> when large spring constant (stiffness coefficient) is applied in order to approximate the rigid body behaviors (Witkin et al., 1992). Thus, the equations of motion of the whole system become numerically ill-conditioned.

When we examine the contact phenomenon physically, there are some properties that can help us to compute the interaction force analytically. Firstly, when we denote the relative normal acceleration between contact points by  $a_n$  and the interaction force as  $f_n$ , two inequality constraints can be obtained:

$$\begin{cases} a_n \geq 0 \\ f_n \geq 0 \end{cases}$$

---

<sup>1</sup>A *stiff* system is one whose dynamical differential equations possess a solution with widely disparate time constants.



These relationships are easy to understand since no inter-penetration is allowed for the contact of rigid bodies. The third characteristic is called the *complementarity constraint*:

$$f_n^T a_n = 0$$

This constraint reflects that if the interaction force is nonzero, the relative acceleration must be zero and the contact is a resting contact. Otherwise, if the interaction force is zero, the acceleration must be positive so the bodies are moving apart. As a result, determination of contact force with above constraints can be solved as a *linear complementarity problem (LCP)* (Cottle et al., 1992). It is one of branches in optimization with wide study. Lötstedt (1982) is the first person who uses this idea to treat friction between contacting rigid bodies. Baraff extends his work to develop a dynamic simulator focusing on the non-penetrating rigid bodies (Baraff, 1993, 1994, 1992). When there is no friction in the system, the LCP method always yield unique solution for the contact force. However, when considering the effect of friction, the corresponding LCP may not have a solution, or if there is solution, it may not be unique. Existence and uniqueness of LCP solution in dynamic simulation are extensively studied in (Pang and Trinkle, 1996a; Trinkle et al., 1996; Pang and Trinkle, 1996b) and they can be guaranteed when strict modification of the problem is applied.

The idea of penalty method is trivial but accuracy of the computed contact force cannot be ensured or unstable equations of motion may be obtained. The LCP method is accurate but it is under restrictions. Also, it is comparative time consuming to solve optimization problem. An alternative to determine the interaction force analytically is to transform the inequality constraints into equality constraints. The bilateral constrained mechanical system is well studied and the constraint force can be easily calculated by, for example, the Lagrangian method (Rosenberg, 1977). Haug et al. (1986) have used similar idea in implementing their general-purpose dynamic simulator. The most difficult part of this approach is how to choose a corresponding bilateral constraint to replace the unilateral constraint when contact is detected. A comprehensive contact transition model is proposed in this dissertation to solve this problem.

The bilateral constraints in contact dynamics represent the geometrical relationships between the contacting rigid bodies or they are called the *contact kinematics* in mechanics. The contact kinematics for rigid bodies is firstly examined in (Cai and Roth, 1987, 1988). Cai and Roth formulate the equations with respect to Cartesian reference frame established at the contact point of the three dimensional contacting rigid

bodies. Montana (1988) generalizes their results by applying differential geometry concepts and people call these generalized equations the *Montana's contact equations*. His result assumes that the contacting bodies are always in contact. The approach developed by Anitescu et al. (1995) allows the non-contacting case based on Montana's work. Sarkar et al. (1996) also extend the velocity level Montana's contact equations to the acceleration level.

In this dissertation, the constraint-based formulation method is applied in modeling different kinds of contacts presented in dextrous manipulation motions. It provides a comparative simple and effective method to determine the constraint force. In this chapter, we will present the contact kinematics occurred in multi-fingered manipulation and review the Montana's contact equations. Based on these equations, we will derive the velocity constraint for the whole multi-fingered robot hand system and the manipulated object. This velocity constraint is in a generalized form that can represent free motion, sticking contact, rolling and sliding contacts. Force related materials will not be considered in this chapter and they will be discussed in next chapter.

## 2.2 Contact Kinematics between Two Rigid Bodies

### 2.2.1 Contact Modes

Before the discussion of contact kinematics in dextrous manipulation, the simple contact kinematics between two rigid bodies is introduced first. When the two rigid bodies are not in touch, we generally call that they are in free motions. If the two rigid bodies are in contact, there are three different contact modes and they may be in sticking contact, rolling or sliding contact. Consider Figure 2.2, coordinate frames  $\Sigma_1$  and  $\Sigma_2$  are located at the contact points on the surfaces of object 1 and object 2 respectively. Let  $v = [v_x \ v_y \ v_z]^T$  and  $\omega = [\omega_x \ \omega_y \ \omega_z]^T$  be the relative translational and rotational velocities of the coordinate frame  $\Sigma_1$  with respect to the coordinate frame  $\Sigma_2$ . When we assume that the two contacting objects' surfaces are smooth, the three contact modes can be defined in terms of the elements of  $v$  and  $\omega$  since some components of the relative contact velocity are constrained for a particular contact mode. The mathematical expressions of the three contact modes are:



1. **Sticking contact:**

$$\begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = 0 \quad \text{and} \quad \begin{bmatrix} \omega_x \\ \omega_y \end{bmatrix} = 0 \quad (2.1)$$

It represents the two contacting objects are sticking together and there is no relative motions between them except the rotation about the normal vector at the contact point.

2. **Rolling contact:**

$$\begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = 0 \quad \text{and} \quad \omega_z = 0 \quad (2.2)$$

It represents the pure rolling contact so translational movement is not allowed.

3. **Sliding contact:**

$$v_z = 0 \quad \text{and} \quad \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = 0 \quad (2.3)$$

Similarly, this contact considers only the pure sliding contact.

### 2.2.2 Montana's Contact Equations

The Montana's contact equations describe the motion of contact points on the surfaces of two contacting objects in response to a relative motion of these objects. With reference to Figure 2.3, the coordinate frame  $\Sigma_1$  and the coordinate frame  $\Sigma_2$  are assigned at the contact points on the surfaces of the two objects. The  $z$ -axes of them are pointing outwardly in the normal directions at the contact point. Local coordinate charts  $c_1$  and  $c_2$  are defined on the surfaces of the two contacting objects. The local coordinate chart  $c_i$  takes a point  $\begin{bmatrix} u_{ix} \\ u_{iy} \end{bmatrix} \in \mathbb{R}^2$  to a point  $\mathcal{X} \in \mathbb{R}^3$  on the surface of an object. The Montana's contact equations governing the geometrical relationships of

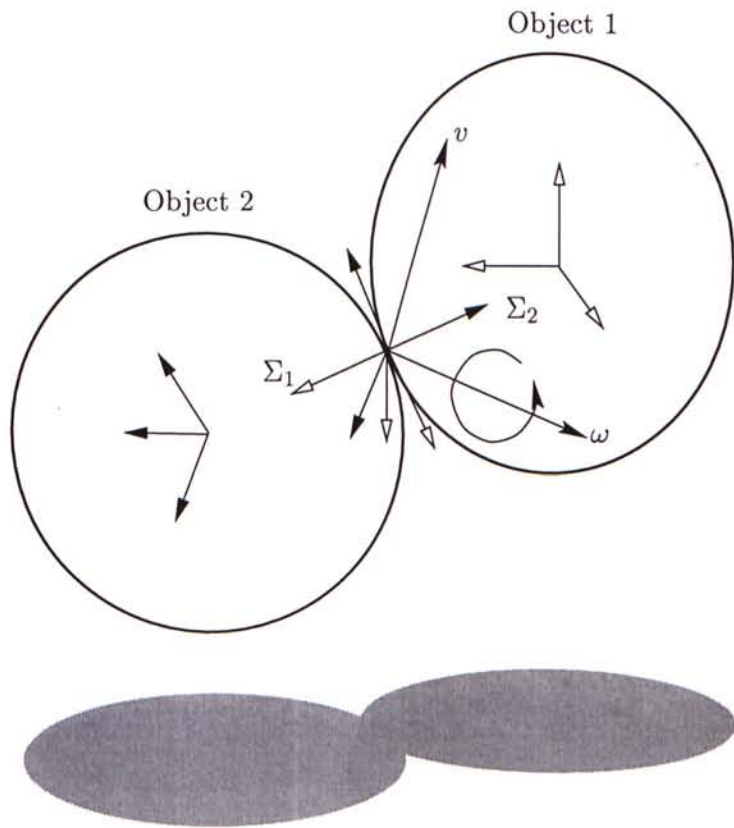


Figure 2.2: Relative contact velocity defined in two contacting rigid bodies.

the two contacting rigid objects are:

$$\begin{cases} \dot{u}_1 = M_1^{-1}(K_1 + \tilde{K}_2)^{-1} \left( \begin{bmatrix} -\omega_y \\ \omega_x \end{bmatrix} - \tilde{K}_2 \begin{bmatrix} v_x \\ v_y \end{bmatrix} \right) \\ \dot{u}_2 = M_2^{-1} {}^2\hat{R} (K_1 + \tilde{K}_2)^{-1} \left( \begin{bmatrix} -\omega_y \\ \omega_x \end{bmatrix} + K_1 \begin{bmatrix} v_x \\ v_y \end{bmatrix} \right) \\ \dot{\psi} = \omega_z + T_1 M_1 \dot{u}_1 + T_2 M_2 \dot{u}_2 \end{cases} \quad (2.4)$$

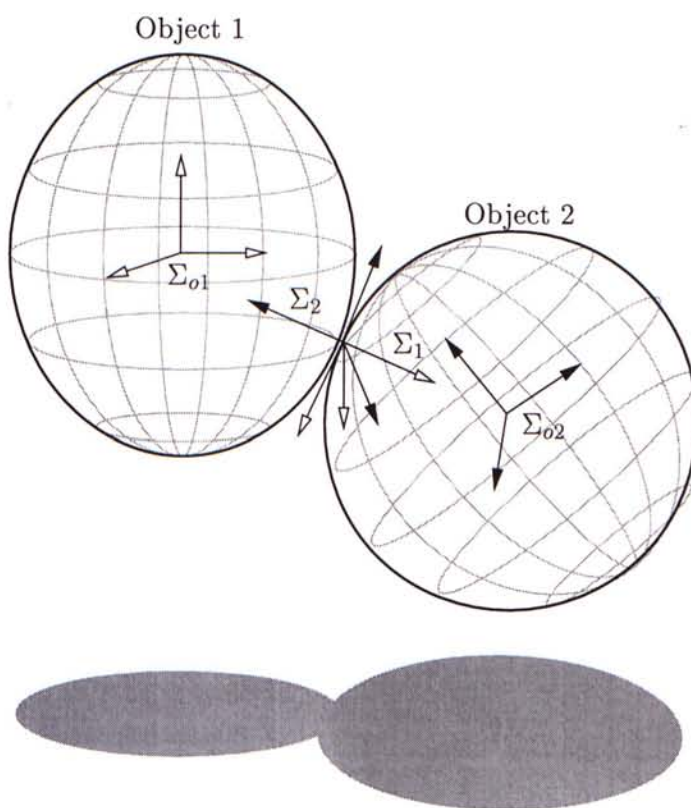


Figure 2.3: Coordinate frames assignment for the Montana's contact equations.

The rotation angle  $\psi$  makes the  $x$ -axes of coordinate frame  $\Sigma_1$  and that of coordinate frame  $\Sigma_2$  aligned. Matrix  ${}^2\hat{R}$  is the sub-matrix of  ${}^2R$  which is the rotational matrix<sup>2</sup> of frame  $\Sigma_1$  relative to the frame  $\Sigma_2$ :

$${}^2R = \begin{bmatrix} {}^2\hat{R} & 0_{2 \times 1} \\ 0_{1 \times 2} & 1 \end{bmatrix} \quad \text{and} \quad {}^2\hat{R} = \begin{bmatrix} \cos \psi & -\sin \psi \\ -\sin \psi & -\cos \psi \end{bmatrix}$$

<sup>2</sup>This rotational matrix can be computed by firstly rotating  $180^\circ$  about the  $x$ -axis and then rotating

Vector  $\begin{bmatrix} v_x & v_y & v_z & \omega_x & \omega_y & \omega_z \end{bmatrix}^T$  is the generalized relative velocity of the coordinate frame  $\Sigma_1$  with respect to the coordinate frame  $\Sigma_2$ . Matrices  $K_i$ ,  $T_i$ , and  $M_i$  are the curvature tensor, torsion tensor, and metric tensor of the  $i$ th contact surface. The matrix  $\tilde{K}_2$  is determined by equation

$$\tilde{K}_2 = {}^2\hat{R} K_2 {}^2\hat{R}$$

Figure 2.4 shows the input-output relationship of the Montana's contact equations. When we have the information of the relative velocity of two contacting bodies, we can determine the contact trajectories on these contacting surfaces. This piece of information is useful when we need to integrate the contact kinematics and the contact dynamics together. The detailed derivation of the Montana's contact equations for the contacting finger and the object in our multi-fingered robot hand system can be referred in appendix A.

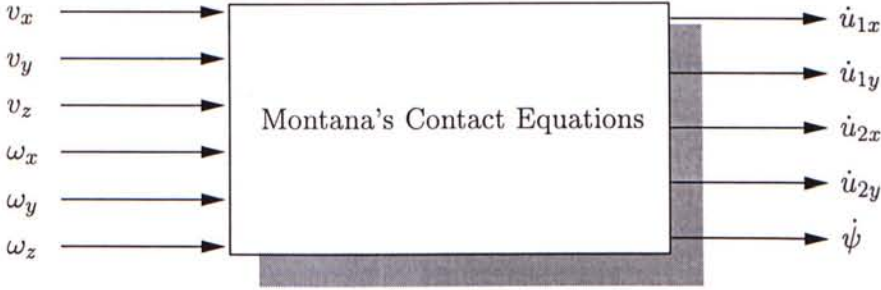


Figure 2.4: Input-output relationship of the Montana's contact equations.

## 2.3 Finger Kinematics

Besides the contact kinematics between two rigid bodies, finger kinematics is also important in formulating the kinematics of the whole multi-fingered robot hand system.

about the  $z$ -axis by angle  $\psi$ . The mathematical expression of this rotational matrix is:

$$\begin{aligned} {}^2R &= R_x(\pi)R_z(\psi) \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \cos \psi & -\sin \psi & 0 \\ \sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos \psi & -\sin \psi & 0 \\ -\sin \psi & -\cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned}$$



Finger kinematics relates the joint space to the Cartesian space. The finger Jacobian matrix further relates the generalized velocities and the generalized forces in the joint and the Cartesian spaces. In this section, the finger forward kinematics and the finger Jacobian are reviewed.

### 2.3.1 Finger Forward Kinematics

The derivation of finger forward kinematics can be divided into two steps. The first step is to assign coordinate frames to each finger link and to find out the kinematic parameters of the finger. Next, the *finger equation*<sup>3</sup> is derived, which is actually the finger forward kinematics, from the link-coordinate transformation matrices.

The coordinate frame for each finger link can be assigned by the *modified*<sup>4</sup> Denavit-Hartenberg method (Craig, 1986) and the assignment is shown in Figure 2.5. After the assignment, the kinematics parameters of the finger can be defined and they include the joint parameters and the link parameters. The joint parameters describe the relative position and orientation of two successive links and the link parameters represent the relative position and orientation of the axes of two success joints. Descriptions of these parameters are summarized in Table 2.1.

The general form of the finger equation is:

$$F(\theta) = \begin{bmatrix} R(\theta) & p(\theta) \\ 0_{1 \times 3} & 1 \end{bmatrix} \quad (2.5)$$

Matrix  $R(\theta) \in \mathfrak{R}^{3 \times 3}$  is the rotational matrix and vector  $p(\theta) \in \mathfrak{R}^{3 \times 1}$  is the translational vector. Vector  $\theta$  represents the generalized joint angles. This finger equation can be determined by the following equation:

$$F(\theta) = {}^b_1T \ {}^1_2T \ \dots \ {}^i_{i+1}T \ \dots \quad (2.6)$$

Index  $i$  represents the  $i$ th coordinate frame and the index  $b$  represents the inertial coordinate frame. Matrix  ${}^i_{i+1}T$  is a  $4 \times 4$  *link-coordinate transformation matrix* which represents the relations between the  $i$ th coordinate frame and the  $i + 1$ th coordinate

<sup>3</sup>Generally, it is called *arm equation* for the manipulator. It is renamed in this dissertation in order to emphasize that it is used for the finger. They are actually identical.

<sup>4</sup>It is the modified version of the popular Denavit-Hartenberg assignment method and its introduction is to avoid the problems arisen in closed kinematic chain.

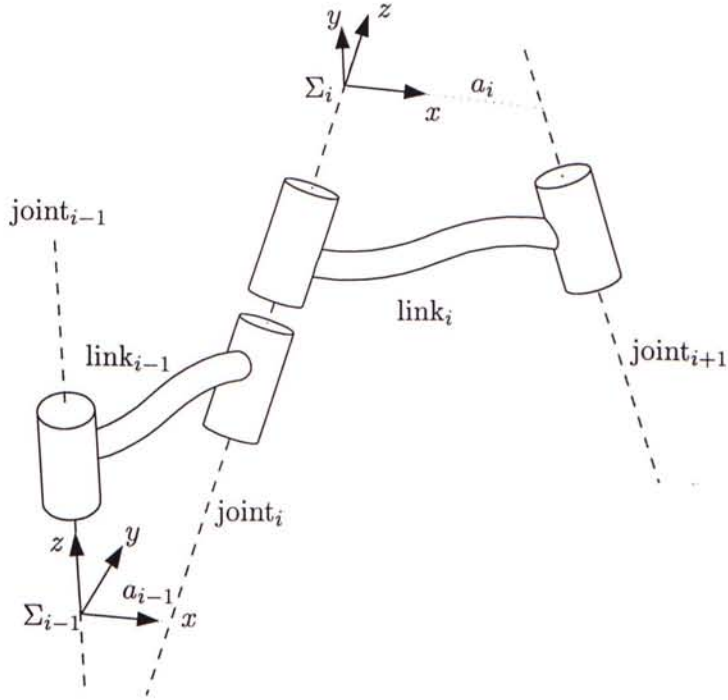


Figure 2.5: Coordinate frame assignment of the modified Denavit-Hartenberg method.

	Name	Description
<b>Joint Parameters</b>	Joint angle, $\theta_i$	The rotation about the $z$ -axis of the $i$ th coordinate frame ( ${}^i z$ ) which is required to make the $x$ -axis of the $i$ th coordinate frame ( ${}^i x$ ) parallel with the $x$ -axis of the $i + 1$ th coordinate frame ( ${}^{i+1} x$ ).
	Joint distance, $d_i$	The translation along ${}^i z$ which is required to make axis ${}^i x$ intersect with axis ${}^{i+1} x$ .
<b>Link Parameters</b>	Link length, $a_i$	The translation along ${}^{i+1} x$ which is needed to make axis ${}^i z$ intersect with axis ${}^{i+1} z$ .
	Link twist angle, $\alpha_i$	The rotation about ${}^{i+1} x$ which is needed to make axis ${}^i z$ parallel with axis ${}^{i+1} z$ .

Table 2.1: Finger parameters.

frame. This transformation has the form of:

$${}_{i+1}^i T = \begin{bmatrix} c_{\theta_i} & -c_{\alpha_i} s_{\theta_i} & s_{\alpha_i} s_{\theta_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\alpha_i} c_{\theta_i} & -s_{\alpha_i} c_{\theta_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.7)$$

The symbols  $s_{\theta_i}$  and  $c_{\theta_i}$  represent  $\sin(\theta_i)$  and  $\cos(\theta_i)$  respectively.

### 2.3.2 Finger Jacobian

The velocity relationship of the kinematics can be obtained by determining the  $i$ th finger Jacobian  $J_i(\theta_i)$ . Assuming  $\Sigma_b$  and  $\Sigma_{f_i}$  are the inertial and the fingertip coordinate frames of the  $i$ th finger, the finger Jacobian relates the velocity of the  $i$ th fingertip coordinate frame  ${}^b \mathcal{V}_{f_i}$  to the joint velocity  $\dot{\theta}_i$  together as

$${}^b \mathcal{V}_{f_i} = J_i(\theta_i) \dot{\theta}_i \quad (2.8)$$

Since the finger Jacobian can be partitioned into two sub-matrices:

$$J_i(\theta_i) = \begin{bmatrix} J_{v_i}(\theta_i) \\ J_{\omega_i}(\theta_i) \end{bmatrix}$$

where  $J_{v_i}(\theta_i)$  represents the Jacobian corresponding to linear velocity and  $J_{\omega_i}(\theta_i)$  represents the Jacobian corresponding to angular velocity. The term  $J_{v_i}(\theta_i)$  can be determined using the vector  $p(\theta)$  in equation (2.5) and the determination of  $J_{\omega_i}(\theta_i)$  requires the information contained in the link-coordinate transformation matrices (equation (2.7)).

## 2.4 Grasp Kinematics between a Finger and an Object

The finger kinematics only considers the relationship between the joint space and the Cartesian space. When the fingers are manipulating an object, several closed chains are formed. It is essential to determine the *grasp kinematics* to find out the relations of the velocities of the fingers, velocities at the contact points and the velocity of the object. In this section, the grasp kinematics between the  $i$ th finger and the object is derived. The derivation not only considers one particular mode of contact but also incorporates different contact modes and different fingertip models into one equation



to facilitate the simulation of dextrous manipulation.

### 2.4.1 Velocity Transformation between Different Coordinate Frames

In order to formulate the grasp kinematics between the  $i$ th finger and the object, velocity transformation methods between different coordinate frames are very important. For example, we always need to know the relationship between the velocity of point  $a$  with respect to coordinate frame  $\Sigma_A$  and the velocity of point  $b$  with respect to coordinate frame  $\Sigma_B$ . The velocities of different parts of the system may be with respect to different coordinate frames and the analysis of the resultant motions requires us to study them in the same coordinate frame. The first Theorem shows us the velocity transformation between two coordinate frames and the second one describes the transformation between three different coordinate frames.

**Theorem 2.1 (Craig, 1986).** *Assume  $\mathcal{V} = \begin{bmatrix} v \\ \omega \end{bmatrix}$  be the generalized velocity of an object and  $T_{\mathcal{V}}$  be the  $6 \times 6$  velocity transformation matrix. This velocity transformation maps the velocity of a point in one coordinate frame into corresponding velocity of another point in another coordinate frame.*

$${}^b\mathcal{V}_b = {}^bT_{\mathcal{V}} {}^a\mathcal{V}_a \quad (2.9)$$

where  ${}^a\mathcal{V}_a$  is the velocity of point  $a$  with respect to coordinate frame  $\Sigma_a$  and  ${}^b\mathcal{V}_b$  is the velocity of point  $b$  with respect to coordinate frame  $\Sigma_b$ , and the velocity transformation matrix  ${}^bT_{\mathcal{V}}$  can be expressed as:

$${}^bT_{\mathcal{V}} = \begin{bmatrix} {}^a_bR & {}^ap_b \times {}^a_bR \\ 0_{3 \times 3} & {}^a_bR \end{bmatrix} \quad (2.10)$$

where  ${}^a_bR$  is the  $3 \times 3$  rotational matrix and  ${}^ap_b$  is the  $3 \times 1$  translational vector.

Based on Theorem 2.1, we can have another velocity transformation equation which involves more than two coordinate frames.

**Theorem 2.2 (Li et al., 1989).** *If there are three coordinate frames  $\Sigma_a$ ,  $\Sigma_b$  and  $\Sigma_c$ , the relative velocities acting on different coordinate frames can be related as*

$${}^a\mathcal{V}_c = {}^cT_{\mathcal{V}} {}^a\mathcal{V}_b + {}^b\mathcal{V}_c \quad (2.11)$$

*Proof.* Consider Figure 2.6, vector  ${}^i\mathcal{V}_{j/k}$  represents the relative velocity from coordinate frame  $\Sigma_k$  to coordinate frame  $\Sigma_j$  with respect to  $\Sigma_i$ . The relative velocity  ${}^c\mathcal{V}_{c/a}$  can

be written as:

$${}^c\mathcal{V}_{c/a} = {}^c\mathcal{V}_{b/a} + {}^c\mathcal{V}_{c/b}$$

Since the term  ${}^c\mathcal{V}_{b/a}$  can be projected to coordinate frame  $\Sigma_b$  by Theorem 2.1 as:

$${}^c\mathcal{V}_{b/a} = {}^cT_b \mathcal{V}_{b/a}$$

The relative velocity  $\mathcal{V}_{j/k}$  can be viewed as  ${}^k\mathcal{V}_j$ . In this way, the equation (2.11) can be obtained.  $\square$

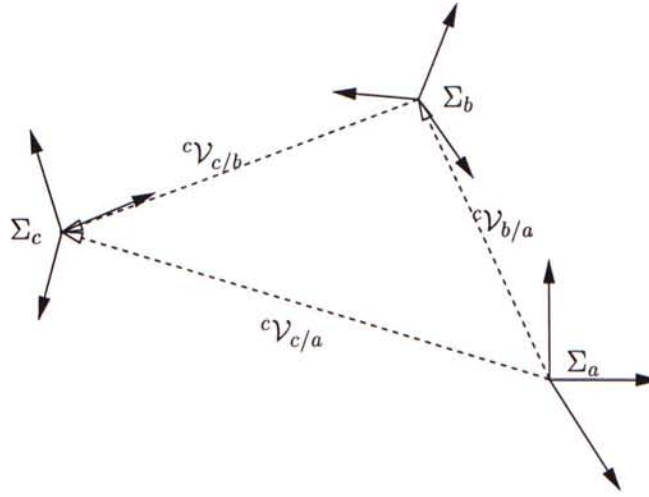


Figure 2.6: Velocities transformation in different coordinate frames.

When the coordinate frame  $\Sigma_c$  is fixed relative to the coordinate frame  $\Sigma_b$ , equation (2.11) can be further simplified as:

$${}^a\mathcal{V}_c = {}^cT_b \mathcal{V}_b \quad (2.12)$$

### 2.4.2 Grasp Kinematics for the $i$ th Contact

Figure 2.7 shows the coordinate frames assignment for the  $i$ th finger and the object that are in contact.  $\Sigma_b$  is the inertial coordinate frame. It is fixed in the workspace such that we can use it as the universal coordinate reference frame.  $\Sigma_{fi}$  and  $\Sigma_o$  are the coordinate frames placed at the fingertip of the  $i$ th fingertip and the object respectively. They are used to describe the configurations<sup>5</sup> of both the  $i$ th fingertip and the object. Both  $\Sigma_{li}$

<sup>5</sup> *Configuration* is a term used to describe both the translational locomotion and the orientation of an object.



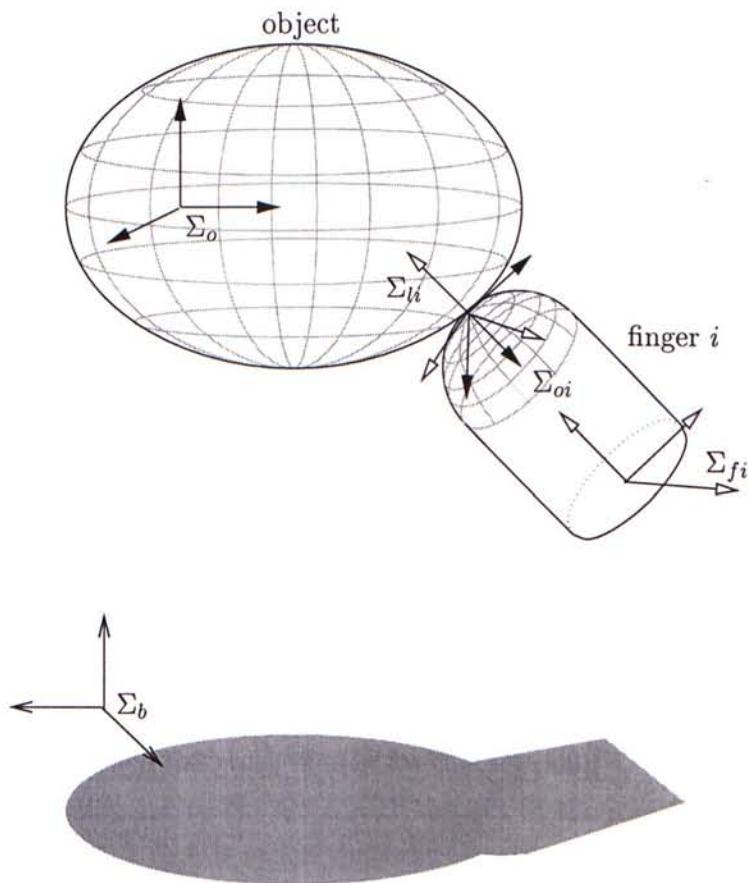


Figure 2.7: Coordinate frames assignment for the contacting  $i$ th finger and object.

and  $\Sigma_{oi}$  are the coordinate frames located at the contact points.  $\Sigma_{li}$  is placed at the contact point on the surface of the  $i$ th fingertip. Similarly,  $\Sigma_{oi}$  is placed at the contact point on the surface of the object. Their  $z$ -axes are pointing in the normal directions of the contact point so they are aligned on the straight line. Using Theorem 2.2, we can establish following velocity relationships for coordinate frames  $\Sigma_b$ ,  $\Sigma_{li}$ , and  $\Sigma_{oi}$ :

$${}^b\mathcal{V}_{oi} = {}_{li}^{oi}T_{\mathcal{V}} {}^b\mathcal{V}_{li} + {}^{li}\mathcal{V}_{oi}$$

This equation implies that the relative velocity  ${}^{li}\mathcal{V}_{oi}$  can be determined by the inertial contact velocities at the contact points on the  $i$ th fingertip surface and on the object surface. It is very important since different expressions of the relative contact velocity  ${}^{li}\mathcal{V}_{oi}$  implies different contact mode as described in Section 2.2.1. However, we are more interested to know how the joint velocities and the object velocity affect this relative contact velocity. Because there is no relative motion between the coordinate frames  $\Sigma_{li}$  and  $\Sigma_{fi}$ , and between the coordinate frames  $\Sigma_{oi}$  and  $\Sigma_o$ , we can apply the equation (2.12) in Theorem 2.2 to obtain following equations:

$${}^b\mathcal{V}_{li} = {}_{fi}^{li}T_{\mathcal{V}} {}^b\mathcal{V}_{fi} \quad \text{and} \quad {}^b\mathcal{V}_{oi} = {}_o^{oi}T_{\mathcal{V}} {}^b\mathcal{V}_o$$

On the other hand, the velocity of coordinate frame  $\Sigma_{fi}$  can be related to the joint velocity of the contacting  $i$ th finger by the finger Jacobian (equation (2.8)). As a result, we can express the grasp kinematics for finger  $i$  as follows:

$$G_i^T {}^b\mathcal{V}_o = J_{fi} \dot{\theta}_i + {}^{li}\mathcal{V}_{oi} \quad (2.13)$$

where  $G_i = {}_o^{oi}T_{\mathcal{V}}^T$  is called the *grasp matrix* for contact point  $i$ . It transforms the force acting at the contact point to the force with respect to the inertial coordinate frame  $\Sigma_b$ . Matrix  $J_{fi} = {}_{li}^{oi}T_{\mathcal{V}} {}_{fi}^{li}T_{\mathcal{V}} J_i$  is called the *hand Jacobian for finger  $i$*  and the term  $J_i$  is the finger Jacobian derived in Section 2.3.2.

### 2.4.3 Different Fingertip Models and Different Contact Modes

There are several fingertip models commonly used in the research community and they are:

1. **Frictionless post contact:** It is assumed that there is no friction between the fingertip and the object. In this case, forces can be only applied in the normal direction of the contact point on the object surface.

2. **Frictional post contact:** For this fingertip model, friction assumed to be presented at the contact point and the *Coulomb friction model* is always applied to model the friction. This model implies that the range of tangential forces which can be applied at a contact is governed by:

$$|f_t| \leq \mu f_n$$

where  $f_t$  and  $f_n$  represents both the tangential and normal forces, and the coefficient  $\mu$  is the *static coefficient of friction*. Sliding occurs when the above inequality cannot be satisfied.

3. **Soft fingertip contact:** This model is similar to the frictional post contact. Other than allowing translational forces applied, torque about the normal component can also be exerted and it is limited by the *torsional friction coefficient*.

These fingertip models impose additional constraints to the grasp kinematics. In order to incorporate them into the kinematics, the grasp kinematic equation (2.13) is rewritten as:

$$B_i^T G_i^T {}^b \mathcal{V}_o = B_i^T J_{f_i} \dot{\theta}_i + B_i^T {}^{li} \mathcal{V}_{o_i} \quad (2.14)$$

The matrix  $B_i$  is called the *fingertip model selection matrix*<sup>6</sup> which is in the form of:

1. **Frictionless post contact:**

$$B_i = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T$$

2. **Frictional post contact:**

$$B_i = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T$$

<sup>6</sup>This matrix is same as the *wrench basis* notated in (Murray et al., 1994).



### 3. Soft fingertip contact:

$$B_i = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T$$

On the other hand, the grasp kinematic equation (2.13) is also constrained by another constraints when the contact is in a particular contact mode. Some of the elements of the relative contact velocity  ${}^i\mathcal{V}_{oi}$  are equal to zero. From equation (2.13), the relative contact velocity can be written as

$${}^i\mathcal{V}_{oi} = C_{Mi}(G_i^T {}^b\mathcal{V}_{oi} - J_{fi}\dot{\theta}_i) \quad (2.15)$$

The matrix *contact mode selection matrix*  $C_{Mi}$  for the  $i$ th contact is introduced in the above equation to represent the effects of different contacting situations:

#### 1. Sticking contact:

$$C_{Mi} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

since  $v_x = v_y = v_z = \omega_x = \omega_y = 0$ .

#### 2. Rolling contact:

$$C_{Mi} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

since only the  $\omega_x$  and  $\omega_y$  of the relative contact velocity are non-zero.

## 3. Sliding contact:

$$C_{Mi} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

since only the elements  $v_x$  and  $v_y$  are non-zero.

As a result, the grasp kinematic equation (2.13) can be extended to incorporate both fingertip model selection matrix and contact mode selection matrix to represent various fingertip models and contacting situations. Combining equation (2.14) and equation (2.15), the grasp kinematic equation becomes

$$B_i^T G_i^T {}^b\mathcal{V}_o = B_i^T J_{fi} \dot{\theta}_i + B_i^T C_{Mi} (G_i^T {}^b\mathcal{V}_{oi} - J_{fi} \dot{\theta}_i) \quad (2.16)$$

## 2.5 Velocity Constraints of the Entire System

The previous section only derives the grasp kinematics for the  $i$ th contacting finger and the object. Typically, there should be more than one finger for a multi-fingered robot hand. We can combine the grasp kinematic equations for all fingers and write out the velocity constraints of the multi-fingered robot hand system. Assume that there are  $m$  fingers for the robot hand. There are  $m$  grasp kinematic equations (equation (2.16)):

$$\begin{aligned} B_1^T G_1^T {}^b\mathcal{V}_o &= B_1^T J_{f1} \dot{\theta}_1 + B_1^T C_{M1} (G_1^T {}^b\mathcal{V}_{o1} - J_{f1} \dot{\theta}_1) \\ B_2^T G_2^T {}^b\mathcal{V}_o &= B_2^T J_{f2} \dot{\theta}_2 + B_2^T C_{M2} (G_2^T {}^b\mathcal{V}_{o2} - J_{f2} \dot{\theta}_2) \\ &\vdots \qquad \qquad \qquad \vdots \qquad \qquad \qquad \vdots \\ B_i^T G_i^T {}^b\mathcal{V}_o &= B_i^T J_{fi} \dot{\theta}_i + B_i^T C_{Mi} (G_i^T {}^b\mathcal{V}_{oi} - J_{fi} \dot{\theta}_i) \\ &\vdots \qquad \qquad \qquad \vdots \qquad \qquad \qquad \vdots \\ B_m^T G_m^T {}^b\mathcal{V}_o &= B_m^T J_{fm} \dot{\theta}_m + B_m^T C_{Mm} (G_m^T {}^b\mathcal{V}_{om} - J_{fm} \dot{\theta}_m) \end{aligned}$$

The matrix form of the above equation can be expressed as

$$G^T {}^b\mathcal{V}_o - J_h \dot{\theta} = 0$$

where

$$G^T = \begin{bmatrix} B_1^T(I - C_{M1})G_1^T \\ B_2^T(I - C_{M2})G_2^T \\ \vdots \\ B_i^T(I - C_{Mi})G_i^T \\ \vdots \\ B_m^T(I - C_{Mm})G_m^T \end{bmatrix}$$

$$J_h = \begin{bmatrix} B_1^T(I - C_{M1})J_{f1} & 0 & \dots & 0 & \dots & 0 \\ 0 & B_2^T(I - C_{M2})J_{f2} & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & B_i^T(I - C_{Mi})J_{fi} & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & \dots & B_m^T(I - C_{Mm})J_{fm} \end{bmatrix}$$

$$\dot{\theta} = [\dot{\theta}_1 \quad \dot{\theta}_2 \quad \dots \quad \dot{\theta}_i \quad \dots \quad \dot{\theta}_m]^T$$

This equation can be further transformed into the standard form of velocity constraint for a robot hand system:

$$A\dot{q} = 0 \quad (2.17)$$

where

$$A = \begin{bmatrix} -J_h & G^T \end{bmatrix} \quad \text{and} \quad \dot{q} = \begin{bmatrix} \dot{\theta} \\ {}^b\mathcal{V}_o \end{bmatrix}$$

Matrix  $A$  is the coefficient of the velocity constraints and vector  $q$  is the generalized velocity of the system. It can be noted that if  $C_{Mi}$  is equal to a  $6 \times 6$  identity matrix, the term  $B_i^T(I - C_{Mi})G_i^T$  and the term  $B_i^T(I - C_{Mi})J_{fi}$  in the constraint coefficient  $A$  are zero so that velocity constraint for the  $i$ th contact does not exist and various grasp configuration can be achieved by setting different  $C_{Mi}$  to an identity matrix.

## 2.6 Summary

In this chapter, we have reviewed existing approaches at modeling contacts occurred in multi-fingered robot hand manipulation. The constraint-based formulation method is



selected in the contact modeling because it is both accurate and computational efficient. The contact kinematics, finger kinematics and grasp kinematics are derived respectively. The velocity constraint of the multi-fingered robot hand system is deduced based on these kinematic equations. It has the form of:

$$A\dot{q} = 0 \quad \text{Equation (2.17)}$$

Various contacting situations and various grasping configuration are allowed in our derivation by imposing the *contact mode selection matrix*  $C_{Mi}$  (equation (2.15)) into the formulation.  $C_{Mi}$  is a  $6 \times 6$  zero matrix except

$$\begin{cases} C_{Mi} = I_{6 \times 6} & \text{for free motion} \\ C_{Mi}(6,6) = 1 & \text{for sticking contact} \\ C_{Mi}(4,4) = C_{Mi}(5,5) = 1 & \text{for rolling contact} \\ C_{Mi}(1,1) = C_{Mi}(2,2) = 1 & \text{for sliding contact} \end{cases}$$

where  $C_{Mi}(i,i)$  is the  $i$ th row,  $i$  column element. Choosing different contact mode selection matrices can reduce the kinematic equations to reflect a certain kind of contacts. They include both the free motion and the sticking, rolling, and sliding contacts.

## Chapter 3

# Contact Modeling: Dynamics

### 3.1 Introduction

For a dynamic simulation system, the fundamental problem to solve is to determine the equations of motion of the system. The dynamics can help us to calculate the acceleration of the system in response to a set of forces (including both external and internal forces) acting on it. Once the acceleration is determined, numerical integration method can be applied to compute the new state<sup>1</sup> of the system. For the multi-fingered robot hand system, after setting up the equations of motion, new states of the fingers and of the object can be determined under the effects of gravitational force and joint torques. The general matrix form of the dynamic equations of a robotic system is:

$$M(q)\ddot{q} + N(q, \dot{q}) = \tau \quad (3.1)$$

Matrix  $M(q)$  is a square inertial matrix of the robotic system. This matrix is always invertible because it is symmetric and positive definite.  $N(q, \dot{q})$  contains the centrifugal, Coriolis, frictional, and gravitational terms of the dynamics. Vector  $q$  and  $\tau$  are the generalized coordinates and the generalized external inputs respectively. Rewriting equation (3.1) can obtain a second order differential equation as:

$$\ddot{q} = M(q)^{-1} [\tau - N(q, \dot{q})] \quad (3.2)$$

This second order differential equation ODE can be further decomposed into two first order equations such that the common ODE numerical integrator can be used to compute the new system state.

---

<sup>1</sup>State represents both location and velocity of a system.

Equation (3.2) shows us a closed form matrix equation to determine the acceleration of a general robotic system. However, due to the speed requirements of real-time control application, computational efficiency in determining the system acceleration is critical because of the limited computational power of digital computer in past. People always rewrite it in a recursive form for the ease of computation. The two most famous recursive methods (Featherstone, 1987; Lilly, 1993) are the *composite-rigid-body method* and the *articulated-body method*. The composite-rigid-body method was firstly proposed by Walker and Orin (1982). It is the most efficient recursive method available for calculating acceleration from the system dynamics in most practical cases when the degree of freedom of the system is less than nine. The main idea of this method is to obtain and to solve a set of simultaneous equations in the unknown system accelerations. In (Featherstone, 1983), the author developed an alternative recursive method, the articulated-body method which uses the spatial operators so that the motion and the force constraints can be obtained and propagated along the mechanical system. Since the arithmetic operations of this algorithm grows linearly with the number of degrees of freedom of bodies, it yields a better performance to simulate a complex dynamic system. Also, the idea used in articulated-body method can be extended to handle the non-serial mechanism and the closed-loop system.

Because the rapid development of digital computer, the computational efficiency in calculating acceleration directly from the system dynamics is greatly improved. Accordingly, the computational form of dynamics derived in this chapter is in closed form rather than in recursive form. This expression can be easier to help us to examine the dynamic behaviors of a system. There are three common methods used by the robotic scientists to formulate the dynamics for a multi-body system: *Newton-Euler formulation*, *Kane's formulation*, and *Lagrange's formulation*. The Newton-Euler method considers the rigid bodies in a multi-body system individually. Each of them has its dynamic equations to describe its motions. These individual dynamic equations are subject to  $n_h$  holonomic constraints and  $n_{nh}$  nonholonomic constraints of the system. This method is simple to setup the required equations; however, sparsity and large number of constraints needed to handle causes this method computational inefficient. On the other hand, Kane's formulation uses the idea of independent generalized coordinates<sup>2</sup> to embed both holonomic and nonholonomic constraints into the dynamics. It yields compact equations to describe the dynamic behaviors but change of constraint requires reformulation of the whole system. The alternative way to apply generalized coordi-

---

<sup>2</sup>Generalized coordinates are given to any set of quantities that completely describes the configuration of a multi-body system.



nates in formulation is the Lagrange's formulation method. It is the most common method used in formulating the dynamics of a robotic system and it can embed the holonomic constraints into the dynamic equations. Moreover, the constraint force can be determined by using the technique of Lagrangian multiplier. More detailed discussion of these methods and their comparisons can be referred to (Gillespie and Colgate, 1997).

In this chapter, the dynamics of the multi-fingered robot hand and the dynamics of the manipulated object are firstly derived. After that, they are integrated together with the constraint equations to form the Lagrange's dynamic equations. Since acceleration level constraint equations are required for the reason of stable numerical integration, the velocity constraints derived in last chapter are differentiated and modified. This modification requires the use of the Montana's contact equations in practice. The final section discusses load distribution problem covered in simulation.

## 3.2 Multi-fingered Robot Hand Dynamics

To derive the dynamic equations of a multi-fingered robot hand, the problem can be broken down to firstly find out the dynamics of each finger. Since in the first step, we assume that the hand is not manipulating an object as well as there are no interactions between fingers. The multi-fingered robot hand is then a solely open loop mechanism. We can derive the dynamics of each finger<sup>3</sup> individually and then integrate them together.

The finger dynamics can be described by a Lagrange's equation and the Lagrange's equations of motion for a conservative system is given by:

$$\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{\theta}} \right) - \frac{\partial L}{\partial \theta} = \tau \quad (3.3)$$

It is conservative because there is no energy loss under the rigid body model assumption. Vector  $\theta$  represents the generalized coordinates and they are the joint angles for the case of finger. Vector  $\tau$  represents the input joint torques, and the Lagrangian  $L$  is defined as the difference between the kinetic and the potential energies of the finger so:

$$L = K - P \quad (3.4)$$

Since a finger is actually a system of rigid bodies, the overall kinetic energy is the

<sup>3</sup>The detailed derivations of the finger dynamics are given in appendix B.

summation of all the rigid body kinetic energies as:

$$K = \sum_j \left( \frac{1}{2} m_j v_j^T v_j + \frac{1}{2} \omega_j I_j \omega_j \right) \quad (3.5)$$

where  $m_j$  is the mass of the  $j$ th rigid body, and  $I_j$  is its inertial tensor. Vectors  $v_j$  and  $\omega_j$  represent the translational and rotational velocities of the center of mass of the  $j$ th rigid body. The potential energy of the finger is only contributed by the gravitational effect and can be expressed as:

$$P = \sum_j (m_j h_j g) \quad (3.6)$$

where  $h_j$  is the height of the center of mass of the  $j$ th rigid body and the term  $g$  is the gravitational acceleration constant.

Using equation (3.3), the dynamics of a finger can be written in the form of:

$$M_{f_i}(\theta_i) \ddot{\theta}_i + N_{f_i}(\theta_i, \dot{\theta}_i) = \tau_i \quad (3.7)$$

Similar to equation (3.1),  $M_{f_i}(\theta_i) \in \mathbb{R}^{n \times n}$  is the inertial tensor of the  $i$ th finger where  $n$  is the number of link of a finger,  $N_{f_i}(\theta_i, \dot{\theta}_i) \in \mathbb{R}^n$  is a vector of gravity, Coriolis, centrifugal, and friction terms, and  $\tau_i \in \mathbb{R}^n$  is the vector of input joint torques.

If there are  $m$  fingers of a multi-fingered robot hand, equation (3.7) can be aggregated to give the dynamics of the multi-fingered robot hand.

$$M_h(\theta) \ddot{\theta} + N_h(\theta, \dot{\theta}) = \tau \quad (3.8)$$

where

$$M_h(\theta) = \begin{bmatrix} M_{f_1}(\theta_1) & 0 & \cdots & 0 \\ 0 & M_{f_2}(\theta_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & M_{f_m}(\theta_m) \end{bmatrix} \in \mathbb{R}^{mn \times mn}$$

$$N_h(\theta, \dot{\theta}) = \begin{bmatrix} N_{f_1}(\theta_1, \dot{\theta}_1) \\ N_{f_2}(\theta_2, \dot{\theta}_2) \\ \vdots \\ N_{f_m}(\theta_m, \dot{\theta}_m) \end{bmatrix} \in \mathbb{R}^{mn}$$

and the vector  $\tau$  is:

$$\tau = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \vdots \\ \tau_m \end{bmatrix} \in \mathbb{R}^{mn}$$

For this dynamic equation (equation (3.8)), there is no external or internal force acting on the system except the joint torques.

### 3.3 Object Dynamics

The manipulated object is assumed to be a rigid body. The Newton-Euler equations are used to derive object dynamics. The Newton equation emphasizes the dynamics of the translational motion of the object, and the Euler equation represents the dynamics of the rotation.

$$\begin{cases} \text{Newton equation:} & m_o \dot{v}_o = f_o \\ \text{Euler equation:} & {}^b I_o \dot{\omega}_o + \omega_o \times {}^b I_o \omega_o = \tau_o \end{cases} \quad (3.9)$$

Scalar  $m_o$  and matrix  ${}^b I_o$  represents the mass and the inertial tensor of the object. Since it is easier to determine the inertial tensor with respect to the object local frame, the inertial tensor with respect to the inertial frame can be calculated by:

$${}^b I_o = {}^b R {}^o I_o {}^b R^T$$

where  ${}^b R$  is the rotational matrix of the object coordinate frame with respect to the inertial coordinate frame. Vectors  $f_o$  and  $\tau_o$  are the external force and torque, and vectors  $v_o$  and  $\omega_o$  are the translational and rotational velocities of the object. These vectors are all referred to the inertial coordinate frame  $\Sigma_b$ .

If we need to determine the configuration and the motion of the manipulated object, the second order differential Newton-Euler equations are required to be integrated twice times. However, there is no physical meaning of the term  $\int \omega_o dt$ . Another variable which is capable to describe the orientation of the object should replace the angular velocity  $\omega_o$ . Since there is a linear relationship between the differential form of the local parameterization of the orientation and the angular velocity of the object, the angular



velocity in equation (3.9) is expressed as:

$$\omega_o = P(\gamma_o)\dot{\gamma}_o \quad (3.10)$$

where  $\gamma_o = [\alpha \ \beta \ \gamma]^T$  is a vector of ZYZ Euler angles<sup>4</sup> and  $P(\gamma_o)$ <sup>5</sup> is a function of the Euler angles:

$$P(\gamma_o) = \begin{bmatrix} 0 & -\sin \alpha & \cos \alpha \sin \beta \\ 0 & \cos \alpha & \sin \alpha \sin \beta \\ 1 & 0 & \cos \beta \end{bmatrix} \in \mathbb{R}^{3 \times 3}$$

As a result, the equations of motion of the object can be written more concise as:

$$M_o(\mathcal{X}_o)\ddot{\mathcal{X}}_o + N_o(\mathcal{X}_o, \dot{\mathcal{X}}_o) = F_o \quad (3.11)$$

where  $M_o(\mathcal{X}_o)$  is the inertial matrix, vectors  $N_o(\mathcal{X}_o, \dot{\mathcal{X}}_o)$  and  $F_o$  are the nonlinear terms and the external force acting on the object. They are in the forms of:

$$\begin{aligned} M_o(\mathcal{X}_o) &= \begin{bmatrix} m_o I_{3 \times 3} & 0_{3 \times 3} \\ 0_{3 \times 3} & {}^b I_o P \end{bmatrix} \in \mathbb{R}^{6 \times 6} \\ N_o(\mathcal{X}_o, \dot{\mathcal{X}}_o) &= \begin{bmatrix} 0_{3 \times 1} \\ {}^b I \dot{P} \dot{\gamma}_o + (P \dot{\gamma}_o) \times {}^b I (P \dot{\gamma}_o) \end{bmatrix} \in \mathbb{R}^6 \\ F_o &= \begin{bmatrix} f_o \\ \tau_o \end{bmatrix} \in \mathbb{R}^6 \\ \mathcal{X}_o &= \begin{bmatrix} x_o \\ \gamma_o \end{bmatrix} \in \mathbb{R}^6 \end{aligned}$$

Similar to the dynamics of the multi-fingered robot hand, the object dynamics shown above is assumed that there is no internal force acting on the object.

<sup>4</sup>For every orientation, it can be specified by three succeeding rotations about the principal axes in a certain order. ZYZ Euler angles represent the orientation that can be replied by rotating the  $z$ -axis by angle  $\alpha$ , rotating the  $y$ -axis by angle  $\beta$ , and rotating  $z$ -axis again by angle  $\gamma$ . It is not a unique Euler angles representation. The rest eleven Euler angles representations can be refer to (Craig, 1989, page 443).

<sup>5</sup>Derivation of matrix  $P(\gamma_o)$  can be referred to (Haug, 1992, page 210).

### 3.4 Constrained System Dynamics

In previous section, we have derived the multi-fingered robot hand dynamics and the object dynamics. In this section, we can combine these dynamics together with the velocity constraints derived in Chapter 2. Since the generalized velocity represented in velocity constraint is in the form of  $\begin{bmatrix} \dot{\theta} & v_o & \omega_o \end{bmatrix}^T$ , we need to modify the velocity constraint so that its generalized velocity is represented by  $\begin{bmatrix} \dot{\theta} & \dot{\mathcal{X}}_o \end{bmatrix}^T$ . The velocity constraint of the system becomes

$$\tilde{A}\dot{\tilde{q}} = 0 \quad (3.12)$$

The overall dynamics of the multi-fingered robot hand manipulation system is:

$$M_s(\tilde{q})\ddot{\tilde{q}} + N_s(\tilde{q}, \dot{\tilde{q}}) = F_s + \tilde{A}^T \lambda \quad (3.13)$$

Vector  $\tilde{q}$  represents the generalized coordinates of the system, which is equivalent to  $\begin{bmatrix} \theta \\ \mathcal{X}_o \end{bmatrix}$ . Matrix  $M_s(\tilde{q})$  is the inertial tensor of the whole system. The nonlinear terms are represented by vectors  $N_s(\tilde{q}, \dot{\tilde{q}})$ . Vectors  $F_s$  represents the external force including both joint torques and force acting on the object. The modified velocity constraint (equation (3.12)) is imposed to the dynamics by using the Lagrangian multiplier  $\lambda$  which actually represents the constraint force  $f_c$ . It also represents the internal force of the system. Matrix  $\tilde{A}$  transforms the effects of constraint force to the joint space of the system.  $M_s(\tilde{q})$ ,  $N_s(\tilde{q}, \dot{\tilde{q}})$ , and  $F_s$  are in the forms of:

$$\begin{aligned} M_s(\tilde{q}) &= \begin{bmatrix} M_h(\theta) & 0_{mn \times 6} \\ 0_{6 \times mn} & M_o(\mathcal{X}_o) \end{bmatrix} \in \mathfrak{R}^{(mn+6) \times (mn+6)} \\ N_s &= \begin{bmatrix} N_h(\theta, \dot{\theta}) \\ N_o(\mathcal{X}_o, \dot{\mathcal{X}}_o) \end{bmatrix} \in \mathfrak{R}^{mn+6} \\ F_s &= \begin{bmatrix} \tau \\ F_o \end{bmatrix} \in \mathfrak{R}^{mn+6} \end{aligned}$$

Therefore, the acceleration of the system can be determined as:

$$\ddot{\tilde{q}} = M_s^{-1}(F_s + \tilde{A}^T f_c - N_s) \quad (3.14)$$

All the terms on the right hand side of the equation (3.14) are known during simu-

lation except the constraint force. In order to determine the constraint force, we need to use the first derivative of the modified velocity constraint equation (3.12). Rather than differentiating the constraint directly, we are required to add one additional term as:

$$\tilde{A}\ddot{\tilde{q}} + \dot{\tilde{A}}\dot{\tilde{q}} + \alpha\tilde{A}\dot{\tilde{q}} = 0 \quad (3.15)$$

The main idea of this modification is similar to use the Baumgarte's constraint stabilization method (Baumgarte, 1972; de Jalón Javier and Eduardo, 1994; Chin, 1995). Direct integration of the system dynamics with velocity constraints is difficult and only in some special cases of it are being studied and solved (Brenan et al., 1989). It is also computational inefficient. As a result, people always reduce this dynamics with velocity constraints into simpler problem by imposing the differential velocity constraints into the dynamics. However, this reduction technique introduces the drift problem during integration. The numerical solutions of such reduced equations may be far away from the manifold defined by the original velocity constraints. The additional term in equation (3.15) function to make integration among the constraint manifold asymptotically stable and  $\alpha$  is a constant coefficient.

Putting the acceleration constraint equation (3.15) into the forward dynamic equation (3.14) and having some algebraic manipulations, we can determine the constraint force  $f_c$  as:

$$f_c = C^{-1}X \quad (3.16)$$

where

$$\begin{aligned} C &= \tilde{A}M_s^{-1}\tilde{A}^T \in \mathfrak{R}^{mn \times mn} \\ X &= \tilde{A}M_s^{-1}(N_s - F_s) - \dot{\tilde{A}}\dot{\tilde{q}} - \alpha\tilde{A}\dot{\tilde{q}} \in \mathfrak{R}^{mn \times 1} \end{aligned}$$

For this equation, we cannot ensure that the matrix  $C$  is non-singular because when there are free motion or sliding contacts, some its rows and columns become zeros and the term  $C$  becomes singular. This means we cannot determine its inverse in all cases. For this reason, the dimension of equation (3.16) is reduced as  $\bar{f}_c = \bar{C}^{-1}\bar{X}$  so that the zero rows and columns are subtracted for the cases of free motion and sliding contact, the constraint force can then be computed for the nonsingular reduced size  $\bar{C}$ . Finally, the constraint force  $f_c$  can be computed based on  $\bar{f}_c$ , that is substituted to the equation (3.14) to determine the acceleration of the system.



Finally, when a multi-fingered robot hand holds or manipulates an object, we must determine how hard each finger grasps on the object. If some of them support nearly all of weight of the object, other fingers do not need to exert so much forces. This problem is called the *load distribution* for the multi-fingered or multiple manipulators system. Since the constraint force  $f_c$  is determined by the inverse of the term  $\tilde{A}M_s^{-1}\tilde{A}^T$ , the payload is optimally distributed and we do not need to handle it manually.

### 3.5 Summary

The dynamics a multi-fingered robot hand system manipulating an object is discussed in this chapter. The robot hand dynamics and the object dynamics are derived independently and are integrated together with the acceleration constraint (equation (3.15)). The major equations are:

- Multi-fingered robot hand dynamics:

$$M_h(\theta)\ddot{\theta} + N_h(\theta, \dot{\theta}) = \tau \quad \text{Equation (3.8)}$$

- Object dynamics:

$$M_o(\mathcal{X}_o)\ddot{\mathcal{X}}_o + N_o(\mathcal{X}_o, \dot{\mathcal{X}}_o) = F_o \quad \text{Equation (3.11)}$$

- Dynamics of the whole system:

$$M_s\ddot{q} + N_s = F_s + \tilde{A}^T f_c \quad \text{Equation (3.14)}$$

Vector  $F_s$  corresponds to the external force and vector  $\tilde{A}^T f_c$  corresponds to the internal force of the system.

## Chapter 4

# Collision Modeling

### 4.1 Introduction

When two rigid body objects collide with each other, there are sudden changes of velocities but their configurations are kept unchanged. These discontinuous changes of velocities are due to a very large interaction force arisen during the collision. This problem has been well studied for over a century with various assumptions made to simplify the problem. For examples, there is no friction between the interaction of two bodies or the interaction is acting along the line jointing the centers of mass of the rigid bodies.

If the above exemplifying assumptions are released, the problem becomes complicated. In many classical mechanics textbooks (Greenwood, 1988), two dimensional impulsive collision problem is introduced to be handled by solving a system of algebraic equations. This method is simple and easy to be extended to solve the three dimensional problem. It is widely used in the computer science community (Hahn, 1988; Moore and Wilhelms, 1988). Treatments of three dimensional impulsive collision with friction using this method are thoroughly studied in (Brach, 1991), which assumes that the direction of the tangential relative velocity between colliding bodies at the collision point is kept constant during collision. As a result, the collision impulse and the post-collision velocities of the colliding bodies can be solved by using a system of algebraic equations.

However, the constant tangential relative velocity assumption used in above algebraic approach was proved to be invalid. This direction is kept changing during collision. Rather than using the algebraic equations to describe the motion, a more accurate collision model is achieved by using a set of differential equations to describe the collision

phenomenon. This idea was firstly introduced in (Routh, 1905) and was further studied by other dynamists (Keller, 1986; Bhatt and Koechling, 1995a,b). Based on these previous work in differential collision modeling, Mirtich (1996) in his Ph.D. work proposed an impulse-based simulation paradigm and developed a general-purpose dynamic simulator to model both rigid body dynamics and mechanism dynamics. His method has been applied to rigid body simulation for haptic display (Chang and Colgate, 1997; Brown and Colgate, 1997), micro-electro-mechanical-system (MEMS) device simulation (Zhuang, 1996), and biped walking robot simulation (Wendlandt, 1997).

In this chapter, we present the impulsive collision modeling equations based on Mirtich's work. The major differences between Mirtich's work and the work included in this chapter are:

- The original work of impulse-based method uses repeated impulse to model all interactions between objects, including both contacts and collisions. The impulse-based method is suitable for simulating the impulsive collision but is not natural to simulate continuous contacts. Mirtich has proposed an ad hoc method to model contact. However, as he suggests, it is more appropriate to model continuous contacts by other methods. We use the impulse-based method to model the impulsive collision only. The contact motion is handled by the constraint-based method discussed in last two chapters.
- Mirtich has discussed the multi-body mechanism collision modeling but some of his expressions are incapable to model the multi-fingered robot hand manipulation motion<sup>1</sup>. Modifications are made in our derivations for our specific system.
- Mirtich used the Stronge's hypothesis (Stronge, 1991) to model the restitution during collision. This is different from Poisson's hypothesis in his previous papers (Mirtich and Canny, 1995a,b). Although using Stronge's hypothesis can improve the accuracy of the simulation, Poisson's hypothesis is more widely used and its formulation is simpler. We give a detailed formulation using the Poisson's restitution model.

In this chapter, the velocities at the collision point on the surfaces of the finger and the object are firstly derived. Under the rigid body assumption, the configurations of the system during impulsive collision are kept constant, and only the velocities are various. We can determine the impulse to find out the post-collision velocities of the

---

<sup>1</sup>This incapability is verified in the personal communication and additional assumptions are required to add in his original work in order to apply it in our multi-fingered robot hand system.



colliding finger and the object. However, it is more convenient to determine the velocity change of collision point first and then to compute those of velocities of the finger and the object. Next, the first order differential collision equation is derived, which is used to describe the process of impulsive collision. Integration of this equation can help us to track through the change of collision point velocities during collision because the collision equation can have some specific forms for different collision conditions. Detailed discussions of different modes of collision integrations in different collision phase are placed at the end of this chapter.

## 4.2 Assumptions of Collision

Before the derivation of relevant equations for the collision, there are three assumptions:

1. **Rigidity:** All physical objects or components of a multi-body system in the simulation are perfectly rigid.
2. **Poisson's hypothesis:** Figure 4.1 shows how the impulse changes during the compression phase and the restitution phase of a collision. Let  $P_n(\gamma_f)$  be the magnitude of the normal component of the impulse imparted by one object onto the other over the entire collision and  $\gamma$  be the generalized *collision parameter*. Let  $P_n(\gamma_{mc})$  be the magnitude of the normal component of the impulse imparted by one object onto the other up to the point of maximum compression. Then,

$$P_n(\gamma_f) = (1 + e)P_n(\gamma_{mc}) \quad (4.1)$$

where  $e$  is a constant between zero and one, dependent on the objects' materials, and is called the *coefficient of restitution*.

3. **Coulomb friction law:** At some instant during a collision between bodies  $i$  and  $j$ , let  ${}^j u_i$  be the collision point velocity of body  $i$  relative to the collision point velocity of body  $j$ . Let  ${}^j u_{it}$  be the tangential component of  ${}^j u_i$ , and let  ${}^j \hat{u}_{it}$  be a unit vector in the direction of  ${}^j u_{it}$ . Let  ${}^j f_{in}$  and  ${}^j f_{it}$  be the normal and tangential components of force exerted by body  $j$  on body  $i$ , respectively. Then,

$$\begin{cases} {}^j f_{it} = -\mu \|{}^j f_{in}\| {}^j \hat{u}_{it} & \text{if } {}^j u_{it} \neq 0 \\ \|{}^j f_{it}\| \leq \mu \|{}^j f_{in}\| & \text{if } {}^j u_{it} = 0 \end{cases} \quad (4.2)$$

where  $\mu$  is the *coefficient of friction*.

The rigidity assumption can ensure that the collision time is infinitesimal and the impulse only instantaneously affects the velocities but not the configurations. The restitution of collision is described by the Poisson's hypothesis, which represents the energy lost effect of the collision. The final assumption is a widely applied friction law to model any frictional interaction.

### 4.3 Collision Point Velocities

#### 4.3.1 Collision Point Velocity of the $i$ th Finger

The coordinate frames assignment in Figure 4.2 is similar to that shown in Figure 2.7. Since the configuration of the system is assumed to be kept constant and there is no relative motion between the two colliding bodies. Coordinate frames  $\Sigma_{li}$  and  $\Sigma_{oi}$  used to describe relative motions in Figure 2.7 are replaced by a collision coordinate frame  $\Sigma_c$ . Vector  $p_{fi}$  and  $p_o$  are the position vectors of the collision point with respect to coordinate frames  $\Sigma_{fi}$  and  $\Sigma_o$  respectively, and vector  $P$  is the corresponding impulse. These vectors are all with respect to the collision coordinate frame  $\Sigma_c$ . The dynamics

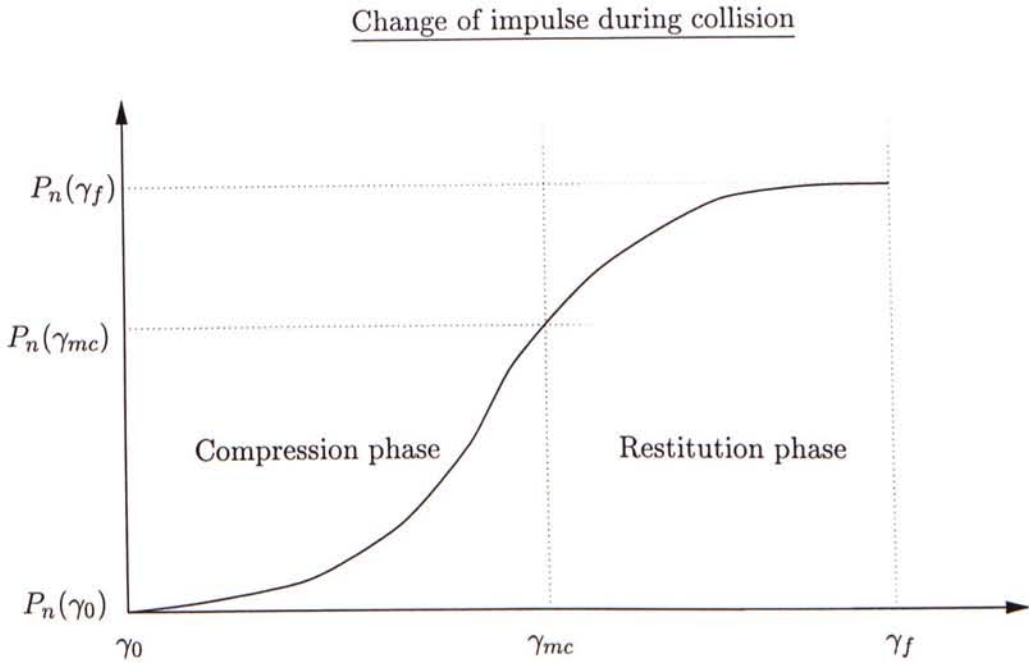


Figure 4.1: Change of the normal component of collision impulse  $P$  in the compression phase and restitution phase of a collision.

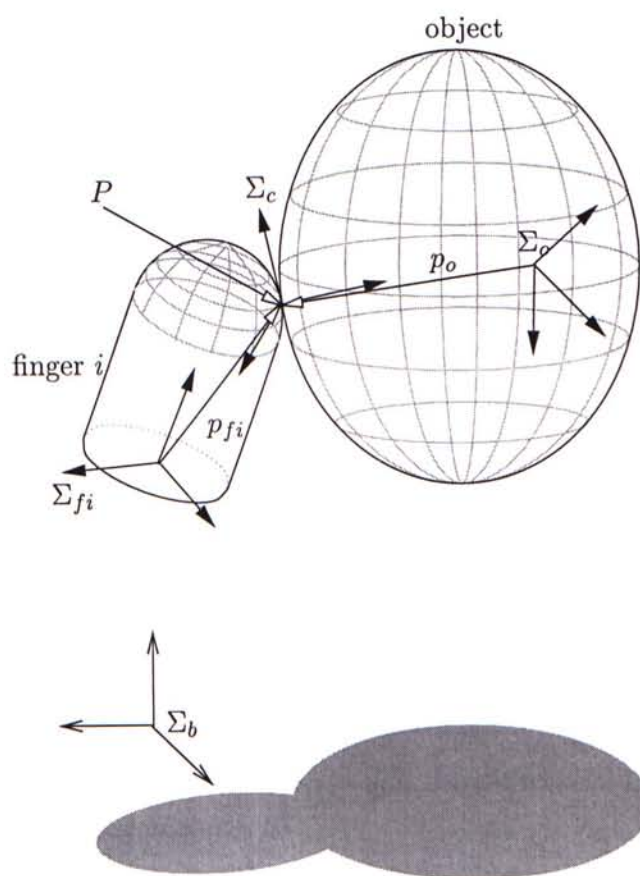


Figure 4.2: Coordinate frame assignment for the colliding  $i$ th finger and the object.



of the  $i$ th finger is:

$$M_{fi}\ddot{\theta}_i + N_{fi} = \tau_i + J_i^T F_c \quad (4.3)$$

This dynamics is similar to the one derived in equation (3.7). The additional term  $J_i^T F_c$  is to describe the external collision force toward the dynamics of the finger.  $F_c$  is a  $3 \times 1$  vector since only the translational force can be applied at the collision point. As a result, the Jacobian  $J_i$  used here is slightly different from the one in equation (2.8). This Jacobian is a  $3 \times 3$  square matrix which acts as the transformation from the joint space to the collision coordinate frame. On the other hand, the finger Jacobian shown in equation (2.8) transforms any vector from joint space to Cartesian space so it is  $6 \times 3$  matrix. Because the whole collision process is examined at the collision point so choosing the collision coordinate frame as the reference frame is more convenient. All the equations derived below are with respect to this coordinate frame.

Since the vectors  $N_{fi}$  and  $\tau_i$  are much smaller than  $J_i^T F_c$  during collision, they become negligible. Integrating equation (4.3) through time results in an expression for the change of joint velocities  $\delta\dot{\theta}_i(t)$ :

$$\delta\dot{\theta}_i(t) = M_{fi}^{-1} J_i^T P(t) \quad (4.4)$$

Since velocity becomes discontinuous function of time during collision under the rigid body assumption, we cannot use it as the collision equation parameter. Thus, time domain is transformed to the  $\gamma$  domain in equation (4.4), where  $\gamma$  is the generalized collision parameter which monotonically increases during the course of collision.

On the other hand, the velocity at the collision point  $u_{fi}$  and the joint velocities  $\dot{\theta}_i$  are related by using the Jacobian  $J_i$  as  $u_{fi} = J_i \dot{\theta}_i$ . Its differential form with respect to the generalized collision parameter  $\gamma$  is:

$$\delta u_{fi}(\gamma) = J_i \delta\dot{\theta}_i(\gamma) \quad (4.5)$$

Combining equations (4.4) and (4.5), the relationship between the impulse and the change of collision point velocity of the  $i$ th fingertip can be obtained as:

$$\delta u_{fi}(\gamma) = K_{fi} P(\gamma) \quad (4.6)$$

where the term  $K_{fi}$  is

$$K_{fi} = J_i M_{fi}^{-1} J_i^T \quad (4.7)$$

which is a positive definite and symmetric matrix.

### 4.3.2 Collision Point Velocity of the Object

The object dynamics can be represented by the Newton-Euler equations (shown in equation (3.9)) and they are rewritten as the forms of:

$$\begin{cases} \text{Newton equation:} & -F_c(t) = m_o \dot{v}_o(t) \\ \text{Euler equation:} & p_o \times (-F_c(t)) = I_o \dot{\omega}_o(t) + \omega_o(t) \times I_o \omega_o(t) \end{cases} \quad (4.8)$$

Since the inertial forces can also be neglected during collision, integration of the above equations through time yields the integral Newton-Euler equations:

$$\begin{cases} \text{Integral Newton equation:} & -P(t) = m_o \delta v_o(t) \\ \text{Integral Euler equation:} & -p_o \times P(t) = I_o \delta \omega_o(t) \end{cases} \quad (4.9)$$

Similar to the case of determining the collision point velocity of the  $i$ th finger, time domain is transformed to the  $\gamma$  domain and equations (4.9) are rewritten as:

$$\begin{cases} \delta v_o(\gamma) & = -\frac{1}{m_o} P(\gamma) \\ \delta \omega_o(\gamma) & = -I_o^{-1} p_o \times P(\gamma) \end{cases} \quad (4.10)$$

Since the collision point velocity of the object can be determined by the object's translational and rotational velocities as  $u_o(\gamma) = v_o(\gamma) + \omega_o(\gamma) \times p_o$ , its differential form is:

$$\delta u_o(\gamma) = \delta v_o(\gamma) + \delta \omega_o(\gamma) \times p_o \quad (4.11)$$

Substituting equation (4.10) into the above equation, we can obtain the relationship of change of collision point velocity of the object and the impulse. It is:

$$\delta u_o(\gamma) = -K_o P(\gamma) \quad (4.12)$$

where matrix  $K_o$  is equal to

$$\frac{1}{m_o} I_{3 \times 3} - \tilde{p}_o I_o^{-1} \tilde{p}_o \quad (4.13)$$

Matrix  $\tilde{p}_o$  is a  $3 \times 3$  skew-symmetric matrix defined as:

$$\tilde{p}_o = \begin{bmatrix} 0 & -p_{oz} & p_{oy} \\ p_{oz} & 0 & -p_{ox} \\ -p_{oy} & p_{ox} & 0 \end{bmatrix}$$

where  $p_{ox}$ ,  $p_{oy}$ , and  $p_{oz}$  are the elements of vector  $p_o$ .

### 4.3.3 Relative Collision Point Velocity

Assuming  $u(\gamma)$  be the relative collision point velocity between the  $i$ th finger and the object, it is equal to  $u(\gamma) = u_{fi}(\gamma) - u_o(\gamma)$  and its differential form with respect to  $\gamma$  is:

$$\delta u(\gamma) = \delta u_{fi}(\gamma) - \delta u_o(\gamma)$$

Putting the collision point velocity of the  $i$ th finger and the collision point velocity of the object (equations (4.6) and (4.12)) into the above equation, we can obtain the relationship between the change of relative collision point velocity and the impulse as:

$$\delta u(\gamma) = KP(\gamma) \quad (4.14)$$

where  $K = K_{fi} + K_o$  is called the *collision matrix*.

## 4.4 Equations of Collision

In previous section, we have derived the relationship between the change of relative collision point velocity and the impulse caused during the collision shown in equation (4.14). The equations of motion which describe this collision process can be obtained by differentiating this equation (4.14). Since the collision matrix is constant with respect to the collision parameter  $\gamma$ , the equation of collision is:

$$\frac{d}{d\gamma} u(\gamma) = K \frac{d}{d\gamma} P(\gamma) \quad (4.15)$$



The above equation is the general form of the collision response. With different collision conditions such as different friction between the colliding objects, the general equation can be modified for the ease of determining the post-collision velocities according to two cases: *sliding mode collision* and *sticking mode collision*.

#### 4.4.1 Sliding Mode Collision

When the tangential relative collision point velocities are nonzero, there is sliding between the colliding finger and the object. When the normal component of impulse  $P_z$  is used to be the collision parameter, it is possible to represent the first derivative of impulse  $\frac{d}{dP_z}P$  in term of the relative collision point velocity as:

$$\frac{d}{dP_z}P = \xi(\theta) \quad (4.16)$$

where

$$\xi(\theta) = \begin{bmatrix} -\mu \frac{u_x}{\sqrt{u_x^2 + u_y^2}} \\ -\mu \frac{u_y}{\sqrt{u_x^2 + u_y^2}} \\ 1 \end{bmatrix} \in \mathfrak{R}^{3 \times 1}$$

Angle  $\theta$  is equivalent to the relative sliding direction at some point during a collision. Scalars  $u_x$ ,  $u_y$ , and  $u_z$  are the  $x$ ,  $y$ , and  $z$  components of the relative collision point velocities  $u$ , and  $\mu$  is the friction coefficient. This equation can be deduced from the Coulomb friction model assumption. As a result, the general form of the collision equation (4.15) can be specified as:

$$\frac{d}{dP_z}u = K\xi(\theta) \quad (4.17)$$

Figure 4.1 shows that there are two phases in a course of collision. The first one is the compression phase and the other is the restitution phase. For the compression phase, the termination condition is much easier to notify if we use the normal component of the relative collision point velocity  $u_z$  as the collision parameter  $\gamma$ . Since at the end of compression phase, the deformation of the colliding objects are finished and there is no relative motion between the colliding objects so  $u_z$  should be equal to zero. Assume that the collision matrix  $K$  is composed if  $k_{i,j}$ . Notation  $k_{2,j}$  represents the second row of matrix  $K$ . Similarly, notation  $k_{i,3}$  represent the third column of matrix  $K$  and

$k_{2,3}$  represents the element in the second row, third column of matrix  $K$ . Using this representation, equation (4.17) can be represented by three equations:

$$\frac{du_x}{dP_z} = k_{1,j}\xi(\theta) \quad (4.18a)$$

$$\frac{du_y}{dP_z} = k_{2,j}\xi(\theta) \quad (4.18b)$$

$$\frac{du_z}{dP_z} = k_{3,j}\xi(\theta) \quad (4.18c)$$

Using these three equations, the following two equations can be obtained:

$$\frac{du_x}{du_z} = \frac{k_{1,j}\xi(\theta)}{k_{3,j}\xi(\theta)}$$

$$\frac{du_y}{du_z} = \frac{k_{2,j}\xi(\theta)}{k_{3,j}\xi(\theta)}$$

Because  $u_z$  is the independent variable in these equations, we would not determine  $\frac{du_x}{du_z}$ . Instead, we determine the value of  $P_z$  at the end of compression phase. This value is useful in determining the  $P_z$  at the end of collision based on the Poisson's hypothesis equation (4.1). Using equation (4.18c), we get:

$$\frac{dP_z}{du_z} = \frac{1}{k_{3,j}\xi(\theta)}$$

Therefore, the sliding mode collision equation in compression phase of the collision process can be concluded as:

$$\frac{1}{du_z} \begin{bmatrix} u_x \\ u_y \\ P_z \end{bmatrix} = \frac{1}{k_{3,j}\xi(\theta)} \begin{bmatrix} k_{1,j}\xi(\theta) \\ k_{2,j}\xi(\theta) \\ 1 \end{bmatrix} \quad (4.19)$$

#### 4.4.2 Sticking Mode Collision

Rather than the sliding mode in the course of collision, the tangential relative collision point velocities can be zero if friction is large enough to maintain the contact. It is the case of sticking mode. Sticking means that there is no relative motion between the colliding bodies. In this case, both first derivatives of the relative collision point tangential velocities vanish,  $\frac{du_x}{dP_z} = \frac{du_y}{dP_z} = 0$ , in equation (4.17); therefore, equation (4.17)

can be written as:

$$\begin{bmatrix} 0 \\ 0 \\ \dot{u}_z \end{bmatrix} = K \frac{d}{dP_z} P$$

where the term  $\frac{d}{dP_z} P$  can be expressed as

$$\frac{d}{dP_z} P = \begin{bmatrix} k_{1,3}^{-1} \\ k_{2,3}^{-1} \\ k_{3,3}^{-1} \end{bmatrix} \dot{u}_z \quad (4.20)$$

where  $k_{i,j}^{-1}$  denotes the  $i$ th row,  $j$ th column element of the inverse of matrix  $K$ . Because the sticking collision may be unstable, there is a test derived in (Mirtich, 1996) to verify the stability of the sticking mode. If the sticking stability test is satisfied, it means friction between the two colliding bodies is sufficient to maintain sticking collision. This situation is called the *stable sticking mode*. On the other hand, if the test fails, sliding immediately resumes because of the insufficient friction and this situation is called *transient sticking mode*. No matter in which sticking mode, equations (4.17) or (4.19) cannot be used any more. We must use another method to determine the information of the collision process in order to determine the post-collision velocities.

### Stable Sticking Mode Collision

When sticking occurs during the compression phase, the value of  $P_z$  at the end of this phase is required to determine. By equation (4.20), we have the equation  $\frac{dP_z}{du_z} = k_{3,3}^{-1}$ , the solution of this first order differential equation can be obtained as:

$$P_z(\gamma_{mc}) = P_z(\gamma_{sticking}) + k_{3,3}^{-1}(\gamma_{mc} - \gamma_{sticking}) \quad (4.21)$$

Scalar  $\gamma_{sticking}$  represents the value of collision parameter at the instant when sticking happens. Equation (4.21) can help us to compute the value of  $P_z$  at the end of restitution phase. Since when the sticking is maintained, velocities  $u_x$  and  $u_y$  at the end of the restitution phase are still zero. Using the relationship that  $\frac{du_z}{dP_z} = \frac{1}{k_{3,3}^{-1}}$ , the velocity  $u_z(\gamma_f)$  can be calculated by the following equation:

$$u_z(\gamma_f) = \frac{1}{k_{3,3}^{-1}}(\gamma_f - \gamma_{mc}) \quad (4.22)$$



Similarly, when the sticking occurs during the restitution phase, we only need to determine  $u_z(\gamma_f)$  since we have the information of  $u_x(\gamma_f)$  and  $u_y(\gamma_f)$  that they are equal to zero. By using the relationship that  $\frac{du_x}{dP_z} = \frac{1}{k_{3,3}^{-1}}$ , the solution of this first order differential equation is:

$$u_z(\gamma_f) = u_z(\gamma_{sticking}) + \frac{1}{k_{3,3}^{-1}}(\gamma_f - \gamma_{sticking}) \quad (4.23)$$

### Transient Sticking Mode Collision

When the sticking is unstable, it is proved that there is exactly one diverging ray along which sliding may resume (Mirtich, 1996). Since it is also proved that the sliding direction is being maintained constant after the transient sticking, the information about the collision can be determined similarly to the method used in stable sticking mode collision. Let  $\beta$  be the resumed direction of the unique diverging ray, if the transient sticking occurs during the compression phase, we have:

$$u_x(\gamma_{mc}) = u_x(\gamma_{sticking}) + \frac{k_{1,j}\xi(\beta)}{k_{3,j}\xi(\beta)}(\gamma_{mc} - \gamma_{sticking}) \quad (4.24a)$$

$$u_y(\gamma_{mc}) = u_y(\gamma_{sticking}) + \frac{k_{2,j}\xi(\beta)}{k_{3,j}\xi(\beta)}(\gamma_{mc} - \gamma_{sticking}) \quad (4.24b)$$

$$P_z(\gamma_{mc}) = P_z(\gamma_{sticking}) + \frac{1}{k_{3,j}\xi(\beta)}(\gamma_{mc} - \gamma_{sticking}) \quad (4.24c)$$

since  $\frac{du_x}{du_z} = \frac{k_{1,j}\xi(\beta)}{k_{3,j}\xi(\beta)}$ ,  $\frac{du_y}{du_z} = \frac{k_{2,j}\xi(\beta)}{k_{3,j}\xi(\beta)}$ , and  $\frac{dP_z}{du_z} = \frac{1}{k_{3,j}\xi(\beta)}$ . These equations (equations (4.24a) to (4.24c)) can help us to determine the post-collision velocities of the system.

On the other hand, if the transient sticking occurs during the restitution phase, the post-collision velocities can be directly obtained by:

$$u_x(\gamma_f) = u_x(\gamma_{sticking}) + k_{1,j}\xi(\beta)(\gamma_f - \gamma_{sticking}) \quad (4.25a)$$

$$u_y(\gamma_f) = u_y(\gamma_{sticking}) + k_{2,j}\xi(\beta)(\gamma_f - \gamma_{sticking}) \quad (4.25b)$$

$$u_z(\gamma_f) = u_z(\gamma_{sticking}) + k_{3,j}\xi(\beta)(\gamma_f - \gamma_{sticking}) \quad (4.25c)$$

## 4.5 Summary

Related work of collision modeling is briefly reviewed at the beginning of this chapter. Based on the work of Mirtich (1996), the modified general first order differential colli-

sion equation is obtained after determining the relative collision point velocities. The collision equations have the form of:

$$\frac{d}{d\gamma}u(\gamma) = K \frac{d}{d\gamma}P(\gamma) \quad \text{Equation (4.15)}$$

This equation provides us a simple and efficient way to handle three dimensional collision with friction occurred in multi-fingered robot hand manipulation motions. The specific forms of this collision equation in different collision modes are further discussed at the last part of this chapter and the detailed implementation of the derived equation are covered in next chapter.

## Chapter 5

# Dynamic Simulation

### 5.1 Introduction

The last three chapters discussed modelings of different interactions presented in the dextrous manipulation motions respectively. The derived equations form the foundation to develop a dynamic simulation system to simulate the grasping or manipulation motions of the multi-fingered robot hand. In this chapter, these generalized dynamic models are applied to develop a dynamic simulation system for the five-fingered robot hand system developed in CUHK.

The traditional robotic dynamic simulators only provide an one way control of the simulation. After the setting the initial conditions and the system parameters, simulation starts and some numerical or graphical results will then be obtained. In our development, we apply the knowledge of human control, computer graphics, and virtual reality to construct the dynamic simulator that allows the simulation done in an interactive way (Figure 5.1). With the feedback information, the human operator can on-line update the simulation parameters.

In this chapter, the details of this dynamic simulation system are presented. The first part of this chapter will cover the architecture of the whole simulation system and the descriptions of its main components. The detailed methodologies used in the dynamic simulator and its program flow will be discussed in the second part. For convenient, we use the term *dynamic simulation system* to describe the entire simulation system and the term *dynamic simulator* to describe the specific module in the system to handle all the numerical computation of the dynamics. The simulation results will be demonstrated in Chapter 6.





Figure 5.1: Interactive control of the dynamic simulation system.

## 5.2 Architecture of the Dynamic Simulation System

The architecture of the dynamic simulation system for the five-fingered robot hand is summarized in Figure 5.2. Except the human operator, there are three main components of the system. Both data glove and graphical control panel are the input devices. The data glove helps us to collect the information of the human dextrous motions measured from the operator. The graphical control panel provides the user some basic commands to control the simulation. The second component is the dynamic simulator, which is the core module. Based on the equations derived in previous contact and collision modelings, the dynamics of the whole system including both the multi-fingered robot hand dynamics and the object dynamics are determined to examine the dynamic behaviors of the system. The output of the dynamic simulator is only some numerical results of the system dynamics. These numerical data are inputted to the virtual environment to provide a realistic three dimensional graphical output. As a result, the graphical results of the dynamic simulation shown in the virtual environment can act as the visual feedback information to the human operator to control and to adjust his dextrous motions.

### 5.2.1 Input Devices

#### Data Glove

The data glove used in the development is the *CyberGlove<sup>TM</sup>* shown in Figure 5.3

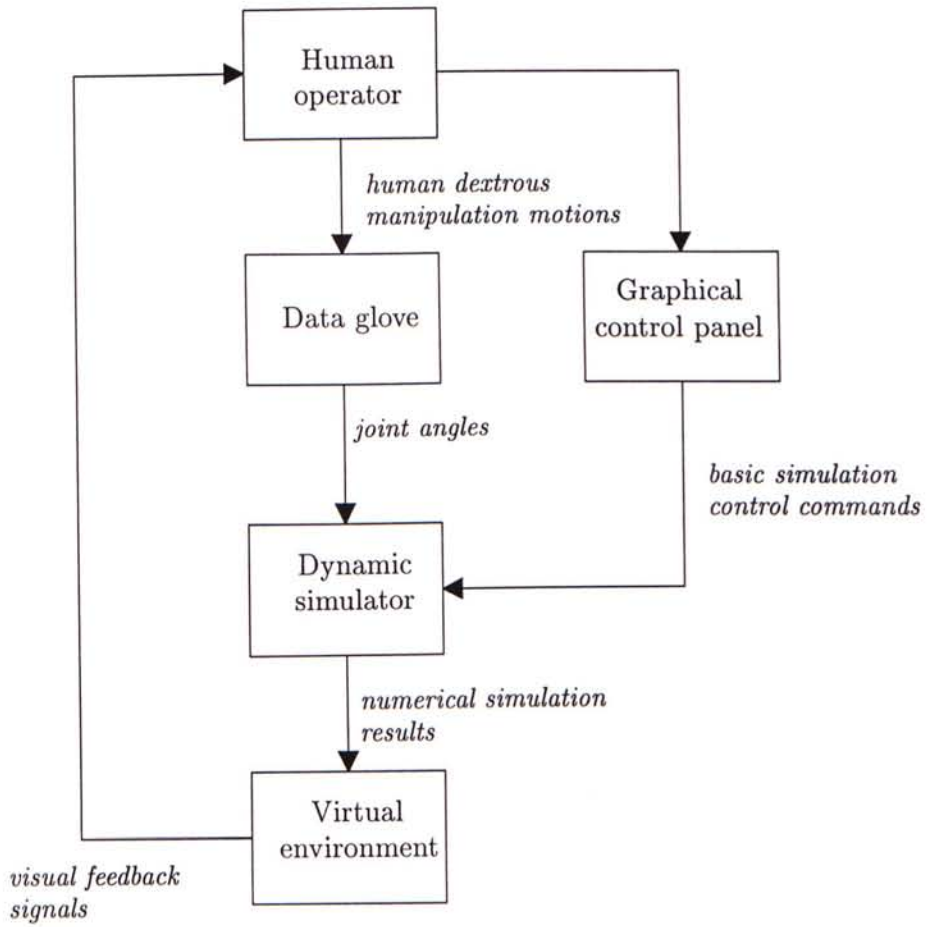


Figure 5.2: Architecture of the dynamic simulation system for the five-fingered robot hand.

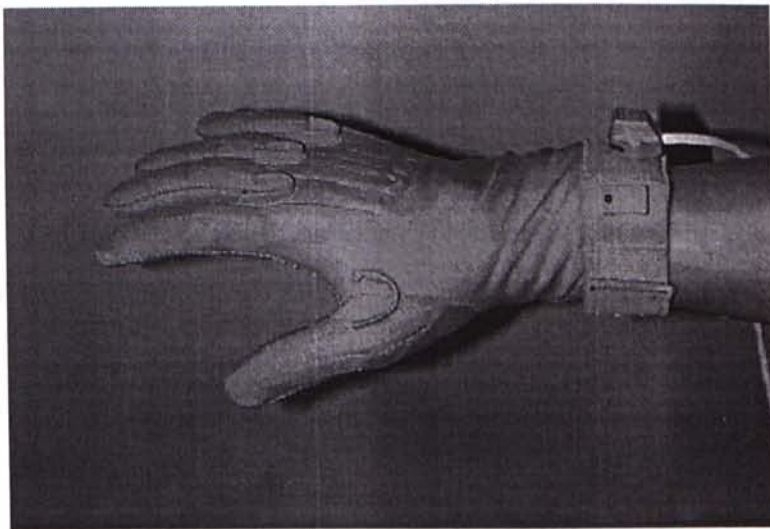


Figure 5.3: *CyberGlove™* used in the dynamic simulator as the input device.



which is a product of the Virtual Technologies Inc<sup>1</sup>. This is a light weight glove composed of flexible sensors to measure the configurations of both operator's fingers and wrist. This data glove is capable to measure the joint angles of the human fingers but not the joint torques of them. In this case, a position control algorithm must be implemented in the dynamic simulation system in order to obtain the joint torques for simulation inputs. The most common position control method is the proportional-plus-derivative (PD) feedback control (Lewis et al., 1993). It is simple, widely used in industry and proved to be stable. However, this method does not consider the effect of gravitation and there is always a steady state error. The improved PD control is applied in our dynamic simulator, which is called the PD plus gravity compensation control method. The gravitational effect is firstly cancelled by a dynamics related term before doing the PD control. Its mathematical expression is:

$$\tau_i = K_{pi}(\theta_{di} - \theta_i) - K_{vi}\dot{\theta}_i + G_i(\theta_i); \quad (5.1)$$

Matrices  $K_{pi}$  and  $K_{vi}$  are the proportional and derivative gains of the control algorithm and both of them are diagonal matrices. They determine the convergence rate of the control. The final term  $G_i(\theta_i)$  is the gravitational terms contained in the finger dynamics. Actually, it is embedded in the term  $N_{fi}(\theta_i, \dot{\theta}_i)$  in equation (3.7). Vectors  $\theta_i$  and  $\dot{\theta}_i$  are the current joint angles and joint velocities in the simulation. Vector  $\theta_{di}$  is the desired joint angles which actually are the angles measured by the data glove. Vector  $\tau_i$  is the corresponding joint torques which is the input data of the dynamic simulator.

Due to the kinematic dissimilarities between the human hand and the five-fingered robot hand, another problem in using the data glove is that of mapping human hand motions into multi-fingered robot hand motions. Different mapping methods have been proposed in the literatures (Hashimoto et al., 1994; Wright and Stanisic, 1990; Speeter, 1992; Rohling et al., 1993). Their aim is to obtain an exact mapping between the human hand motions and the robot hand motions. For example, when the thumb and index are touched to show the victory gesture, there should be a similar interaction between the thumb and the index of the robot hand. However, it is not the main objective in this dissertation to study this mapping problem. A good mapping can help us to control the virtual robot hand in the virtual environment more easily but it does not affect the physical accuracy of the dynamic simulation system. As a result, a simple mapping method is implemented in our system. Since there 22 sensors for the data

<sup>1</sup>Home page: <http://www.virtex.com>



glove but there are 15 joints for the five-fingered robot hand (3 joints for each finger), not all sensory data are required to choose to control the robot hand. After comparing the structure of the data glove and that of the five-fingered hand, The 14 sensory data shown in Table 5.1<sup>2</sup> are used and the simple mapping method is summarized in Table 5.2.

Finger (glove)	Joint(glove)	Sensor Name (Description)
thumb, 0	1	thumb MPJ, $S_{0,1}$ (joint where the thumb meets the palm)
thumb, 0	2	thumb IJ, $S_{0,2}$ (outer thumb joint)
thumb, 0	3	thumb abduction, $S_{0,3}$ (angle between thumb and index finger)
index, 1	0	index MPJ, $S_{1,0}$ (joint where the index meets the palm)
index, 1	1	index PIJ, $S_{1,1}$ (joint second from fingertip)
middle, 2	0	middle MPJ, $S_{2,0}$
middle, 2	1	middle PIJ, $S_{2,1}$
middle, 2	3	middle-index abduction, $S_{2,3}$ (angle between middle and index fingers)
ring, 3	0	ring MPJ, $S_{3,0}$
ring, 3	1	ring PIJ, $S_{3,1}$
ring, 3	3	ring-middle abduction, $S_{3,3}$ (angle between ring and middle fingers)
pinkie, 4	0	pinkie MPJ, $S_{4,0}$
pinkie, 4	1	pinkie PIJ, $S_{4,1}$
pinkie, 4	3	pinkie-ring abduction, $S_{4,3}$ (angle between pinkie and ring fingers)

Table 5.1: 14 sensory data used in the simulation system.

### Graphical Control Panel

Another input of the dynamic simulator is the graphical control panel. It acts as the graphical user interface (GUI) of the system. Its screen shot is shown in Figure 5.4. This control panel provides the basic controls of the simulation such as **START**, **PAUSE**, and **TERMINATE**. Also, using the dialers and sliders can help the human operator to

<sup>2</sup>MPJ stands for the metacarpophalangeal joint, PIJ is the proximal interphalangeal joint, DIJ is the distal interphalangeal joint, TMJ is the trapeziometacarpal joint.

Finger (robot hand)	Joint (robot hand)	Angle
thumb, 0	1 (nearest to the base)	$-S_{0,3}/2$
thumb, 0	2	$S_{0,1}$
thumb, 0	3	$S_{0,2}$
index, 1	1	$-(S_{2,3} - S_{0,3})/2$
index, 1	2	$S_{1,0}$
index, 1	3	$S_{1,1}$
middle, 2	1	$-(S_{3,3} - S_{2,3})/2$
middle, 2	2	$S_{2,0}$
middle, 2	3	$S_{2,1}$
ring, 3	1	$-(S_{4,3} - S_{3,3})/2$
ring, 3	2	$S_{3,0}$
ring, 3	3	$S_{3,1}$
pinkie, 4	1	$S_{4,3}/2$
pinkie, 4	2	$S_{4,0}$
pinkie, 4	3	$S_{4,1}$

Table 5.2: Mapping between data glove and five-fingered robot hand.

specify the initial configuration of the system before starting the simulation. The recording utilities and the mechanism to modify the specifications of the system help us to examine different simulation situations. This graphical control panel was mainly constructed by using the *Forms library* (Overmars, 1995). It provides a rich set of subroutines to construct GUI in the *Silicon Graphics Workstation*.

### 5.2.2 Dynamic Simulator

As mentioned in the previous sections, the dynamic simulator is the nucleus of the dynamic simulation system. Based on the equations formulated in Chapters 2, 3, and 4, we can accurately describe the dynamic behaviors of the multi-fingered robot hand motions with consideration of different type of interactions between fingers and object. In the contact modeling, the constraint-based formulation is actually used to develop the relevant kinematic and dynamic equations. Otherwise, in the collision modeling, the impulse-based formulation is applied to formulate the first order differential collision equation to describe the collision process. Combining the constraint-based simulation and the impulse-based simulation can help us to examine the free motion, collision, sticking contact, rolling and sliding contacts of the system. The detailed descriptions of dynamic simulator will be discussed at the second part of this chapter.

The dynamic simulation system is developed in the platform of Silicon Graphics



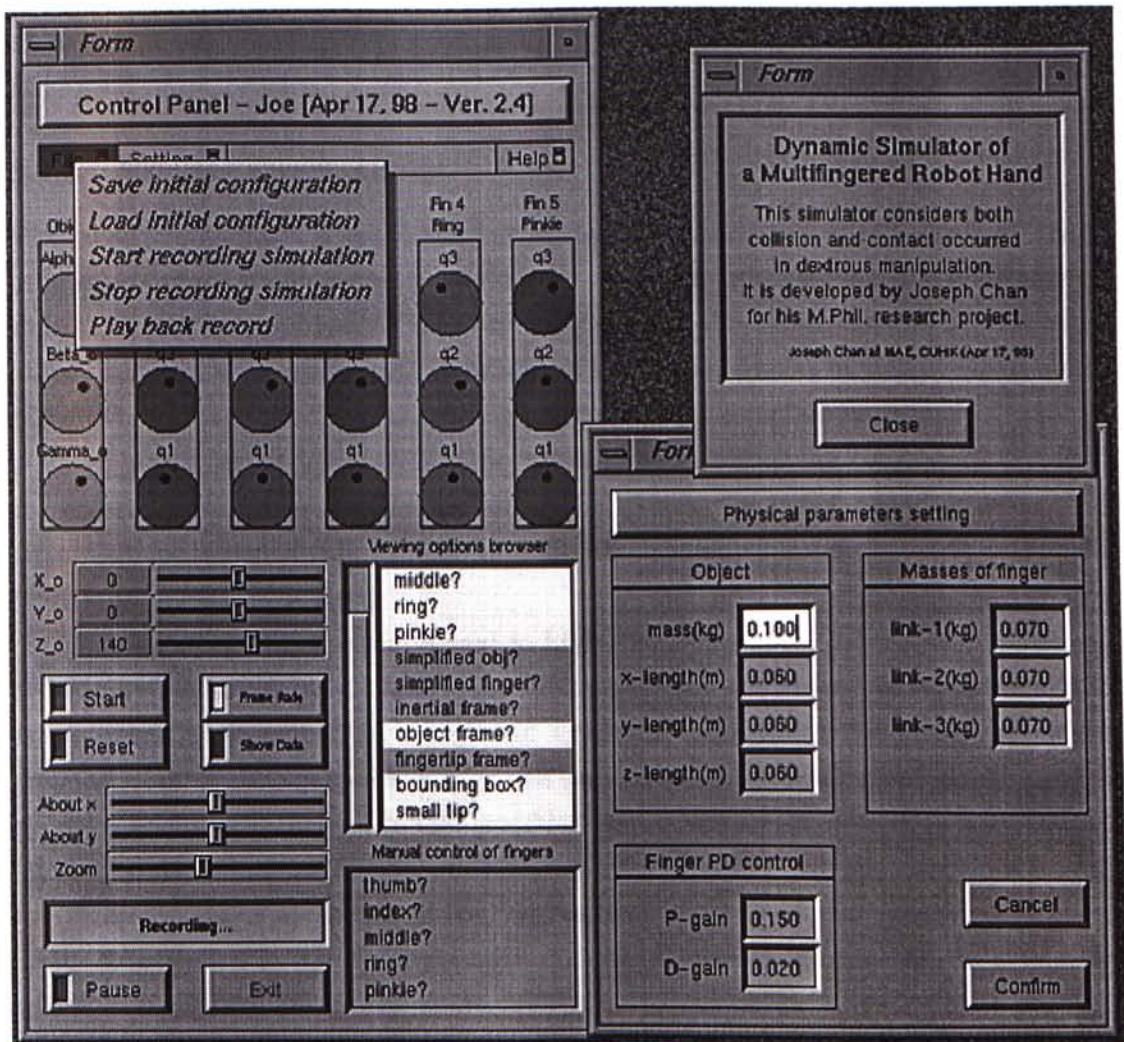


Figure 5.4: Screen shot of the graphical control panel of the dynamic simulator.



graphical workstation (INDIGO<sup>2</sup>) using C programming language. Because of the intensive use of matrix equations in formulating the system equations, it is time consuming to implement them by using C language directly for this part of implementation. All the equations of the system were firstly written in the MATLAB environment and then converted into C source code. MATLAB provides the ideal matrix-based environment to implement the equations. The conversion from MATLAB code to C code and the usage of this converted code are accomplished by the *MATLAB compiler* and the *MATLAB Math Library*.

### 5.2.3 Virtual Environment

Virtual environment is a simulated environment in computer. The numerical results from the dynamic simulator are passed to the virtual environment module to display the simulation results graphically. The graphical output is three dimensional, in perspective projection, and with rendering. Showing the simulation results in virtual environment can provide a visual feedback to the human operator for a certain motion. For example, if the operator wants to rotate a cube, there are some grasping configurations which are not stable to grasp the manipulated cube. Then, the cube may slide away. The virtual environment in this way can provide information to the operator to correct his movement.

Running of a virtual environment is highly computational intensive. It does not provide a satisfactory result to develop in the personal computer. Instead, the virtual environment is developed in a high-end graphical workstation by using the default graphics library, *IRIS* graphics library. All the realistic graphical output can be handled by the subroutines provided in this graphics library and the objects drawn are defined by their vertices and faces. Since this library only supports primitive way to construct the three dimensional models of the five-fingered robot hand, the geometric model is firstly constructed by the solid modeler, AutoCAD, and then converted to C code by a three dimensional file format converter<sup>3</sup>.

## 5.3 Methodologies and Program Flow of the Dynamic Simulator

We have just presented the overview of the dynamic simulation system for the five-fingered robot hand. In this part, we will concentrate to discuss the implementation

<sup>3</sup>The 3D file format converter used is called the *wcvt2pov*. It is a freeware found in internet and it can be download from <http://www.europa.com/~keithr/free.htm>.

issues of the dynamic simulator. The overview program flow of the dynamic simulator is shown in Algorithm 5.1. There are three main components of this simulator. The interference detection is the routine to detect whether there is interaction in the system. The interaction includes both contact and collision. The constraint-based simulation and the impulse-based simulation are used to determine the dynamic responses of the system.

---

**Algorithm 5.1** Program flow of the dynamic simulator.

---

**Input:** initial system state

```
1: loop
2:   interference detection {refer to Algorithm 5.2}
3:   constraint-based simulation {refer to Algorithm 5.3}
4:   if collision detected then
5:     impulse-based simulation {refer to Algorithm 5.4}
6:   end if
7: end loop
```

---

### 5.3.1 Interference Detection

Development of a robust and efficient interference detection algorithm is one of the hottest research areas in virtual reality, motion planning, and dynamic simulation (Lin, 1993; Garcia Alonso et al., 1994; Held et al., 1995; Hubbard, 1995; Leung, 1996; Mirtich, 1997). It may be called in different names such as contact detection or collision detection. In order not to mix up the terms, contact and collision, in dynamic modeling, interference detection is used to describe the detection of interactions. Various techniques have been proposed for convex polyhedral, concave polyhedral, curved object interaction detection, and software libraries are also available.

For the dynamic simulation system of the five-fingered robot hand, the interference detection problem is limited to detect whether there is interaction between fingertips and a rectangular object. We do not consider the interactions between finger-to-finger or link-to-link. Furthermore, interaction further narrows to allow only face-to-face interaction, but not the face-to-edge, edge-to-edge, face-to-vertex, or edge-to-vertex problem. These assumptions greatly help to simplify the problem but do not lose the object of simulation to examine the dynamic behaviors. This limited situation is capable to handle most of the multi-fingered grasping or manipulation motions. As a result, a pure geometric interference detection method has been implemented.

The idea of the interference detection is simple. Consider Figure 5.5, the interesting part of the fingertip is its end part and it is actually a hemisphere. In such situation, the



interference detection is to detect whether there is interaction between the hemisphere and the rectangular object if the object is restituted to be rectangular in shape. This problem can be further simplified by “blow up” the size of the rectangular object by the length of radius of the hemisphere in each side. Therefore, the interaction detection is simply to determine whether the center of the hemisphere is placed inside the “growth” object. The program flow of this interference detection is summarized in Algorithm 5.2.

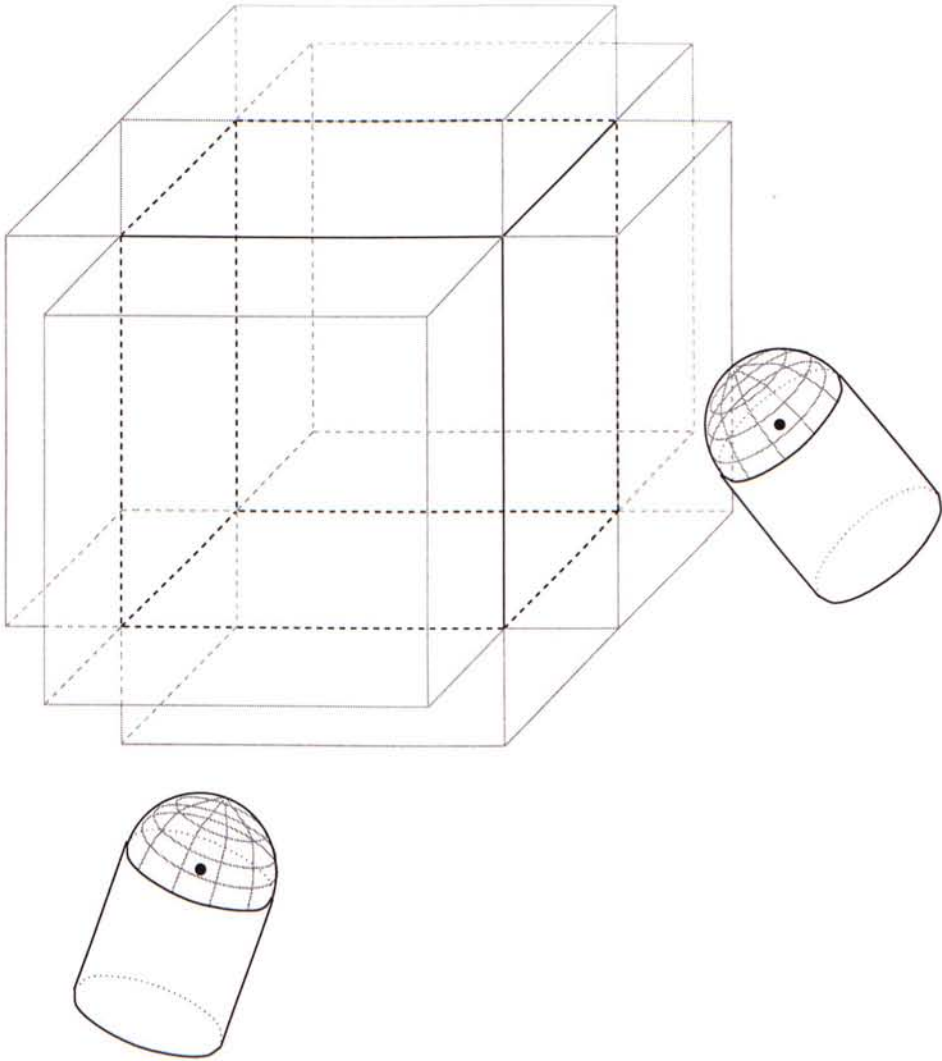


Figure 5.5: The “growth” rectangular object and the object in hemisphere shape used in the interference detection of the five-fingered robot hand dynamic simulation system.



---

**Algorithm 5.2** Interference detection of the dynamic simulator.

---

**Input:** current system state

- 1: compute fingertip state from joint state {using finger forward kinematic equation (2.5) and finger Jacobian (2.8)}
  - 2: apply object coordinate frame  $\Sigma_o$  as reference coordinate frame
  - 3: **for** finger- $i = 1$  to 5 **do**
  - 4:   compute distance between fingertip  $i$  and object
  - 5:   **if** distance  $\leq 0$  **then**
  - 6:     compute relative contact velocity,  $u$
  - 7:     **if**  $u < \text{threshold}_{\text{collision}}$  **then**
  - 8:       write finger- $i$ -flag = contact {contact detected for finger  $i$ }
  - 9:     **else**
  - 10:      write finger- $i$ -flag = collision {collision detected for finger  $i$ }
  - 11:     **end if**
  - 12:   **end if**
  - 13: **end for**
- 

### 5.3.2 Constraint-based Simulation

The constraint-based simulation is to calculate the dynamic response of the system when the hand and the object are stably in contact. The state of the system is updated by numerically integrating the system dynamic equation (3.14). The program flow of the constraint-based simulation is shown in Algorithm 5.3.

---

**Algorithm 5.3** Constraint-based simulation of the dynamic simulator.

---

- 1: estimate and update current contact mode {using contact transition model in Figure 5.6}
  - 2: compute velocity constraints {equation (2.17)}
  - 3: compute joint input torques {using PD+gravity control, equation (5.1)}
  - 4: compute constraint force {equation (3.16)}
  - 5: compute system dynamics {equation (3.14)}
  - 6: compute new system state {numerical integration described by equation (5.2)}
  - 7: update system state
- 

#### Contact Transition Model

When the flag of contact mode is true, it represents there is contact in the simulation. However, it does not give us any other information about what kind of contact should be. Selection of an appropriate kind of contact is based on the system states, and the previous and the current contact information. This selection is summarized in the state diagram of the contact transition model shown in Figure 5.6.



This contact transition model is developed based on another simple contact transition model proposed in (Wendlandt, 1997). It helps to handle the change of contacts such as sticking contact and sliding contact occurred between the robot foot and the ground in biped walking robot simulation. This simple contact transition model is extended to handle the change of contact modes occurred specially in dextrous manipulation motions. Circle in the diagram represents the state of contact and the rounded corner box represents to the condition of transition between different contact states. For example, if there is no contact between the  $i$ th finger and the object in last simulation instance but contact is detected in the current state, this contact is classified to be the rolling contact if it satisfies following criteria:

- previous contact mode is sticking contact or rolling contact, and
- constraint force is within the friction cone, and
- relative angular contact velocity is large than zero

After determining the current contact mode, we can choose the appropriate contact mode selection matrix  $C_{Mi}$  (equation (2.15)) in formulating the velocity constraint of the system.

### Numerical Integration

After the construction of the velocity constraint, we can use the information of joint input torques and constraint force to determine the system dynamic response. As mentioned before, the acceleration of the system is required to integrate numerically in order to obtain the system state in next instance. This integration algorithm should be accurate and efficient, and the fixed step size fourth order Runge-Kutta method is chosen as the integration algorithm. Let  $\dot{x} = f(x, t)$  be the general form of first order differential equation,  $x_0$ ,  $t_0$  and  $h$  are the initial state value, current time instance, and the integration step size, the new state value  $x(t_0 + h)$  can be determined by the following equation:

$$x(t_0 + h) = x_0 + \frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4 \quad (5.2)$$



where

$$\begin{aligned} k_1 &= hf(x_0, t_0) \\ k_2 &= hf\left(x_0 + \frac{k_1}{2}, t_0 + \frac{h}{2}\right) \\ k_3 &= hf\left(x_0 + \frac{k_2}{2}, t_0 + \frac{h}{2}\right) \\ k_4 &= hf(x_0 + k_3, t_0 + h) \end{aligned}$$

### 5.3.3 Impulse-based Simulation

The core of the impulse-based simulation is to integrate the collision equation (4.15) in order to determine the post-collision velocities of the system. This collision integration may involve two different integrations which are the sliding mode collision integration and the sticking mode collision integration. Selection of these two integrations depends on the pre-collision velocities and the friction coefficient. The detailed program flow and implementations of these simulations are shown in Algorithms 5.4, 5.5, and 5.6.

---

**Algorithm 5.4** Impulse-based simulation of the dynamic simulator.

---

- 1: **for** each collision **do**
- 2:   determine collision coordinate frame,  $\Sigma_c$
- 3:   compute initial relative collision point velocity,  $u(\gamma_o)$

$$u(\gamma_o) = u_{fi}(\gamma_o) - u_o(\gamma_o)$$

- 4:   compute collision matrix,  $K$  {equations (4.7), (4.13) and (4.14)}
- 5:   compute final relative collision point velocity,  $u(\gamma_f)$  {refer to Algorithm 5.5}
- 6:   compute corresponding impulse {using  $u(\gamma_o)$ ,  $u(\gamma_f)$  and  $K$  in equation (4.14)}

$$P(\gamma) = K^{-1}(u(\gamma_f) - u(\gamma_o))$$

- 7:   determine post-collision velocities
  - 8: **end for**
  - 9: **for** each contact **do**
  - 10:   propagate change of object velocity to joint velocity
  - 11: **end for**
- 

### Post-collision Velocity

In Algorithm 5.4, we can determine the post-collision velocities of the system after computing the impulse of collision. These post-collision velocities consist of the post-

---

**Algorithm 5.5** Sliding mode collision integration.

---

```

1: repeat {compression phase of sliding mode collision integration}
2:   if sticking mode test success,  $u_t < \text{threshold}_{sliding}$  then
3:     sticking mode collision integration {refer to Algorithm 5.6}
4:     write sticking-mode-flag = true
5:     break
6:   end if
7:   sliding mode collision integration in compression phase {equation (4.19)}
8: until end of compression phase,  $u_z \geq 0$ 
9: if sticking-mode-flag  $\neq$  true then {continue sliding mode collision integration}
10:  compute terminating condition of restitution phase,  $P_z(\gamma_f)$  {using Poisson's hypothesis, equation (4.1)}
11:  repeat {restitution phase of sliding mode collision integration}
12:    if sticking mode test success,  $u_t < \text{threshold}_{sliding}$  then
13:      sticking mode collision integration {refer to Algorithm 5.6}
14:      break
15:    end if
16:    sliding mode collision integration in restitution phase {equation (4.16)}
17:  until end of restitution phase,  $P_z(\gamma) \geq P_z(\gamma_f)$ 
18: end if

```

---

collision joint velocities of the colliding finger and the post-collision object velocity. The required equations in determining these post-collision velocities are derived in this section.

By equation (4.6), we can write the resultant collision point velocity of the  $i$ th colliding finger as:

$$u_{fi}(\gamma_f) = K_{fi}P(\gamma) + u_{fi}(\gamma_0) \quad (5.3)$$

Since this post-collision velocity  $u_{fi}(\gamma_f)$  can be related to the joint velocities of the finger by the finger Jacobian as:

$$u_{fi}(\gamma_f) = J_i \dot{\theta}_i(\gamma_f)$$

Assuming the  $i$ th finger is not in a singular configuration so the finger Jacobian  $J_i$  is invertible. The joint velocities can be written as:

$$\dot{\theta}_i(\gamma_f) = J_i^{-1} u_{fi}(\gamma_f) \quad (5.4)$$

By equations (5.3) and (5.4), the post-collision joint velocities of the  $i$ th colliding finger

---

**Algorithm 5.6** Sticking mode collision integration.

---

- 1: do sticking stability test
- 2: **if** stable sticking mode **then**
- 3:   **if** occur in compression phase **then**
- 4:     compute  $P_z(\gamma_{mc})$  {equation (4.21)}
- 5:     compute  $P_z(\gamma_f)$  {equation (4.1)}
- 6:     compute  $u(\gamma_f)$  {equation (4.22)}

$$u(\gamma_f) = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{k_{3,3}}(\gamma_f - \gamma_{mc}) \end{bmatrix}$$

where  $\gamma_{mc} = P_z(\gamma_{mc})$  and  $\gamma_f = P_z(\gamma_f)$

- 7:   **else if** occur in restitution phase **then**
- 8:     compute  $u(\gamma_f)$  {equation (4.23)}

$$u(\gamma_f) = \begin{bmatrix} 0 \\ 0 \\ u_z(\gamma_{sticking}) + \frac{1}{k_{3,3}}(\gamma_f - \gamma_{sticking}) \end{bmatrix}$$

- 9:   **end if**
- 10: **else if** transient sticking mode **then**
- 11:   compute direction of diverging ray,  $\beta$
- 12:   **if** occur in compression phase **then**
- 13:     compute  $u_x(\gamma_{mc})$ ,  $u_y(\gamma_{mc})$ , and  $P_z(\gamma_{mc})$  {equations (4.24a) to (4.24c)}
- 14:     compute  $P_z(\gamma_f)$  {equation (4.1)}
- 15:     compute  $u(\gamma_f)$  {similar to equations (4.25a) to (4.25c)}

$$u(\gamma_f) = \begin{bmatrix} u_x(\gamma_{mc}) + k_{1,j}\xi(\beta)(\gamma_f - \gamma_{mc}) \\ u_y(\gamma_{mc}) + k_{2,j}\xi(\beta)(\gamma_f - \gamma_{mc}) \\ k_{3,j}\xi(\beta)(\gamma_f - \gamma_{mc}) \end{bmatrix}$$

- 16:   **else if** occur in restitution phase **then**
- 17:     compute  $u(\gamma_f)$  {equations (4.25a) to (4.25c)}

$$u(\gamma_f) = \begin{bmatrix} k_{1,j}\xi(\beta)(\gamma_f - \gamma_{sticking}) \\ k_{2,j}\xi(\beta)(\gamma_f - \gamma_{sticking}) \\ u_z(\gamma_{sticking}) + k_{3,j}\xi(\beta)(\gamma_f - \gamma_{sticking}) \end{bmatrix}$$

- 18:   **end if**
  - 19: **end if**
-



can be computed by:

$$\dot{\theta}_i(\gamma_f) = J_i^{-1} [K_{fi}P(\gamma) + u_{fi}(\gamma_0)] \quad (5.5)$$

On the other hand, the post-collision object velocity can be deduced from equation (4.10). Let  $\mathcal{V}_o = \begin{bmatrix} v_o \\ \omega_o \end{bmatrix}$  be the generalized object velocity,  $\mathcal{V}_o(\gamma_f)$  is:

$$\mathcal{V}_o(\gamma_f) = \begin{bmatrix} -\frac{1}{m_o}P(\gamma_f) + v_o(\gamma_0) \\ -I^{-1}p_o \times P(\gamma_f) + \omega_o(\gamma_0) \end{bmatrix} \quad (5.6)$$

### Velocity Propagation

Because the impulse-based simulation does not involve any change in configurations of the colliding objects, multiple collisions can be partitioned and solved one by one by equation (4.15). When a finger is colliding with the object grasped or manipulated by other fingers, some special treatments are necessary to handle this situation. Compared to the impulsive force caused by a collision, the grasping forces are very small so that they were ignored in determining collision response of the system by using equation (4.15). Since the object and other contacted fingers are actually connected together, this change of the object velocity must be propagated to other contacting fingers. As a result, there are also instantaneous changes in joint velocities of all contacting fingers.

When we have the post-collision object velocity, we can determine the new contact velocity  $u_{ci} = u_{fi}$  at the contact point of the  $i$ th contacting finger. Using equation (4.5),

$$u_{fi}(\gamma_f) - u_{fi}(\gamma_0) = J_i [\dot{\theta}_i(\gamma_f) - \dot{\theta}_i(\gamma_0)]$$

Therefore, the new propagated joint velocities of the  $i$ th contacting finger can be written as:

$$\dot{\theta}_i(\gamma_f) = J_i^{-1} [u_{fi}(\gamma_f) - u_{fi}(\gamma_0) + J_i \dot{\theta}_i(\gamma_0)] \quad (5.7)$$

## 5.4 Summary

In this chapter, we have presented the details of the dynamic simulation system for the five-fingered robot hand system. It is developed based on the equations derived in last three chapters. The design of the system facilitates us to do the dynamic

simulation interactively so that the dynamic behaviors of the multi-fingered robot hand system can be examined on-line. The simulation inputs are obtained directly from transferring the human operator skills to the system.

The architecture of the dynamic simulation system of the five-fingered robot hand system is presented at the beginning of this chapter. It mainly includes the inputs, dynamic simulator, and the virtual environment. The implementation issues of these components are also reviewed. In the next part, detailed descriptions of the dynamic simulator are discussed since it is the heart of the whole dynamic simulation. Inference detection is done by using geometric relationships between fingers and object. The constraint-based simulation is outlined. The detailed implementation of the impulse-based simulation is discussed. For the multiple collisions, each of them can be treated individually. The mixed contact collision situation is simplified to ignore the contact interactions first and then propagate the velocity change effect to the contacting fingers after collision.

## Chapter 6

# Simulation Results

### 6.1 Introduction

The dynamic simulation system based on the full kinematic and dynamic models of a five-fingered robot hand system has been developed. In this chapter, various capabilities of the simulator mentioned in the previous chapters will be demonstrated. The simulation results are presented in three main parts. The first part in Section 6.2, 6.3 and 6.4 shows the results of the constraint-based simulation based on the equations formulated in Chapters 2 and 3. Changes in grasping, rolling contact, sliding contact are demonstrated with numerical data. The second part presents the impulse-based simulation results. Collision responses of a finger collided with different object are discussed in Section 6.5 and the effect of velocity propagation discussed in Section 5.3.3 is illustrated in an example (Example 8) involving mixed contacts and collisions. Furthermore, two examples (Examples 9 and 10) showing dextrous manipulations are presented in Section 6.6. These manipulations involve more than one particular type of interactions such as contacts and collision, as well as grasping configurations. Finally, the chapter is ended with a summary of the simulation results.

The used physical and geometric parameters of the object are listed in Tables 6.1. The other simulation settings are provided in Appendix C.

### 6.2 Change of Grasping Configurations

Figure 6.1 presents an example in changing grasping configuration. Initially, there are four fingers grasping the rectangular object and the third (middle) finger is in free motion. The third finger is moving to touch the object while the second (index) fin-



Example	Object mass, $m_o$ (kg)	Object Dimensions, (m)
1	0.1	$0.1 \times 0.18 \times 0.1$
2	0.1	$0.06 \times 0.06 \times 0.06$
3	0.1	$0.08 \times 0.06 \times 0.03$
4	0.1	$0.16 \times 0.06 \times 0.06$
5	0.01	$0.06 \times 0.06 \times 0.06$
6	0.1	$0.06 \times 0.06 \times 0.06$
7	0.5	$0.06 \times 0.06 \times 0.06$
8	0.1	$0.06 \times 0.06 \times 0.06$
9	0.1	$0.1 \times 0.08 \times 0.1$
10	0.05	$0.1 \times 0.1 \times 0.08$

Table 6.1: Conditions used in the simulation results demonstration.

ger and the fourth (ring) finger are blending outward. These fingers eventually lose their contacts with the object. This example shows the four-fingered grasping configuration is changed to the three-fingered grasping configuration. This kind of grasping configuration changing is achieved by imposing different contact mode selection matrix  $C_{Mi}$  derived in Chapter 2 into the system dynamics to represent both constrained and unconstrained motions.

The second example shown in Figure 6.2 demonstrates the change in grasping configuration from a five-fingered grasp to a three-fingered grasp. Unlike the example 1, instability is introduced in this grasping configuration change, that the force-closure grasp cannot be maintained. The object becomes to rotate after the withdrawals of the third and the fourth fingers at  $t = 0.2s$ . The joint angles of these fingers are shown in Figure 6.3 and the corresponding change of object configuration can be referred to Figure 6.4.

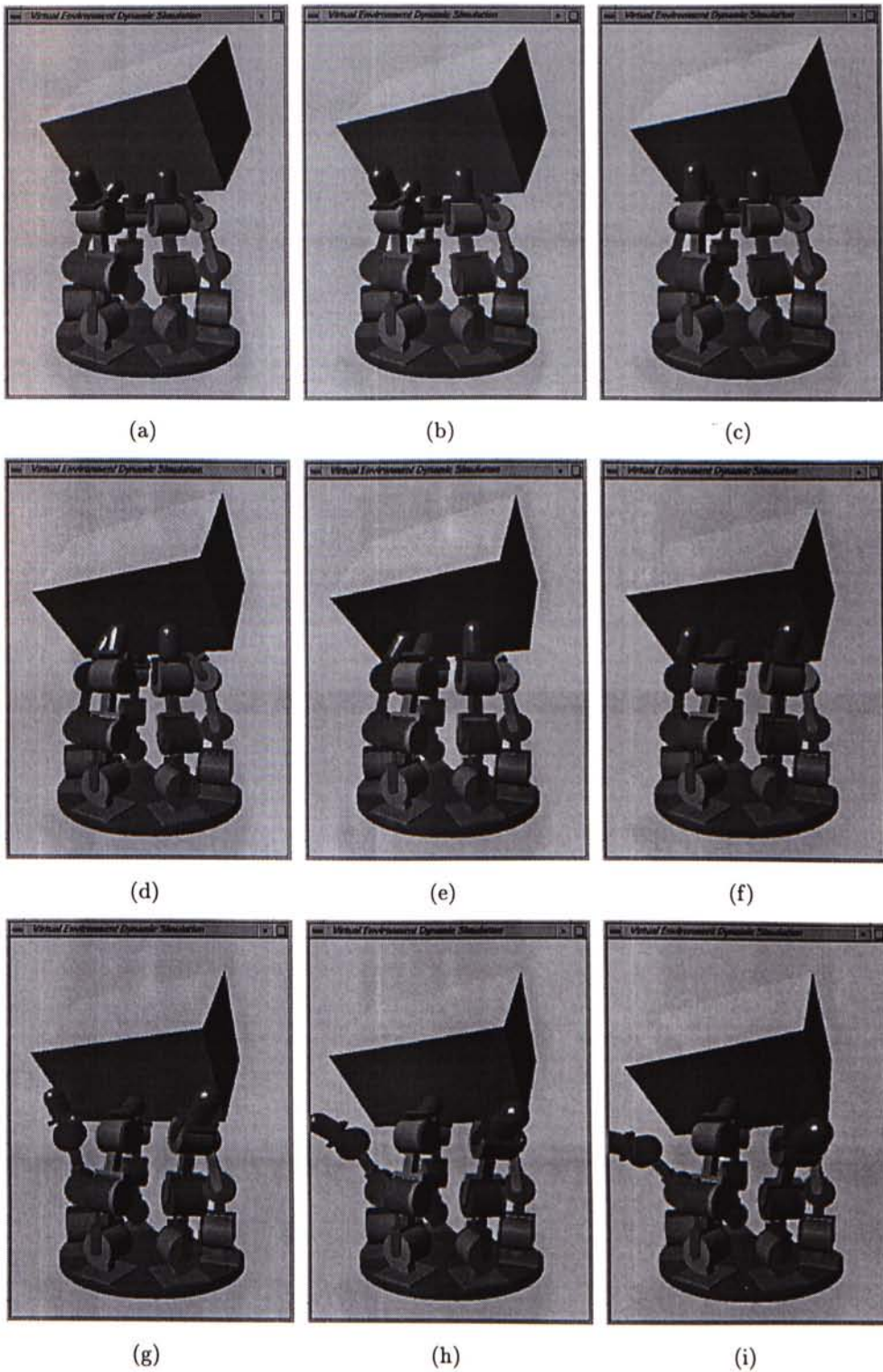


Figure 6.1: Example 1: Change in grasping configuration from a four-fingered grasp to a three-fingered grasp.



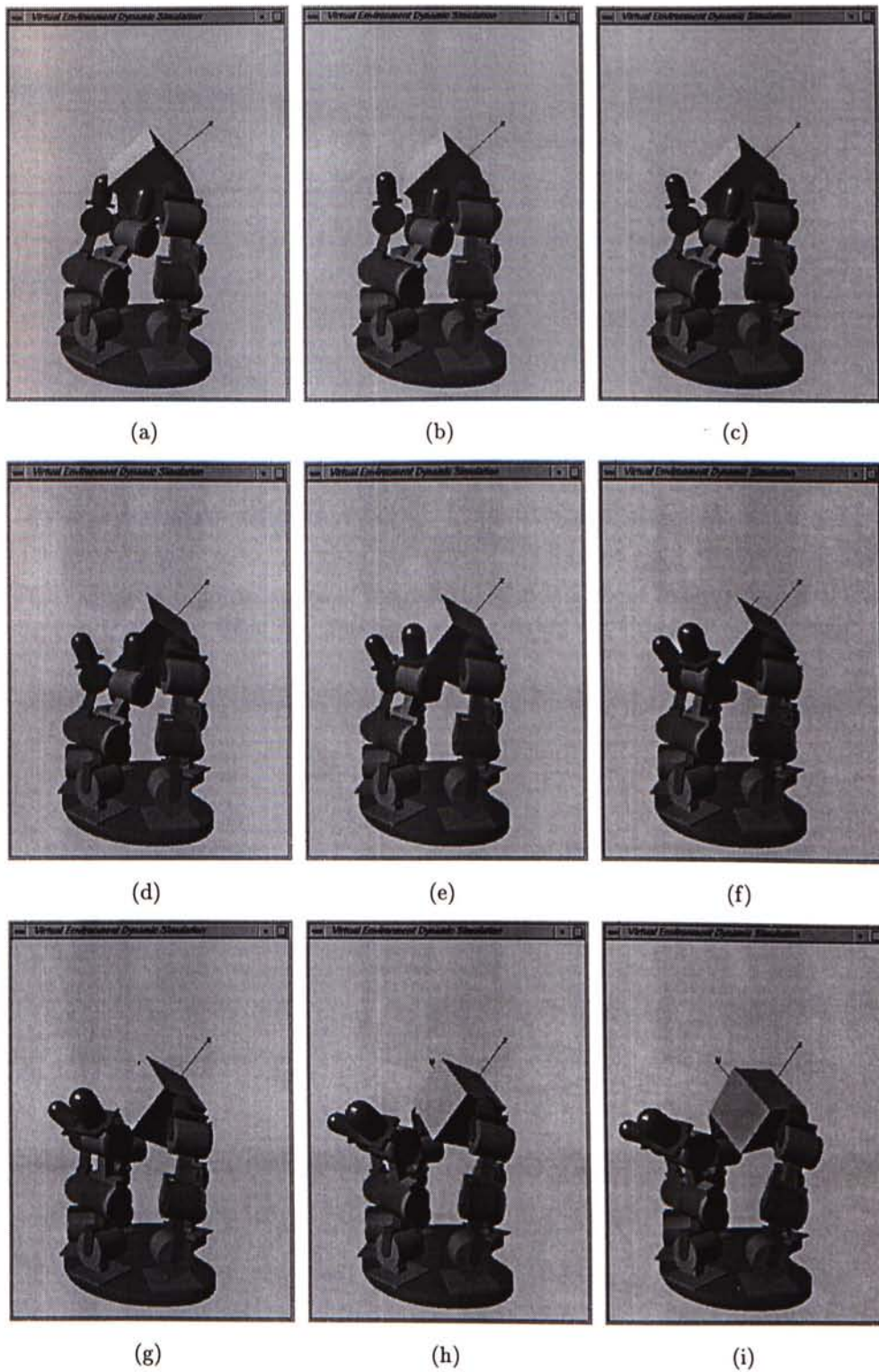
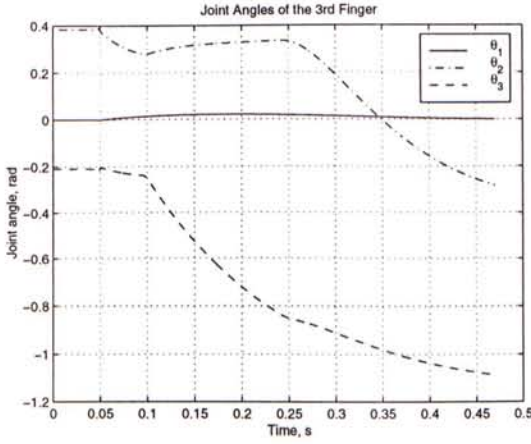
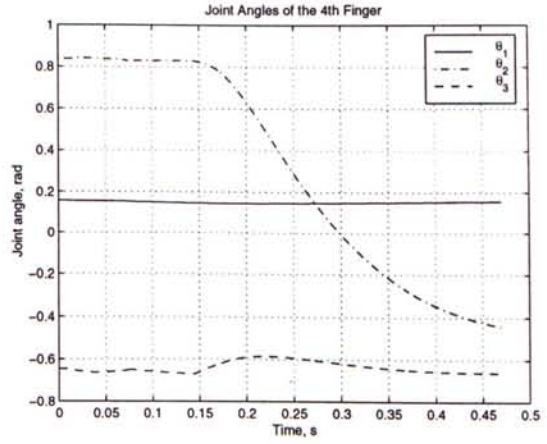


Figure 6.2: Example 2: Change in grasping configuration from a five-fingered grasp to a three-fingered grasp.



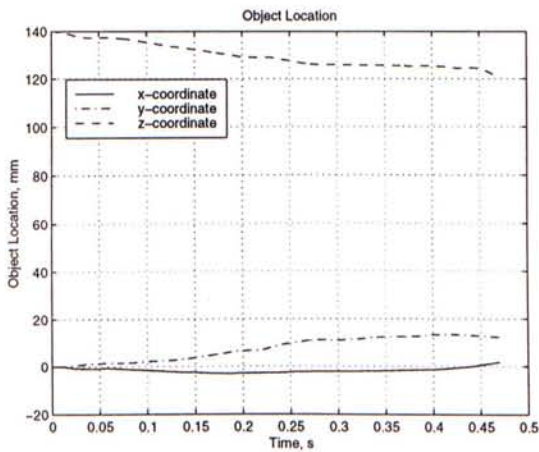


(a) Change of joint angles of the 3rd (middle) finger.

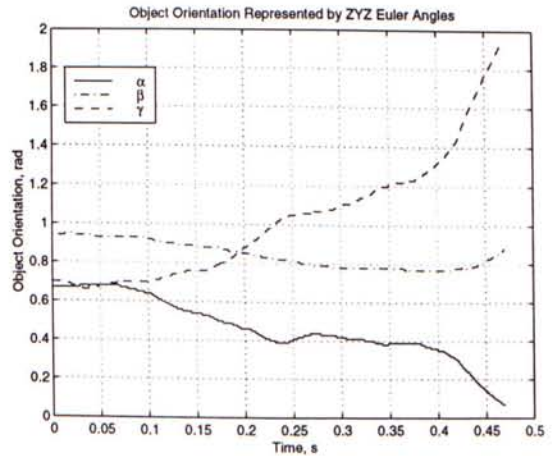


(b) Change of joint angles of the 4th (ring) finger.

Figure 6.3: Change of joint angles of the 3rd and the 4th finger in the process of changing grasping configuration from the five-fingered grasp to the three-fingered grasp.



(a) Change of object location.



(b) Change of object orientation.

Figure 6.4: Change of object configuration in the process of changing grasping configuration from the five-fingered grasp to the three-fingered grasp.

### 6.3 Rolling Contact

Rolling contact is presented between a finger and an object in the simulation shown in Figure 6.5. At the beginning, the rectangular is placed on the top of the finger. Since friction between the contacting finger and the object is sufficient to prevent sliding motion, the object rolls about the spherical part of the fingertip. Figure 6.6 shows the change of joint angles under the effect of the contacting object. Figure 6.7 shows the object configuration during the simulation. Both the  $z$ -direction and ZYZ  $\beta$  Euler angle of the object are changed most vigorous. Because the object moves nearly only in the  $xz$ -plane of the system<sup>1</sup>, the value of the  $y$  component of the constraint force is much smaller than that of the  $x$  and the  $z$  components. Figure 6.9 shows the changes of the parameters defined in the Montana's contact equation (equation (2.4)). The figures in the first row show the local coordinates,  $c_{fix}$  and  $c_{fiy}$ , changes of the fingertip. These coordinates are defined similar to the spherical coordinates, which represent location on the surface of the fingertip. The figures in the second row show the local coordinates,  $c_{ox}$  and  $c_{oy}$  changes on the contacting surface of the object. The origin point of this local coordinate chart is defined at the center of the rectangular face. From these four figures, changes of the fingertip local coordinates are larger than that of the object local coordinates. This situation represents that the interaction between the finger and the object is in rolling contact in most of the time<sup>2</sup>.

### 6.4 Sliding Contact

Sliding motion between a finger and an object is demonstrated in Figure 6.10. Figure 6.11 shows the changes of joint angles of the finger during the simulation. Figure 6.12 shows the changes of object configuration. The sliding motion presents mainly in the  $xz$ -plane of the system. Changes of constraint force arisen in the sliding interaction are shown in Figure 6.13 and Figure 6.14 shows the local contact parameters of the two contacting bodies. Compared to Figure 6.9, we can see that the changes of the object local contact parameters are much larger than that of the fingertip local contact parameters. This situation implies that the interaction between the finger and the object is in sliding contact motion in most of the time.

<sup>1</sup>The inertial coordinate frame  $\Sigma_b$  used for the five-fingered robot hand is shown in Figure C.2.

<sup>2</sup>For pure rolling motion, both changes of  $c_{ox}$  and  $c_{oy}$  should be equal to zero. Similarly, for pure sliding motion, both changes of  $c_{fix}$  and  $c_{fiy}$  should be equal to zero.



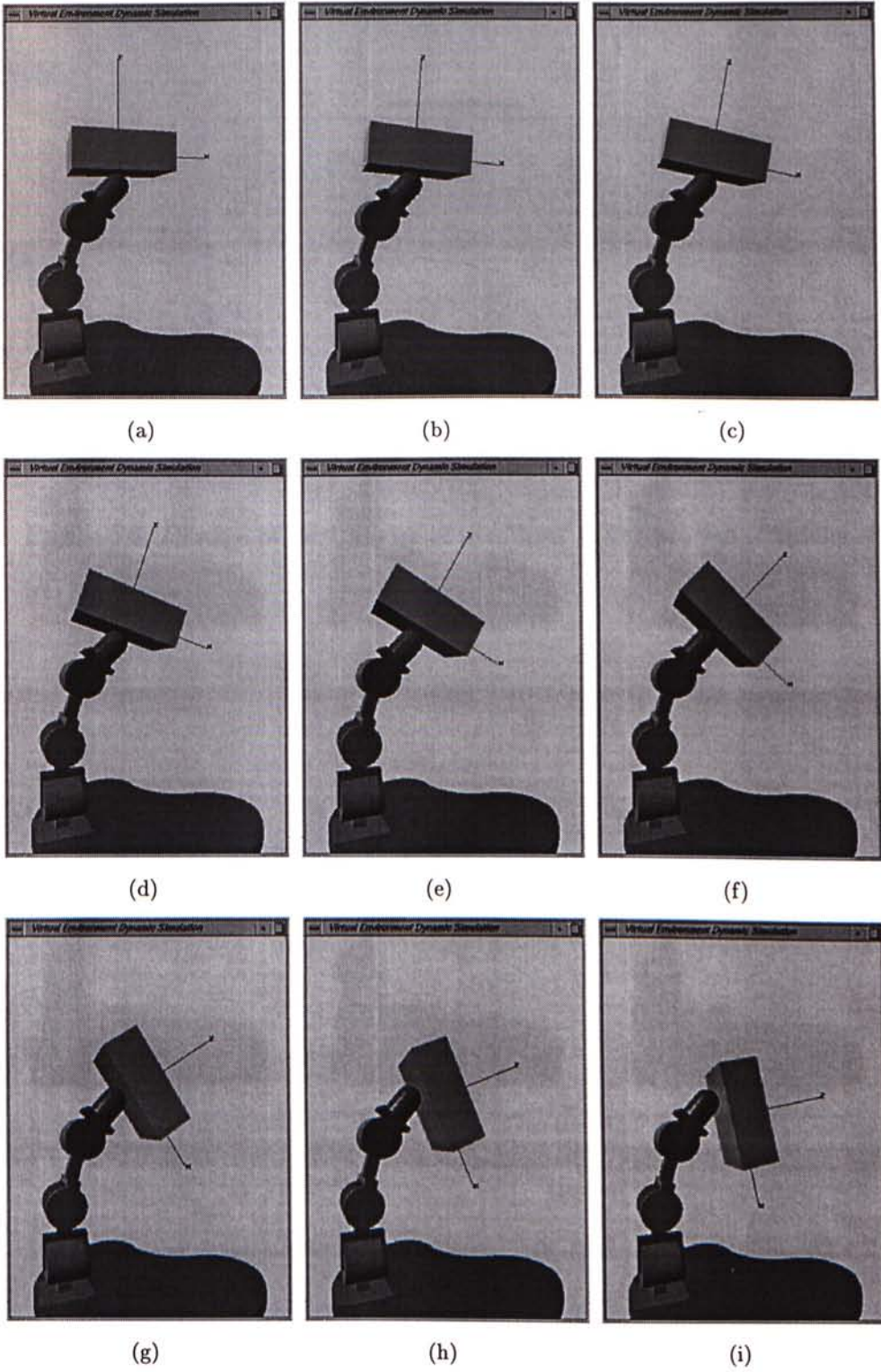


Figure 6.5: Example 3: Rolling between a finger and an object.



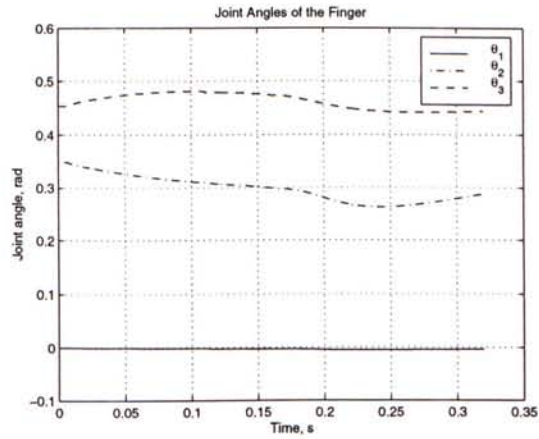
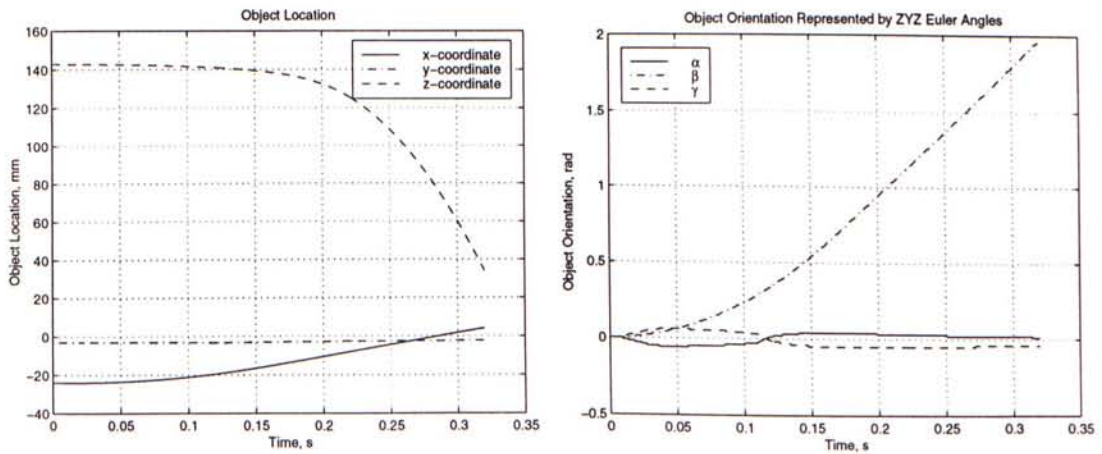


Figure 6.6: Change of joint angles of the finger in the process of rolling.



(a) Change of object location.

(b) Change of object orientation.

Figure 6.7: Change of object configuration in the process of rolling.

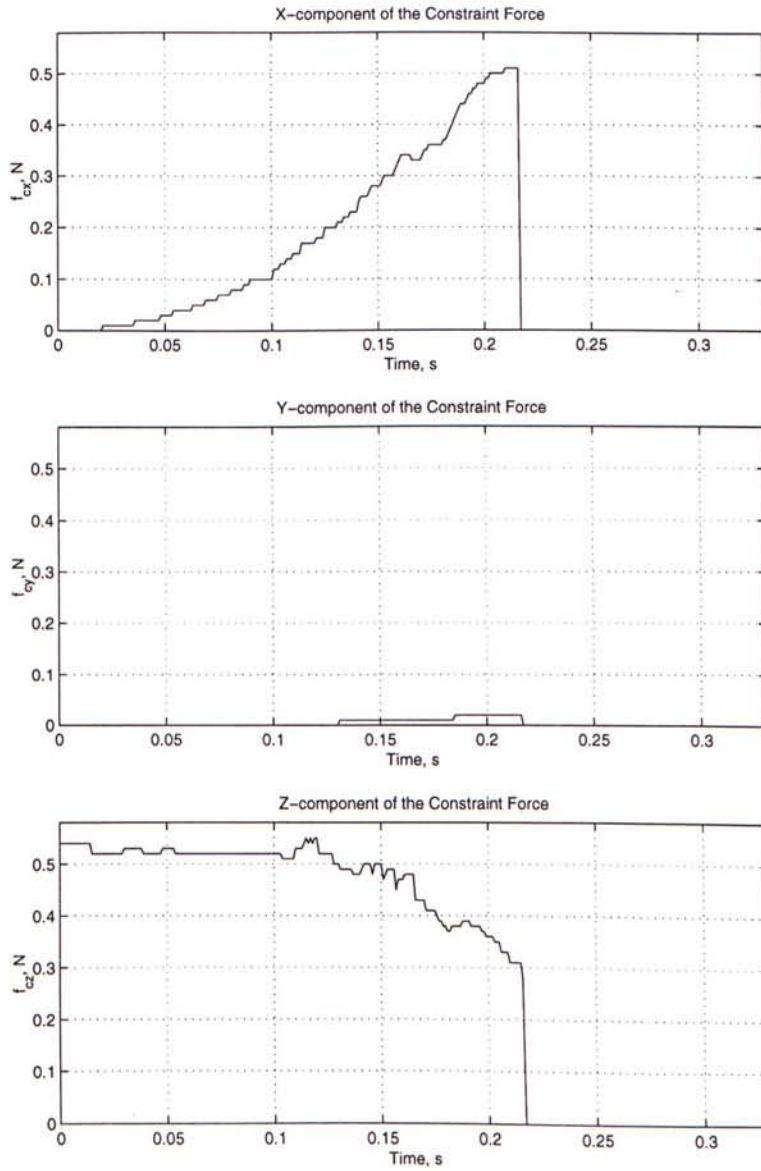


Figure 6.8: Change of constraint force  $f_c$  arisen in the process of rolling.

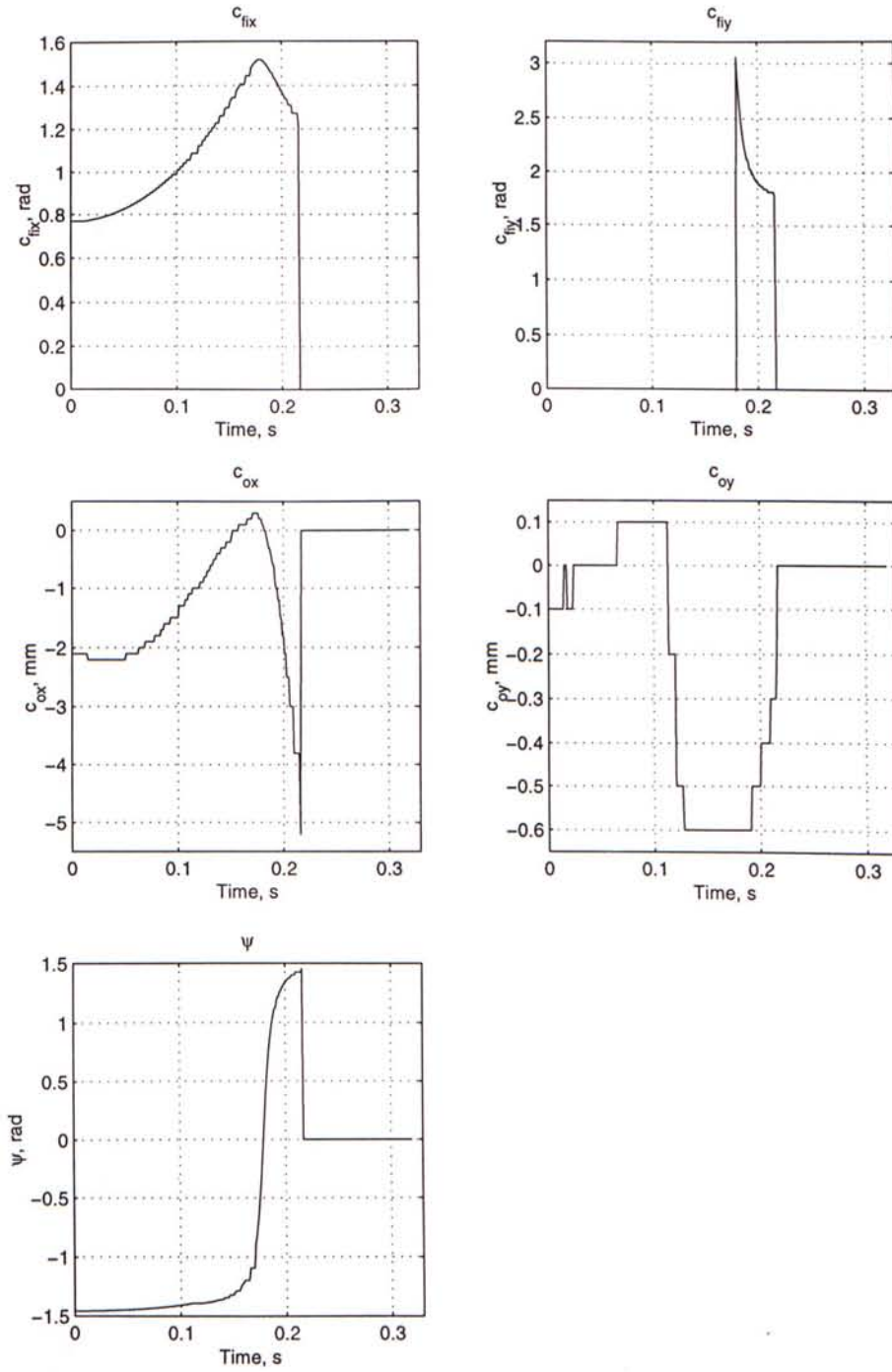


Figure 6.9: Change of local contact parameters  $[u_{li} \ u_{oi} \ \psi]^T$  in the process of rolling.



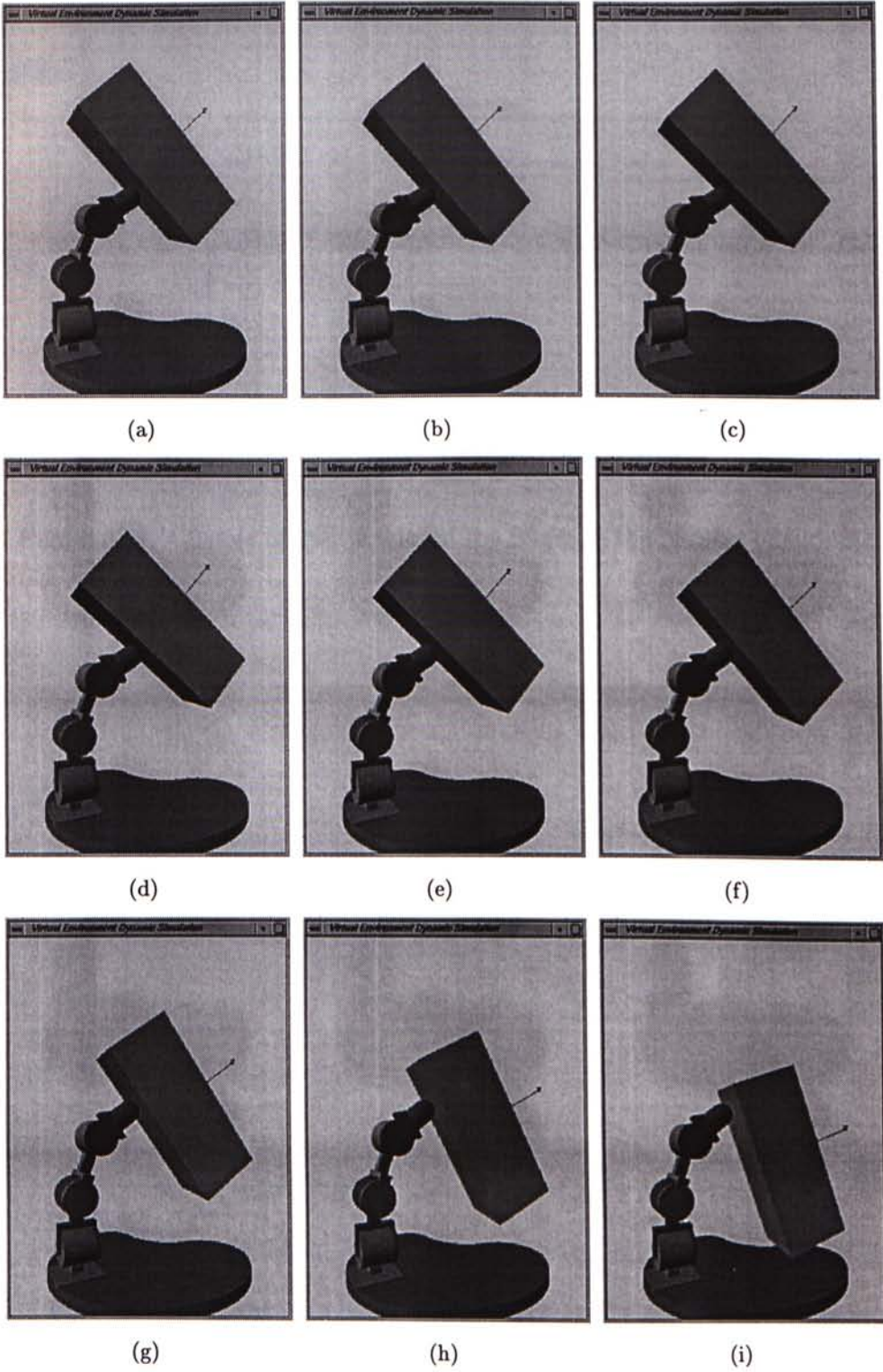


Figure 6.10: Example 4: Sliding between a finger and an object.

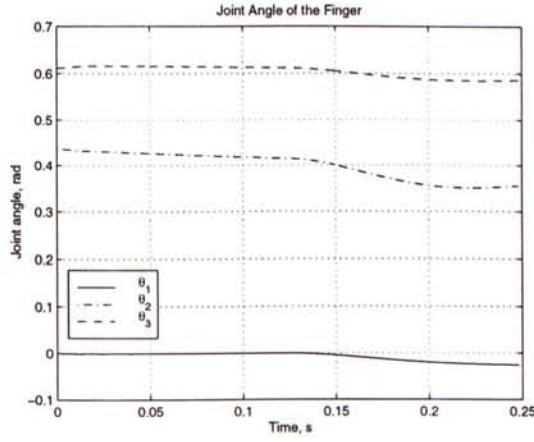
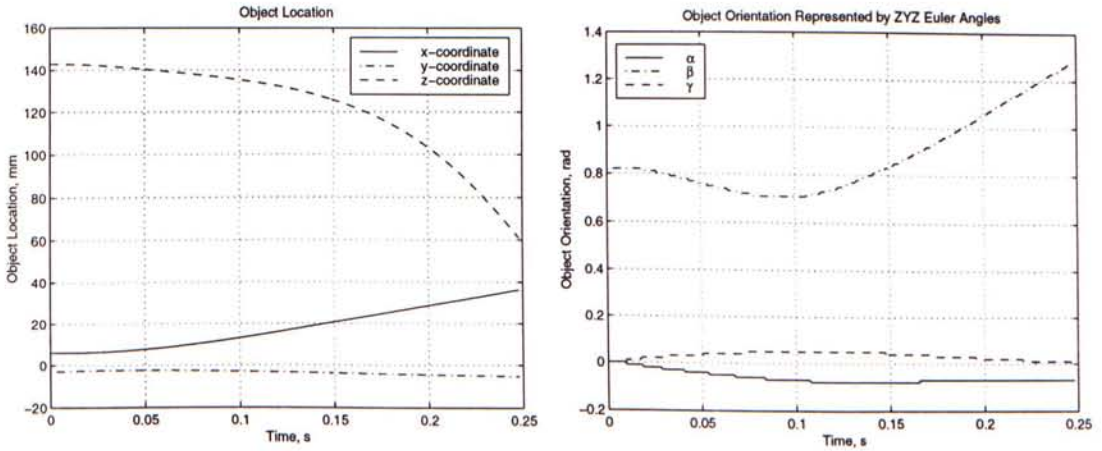


Figure 6.11: Change of joint angles of the finger in the process of sliding.



(a) Change of object location.

(b) Change of object orientation.

Figure 6.12: Change of object configuration in the process of sliding.

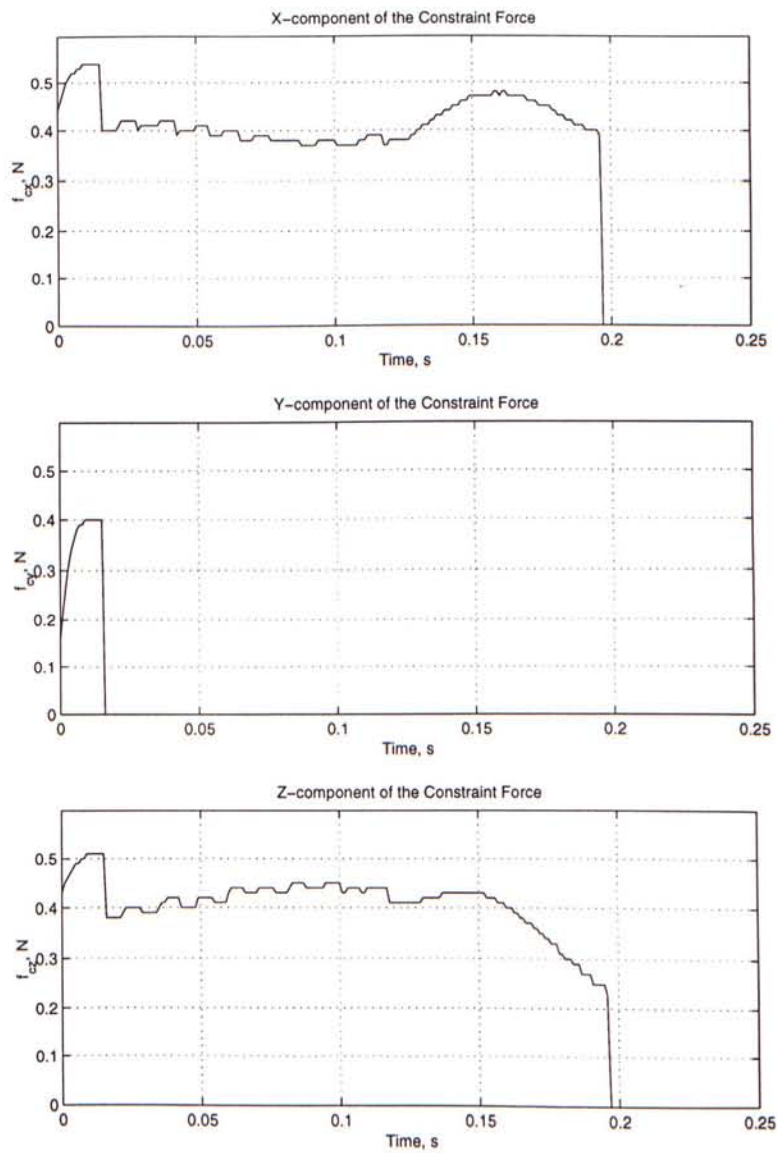


Figure 6.13: Change of constraint force  $f_c$  arisen in the process of sliding.



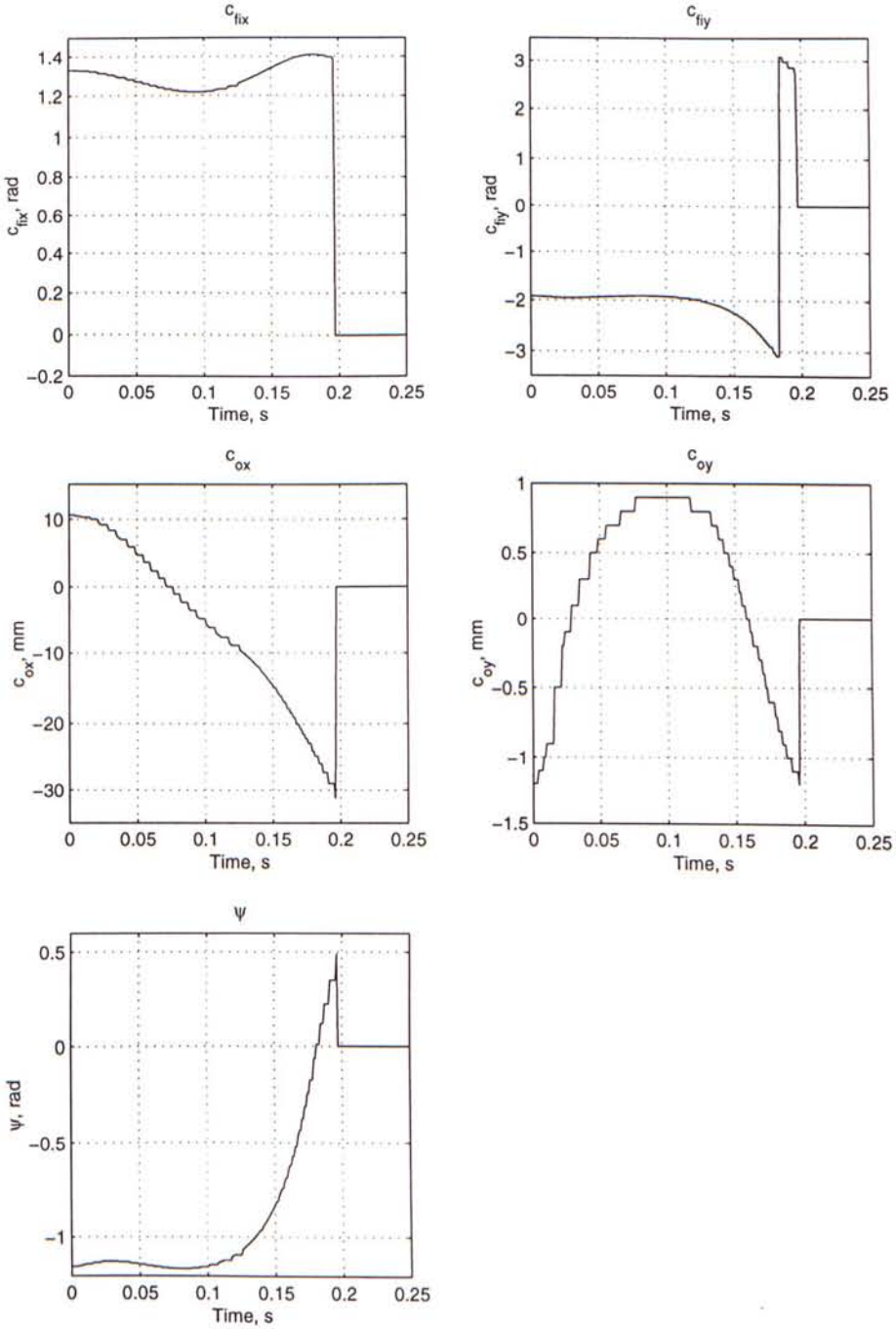


Figure 6.14: Change of local contact parameters  $[u_{li} \ u_{oi} \ \psi]^T$  in the process of sliding.

## 6.5 Collisions

The sudden change of velocity occurred when there is impulsive force applied to the system. This change is discontinuous and is handled by the collision model presented in Chapter 4. Figures 6.15, 6.16 and 6.17 show the collision responses of a finger collided with an object with different masses. Figures 6.18, 6.19 and 6.20 shows the velocity responses of the collision. The collision happens at about  $t = 0.12s$  and there are rapid changes of joint velocities, object translational velocities, and differential ZYZ Euler angles. Furthermore, there is one more velocity change for the case when  $m_o = 0.5$  at  $t = 0.175s$  since the first collision introduces an another collision between the finger and the object.

Last three examples demonstrate the dynamic effect of a finger collided with an object. Example 8 shown in Figure 6.21 illustrates that the collision response of the system when there are mixed contacts and collisions. The second (index) finger is colliding with the object that is grasped by other fingers. Figure 6.22 shows the joint velocities changes of different fingers. The joint velocities of the third (middle) finger are kept to be zero, which implies that there is no contact between this finger and the object. Since velocity propagation mechanism described in Section 5.3.3 is implemented in our system, joint velocities change of a colliding finger can be propagated to other contacting fingers. For examples, there is a coupling effect for the first (thumb) finger and the fifth (pinkie) finger at  $t = 0.16s$ . Furthermore, the change of joint velocities of the second (index) finger at  $t = 0.3s$  are propagated to the fourth (ring) finger so there is a change of joint velocities.

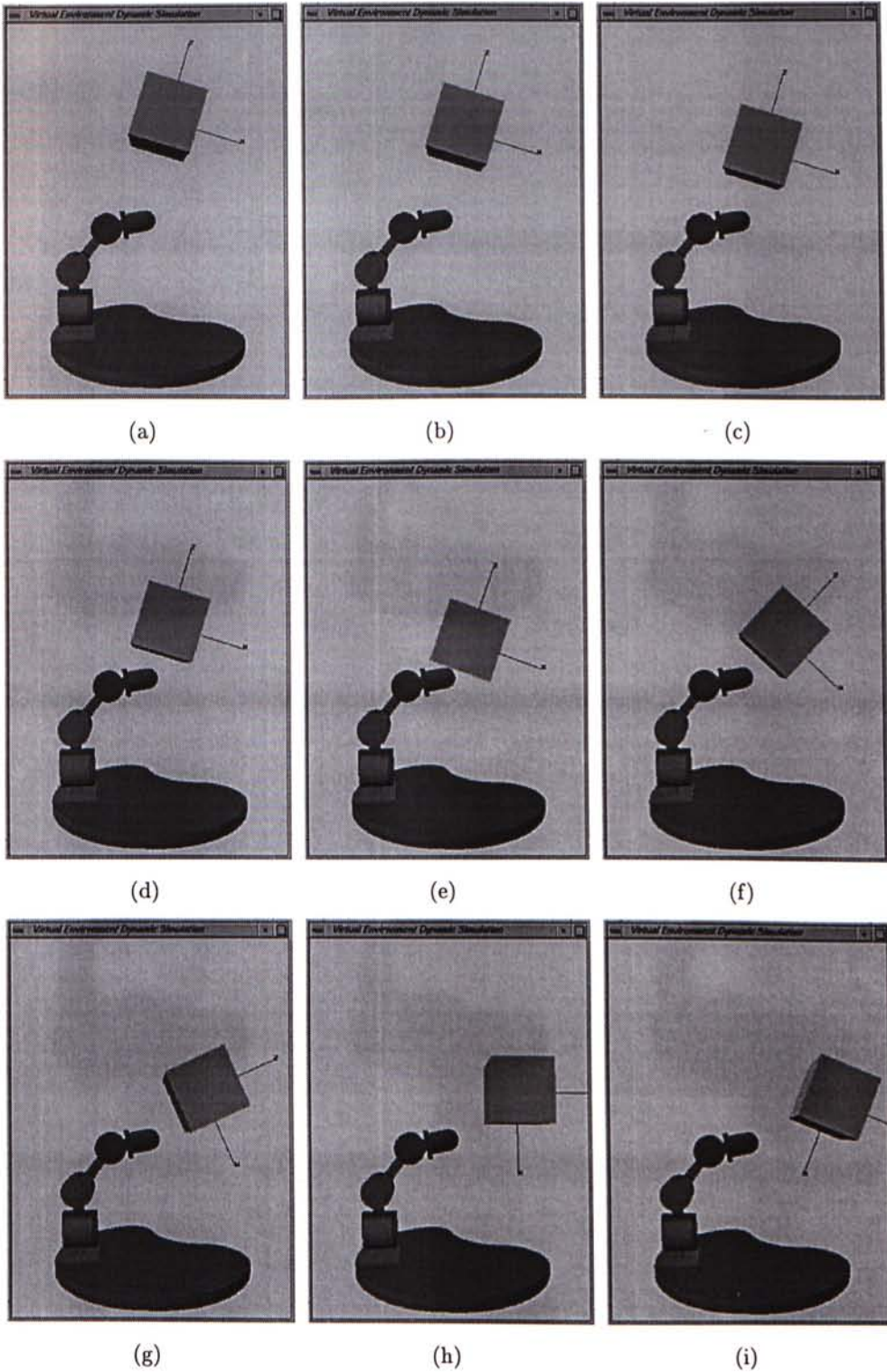


Figure 6.15: Example 5: Collision between a finger and an object with mass  $m_o = 0.01kg$ .



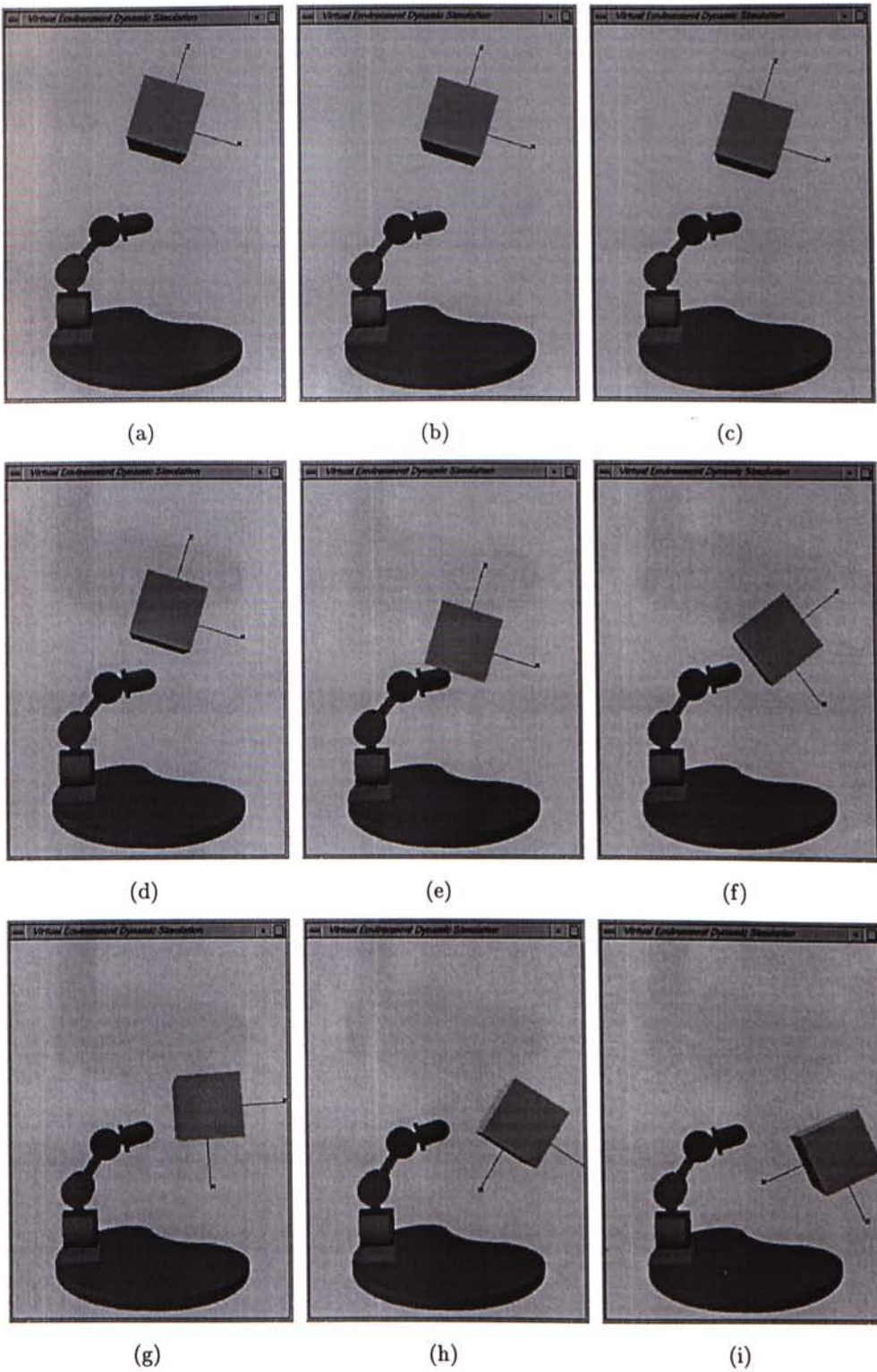


Figure 6.16: Example 6: Collision between a finger and an object with mass  $m_o = 0.1kg$ .

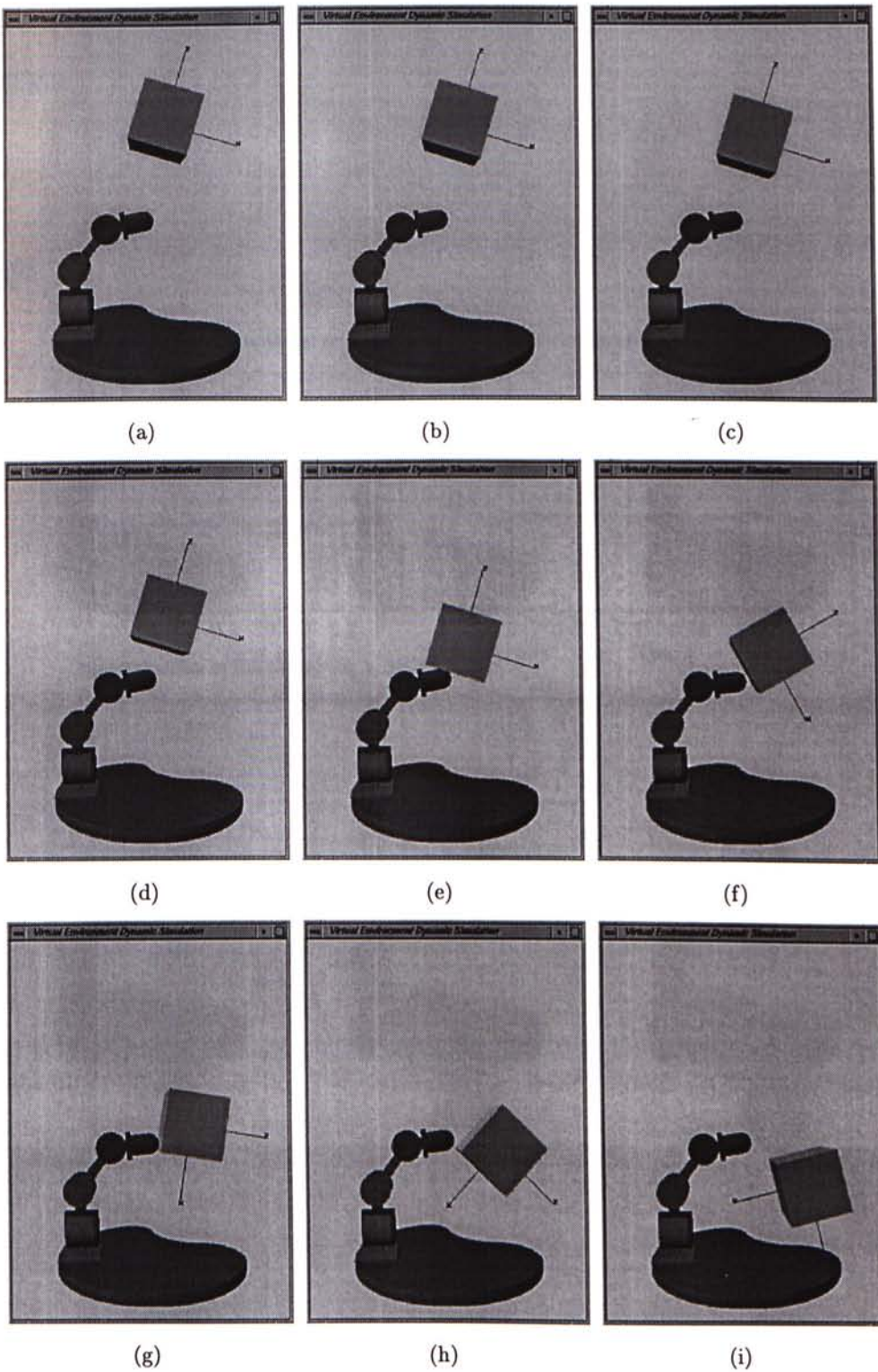


Figure 6.17: Example 7: Collision between a finger and an object with mass  $m_o = 0.5kg$ .

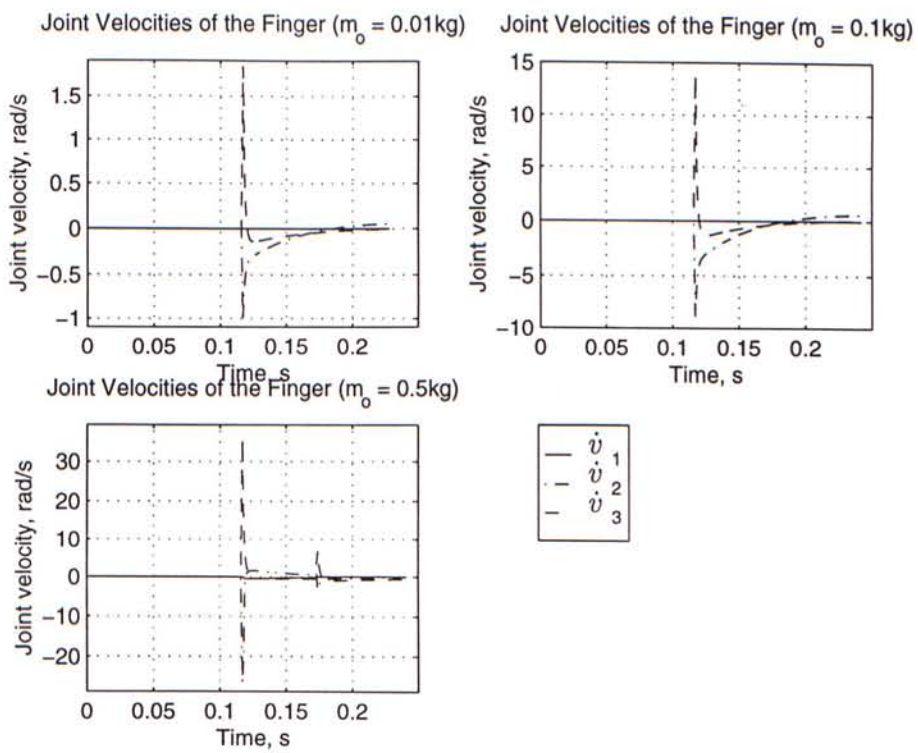


Figure 6.18: Change of joint velocities of the finger in the process of collision with different object mass.



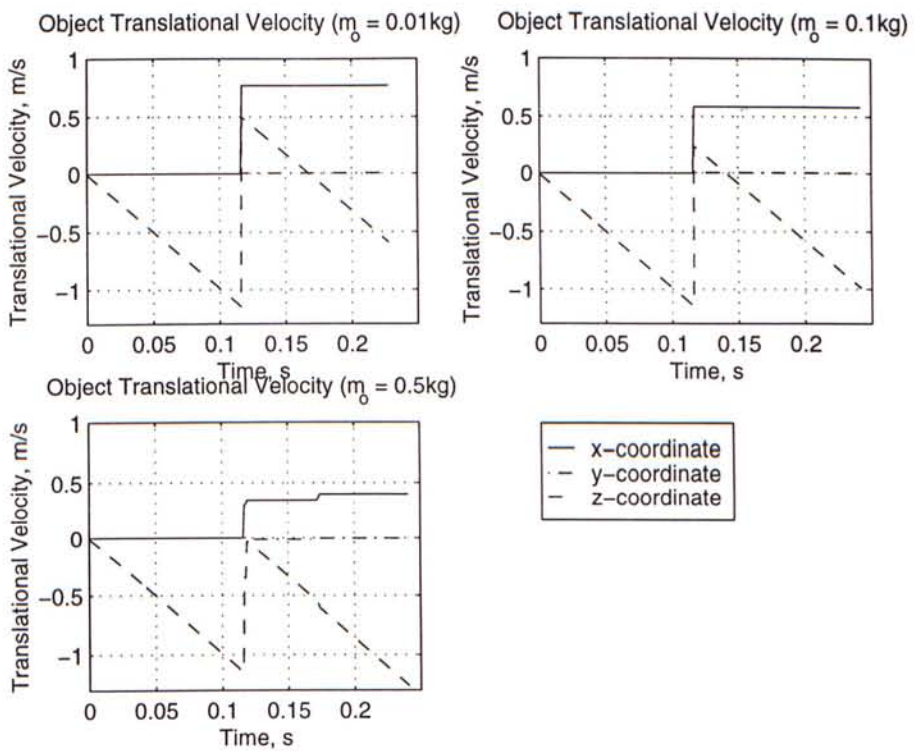


Figure 6.19: Change of object translational velocity in the process of collision with different object mass.

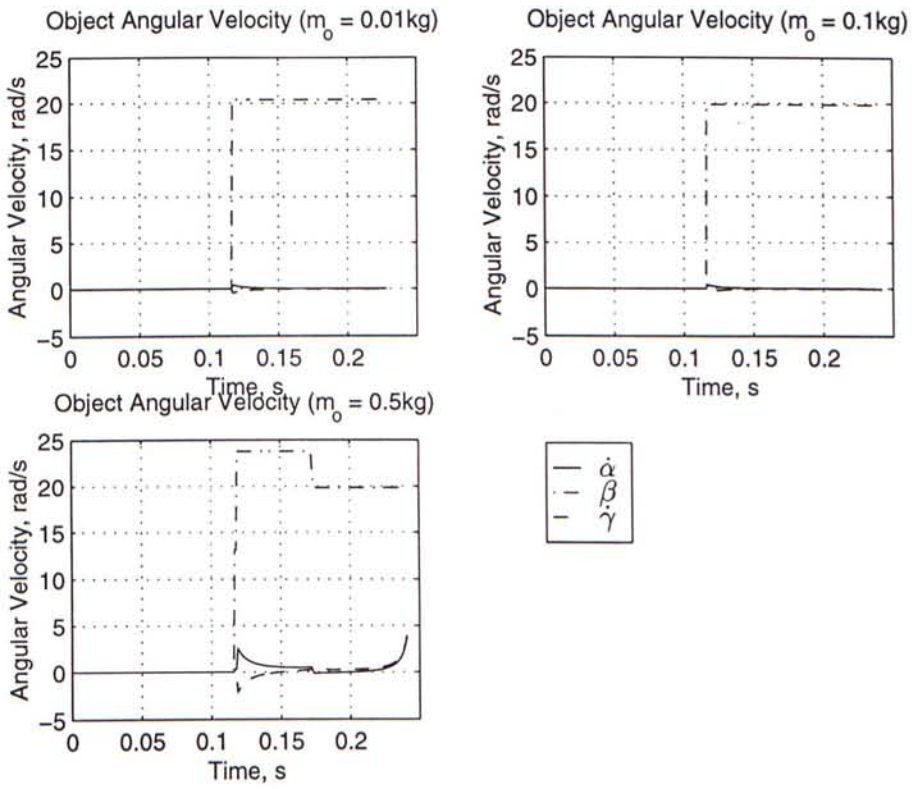


Figure 6.20: Change of object differential ZYZ Euler angles in the process of collision with different object mass.

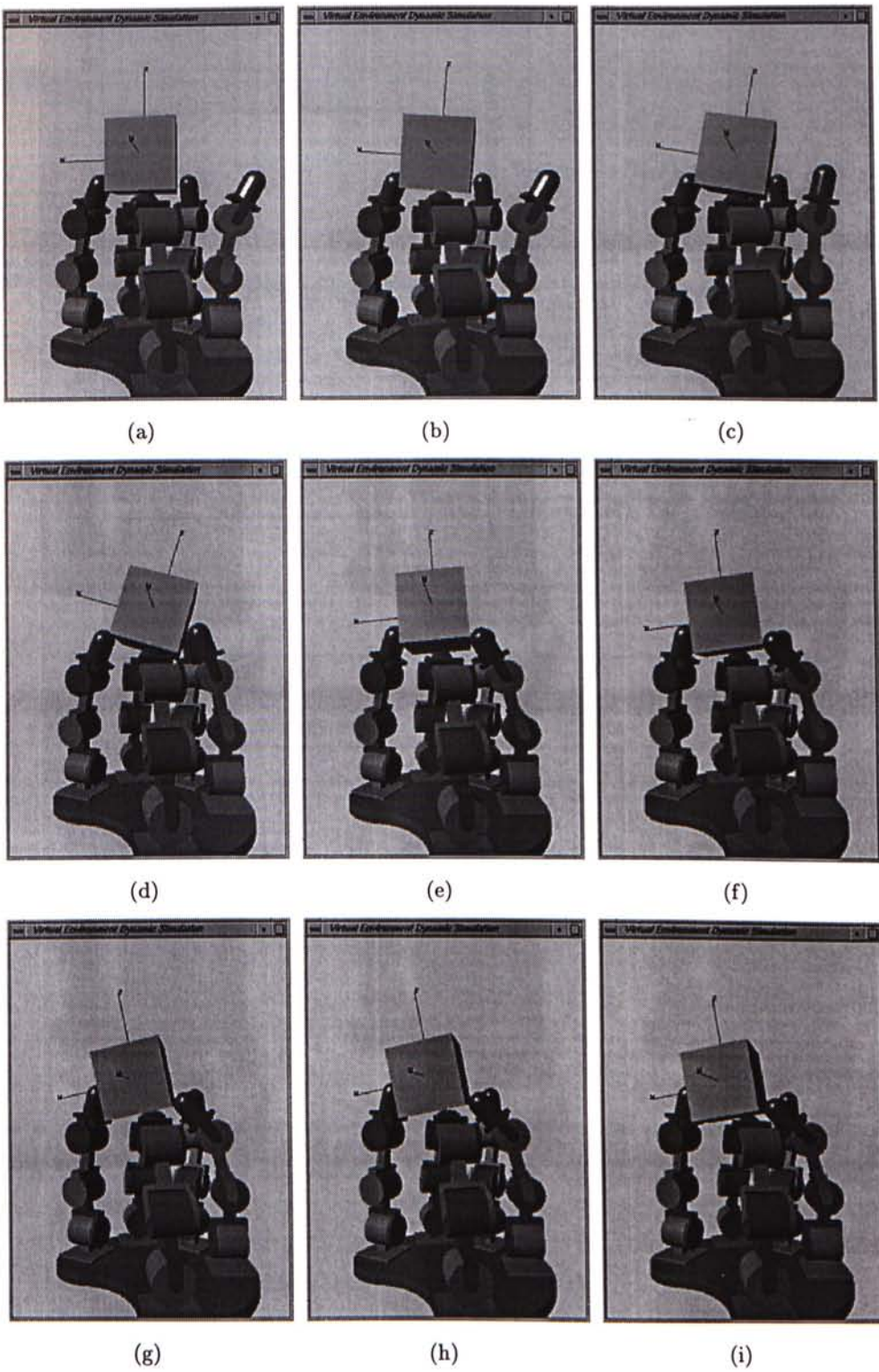


Figure 6.21: Example 8: Dexterous manipulation with mixed contacts and collisions.



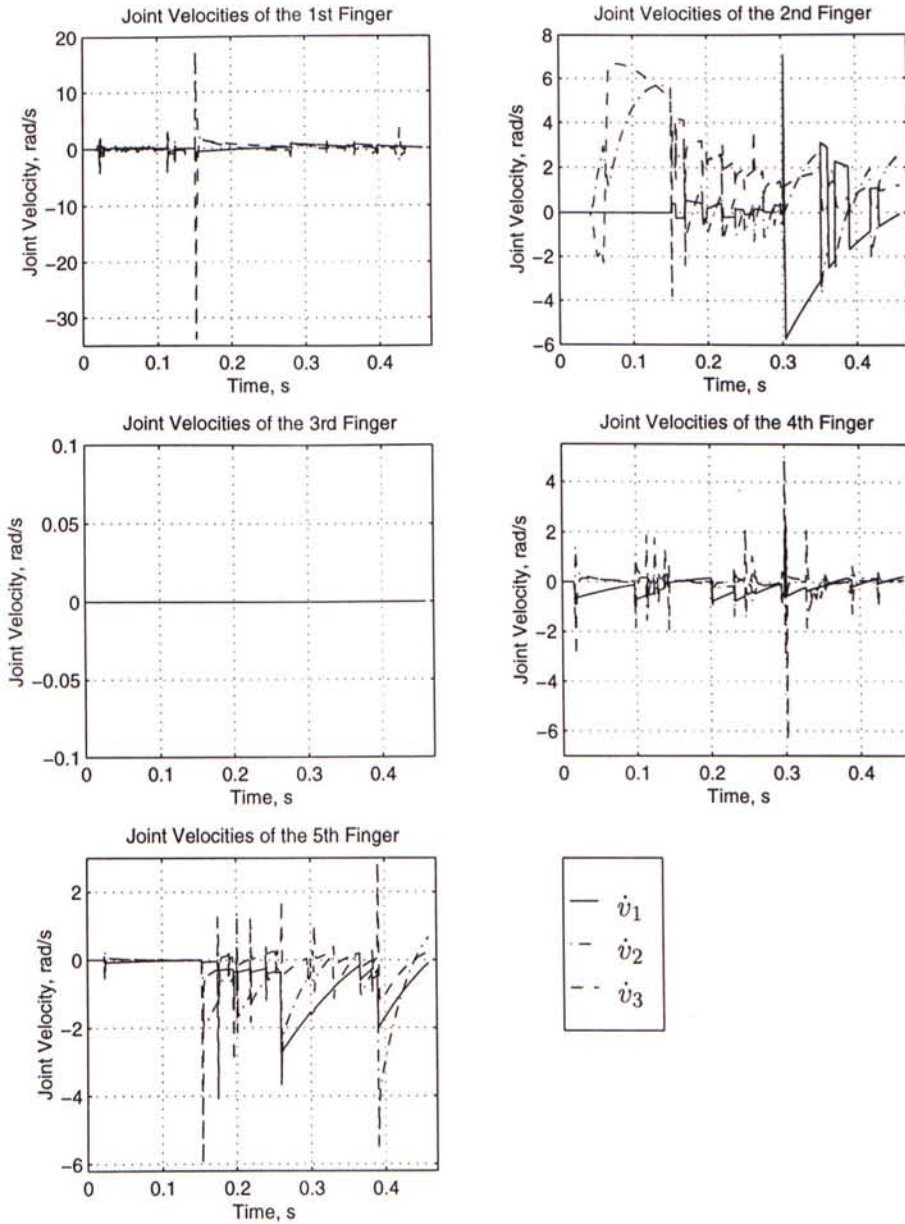


Figure 6.22: Change of joint velocities of the finger in the process of mixed contacts and collisions.

## 6.6 Dexterous Manipulation Motions

Two dextrous manipulation motions of the simulated five-fingered robot hand are shown in Figures 6.23 and 6.25. The changes of the object configuration are shown in Figures 6.24 and 6.26. Example 9 shows the object are being pushed by two fingers to the

other sides of the robot hand. Example 10 shows that the robot hand is rotating the object. During the rotation, the contact positions of many fingers are kept changing. For example, the last (pinkie) finger initially touches the center part of the bottom edge. At the end of simulation, it slides to the side of the object.

## 6.7 Summary

In this chapter, various motions of the simulated five-fingered robot hand are simulated. They involve the changes in grasping configuration, rolling and sliding contacts, collision, as well as simple dextrous manipulations. The dynamic responses of these motions are also presented such as the changes of constraint force and that of the local contact parameters. These simulation results verify that our dynamic simulation is both efficient and accurate compared to other dynamic simulators.

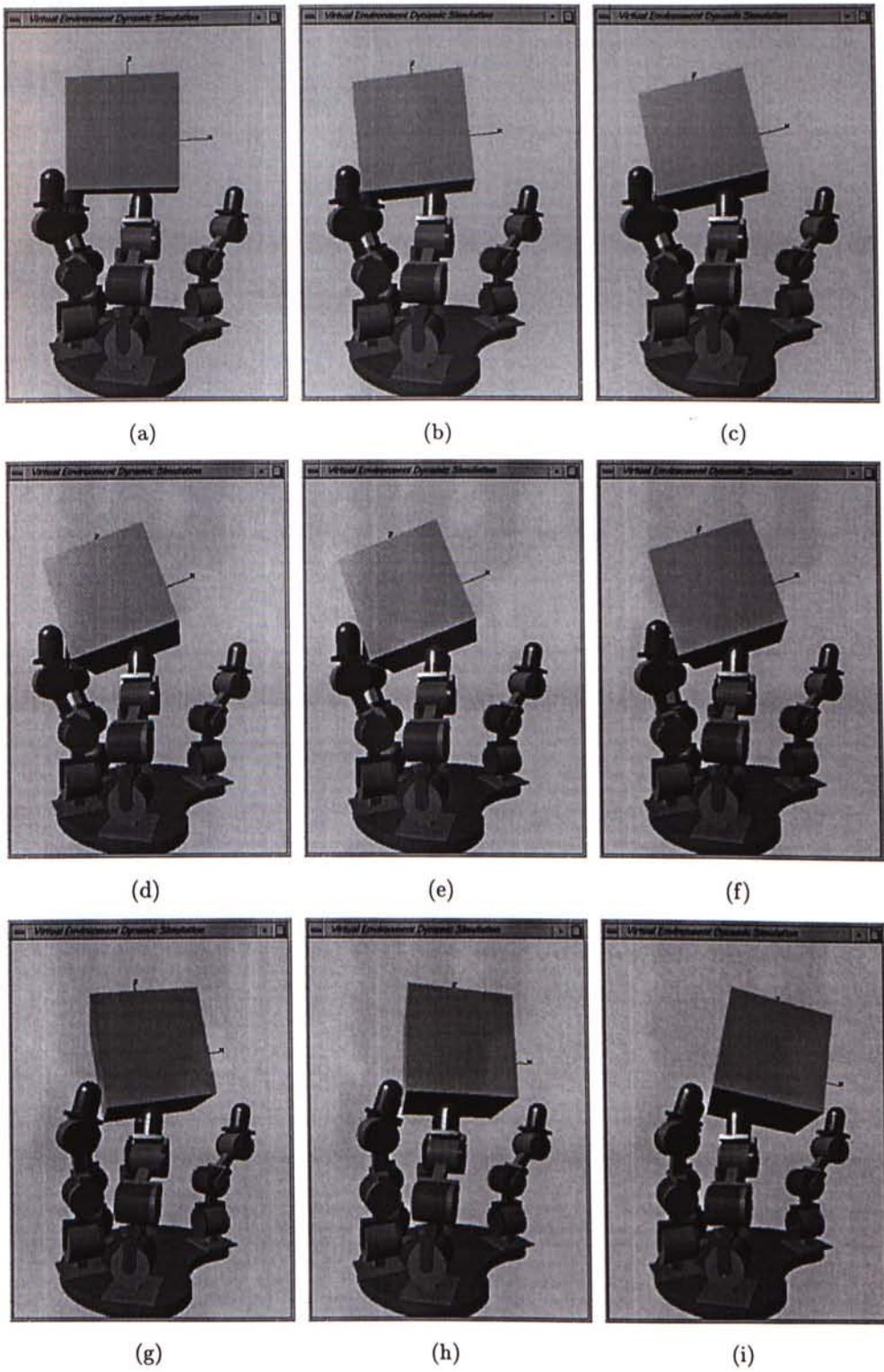
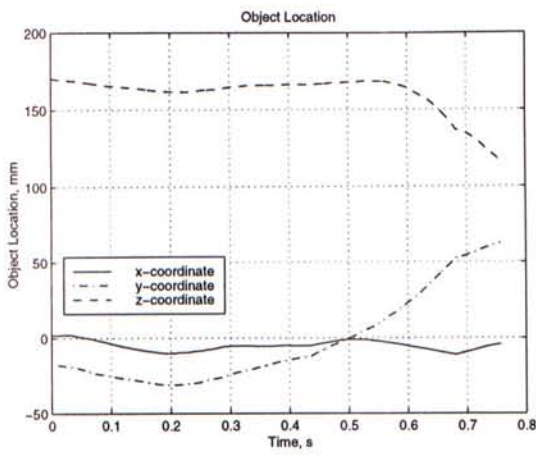
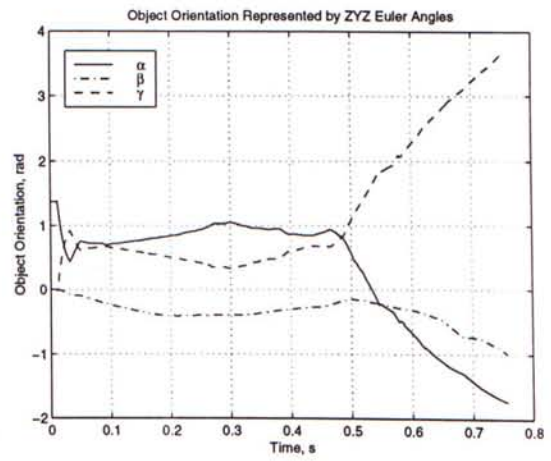


Figure 6.23: Example 9: Simple dextrous manipulation.





(a) Change of object location.



(b) Change of object orientation.

Figure 6.24: Change of object configuration in the process of a simple dextrous manipulation shown in Example 9.

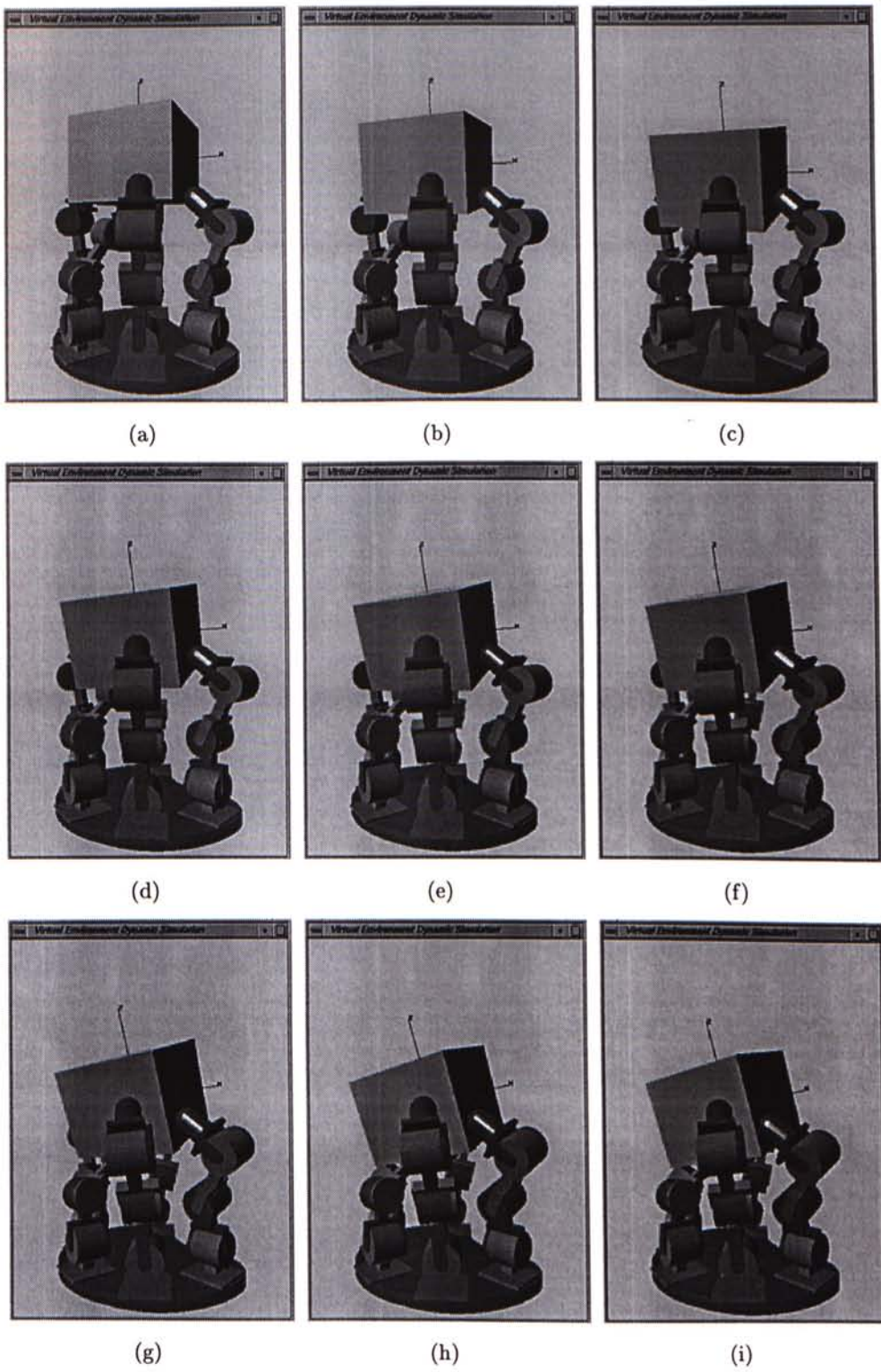
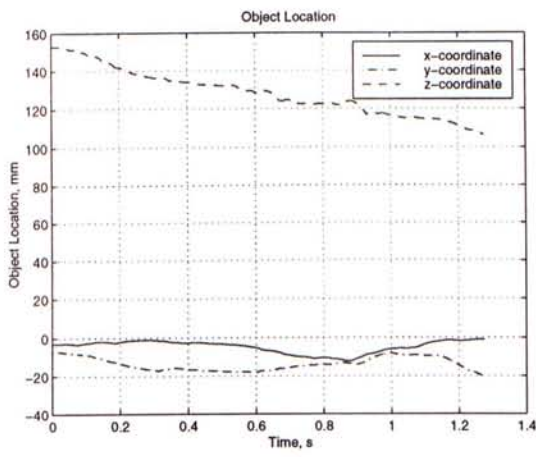
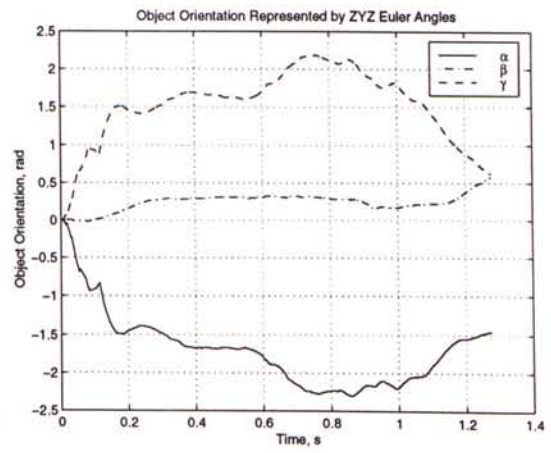


Figure 6.25: Example 10: Simple dextrous manipulation.



(a) Change of object location.



(b) Change of object orientation.

Figure 6.26: Change of object configuration in the process of a simple dextrous manipulation shown in Example 10.



# Chapter 7

## Conclusions

### 7.1 Summary of Contributions

In this dissertation, we have presented a new dynamic modeling method for the general multi-fingered robot hand system, which combines free motion, collision, and contact motions. Furthermore, we have developed a dynamic simulator for a multi-fingered robot hand system based on its full kinematics and dynamics.

The most difficult part in the dynamic modeling for the multi-fingered robot hand system is to handle the interactions between the fingers and the manipulated object. These interactions may be collision, sticking contact, rolling or sliding contact and they always change rapidly in dextrous grasping and manipulation motions. In this dissertation, the dynamics formulation is divided into two parts: *contact modeling* and *collision modeling*. Firstly, The kinematics and dynamics of a multi-fingered robot hand manipulating an object are derived under different contact motions. Secondly, the velocity discontinuity caused by collision is handled by an improved collision model.

In the contact modeling, we have derived a unified formulation of different contact constraints. Using this new formulation method, we derive a general contact constraint to represent all contact cases occurred in multi-fingered robot hand manipulation. They include free motion, sticking contact, rolling and sliding contacts. It is achieved by using different *contact mode selection matrix*  $C_{Mi}$  (equation (2.15)) in the general contact constraint. Based on this formulation, we are able to simulate the rapid changing of grasping configurations or contact kinematic constraints effectively. The contact constraints of the whole system are then extended to the general velocity and acceleration constraints of the system by applying the Montana's contact equations (equation (2.4)).

To model the collision, the impulse-based method proposed by Mirtich (1996) is

modified and extended. This method is mainly used to handle the interaction that the relative interacting velocity is large so there are sudden changes of velocities. The derived equations provide a simple but accurate model to handle the collision arisen in multi-fingered robot hand grasping or manipulation motions. The first order differential collision equations for the finger-object system are derived and different collision integration methods are presented. Special treatments are presented to handle the multiple collisions and the mixed contact-collision cases.

Based on the equations derived in the contact modeling and the collision modeling, a dynamic simulation system of a five-fingered robot hand has been developed. This system incorporates the knowledge of multi-body dynamics, contact modeling, physics-based simulation, human control, and virtual reality. It is able to efficiently simulate both the multi-fingered grasping and dextrous manipulation of the five-fingered robot hand compared to other dynamic simulators. In the constraint-based simulation, a new contact model is presented to describe the change of contact constraints during simulation. Combining the contact modeling equations and this contact model, the dynamic simulation system is successful to handle the rapid changing of grasp configuration and various contact constraints. Using the data glove as one of the input devices and the virtual environment as the output medium, the system can easily simulate and examine the dynamic behaviors of the transferred human skills to the virtual robot hand interactively. The performance of the dynamic simulation is verified by the simulation examples and results presented in Chapter 6.

## 7.2 Future Work

The work in this dissertation can be further extended in the improvement of the current system and its potential applications. They are:

### 7.2.1 Improvement of Current System

- **Force feedback:** In the current dynamic simulation system, only visual information of the simulation results is feedback to the human operator for motion guidance and error adjustment. It is one kind of information to help us to do the dextrous manipulation motion. However, the operator does not feel that he is manipulating an object since no force feedback actuator is available on the data glove. In order to have a better quality of the feedback loop, the force feedback can be developed in the dynamic simulation system. There are some



commercial force actuators for data glove have been already available such as the *CyberGrasp*<sup>1</sup>.

- **Speed Improvement:** The computational requirement of the dynamic simulation is quite high. The graphical workstation needs to support the three dimensional fully rendered virtual environment, enormous number of matrix computations, and several computational intensive numerical integration subroutines. The speed performance of current dynamic simulation is only satisfactory and it may affect the interactive performance of the system. The speed of the system can be improved to reformulate all the kinematic and dynamic equation in recursive forms or port the current system to the multi-processes workstation for parallel computation.
- **Support various shapes of grasped object:** As described in Chapter 5, the current interference detection is done based on simple geometric relationships of the fingertips and the grasped object. They are restricted in certain shapes. The dynamic simulation system can be extended to handle various shapes of grasped object or fingertip by incorporating a much sophisticated interference detection algorithm such as that proposed in (Lin, 1993) and (Mirtich, 1997). They are effective to detect interaction between convex polyhedral objects. Using these interaction detection algorithms, the handling of interactions between the fingers and the grasped object can be also extended to the interactions between fingers to fingers or links to links.
- **Experimental analysis of the simulation system performance:** In order to have a more analytical evaluation of the performance of the dynamic simulation system, experiments on multi-fingered grasping and dextrous manipulation can be done based on the real five-fingered robot hand. The experimental data can be used to compare to the simulation results for further analysis.

### 7.2.2 Applications

- **Embed in task teaching system:** The current developed dynamic simulation system working in the virtual environment for the real five-fingered robot hand system can be embedded to the task teaching system DMTS proposed in (Liu, 1995). Learning and optimization capabilities are required to added to the current simulation system.

---

<sup>1</sup>Further information of this commercial product can be referred to its internet homepage [http://www.virtex.com/prod\\_cybergrasp.html](http://www.virtex.com/prod_cybergrasp.html).



- **Testbed for testing control algorithms:** Our proposed dynamic modeling methods can help us to develop another dynamic simulation system for multiple manipulators cooperation system and the multi-legged walking robot since they are similar to the multi-fingered robot hand in mechanical structure and behaviors. The dynamic simulation system can be used as a testbed to evaluate the performance of different control algorithms such as (Qin et al., 1997) and (Liu and Arimoto, 1998) applied to these closed chain mechanisms. It is more safe, cost-effective, and efficient comparing to perform experiments in real mechanisms.
- **Internet-based simulation system:** In the final project of Jong (1998), an internet-based robot control system was developed. The control process can be done remotely through the world wide web. The experimental and research resources can then be fully utilized. Using the same idea, the current dynamic simulation can be ported to the internet and makes it to be the internet-based simulation system. In this way, the dynamic simulation of the multi-fingered robot hand can be done in the world wide web browser and it can facilitate the development of multi-fingered robot hand.

## Appendix A

# Montana's Contact Equations for Finger-object Contact

Using the notations shown in Figure 2.7, the Montana's contact equations are written as following form:

$$\begin{cases} \dot{u}_{li} &= M_{li}^{-1}(K_{li} + \tilde{K}_{oi})^{-1} \left( \begin{bmatrix} -\omega_y \\ \omega_x \end{bmatrix} - \tilde{K}_{oi} \begin{bmatrix} v_x \\ v_y \end{bmatrix} \right) \\ \dot{u}_{oi} &= M_{oi}^{-1} {}_{oi}^{\hat{R}} {}_{li}^{\hat{R}} (K_{li} + \tilde{K}_{oi})^{-1} \left( \begin{bmatrix} -\omega_y \\ \omega_x \end{bmatrix} + K_{li} \begin{bmatrix} v_x \\ v_y \end{bmatrix} \right) \\ \dot{\psi} &= \omega_z + T_{li} M_{li} \dot{u}_{li} + T_{oi} M_{oi} \dot{u}_{oi} \end{cases} \quad (\text{A.1})$$

### A.1 Local Coordinates Charts

Let  $c_{li}$  and  $c_{oi}$  be the local coordinate chart with respect to the fingertip surface and the object surface if the fingertip is modeled as a hemisphere and the object is modeled as a rectangular object, therefore:

$$c_{li}(u_{lix}, u_{liy}) = \begin{bmatrix} r \sin(u_{lix}) \\ r \cos(u_{lix}) \sin(u_{liy}) \\ r \cos(u_{lix}) \cos(u_{liy}) \end{bmatrix} \quad (\text{A.2})$$

where  $r$  is the radius of the fingertip, and:

$$c_{oi}(u_{oix}, u_{oiy}) = \begin{bmatrix} u_{oix} \\ u_{oiy} \\ 0 \end{bmatrix} \quad (\text{A.3})$$

To simplify the notations, we use  $\begin{bmatrix} u_f \\ v_f \end{bmatrix} = \begin{bmatrix} u_{lix} \\ u_{liy} \end{bmatrix}$ ,  $\begin{bmatrix} u_o \\ v_o \end{bmatrix} = \begin{bmatrix} u_{oix} \\ u_{oiy} \end{bmatrix}$ ,  $c_{li} = c_f$ , and  $c_{oi} = c_o$  to rewrite equations (A.2) and (A.3):

$$c_f(u_f, v_f) = \begin{bmatrix} r \sin(u_f) \\ r \cos(u_f) \sin(v_f) \\ r \cos(u_f) \cos(v_f) \end{bmatrix}, \text{ and } c_o(u_o, v_o) = \begin{bmatrix} u_o \\ v_o \\ 0 \end{bmatrix} \quad (\text{A.4})$$

## A.2 Curvature, Torsion and Metric Tensors

The definitions of the metric, curvature, and torsion tensor are shown as follows:

- **Metric tensor:**

$$M_i = \begin{bmatrix} \left\| \frac{\partial c_i}{\partial u} \right\| & 0 \\ 0 & \left\| \frac{\partial c_i}{\partial v} \right\| \end{bmatrix}$$

- **Curvature tensor:**

$$K_i = \begin{bmatrix} \frac{\frac{\partial c_i}{\partial u} \cdot \frac{\partial n_i}{\partial u}}{\left\| \frac{\partial c_i}{\partial u} \right\|^2} & \frac{\frac{\partial c_i}{\partial u} \cdot \frac{\partial n_i}{\partial v}}{\left\| \frac{\partial c_i}{\partial u} \right\| \left\| \frac{\partial c_i}{\partial v} \right\|} \\ \frac{\frac{\partial c_i}{\partial v} \cdot \frac{\partial n_i}{\partial u}}{\left\| \frac{\partial c_i}{\partial u} \right\| \left\| \frac{\partial c_i}{\partial v} \right\|} & \frac{\frac{\partial c_i}{\partial v} \cdot \frac{\partial n_i}{\partial v}}{\left\| \frac{\partial c_i}{\partial v} \right\|^2} \end{bmatrix}$$

where

$$n_i = \frac{\frac{\partial c_i}{\partial u} \times \frac{\partial c_i}{\partial v}}{\left| \frac{\partial c_i}{\partial u} \times \frac{\partial c_i}{\partial v} \right|}$$



- Torsion tensor:

$$T_i = \begin{bmatrix} \frac{\partial c_i}{\partial v} \cdot \frac{\partial^2 c_i}{\partial u^2} & \frac{\partial c_i}{\partial v} \cdot \frac{\partial^2 c_i}{\partial u \partial v} \\ \left\| \frac{\partial c_i}{\partial u} \right\|^2 \frac{\partial c_i}{\partial v} & \left\| \frac{\partial c_i}{\partial u} \right\| \left\| \frac{\partial c_i}{\partial v} \right\|^2 \end{bmatrix}$$

Using these definitions, we can determine the metric, curvature, and torsion tensors of the fingertip surface and they are:

- Metric tensor:

$$M_{li} = \begin{bmatrix} r & 0 \\ 0 & r \cos(u_f) \end{bmatrix}$$

- Curvature tensor:

$$K_{li} = \begin{bmatrix} \frac{1}{r} & 0 \\ 0 & \frac{1}{r} \end{bmatrix}$$

- Torsion tensor:

$$T_{li} = \begin{bmatrix} 0 & -\frac{1}{r} \tan(u_f) \end{bmatrix}$$

Similarly, the metric, curvature, and torsion tensors of the object surface and they are:

- Metric tensor:

$$M_{oi} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

- Curvature tensor:

$$K_{oi} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

- Torsion tensor:

$$T_{oi} = \begin{bmatrix} 0 & 0 \end{bmatrix}$$

### A.3 Montana's Contact Equations

Since  ${}^{oi}\hat{R}$  is equal to:

$${}^{oi}\hat{R} = \begin{bmatrix} \cos \psi & -\sin \psi \\ -\sin \psi & -\cos \psi \end{bmatrix}$$

Therefore, the term  $\tilde{K}_{oi}$  can be determined as:

$$\begin{aligned} \tilde{K}_{oi} &= \begin{bmatrix} c_\psi & -s_\psi \\ -s_\psi & -c_\psi \end{bmatrix} K_{oi} \begin{bmatrix} c_\psi & -s_\psi \\ -s_\psi & -c_\psi \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \end{aligned} \tag{A.5}$$

Using equations (A.1), we can compute  $\begin{bmatrix} \dot{u}_f \\ \dot{v}_f \end{bmatrix}$ ,  $\begin{bmatrix} \dot{u}_o \\ \dot{v}_o \end{bmatrix}$ , and  $\dot{\psi}$  as:

- $\begin{bmatrix} \dot{u}_f \\ \dot{v}_f \end{bmatrix}$ :

$$\begin{aligned} \begin{bmatrix} \dot{u}_f \\ \dot{v}_f \end{bmatrix} &= \begin{bmatrix} r & 0 \\ 0 & r \cos(u_f) \end{bmatrix}^{-1} \begin{bmatrix} \frac{1}{r} & 0 \\ 0 & \frac{1}{r} \end{bmatrix}^{-1} \begin{bmatrix} -\omega_y \\ \omega_x \end{bmatrix} \\ &= \begin{bmatrix} -\omega_y \\ \frac{\omega_x}{\cos(u_f)} \end{bmatrix} \end{aligned}$$

- $\begin{bmatrix} \dot{u}_o \\ \dot{v}_o \end{bmatrix}$ :

$$\begin{aligned} \begin{bmatrix} \dot{u}_o \\ \dot{v}_o \end{bmatrix} &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} c_\psi & -s_\psi \\ -s_\psi & -c_\psi \end{bmatrix} \begin{bmatrix} \frac{1}{r} & 0 \\ 0 & \frac{1}{r} \end{bmatrix}^{-1} \left( \begin{bmatrix} -\omega_y \\ \omega_x \end{bmatrix} + \begin{bmatrix} \frac{1}{r} & 0 \\ 0 & \frac{1}{r} \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} \right) \\ &= \begin{bmatrix} (v_x - r\omega_y)c_\psi - (v_y + r\omega_x)s_\psi \\ -(v_y + r\omega_x)c_\psi + (-v_x + r\omega_y)s_\psi \end{bmatrix} \end{aligned}$$

- $\dot{\psi}$ :

$$\begin{aligned}\dot{\psi} &= \omega_z + \begin{bmatrix} 0 & -\frac{1}{r} \tan(u_f) \end{bmatrix} \begin{bmatrix} r & 0 \\ 0 & r \cos(u_f) \end{bmatrix} \begin{bmatrix} -\omega_y \\ \frac{\omega_x}{\cos(u_f)} \end{bmatrix} \\ &= \omega_z - \omega_x \tan(u_f)\end{aligned}$$

The matrix form of the Montana's contact equations are thus:

$$\begin{bmatrix} \dot{u}_f \\ \dot{v}_f \\ \dot{u}_o \\ \dot{v}_o \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & \sec(u_f) & 0 & 0 \\ c_\psi & -s_\psi & 0 & -rs_\psi & -rc_\psi & 0 \\ -s_\psi & -c_\psi & 0 & -rc_\psi & rs_\psi & 0 \\ 0 & 0 & 0 & -\tan(u_f) & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$



# Appendix B

## Finger Dynamics

### B.1 Forward Kinematics of a Robot Finger

Figure (B.1) shows a finger of the five-fingered robot hand in our department. The coordinate frame for each link is assigned by the modified Denavit-Hartenberg method. The link masses are assumed to be concentrated at the ends of the links and the joint variable is  $\theta = [\theta_1 \ \theta_2 \ \theta_3]^T$

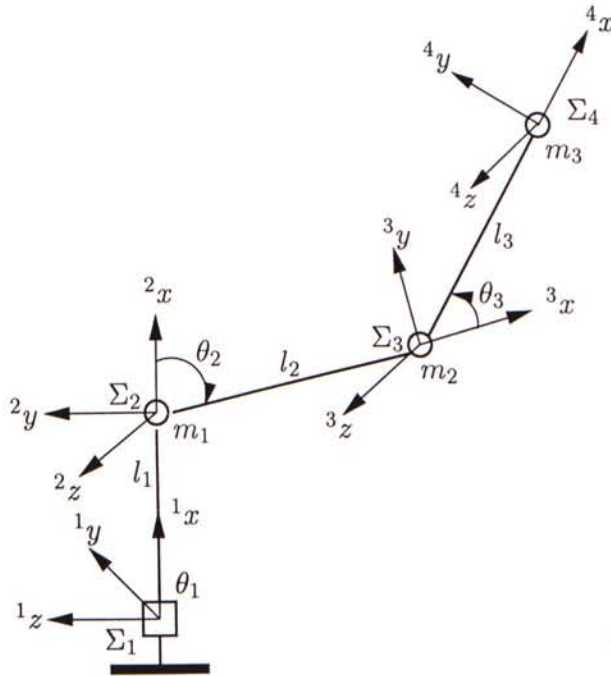


Figure B.1: Coordinate frames assignment of a finger of the five-fingered robot hand.

### B.1.1 Link-coordinate Transformation

By using equation (2.7), the link-coordinate transformation matrix coordinate frame  $\Sigma_2$  to coordinate frame  $\Sigma_1$  is:

$${}^1_2T = \begin{bmatrix} c_1 & 0 & s_1 & l_1c_1 \\ s_1 & 0 & -c_1 & l_1s_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

since  $d_1 = 0$  and  $\alpha_1 = \pi/2$ . Similarly, the link-coordinate transformation matrix coordinate frame  $\Sigma_3$  to coordinate frame  $\Sigma_2$  and the link-coordinate transformation matrix coordinate frame  $\Sigma_4$  to coordinate frame  $\Sigma_3$  are:

$${}^2_3T = \begin{bmatrix} c_2 & -s_2 & 0 & l_2c_2 \\ s_2 & c_2 & 0 & l_2s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{B.1})$$

$${}^3_4T = \begin{bmatrix} c_3 & -s_3 & 0 & l_3c_3 \\ s_3 & c_3 & 0 & l_3s_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{B.2})$$

since  $d_2 = d_3 = 0$  and  $\alpha_2 = \alpha_3 = 0$ .

### B.1.2 Forward Kinematics

The forward kinematics of the finger can be determined by equation (2.6) as:

$$F(\theta) = \begin{bmatrix} R(\theta) & p(\theta) \\ 0_{1 \times 3} & 1 \end{bmatrix}$$

where

$$R(\theta) = \begin{bmatrix} c_1c_{23} & -c_1s_{23} & s_1 \\ s_1c_{23} & -s_1s_{23} & -c_1 \\ s_{23} & c_{23} & 0 \end{bmatrix} \quad (\text{B.3})$$

and

$$p(\theta) = \begin{bmatrix} c_1(l_1 + l_2c_2 + l_3c_{23}) \\ s_1(l_1 + l_2c_2 + l_3c_{23}) \\ l_2s_2 + l_3s_{23} \end{bmatrix}$$

## B.2 Dynamic Equation of a Robot Finger

### B.2.1 Kinetic and Potential Energy

After obtaining the homogeneous transformation matrix in the above section, the kinetic and potential energies of each link can be determined.

**Consider  $m_1$ :** Coordinates of  $m_1$  respect to coordinate frame  $\Sigma_1$  is:

$$\begin{bmatrix} x_{m1} \\ y_{m1} \\ z_{m1} \\ 1 \end{bmatrix} = {}^1_2T \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} l_1c_1 \\ l_1s_1 \\ 0 \\ 1 \end{bmatrix}$$

and its corresponding velocity respect to coordinate frame  $\Sigma_1$  is:

$$\begin{bmatrix} \dot{x}_{m1} \\ \dot{y}_{m1} \\ \dot{z}_{m1} \\ 1 \end{bmatrix} = \begin{bmatrix} -l_1s_1\dot{\theta}_1 \\ -l_1c_1\dot{\theta}_1 \\ 0 \\ 1 \end{bmatrix}$$

Therefore, the kinetic and potential energies are:

$$K_1 = \frac{1}{2}m_1(\dot{x}_{m1}^2 + \dot{y}_{m1}^2 + \dot{z}_{m1}^2) = \frac{1}{2}m_1l_1^2\dot{\theta}_1^2$$

$$P_1 = m_1gx_{m1} = m_1gl_1c_1$$

**Consider  $m_2$ :** Coordinates of  $m_2$  respect to coordinate frame  $\Sigma_1$  is:

$$\begin{bmatrix} x_{m2} \\ y_{m2} \\ z_{m2} \\ 1 \end{bmatrix} = {}^1_3T \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$



The corresponding kinetic and potential energies are:

$$\begin{aligned} K_2 &= \frac{1}{2}m_2(\dot{x}_{m2}^2 + \dot{y}_{m2}^2 + \dot{z}_{m2}^2) \\ &= \frac{1}{2}m_2 \left[ \left( l_1^2 + \frac{1}{2}l_2^2 + 2l_1l_2c_2 + \frac{1}{2}l_2^2 \cos(2\theta_2) \right) \dot{\theta}_1^2 + l_2^2 \dot{\theta}_2^2 \right] \\ P_2 &= m_2gx_{m2} = m_2gc_1(l_1 + l_2c_2) \end{aligned}$$

**Consider  $m_3$ :** Coordinates of  $m_3$  respect to coordinate frame  $\Sigma_1$  is:

$$\begin{bmatrix} x_{m3} \\ y_{m3} \\ z_{m3} \\ 1 \end{bmatrix} = {}^1_4T \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} c_1(l_1 + l_2c_2 + l_3c_{23}) \\ s_1(l_1 + l_2c_2 + l_3c_{23}) \\ l_2s_2 + l_3s_{23} \\ 1 \end{bmatrix}$$

The kinetic and potential energies are:

$$\begin{aligned} K_3 &= \frac{1}{2}m_3(\dot{x}_{m3}^2 + \dot{y}_{m3}^2 + \dot{z}_{m3}^2) \\ P_3 &= m_3gx_{m3} = m_3gc_1(l_1 + l_2c_2 + l_3c_{23}) \end{aligned}$$

since  $K_3$  is so complicated and its closed form is not shown here.

### B.2.2 Lagrange's Equation

Using Lagrange's formulation method for a conservative system:

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{\theta}} - \frac{\partial L}{\partial \theta} = \tau \tag{B.4}$$

The dynamics of the finger can be formulated in the standard form of:

$$M(\theta)\ddot{\theta} + V(\theta, \dot{\theta}) + G(\theta) = \tau$$

where

$$M_{11} = m_1 l_1^2 + m_2 (l_1 + l_2 c_2)^2 + m_3 [l_1 + l_2 c_2 + l_3 c_{23}]^2$$

$$M_{12} = M_{13} = M_{21} = M_{31} = 0$$

$$M_{22} = m_2 l_2^2 + m_3 (l_2^2 + l_3^2 + 2l_2 l_3 c_3)$$

$$M_{23} = M_{32} = m_3 l_3 (l_2 c_3 + l_3)$$

$$M_{33} = m_3 l_3^2$$

$$V_1 = -2m_2 l_2 s_2 (l_1 + l_2 c_2) \dot{\theta}_1 \dot{\theta}_2 - 2m_3 (l_1 + l_2 c_2 + l_3 c_{23}) (l_2 s_2 + l_3 s_{23}) \dot{\theta}_1 \dot{\theta}_2 - 2l_3 m_3 s_{23} (l_1 + l_2 c_2 + l_3 c_{23}) \dot{\theta}_1 \dot{\theta}_3$$

$$V_2 = m_2 l_2 s_2 (l_1 + l_2 c_2) \dot{\theta}_1^2 + m_3 (l_1 + l_2 c_2 + l_3 c_{23}) (l_2 s_2 + l_3 s_{23}) \dot{\theta}_1^2 - m_3 l_2 l_3 s_3 (2\dot{\theta}_2 \dot{\theta}_3 + \dot{\theta}_3^2)$$

$$V_3 = m_3 l_3 \left[ \frac{1}{2} l_2 s_3 + l_1 s_{23} + \frac{1}{2} l_3 \sin[2(\theta_2 + \theta_3)] + \frac{1}{2} l_2 \sin(2\theta_2 + \theta_3) \right] \dot{\theta}_1^2 + m_3 l_2 l_3 s_3 \dot{\theta}_2^2$$

$$G_1 = -g s_1 [m_1 l_1 + m_2 (l_1 + l_2 c_2) + m_3 (l_1 + l_2 c_2 + l_3 c_{23})]$$

$$G_2 = -g c_1 [m_2 l_2 s_2 + m_3 (l_2 s_2 + l_3 s_{23})]$$

$$G_3 = -g m_3 l_3 c_1 s_{23}$$

# Appendix C

## Simulation Configurations

### C.1 Geometric models

The specification of the geometric model of a finger of the five-fingered robot hand used in the simulation is shown in Figure C.1. The dimensions of this model are fully based on the real five-fingered robot hand and the unit shown in figure is in millimeter (mm). The corresponding three dimensional solid model of the robot hand constructed by AutoCAD is shown in Figure C.2.

### C.2 Physical Parameters

The setting of physical parameters applied in the simulation are:

- Mass of the three links of each finger are  $0.07kg$ .
- Gravitational constant,  $g = 9.8m/s^2$
- Coefficient of friction,  $\mu = 0.6$
- Coefficient of restitution,  $e = 0.8744 + 0.0756 \times 2.7182^{k1}$  where  $k$  is the inter-penetration distance between the contacted finger and the object.

---

<sup>1</sup>If  $e$  is less than one, it implies the collision between rigid bodies is not perfectly elastic and energy is lost during collision. In our simulation, if collision appears for a long period to time, the finger will penetrate into the object because of the energy loss. In order to prevent the severe inter-penetration happened, the restitution is set to be an exponential function (Mirtich, 1996) of the penetration distance rather than to end a constant.



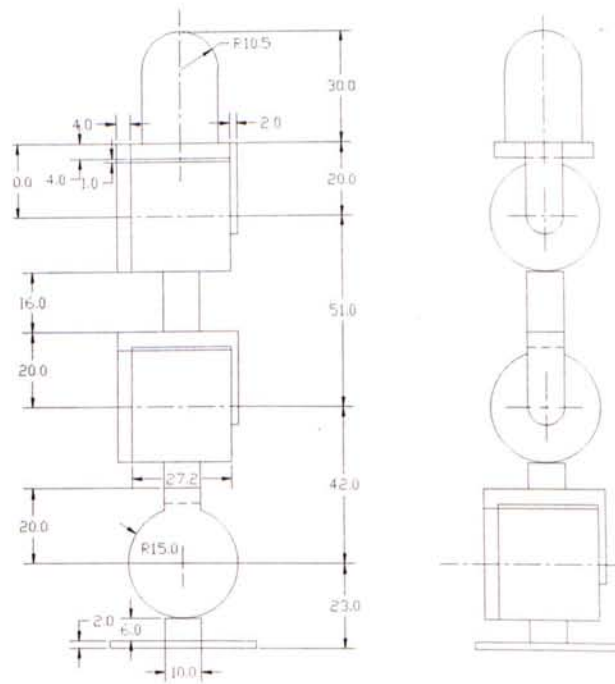


Figure C.1: Specification of the geometric model of a finger.

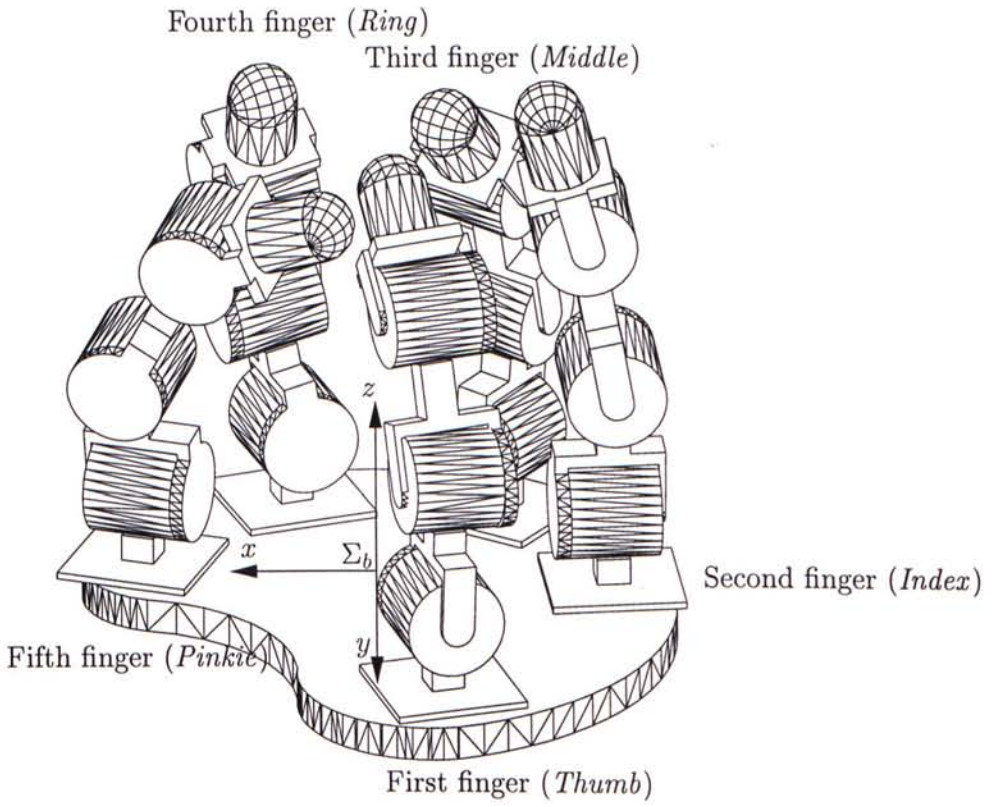


Figure C.2: Three dimensional solid model of the five-fingered robot hand constructed by AutoCAD.

## C.3 Simulation Parameters

The simulation parameters setting are:

- P-gain of the PD+gravity finger control law is a  $3 \times 3$  diagonal matrix and all the diagonal terms are 0.15.
- D-gain of the PD+gravity finger control law is also a  $3 \times 3$  diagonal matrix with diagonal values equal to 0.02.
- $\alpha$  used in stabilizing the velocity constraint is set to be 0.1.
- Collision threshold is  $0.01m/s$ .
- Both time steps used in the constraint-based and impulse-based simulations are equivalent and they are 0.001s.



# Bibliography

- Ali, M. S., Kyriakopoulos, K. J., and Stephanou, H. E. (1993). The kinematics of the anthropot-2 dextrous hand. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3705–3710.
- Anitescu, M., Cremer, J. F., and Potra, F. A. (1995). Formulating 3D contact dynamics problems. Technical Report Computational Mathematics 80/1995, Department of Mathematics, The University of Iowa.
- Baiardi, P., Cannata, G., Casalino, G., and Pagano, P. (1993). Modelling control phenomena within the dynamic simulation of advanced robotic structures. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 196–203.
- Baraff, D. (1992). *Dynamic Simulation of Non-penetrating Rigid Bodies*. PhD thesis, Cornell University.
- Baraff, D. (1993). Issues in computing contact forces for non-penetrating rigid bodies. *Algorithmica*, 10:292–352.
- Baraff, D. (1994). Fast contact force computation for non-penetrating rigid bodies. In *SIGGRAPH 94*, pages 23–34.
- Baumgarte, J. (1972). Stabilization of constraints and integrals of motion in dynamical systems. *Computer Methods in Applied Mechanics and Engineering*, 1:1–16.
- Bergamasco, M., Degl’Innocenti, P., and Bucciarelli, D. (1994). A realistic approach for grasping and moving virtual objects. In *IROS Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems: Advanced Robotic Systems and the Real World*, pages 717–724.
- Bhatt, V. and Koechling, J. (1995a). Partitioning the parameter space according to different behaviors during three-dimensional impacts. *Transactions of the ASME*, 62:740–746.

- Bhatt, V. and Koechling, J. (1995b). Three-dimensional frictional rigid-body impact. *Journal of Applied Mechanics*, 62.
- Brach, R. M. (1991). *Mechanical impact dynamics: rigid body collisions*. John Wiley and Sons.
- Brenan, K., Campbell, S., and Petzold, L. (1989). *Numerical solution of initial-value problems in differential-algebraic equations*. North-Holland.
- Brown, J. M. and Colgate, J. E. (1997). Passive implementation of multibody simulations for haptic display. In *Proceedings of the 1997 ASME international mechanical engineering congress and exhibition*, pages 85–92.
- Cadoz, C., Luciani, A., and Florens, J. L. (1993). Cordis-anima: A modeling and simulation system for sound and image synthesis - the general formalism. *Computer Music Journal*, 17:19–29.
- Cai, C. and Roth, B. (1987). On the spatial motion of rigid bodies with point contact. In *Proceedings of the International Conference on Robotics and Automation*, pages 686–695.
- Cai, C. and Roth, B. (1988). On the spatial motion of rigid bodies with line contact. In *Proceedings of the International Conference on Robotics and Automation*, pages 1036–1041.
- Chang, B. and Colgate, J. E. (1997). Real-time impulse-based simulation of rigid body systems for haptic display. In *Proceedings of the 1997 ASME international mechanical engineering congress and exhibition*.
- Chin, H. S. (1995). *Stabilization methods for simulations of constrained multibody system*. PhD thesis, The University of British Columbia, Vancouver, Canada.
- Cole, A. A., Hauser, J. E., and Sastry, S. S. (1989). Kinematics and control of multi-fingered hands with rolling contact. *IEEE Transaction on Robotics and Automation*, 34(4):398–404.
- Cole, A. A., Hsu, P., and Sastry, S. S. (1992). Dynamic control of sliding by robot hands for regrasping. *IEEE Transaction on Robotics and Automation*, 8(1):42–52.
- Cottle, R. K., Pang, J. S., and Stone, R. E. (1992). *The Linear Complementarity Problem*. Academic Press.



- Craig, J. J. (1986). *Introduction to robotics: mechanics and control*. Addison Wesley Publishing Company.
- Craig, J. J. (1989). *Introduction to Robotics: Mechanics and Control*. Addison-Wesley, 2nd edition.
- Cremer, J. F. (1989). An architecture for general purpose physical system simulation - integrating geometry, dynamics and control. Technical Report 89-987, Department of Computer Science, Cornell University.
- Cremer, J. F. and Stewart, A. J. (1989). The architecture of newton: A general purpose dynamic simulator. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1806–1811.
- Cremer, J. F. and Vaněček, G. (1994). Isaac: Building simulations for virtual environments. In *Proceedings of the IFIPTC 5 WG 5.10 International Workshop on Virtual Simulation Workshop*.
- de Jalón Javier, G. and Eduardo, B. (1994). *Kinematic and dynamic simulation of multibody systems: The real-time challenge*. Springer-Verlag.
- Featherstone, R. (1983). The calculation of robot dynamics using articulated body inertias. *The International Journal of Robotic Research*, 2:13–30.
- Featherstone, R. (1987). *Robot dynamics algorithms*. Kluwer Academic Publishers.
- Fujimoto, Y. and Kawamura, A. (1995). Three dimensional digital simulation and autonomous walking control for eight-axis biped robot. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 2877–2884.
- Garcia Alonso, A., Serrano, N., and Flaquer, J. (1994). Solving the collision detection problem. *IEEE computer graphics and applications*, 14(3):36–43.
- Gillespie, R. B. (1996). *Haptic Display of Systems with Changing Kinematic Constraints: The Virtual Piano Actions*. PhD thesis, Stanford University.
- Gillespie, R. B. and Colgate, J. E. (1997). A survey of multibody dynamics for virtual environment. In *Proceedings of the ASME International Mechanical Engineering Congress and Exhibition*.
- Goyal, S., Pinson, E. N., and Sinden, F. W. (1994a). Simulation of dynamics of interacting rigid bodies including friction i: General problem and contact model. *Engineering with Computers*, 10(3):162–174.



- Goyal, S., Pinson, E. N., and Sinden, F. W. (1994b). Simulation of dynamics of interacting rigid bodies including friction ii: Software system design and implementation. *Engineering with Computers*, 10(3):175–195.
- Greenwood, D. T. (1988). *Principles of dynamics*. Prentice Hall, 2 edition.
- Hahn, J. K. (1988). Realistic animation of rigid bodies. *Computer Graphics*, 22(4):299–308.
- Hashimoto, H., Buss, M., Kunii, Y., and Harashima, F. (1994). Intelligent cooperative manipulation system using dynamic force simulator. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 2598–2603.
- Haug, E. J. (1992). *Intermediate dynamics*. Prentice-Hall International.
- Haug, E. J., Wu, S. C., and Yang, S. M. (1986). Dynamics of mechanical systems with coulomb friction, stiction, impact and constraint addition-deletion I, II and III. *Mechanism and Machine Theory*, 21(5):401–425.
- Held, M., Klosowski, J. T., and Mitchell, J. S. B. (1995). Evaluation of collision detection methods for virtual reality fly-throughs. In *Proceedings of the 7th Canadian Conference of Computational Geometry*, pages 205–210.
- Hubbard, P. M. (1995). Collision detection for interactive graphics applications. *IEEE transactions on visualization and computer graphics*, 1(3):218–230.
- Ip, H. H. S. and Chan, C. S. (1997). Dynamic simulation of human hand motion using an anatomical correct hierarchical approach. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, pages 1307–1312.
- Iwata, H. (1990). Artificial reality with force-feedback development of desktop virtual space with compact master manipulator. *Computer Graphics*, 24(4):59–64.
- Jacobsen, S. C., Iverson, E. K., Knutti, D. F., Johnson, R. T., and Biggers, K. B. (1986). Design of the Utah/MIT dextrous hand. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1520–1532.
- Jong, C. W. (1998). Robot control through internet. Final year project, Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong.
- Kane, T. R. and Levinson, D. A. (1985). *Dynamics: Theory and Applications*. McGraw-Hill Publishing Company.

- Kang, S. B. (1994). *Robot Instruction by Human Demonstration*. PhD thesis, Carnegie Mellon University.
- Kasinski, A., Bonivento, C., and Vassura, G. (1991). Computer-graphic support for interactive grasp-planning. In *IEEE*, pages 1182–1186.
- Keller, H., Stolz, H., Ziegler, A., and Bräunl, T. (1995). *Virtual Mechanics: Simulation and Animation of Rigid Body Systems*. Univ. Stuttgart.
- Keller, J. B. (1986). Impact with friction. *Journal of Applied Mechanics*, 53:1–4.
- Kraus, P. R., Fredrikssona, A., and Kumar, V. (1997). Modeling of frictional contacts for dynamic simulation. In *IROS 1997 Workshops on Dynamic Simulation: Methods and Applications*.
- Kraus, P. R. and Kumar, V. (1997). Compliant contact models for rigid body collisions. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1382–1387.
- Kunii, Y. and Hashimoto, H. (1996). Dynamic force simulator for multifinger force display. *IEEE Transaction on Industrial Electronics*, 43(1):74–80.
- Lam, P. C. (1998). A friendly teaching system for dextrous manipulation tasks of multifingered hands. Master's thesis, Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong.
- Lee, P. U., Ruspini, D. C., and Khatib, O. (1994). Dynamic simulation of interactive robotic environment. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 1147–1152.
- Leung, K. C. T. (1996). An efficient collision detection algorithm for polytope in virtual environments. Master's thesis, The University of Hong Kong.
- Lewis, F. L., Abdallah, C. T., and Dawson, D. M. (1993). *Control of robot manipulator*. Macmillan Publishing Company.
- Li, Z. and Canny, J. F. (1993). *Nonholonomic Motion Planning*. Kluwer Academic.
- Li, Z., Hsu, P., and Sastry, S. (1989). Grasping and coordinated manipulation by a multifingered robot hand. *International Journal of Robotic Research*, 8(4):33–49.
- Lilly, K. W. (1993). *Efficient dynamic simulation of robotic mechanisms*. Kluwer Academic Publishers.



- Lin, M. C. (1993). *Efficient collision detection for animation and robotics*. PhD thesis, University of California at Berkeley.
- Liu, Y. (1995). *Proposal of the Dextrous Manipulation Teaching System*. CUHK.
- Liu, Y. H. and Arimoto, S. (1998). Decentralized adaptive and nonadaptive position/force controllers for redundant manipulators in cooperations. *International Journal of Robotics Research*, 17(3):232–247.
- Lötstedt, P. (1982). Mechanical systems of rigid bodies subject to unilateral constraints. *Journal of Applied Mechanics*, 42(2):281–296.
- Luciani, A., Jimenez, S., Florens, J. L., Cadoz, C., and Raoult, O. (1991). Computational physics: A modeler-simulator for animated physical objects. In *International Computer Music Conference*.
- McMillan, S. (1995). *Computational Dynamics for Robotic Systems on Land and under Water*. PhD thesis, The Ohio State University.
- Mirtich, B. (1997). V-Clip: fast and robust polyhedral collision detection. Technical Report TR97-05, A Mitsubishi Electric Research Laboratory.
- Mirtich, B. and Canny, J. (1995a). Impulse-based dynamic simulation. In *Proceedings of the Algorithmic Foundations of Robotics*.
- Mirtich, B. and Canny, J. (1995b). Impulse-based simulation of rigid bodies. In *Symposium on interactive 3D graphics*, pages 181–188.
- Mirtich, B. V. (1996). *Impulse-based Dynamic Simulation of Rigid Body Systems*. PhD thesis, University of California at Berkeley.
- Mirza, K., Hanes, M. D., and Orin, D. E. (1993). Dynamic simulation of enveloping power grasps. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 430–435.
- Montana, D. J. (1988). The kinematics of contact and grasp. *The International Journal of Robotics Research*, 7(3):17–32.
- Montana, D. J. (1995). The kinematics of multi-fingered manipulation. *IEEE Transaction on Robotics and Automation*, 11(4):491–503.
- Moore, M. and Wilhelms, J. (1988). Collision detection and response for computer animation. *Computer Graphics*, 22:289–298.



- Muñoz, L. A., Bard, C., and Nájera, J. (1995). Dextrous manipulation: A geometrical reasoning point of view. In *IEEE International Conference on Robotics and Automation*, pages 458–463.
- Murray, R. M., Li, Z., and Sastry, S. S. (1994). *A Mathematical Introduction to Robotic Manipulation*. CRC Press.
- Nelson, R. C., Jägersand, M., and Fuentes, O. (1995). Virtual tools: A framework for simplifying sensory-motor control in complex robotic systems. Technical Report TR576, Department of Computer Science, University of Rochester.
- Overmars, M. H. (1995). *Forms library: a graphical user interface toolkit for Silicon Graphics workstations*. Department of Computer Science, Utrecht University, Netherlands. <http://www.cs.ruu.nl/~markov>.
- Pang, J. S. and Trinkle, J. C. (1996a). Complementarity formulations and existence of solutions of dynamic multi-rigid-body contact problems with Coulomb friction. *Mathematical Programming*. To appear.
- Pang, J. S. and Trinkle, J. C. (1996b). Dynamic multi-rigid-body systems with concurrent distributed contacts. *Journal of Applied Mechanics*. Submitted.
- Qin, Z., Li, Z. X., and Jiang, S. (1997). CoSAM<sup>2</sup>: A unified control system architecture for multifingered manipulation. *IEEE Transactions on Robotics and Automation*. submitted.
- Reznik, D. and Laugier, C. (1996). Dynamic simulation and virtual control of a deformable fingertip. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1669–1674.
- Rohling, R. N., Hollerbach, J. M., and Jacobsen, S. C. (1993). Optimized fingertip mapping: a general algorithm for robotic hand teleoperation. *Presence: teleoperators and virtual environments*, 2(3):203–220.
- Rosenberg, R. M. (1977). *Analytical Dynamics of Discrete Systems*. Plenum Press.
- Routh, E. J. (1905). *Elementary rigid dynamics*. Macmillan.
- Salisbury, J. K. and Roth, B. (1983). Kinematic and force analysis of articulated mechanical hands. *Journal of Mechanisms, Transmissions and Automation in Design*, 105:35–41.

- Sanso, R. M. and Thalmann, D. (1994). A hand control and automatic grasping system for synthetic actors. *EUROGRAPHICS*, 13(3):167–177.
- Sarkar, N., Kumar, V., and Yun, X. (1996). Velocity and acceleration analysis of contact between three-dimensional rigid bodies. *Transactions of the American Society of Mechanical Engineers*, 63:974–984.
- Shimoga, K. B. (1996). Robot grasp synthesis algorithms: A survey. *International Journal of Robotic Research*, 15(3):230–266.
- Speeter, T. H. (1992). Transforming human hand motion for telemanipulation. *Presence: teleoperators and virtual environments*, 1(1):63–79.
- Stronge, W. J. (1991). Unraveling paradoxical theories for rigid body collisions. *Journal of applied mechanics*, 58:1049–1055.
- Trinkle, J. C., Pang, J. S., Sudarsky, S., and Lo, G. (1996). On dynamic multi-rigid-body contact problems with Coulomb friction. *Zeitschrift für Angewandte Mathematik und Mechanik*. Accepted.
- Umetsu, M. and Oniki, K. (1993). Compliant motion control of arm-hand system. In *Proceedings of JSME International Conference on Advanced Mechanics*, pages 429–432.
- Walker, M. W. and Orin, D. E. (1982). Efficient dynamic computer simulation of robotic mechanisms. *Transactions of ASME Journal of Dynamic Systems Measurement and Control*, 104:205–211.
- Wendlandt, J. M. (1997). *Control and Simulation of Multibody Systems*. PhD thesis, University of California at Berkeley.
- Witkin, A., Gleichler, M., and Welch, W. (1992). Interactive dynamics. *Computer Graphics*, 24(2):11–22.
- Wright, A. K. and Stanisic, M. M. (1990). Kinematic mapping between the exos handmaster exoskeleton and the Utah/MIT dextrous hand. In *Proceedings of IEEE International Conference on Systems Engineering*.
- Zhuang, Y. (1996). Simulation of linear electrostatic stepper motor (a MEMS device) using impulse. Course Project, Computer Science Department, California University at Berkeley.





CUHK Libraries



003703770