This is to certify that the

dissertation entitled

DEVELOPMENT OF MINIATURE CLIMBING ROBOTS - MODELING, CONTROL AND MOTION PLANNING

presented by

JIZHONG XIAO

has been accepted towards fulfillment of the requirements for

PHD degree in ELECTRICAL ENGINEERING

Major professor

Date 7/30/02

DATE DUE	DATE DUE	DATE DUE

DEVELOPMENT OF MINIATURE CLIMBING ROBOTS — MODELING, CONTROL AND MOTION PLANNING

By

JIZHONG XIAO

A DISSERTATION

Submitted to

Michigan State University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

2002

ABSTRACT

Development of Miniature Climbing Robots

—Modeling, Control and Motion Planning

By

Jizhong Xiao

An increasing interest in the development of special climbing robots has been witnessed in the last decade. Cleaning high-rise buildings, spray painting and sand blasting of gas tanks, inspecting and maintaining nuclear facilities, assisting fire fighting and rescue operations, remote monitoring of hazardous environment, etc. — all of these practical problems have an immediate need of automation. Climbing robots, with their ability to adhere to wall surfaces and move around carrying appropriate sensors or tools, are the best candidates for these kinds of jobs. The special characteristics and capabilities of climbing robots would not only allow them to replace human workers in these dangerous duties but also eliminate costly erection of scaffolding.

This dissertation describes the development of miniature bipedal climbing robots, with an emphasis on the CRAWLER robot. This robot is the smallest such robot to date, able to climb walls, walk on ceilings, travel through pipes, and transit between two inclined surfaces. Two active suction feet, where pump motors vacuum air out of suction cups, are used to support the robot on surfaces. The robot adopts the bipedal structure and an under-actuated mechanism to provide the robot with versatile mobility and multiple locomotion modes. The under-actuated mechanism reduces the number of motors required and thereby the robot size, weight, and power consumption, but it imposes challenges on robot control and motion planning.

The robot kinematic model and dynamic model are derived. The analysis of the dynamic model reveals that the robot link gravity is a dominant term in robot dynamic effects. A joint level PD (proportional + derivative) control plus feedforward

gravity compensation method is proposed. This method outperforms the conventional PD control method because it not only achieves the joint level control but also compensates for the gravity effects which depend on the configuration of all the robot links. A DSP-based embedded control system is designed and installed on-board to control the robot. By using the DSP (digital signal processor) chip, the number of electrical components is minimized and the control system is self-contained.

In motion planning analysis, a hybrid configuration space concept is proposed which incorporates the continuous configuration space with the discrete motion status space (*i.e.*, standing foot, motion mode). The hybrid configuration space is an effective tool to analyze the motion planning of the climbing robot since the robot motion is uniquely determined in the hybrid configuration space framework. Based on motion pattern analysis, a motion planning method is developed that consists of a global planner and a local planner. The global planner generates a possible path by simplifying the robot as a rectangular rigid object with no kinematic constraints. By using an approach called trapezoidal decomposition and a searching algorithm known as A^*, it is easy for the global planner to smooth the possible path and allow the robot to move effectively in translation mode. The possible path describes the global motion of the robot and minimizes the turns. The purpose of the local planner is to generate a feasible motion sequence around the turning point by considering the robot constraints. A cost function is defined based on the motion status information and a number of heuristics to help the search of an optimal motion sequence. This approach using global and local levels of refinement reduces the overall complexity and simplifies implementation. Experiments and simulations demonstrate the effectiveness of our robot system.

To my parents, my wife—Jing Ye, and my son—Bowen Xiao.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Dr. Ning Xi, for his insightful guidance, enthusiastic encouragement and kind support during the research and the development of this dissertation. Without his assistance and advice, this research would not be possible.

I would like to thank Dr. R. Lal Tummala, who is the principal investigator of the micro-robot project, for his supervision, and helpful feedback and suggestions after reading many of my papers. I am very grateful to other guidance committee members: Dr. Fathi Salam, and Dr. Gang Bao. Their insightful comments and suggestions will enhance the technical soundness of this dissertation. I also would like to thank Dr. Zengyu Yu from Texas Instruments Inc. for his valuable help on the TI DSP implementation.

Special thanks are dedicated to Dr. Hans Dulimarta for his help on the software coding and the collaborative contributions to the work on motion planning. I am grateful to other members of the micro-robot project and the colleagues from Robotics and Automation Lab at Michigan State University. The discussions with Mark Minor, Meng Yue, Jindong Tan, Weihua Sheng, Yu Sun, Heping Chen, Amit Goradia, and Imad Elhajj, substantially contributed to my work and broadened my knowledge.

I also would like to thank the facilitators of the NSC840 writing course, Dr. Renat Snider, Dr. Ruth Barton, Dr. James Resh, and Dr. Timothy Grotjohn, for their great help in polishing my papers and dissertation.

I am greatly indebted to my parents for their continuous encouragement and support throughout my academic life. Thanks also go to my wife, Jing Ye, for her patience, understanding and support during my time in graduate school.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background

Interest in the development of special climbing and walking robots is growing rapidly. To increase the operation efficiency and to protect human health and safety in hazardous tasks are the motivations behind the development of climbing robot. Climbing robots with the ability to adhere to wall surfaces and move around carrying appropriate tools are currently being strongly requested by various industries in order to perform dangerous operations such as cleaning of high-rise buildings, spray painting and sand blasting of gas tanks, maintenance of nuclear facilities, aircraft inspection, surveillance in hostile environments, assistance for fire fighting and rescue operations, etc. Such capabilities of climbing robots would not only allow them to replace human workers in those dangerous duties but also eliminate costly scaffolding.

By now, applications of climbing robots have been found in many industries. The chemical and nuclear industries are the two primary fields where climbing robots are expected to perform remote inspection and maintenance tasks in highly contaminated areas or radioactive environments. Application examples include: the retrieval of irradiated material samples [20], the inspection of storage tanks [76], the handling and manipulation of nuclear fuel [12] [65], and the dismantling of radioactive facilties by underwater climbing robots [5].

In construction and shipbuilding industries, applications of climbing robots are also apparent. The most relevant examples are the cleaning of windows or walls [76], the painting or sanding of ship hulls, the inspection of the metallic skeletons of bridges [1]. In military applications, climbing robots have been used for the inspection of the exterior of large aircraft [6].

In addition to those areas, other fields have benefited from the development of

climbing robots. It has been shown that tiny pipe and duct climbing robots are very useful in weld inspection, leak detection, checking and cleaning of unwanted deposits in gas ducts, and sewage pipes [36], [50], [68, 69], [72].

Low-cost miniature climbing robots, with versatile maneuverability and the capability to operate in areas where access is tight, have drawn much attention from national research agencies. This dissertation is based on the research work of a micro-robot project at Michigan State University funded by DARPA (Defense Advanced Research Project Agency) Distributed Robotics Program. The purpose of the project is to design, build and test prototype micro-robots that incorporate control, sensing and planning to perform different missions. The mission scope includes inspection in constrained environments, such as pipes and ventilation ducts, and gathering information about a hostile situation within a building (Tummala et al) [71]. Thus, the robots must be able to crawl through pipes, travel on surfaces with varying inclinations, such as floors, walls, and ceilings, and be able to traverse between such surfaces.

1.2 Literature Review

1.2.1 Climbing Robots in Literatures

A number of climbing robots have been developed in recent years. Motivations are typically inspection or maintenance in dangerous environments like the exteriors of tall buildings, airplanes or ships; or in nuclear facilities or pipelines.

One of the most important issues for climbing robots is to design a proper adhesive mechanism to ensure that the robot sticks to wall surfaces reliably. So far, three kinds of adhesive devices have been used: magnetic devices, vacuum suckers and attraction force generators based on aerodynamic principles. Magnetic adhesive devices are the most promising for robots moving around on steel structures. Robots using permanent

magnets or electromagnets can be found in [23], [24], [27], [76] for climbing large steel structures and in [36], [68, 69] for the internal inspection of iron pipes. However, their applications are limited to steel walls due to the nature of magnets.

In applications for non-ferromagnetic wall surfaces, robots most generally use vacuum suckers to produce the adhesive force. Example of such robots include the Robug [19, 20, 43, 44] at University of Portsmouth, UK, NINJA-1 robot [26, 49] at Tokyo Institute of Technology, Japan, and ROBIN [57] at Vanderbilt University, USA. Besides those robots built in academic institutes, some robots have been put into practical use. For example, MACS robots [6] at the Jet Propulsion Laboratory (JPL) use suction cups for surface adherence when inspecting the exterior of large military aircraft; Robicen robots [12, 65] use pneumatic actuators and suction pads for remote inspection in nuclear power plants; SADIE robots [72] use a sliding frame mechanism and vacuum gripper feet for weld inspection of gas duct internals at nuclear power stations. A wall climbing robot with scanning type suction cups is reported in [77]. An underwater climbing robot with vacuum grippers for contact arc metal drilling and cutting is reported in [5]. Other climbing robots based on pneumatic actuators and vacuum suction cups are reported in [64] and [76]. The great defect of the suction foot is limited locomotion speed, but advantages are apparent and outweigh this drawback. A suction foot is capable of producing a powerful adhesive force and their pneumatic components have an excellent weight:power ratio. A climbing robot with articulate structure and equipped with suction feet is capable of moving between surfaces by using the degrees of motion freedom of the legs.

Apart from the aforementioned adhesive mechanisms, the third choice is to create attraction force based on aerodynamics including the use of propeller [52, 53, 54] and reversed hovercraft technique. In this case, the robot is normally driven by wheels and is suitable to move quickly on the wall surface. But it has two major disadvantages. The first one is that it is difficult to produce a sufficient adhesive force due to the

aperture between the robot and the wall. The second is that the transition of robot between surfaces is very difficult if not impossible.

Besides the suction mechanism, mechanical structure and mobility are other design factors determined by specific applications. Climbing up large, nearly flat surfaces like ship hulls or reactor tanks or exterior of aircraft will demand simple machines with fewer degrees of freedom in their motion. Wheeled vehicles with vacuum grippers [12] [76] or robots using sliding frame systems [5] [6] may satisfy this requirement. On the other hand, in more complicated tasks where robots are required to transit from a floor to a wall or from a wall to a ceiling, articulated limbed structures with two to eight legs are predominant. Bipedal robots with the ability to move between surfaces include ROSTAM III [7], ROBIN[57], and robots designed by Nishi [51]. As alternatives for increased safety and load capacity, quadruped and more legged robots are adopted. ROBUG II [43] and NINJA-1 [26, 49] are quadruped robots featuring multiple degree-of-freedom (DOF) legs protruding from a central body and carrying suction feet. With even greater complexity and larger size, the robot in [23] has hexapod configuration and the ROBUG III [19, 20] use eight legs to provide increased stability. A spider-like pipe robot is developed in [50]. More limbs typically provide redundant support and often increase load capacity and safety. However, these benefits are achieved at the cost of increased complexity, size and weight.

Thus, when compactness and efficiency are critical, as in the case of miniature robots, a structure with minimal weight and complexity is best applied. For these reasons, the biped format is an excellent candidate. Among the three adhesive mechanisms, each of them has its advantages and disadvantages. The vacuum sucker is the most versatile device and is commonly used in the climbing robot. Our miniature climbing robots adopt suction foot and bipedal articulated structure to achieve good balance between compactness and desired mobility.

1.2.2 Overview of Robot Motion Planning

The analysis techniques and algorithms for motion planning have become quite valuable in many applications such as robotics, virtual reality systems, graphical animation, and computer-aided design. A robot with motion planning ability can autonomously determine the detailed motion sequences to accomplish a particular task without collision with obstacles. This relieves the human operators from tediously issuing the detailed commands for the robot.

The configuration space concept developed by Lozano-Perez [59], [60], and J.C. Latombe [39], etc, served as a powerful representational tool for both the development and the analysis of motion planning algorithms. In the configuration space framework, it becomes possible to develop algorithms that are generalizable and adaptable to a wide variety of applications. There exists a large number of techniques for solving the basic motion planning problem and they typically follow one of the three approaches: roadmap, cell decomposition and artificial potential field (see Latombe [39] for a complete survey).

Cell decomposition methods partition the configuration space into regions within which a collision-free path is easy to determine, and this path can be easily connected to collision-free paths of adjacent regions. A solution to the motion planning problem is determined by searching for a sequence of collision-free cells that initially include the initial configuration and finally includes goal configuration. Approaches to configuration space partitioning can be categorized either as *exact cell decomposition*, in which cells are typically constructed by determining critical sets in an algebraic description of the configuration space [39, 66], or *approximate cell decomposition*, in which each cell is assumed to be of the same basic shape, and the configuration space is usually decomposed in a hierarchical manner [8, 58].

In a roadmap approach, a one-dimensional network of paths is constructed that preserves the connectivity of the free configuration space. A solution path is con-

structed by connecting initial and goal configurations to the roadmap and performing a graph search on the extended roadmap. The *visibility graph* approach generates a roadmap by connecting certain vertices of the boundary of the free configuration space, and is primarily suitable for two-dimensional, polygonal configuration-space planning problems [17] [21] [61] [63]. The topological *retraction* operation has been used in a roadmap generation approach that continuously retracts free configuration space onto its generalized Voronoi diagram [56]. Other roadmap methods are described in [13, 14, 15].

In the prior two approaches, a path is constructed by analyzing the global structure of free configuration space; however, in most artificial potential field methods, the robot moves locally according to some forces that are defined as the negative gradient of an artificial potential function. This approach was taken in [37] for real-time collision avoidance. The potential function is typically defined on free configuration space as the sum of an attractive potential that pulls the robot toward the goal, and a repulsive potential that pushes the robot away from obstacles. One of the primary concerns in this approach is the existence of local minima in the potential function. In [30] heuristic search is performed over nominal paths at a global level, and in [9, 10] Brownian motion is combined with motion planning to escape local extrema. In [62] it has been shown that for some problems, a global navigation function can be constructed that ensures the existence of a single minimum that lies at the goal configuration.

Approaches to the basic motion planning problem are often evaluated with regard to completeness and computational complexity. An algorithm is considered to be *complete* if it is guaranteed to find a solution path whenever one exists. The exact cell decomposition method is complete, but extremely expensive with doubly-exponential time complexity in the dimension of the configuration space. The best known complete method constructs a roadmap through geometric stratifications and has time

complexity that is exponential in dimension [14]. Approximate cell decomposition methods usually achieve *resolution completeness*, which means that the algorithm can dynamically adjust the resolution until a solution is found; however, due to the hierarchical partitioning of n-dimensional cells, the time and space complexity grows quickly with the dimension of the configuration space [39]. Potential field methods are typically far more computationally efficient than the other approaches; however, completeness is usually sacrificed.

In recent years, random sampling has emerged as a promising new approach for motion planning. Randomized motion planners are capable of solving complex motion planning problems in high-dimensional spaces. Planners based on random sampling are not complete. Some of them satisfy a weaker, but still interesting property called *probabilistic completeness*: if a path exists, the probability that the planner finds it converges to 1 quickly, as running time increases. Two of the most popular approaches include the probabilistic roadmap (PRM) algorithm [3, 35], and the Rapidly-Exploring Random Tree (RRT) [38, 41] algorithm.

Probabilistic roadmap method proceeds in two stages. In the preprocessing stage, it samples collision-free configurations at random and connects them by simple canonical paths, thus creating a probabilistic roadmap. In the query stage, it connects the two query configurations to the roadmap and searches the roadmap for a path. The first PRM planners [28, 34, 70] use a straightforward uniform distribution for sampling new configuration, possibly followed by an enhancement step to increase the sampling density in critical regions [34]. Recently a number of other sampling strategies have been developed, including shrinking the obstacles [29], sampling near the free space boundaries [2] or medial axis of the configuration space [73], using "guards" to reject unwanted samples [55]. The "lazy PRM" [11] is the variation of PRM method built with all collision checking postponed until a query is presented. The RRT method starts from a single collision free node and grows out from it in fixed-size randomly

directed steps until the unexplored space is filled. RRT is particularly suited for problems that involve differential constraints.

Recently, there is a trend to expand configuration space framework to handle more complicated motion planning problems, such as planning under uncertainties, planning with kinematic and dynamic constraints, and multiple robot motion planning, etc. LaValle [40, 42] conducted a systematic research and extended the basic motion planning problem with game theory. He advocated the expansion from *configuration space* to *state space* and to broader space — *information space* to cover general motion strategy problems, involving uncertainty in sensing and control, environment uncertainties, and the coordination of multiple robots. Sharma and Sutanto [67] proposed a vision-based motion planning framework in which they unified the configuration space and an image feature space and considered sensors as an integral part of the motion goal. In [39], the dynamic motion planning problem, where the obstacles in the workspace are moving, was studied in *configuration-time space.*

1.3 Our Climbing Robots

We have built two prototype climbing robots. The first generation robot is called "FLIPPER" and its picture is shown in Figure 1.1. The mechanical design of the FLIPPER robot is reported in [45, 46] by Minor, Dulimarta, et al. Dangi, Stam and Aslam reported the design and testing of the Smart Robotic Foot in [16]. The kinematic model of the FLIPPER is reported by Yue, Minor et al. in [79]. The DSP-based controller design of the FLIPPER robot is reported in [74] by Xiao et al. When the robot stands vertically, it measures approximately $45mm \times 45mm \times 248mm$ and weights 335 grams (11.8 Oz).

The FLIPPER is a biped robot with two suction feet located at the robot extremities. The middle joint is of revolute type. The motion of the FLIPPER is achieved by sticking one foot on the surface and "flipping" its body as shown in Figure 1.2.

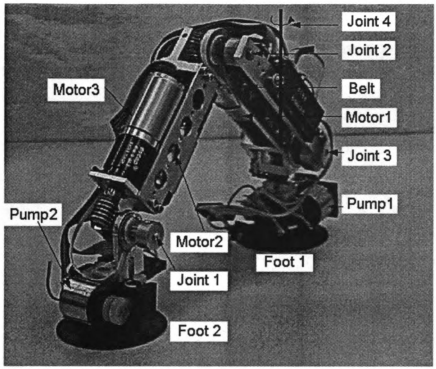

Figure 1.1: Picture of the FLIPPER robot

The flipping mode of locomotion requires space twice of the height of the robot leg, and thus it is not suitable when the robot has to travel through confined spaces, such as ventilation ducts.

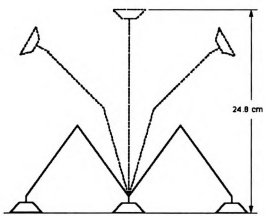

Figure 1.2: Flipping mode of locomotion

In response to the requirement of performing missions in a constrained environment, the second generation of the microrobot, "CRAWLER", has been constructed.

The picture of the CRAWLER is shown in Figure 1.3.

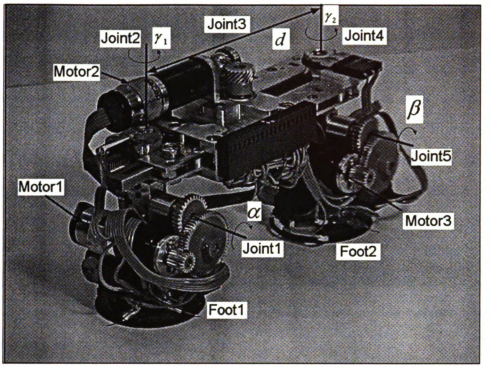

Figure 1.3: Picture of the CRAWLER robot

The CRAWLER robot achieves linear motion through extension and contraction of the robot legs. It is used to travel through inside pipes. The size and weight of the robot are minimized through under-actuated mechanism, wherein 5 joints are driven by only 3 motors. This special mechanical structure reduces robot size and weight but poses challenges in control and motion planning. The dimension of the robot is approximately 80 mm in height, and 50 mm in width. It weights about 450 grams (15.8 Oz). Our robots are the smallest of the bipedal climbing robots found in the literatures. The next larger climber is Yano's robot [77], which measures 380mm$\times$ 250mm$\times$ 170mm and weighs 10 kg.

This dissertation will only present the modeling, control and motion planning of the CRAWLER robot.

1.4 Challenges and Difficulties

To design a miniature robot that can climb walls, walk on ceilings, crawl through pipes, and transit between different inclined surfaces is a challenging task. The requirements of small size, light weight and low power consumption must be satisfied.

Unlike normal wheeled vehicle robots, climbing robots must have a proper adhesive mechanism to ensure that the robots grip wall surfaces firmly and without sacrificing mobility. The desired capabilities of the robot, such as the transition between different surfaces and traveling through pipes, imposes further difficulties on the design and control. To achieve its mission, a robot with multiple forms of locomotion is imperative. By adopting suction feet, bipedal articulated structure, and an under-actuated mechanism, our climbing robots achieve a good balance between compactness and maneuverability.

As a self-contained system, the climbing robot needs to carry its own power source, sensors, control system and associated hardware. Thus minimization of power consumption and weight is critical to prolonged operation. The challenges in designing the robot controller are to use as few components as possible and take any measures to reduce power usage. Thus the selection of proper sensors, controller chips, components and the design of small and efficient electrical circuit becomes very important. By efficiently utilizing the resources of a DSP chip, our embedded control system minimizes the component count, reduces the power consumption, and becomes self-contained and tetherless.

From a mechanical design point of view, an under-actuated mechanism reduces both weight and power consumption of a robot, since it requires fewer motors than a conventional robot. However, the under-actuated mechanism increases control complexity and imposes challenges in motion planning.

In order to generate smooth locomotion, the robot requires a motion controller capable of computing the robot kinematic model in real-time and generating syn-

chronized motion control of multiple joints. The effects of gravity on the robot joints change with varying inclinations as the robot moves across different surfaces. This makes gravity compensation much more difficult than for fixed-base robots or mobile robots moving on the ground. We derive the robot kinematic and dynamic models and utilize them to plan the robot motion and propose a PD+gravity compensation method to achieve a good performance.

The motion planning of climbing robots with under-actuated mechanism is more complicated than that of wheeled robots. Due to the constraints imposed by the mechanical structure and the under-actuation, the climbing robot can perform only restricted motions even in the absence of obstacles. The coupling between motion joints and the switching between different motion modes further complicate motion planning. We introduce a hybrid configuration space concept and propose an effective motion planning approach to generate a path and a motion sequence allowing the robot to travel through an environment without collision with obstacles.

1.5 Outline of the Dissertation

The scope of this dissertation is to investigate the modeling, control and motion planning of the CRAWLER climbing robot.

Chapter 2 describes the modeling of the CRAWLER robot. The mechanical structure and locomotion modes of the robot are introduced first. Then, the robot kinematic and dynamic models are derived. Locomotion analysis is also presented in this chapter.

Chapter 3 discusses the embedded control system design based on a DSP chip. Overview of the control system architecture, the hardware design and software development are presented. A joint level PD+gravity compensation method is proposed. This control method outperforms the conventional PD controller because it not only achieves joint level control but also compensates for the gravity effects which is de-

termined by the configuration of all the robot links. The experimental results are provided with snapshots showing the robot capabilities.

Chapter 4 studies the motion planning of the CRAWLER robot. The hybrid configuration space concept is proposed to incorporate the continuous configuration space with discrete motion status (*i.e.*, standing foot, motion mode) imposed by the under-actuated mechanism of the robot. The motion pattern of the CRAWLER robot is analyzed under the hybrid configuration space framework. A motion planning algorithm is developed which consists of a global planner and a local planner to generate a collision-free path and a feasible motion sequence to travel along the path. A cost function is defined based on the motion status information to guide the search for an optimal path. Simulations and experiments have been conducted and the results verified the efficency of the method.

Chapter 5 summarizes the contributions of the research and concludes the dissertation with recommendations for the improvement of the CRAWLER robot and for future research directions of climbing robots.

CHAPTER 2

MODELING OF THE CRAWLER ROBOT

In this chapter, the mechanical structure of the CRAWLER robot is introduced, the robot kinematic model and dynamic model are derived, and the locomotion analysis is presented.

2.1 Mechanical Structure

The purpose of the climbing robot is semi-autonomous reconnaissance in confined environments. The robot will be deployed on the exterior or interior surfaces of buildings and structures. It will traverse horizontal and vertically inclined surfaces and climb between them in order to provide video surveillance or deliver reconnaissance sensors to specified locations. The robot must be sufficiently small to travel through pipes, ventilation ducts and to avert visual detection. For effective and prolonged operation, the robot needs to be small in size, light in weight, and consume as less power as possible.

The CRAWLER robot is designed as a bipedal robot with an under-actuated mechanism to achieve good balance between compactness and maneuverability. The robot is supported by two suction feet which can hold on anticipated smooth and non-porous surfaces. The dimension of the prototype robot is approximately 80 mm in height and 50 mm in width. It weights 450 grams. It has the capability to climb on a wall, walk on a horizontal surface, crawl through pipes, and transit between two inclined surfaces.

2.1.1 Description of Mechanism

The mechanical structure of the CRAWLER robot is shown in Figure 1.3. The under-actuated mechanism enables the robot to drive 5 joints using only 3 motors

thus reducing both the weight and the power consumption of the robot. Motors 1 and 3 independently drive joints 1 and 5, respectively; thereby adjusting the tilt angle of the suction foot 1 and foot 2 so that the robot can grip the surface firmly. Motor 2 is responsible for controlling joints 2, 3, and 4. Joint 2 and 4 are revolute joints providing steering capability of the feet relative to the legs. Joint 3 represents the prismatic motion of the legs that allows the robot expanding and contracting its legs. The clock-wise (CW) rotation of motor 2 causes contraction, *i.e.*, both legs slide into the robot body; while the counter-clock-wise (CCW) rotation of motor 2 causes expansion, *i.e.*, both legs translate out of the robot body.

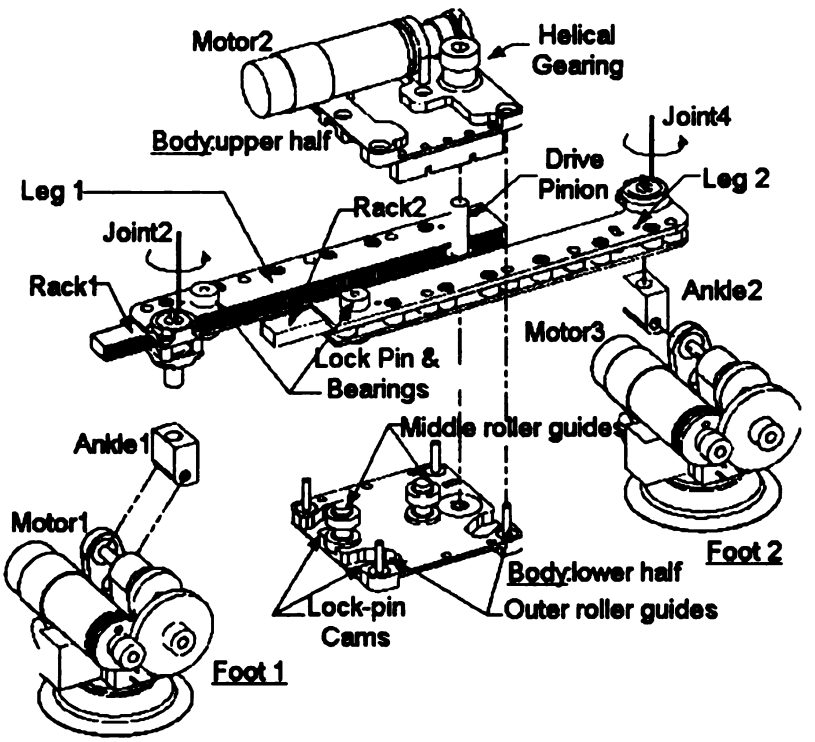

Figure 2.1: Exploded view of the CRAWLER robot

A partially exploded view of the CRAWLER robot is illustrated in Figure 2.1. The robot consists of identical pairs of legs, racks, ankles and suction feet. Motor 2 drives the helical gear, through the drive pinion, making the two racks sliding in opposite directions. The innovation of the design lies in the structure of the rack/leg pair and the lock-pin cams. Each rack has a log-pin notch (see Figure 2.2). In the normal case,

the lock pin passes through a slot on the leg and engages the notch by an elastic plate, resulting in the rack/leg pair to slide together. When a rack/leg combination slides in and pulls the lock-pin bearing into the cam slot, the lock-pin cams contained within the upper and lower halves of the body force the bearing to move along the special curve and push the lock pin outside the notch. This disengaging effect separates the leg/rack pair and prevents the leg from moving but allows the rack to continue sliding. The rack then drives the corresponding robot foot to rotate along joint 2 or joint 4.

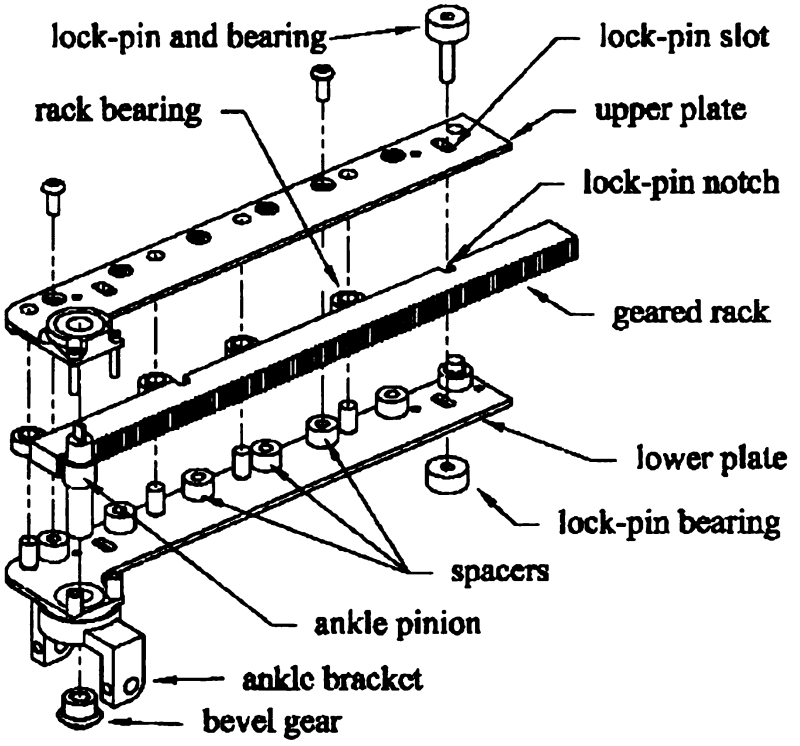

Figure 2.2: Exploded view of the CRAWLER rack/leg pair

2.1.2 Locomotion Modes

The CRAWLER robot has three motion modes and the capability to switch between them. Motor 2 drives a set of joints (joint 2,3,4) but not all of them simultaneously. In each of the three modes, a particular subset of joints is driven and the remaining joints are locked to prevent rotation. Figure 2.3 shows the top-down view of the CRAWLER robot and their locomotion modes. Notice that the legs are essentially

identical, with the exception of their lock-pin locations. In the case of leg 1, the pin is adjacent to the ankle and enters its cam slot when both legs are contracted. In the case of leg 2, the lock-pin is mounted at the end opposite from the ankle and it enters its cam slot when both legs are extended. The switching between motion modes is achieved by the engaging/disengaging of the lock-pin.

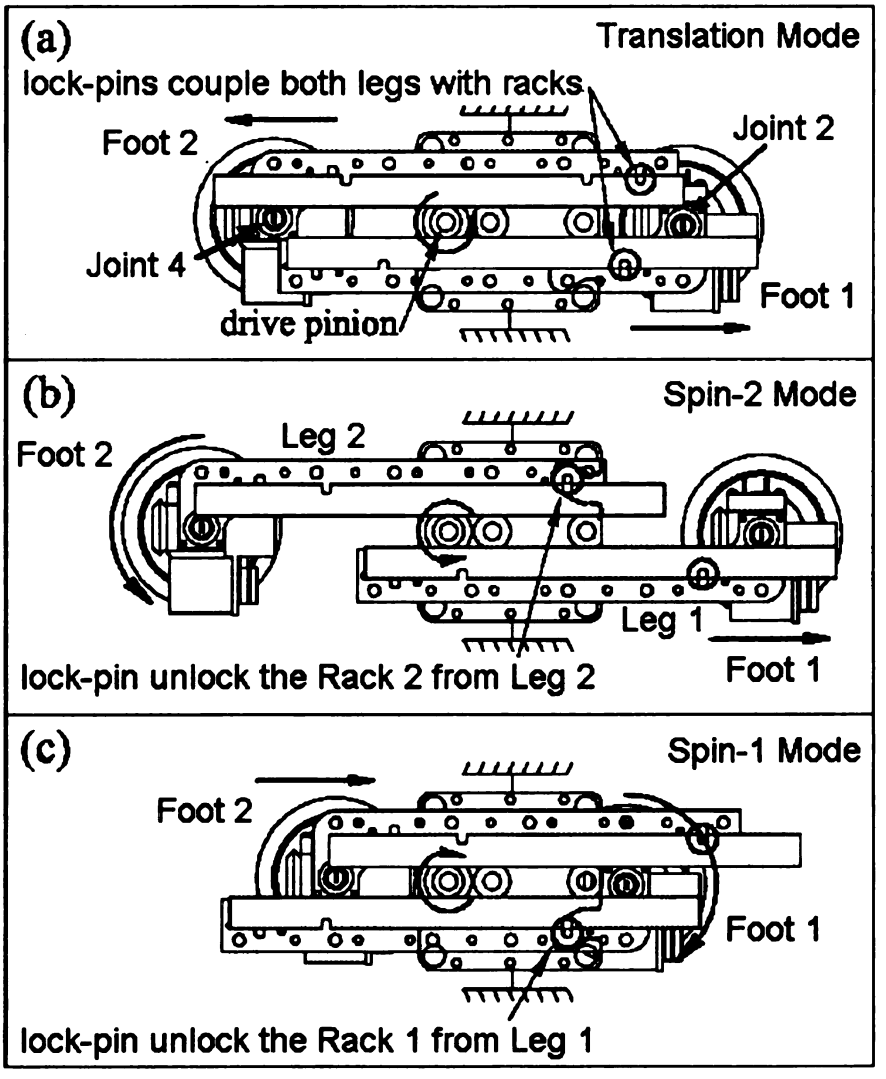

Figure 2.3: Locomotion modes of the CRAWLER robot

- **Translation Mode**: When both the lock-pin bearings are outside the cam, *i.e.*, the legs and their corresponding racks are locked together, joint 2 and 4 are prevented from rotating; thereby the rotation of motor 2 causes translation motion of the legs. A CCW rotation of motor 2 causes the legs to extend while

a CW rotation causes them to contract. If the translation motion continues beyond a certain range, both in extension and contraction, one of the lock-pins will enter its cam slot on the body, causing the mode switch from the translation mode to spin-1 mode or spin-2 mode.

- **Spin-1 Mode**: When lock-pin on leg 1 enters its cam-slot during contraction, it disengages the rack 1 from leg 1 and allows the CW rotation of foot 1 relative to leg 1 about joint 2. Meanwhile, since the lock-pin on leg 2 still couples the leg and rack motion, leg 2 will continue to contract and joint 4 is held fixed.

- **Spin-2 Mode**: If the legs of the robot keep extending in translation mode, the lock-pin bearing on leg 2 will enter its cam slot and unlock the rack 2 from leg 2 causing the CCW rotation of foot 2 about joint 4. At mean time, leg1/rack1 pair continues to extend along joint 3 while joint 2 is held fixed.

2.1.3 Robot Foot

The two legs of the robot are supported by the suction feet that provide the robot with the ability to walk on a horizontal surface as well as climb a vertical wall. Unlike the large suction feet reported in [12], [65], where the power and air are supplied externally through a tether, our miniature suction feet are self-contained and tetherless without limitations in motion range [16] (Dangi, Stam and Aslam). The suction foot is shown schematically in Figure 2.4. Its main components are a diaphragm-type vacuum pump, a suction cup, a pressure sensor and a micro machined shape memory alloy valve. The pump is connected to the suction cup through a custom designed aluminum connector. The connector integrates the foot components and serves as a mounting platform for the robot body. The 40 mm diameter suction cup with cleats is used for adherence. The cleats increase the rigidity of the grip and provide a larger contact area to reduce slippage. The pressure sensor monitors the

pressure level inside the suction cup to ensure that the foot is firmly attached to the object surface. The foot is released through the actuation of the valve by a signal from the robot controller. Several touch sensors are also attached to the suction cup in different radial directions. This gives information on which part of the suction cup has touched the contact surface and facilitates the robot in adjusting the suction foot orientation. The fully assembled foot measures $40mm \times 40mm \times 25mm$ and weighs 35 grams.

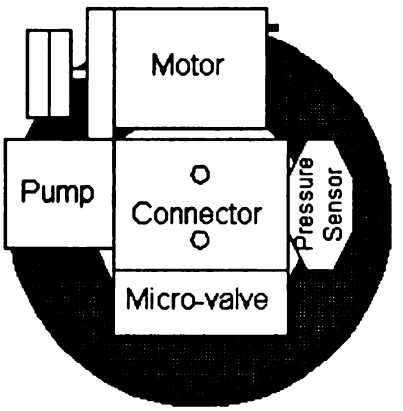

Figure 2.4: Schematics of the suction foot

Comprehensive tests were conducted on different types of surfaces with different cup diameters and the results were reported in [16] by Dangi, Stam and Aslam. The foot was tested in both parallel and perpendicular configurations. In the parallel configuration, as shown in Figure 2.5 (a), the load is applied via a rod at a distance D from the vertical testing surface. In the perpendicular configuration, the load direction was kept perpendicular to the horizontal surface as shown in Figure 2.5 (b).

Table 2.1 shows the experimental results of the suction foot with 40 mm diameter cup. Three types of sample surfaces viz. glass, painted steel cabinet, and spray painted plastic, with increasing order of surface roughness, were used for testing. The maximum load value is obtained by gradually increasing the load until the suction foot fails to support it. The surface conditions were alternated between dry and wet.

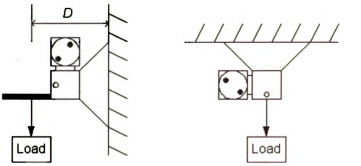

Figure 2.5: Suction foot testing configuration

The wet condition was simulated by spraying of de-ionized water from a distance of 150 mm. It can be seen that for the same surface condition (*e.g.* dry) the load supported decreased with increasing surface roughness. This can be attributed to loss of vacuum with rougher surfaces. It can also be seen that a wet surface supports more weight than a dry surface. A possible explanation for this phenomenon can be that water occupies the gap between the cup and test surface resulting in lower vacuum loss than dry condition.

Table 2.1: Suction cup test results

Test Configuration	D (mm)	Maximum Load Supported (g)					
		Glass		Metal Cabinet		Painted Plastic	
		Wet	Dry	Wet	Dry	Wet	Dry
Parallel	40	1172.4	1161.05	1159.5	1155	1167.25	1108.1
Perpendicular	-	3674.5	3598.4	3622.55	3584.2	3275.25	3251.8

Considering the 450 grams self weight of the robot, the extra holding capacity of one foot is in the range of 650-3200 grams, more than enough to carry controller hardware, a 9V battery (40 grams), miniature reconnaissance sensors such as wireless pin-hole camera, micro-phone and transmitters, etc. Figure 2.6 are some snapshots showing where the climbing robot can go in the engineering building at Michigan

State University.

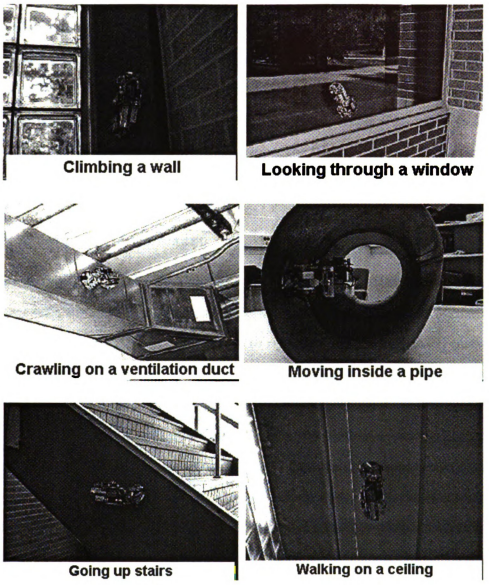

Figure 2.6: Whereabouts of the CRAWLER robot

2.2 Kinematic Model

In robot systems, the control is realized in the joint space, whereas the task level commands are normally expressed in world coordinate space. For the climbing robot, the reconnaissance camera is mounted at one of its feet to permit the robot to either look through a glass window or to use the camera like a periscope when the

alternative foot supports the robot. The tilt angle of the camera is determined by the position/orientation of the robot's free foot relative to its standing foot. Thus it is imperative to derive the robot kinematic model which describes the relation between the robot joint variables and the position/orientation of the robot's free foot with respect to a fixed reference coordinate frame located at the center of standing foot. The kinematic model is also very important in solving the robot motion planning problem. This section derives the kinematic model of the CRAWLER climbing robot.

2.2.1 Coordinate Assignment

Because the structure of the CRAWLER robot requires that at least one foot remain in contact with the surface at all times, the setup of the coordinate frames is conducted in the three-dimensional space with respect to right-foot supporting (RFS) phase and left-foot supporting (LFS) phase. In each phase, the robot has two motion modes. One is the translation mode, which means both rack and leg pairs are locked together and thus the middle joint motor 2 drives the two legs sliding in opposite direction. In this mode, only two rotational joints (J1 and J5) and the prismatic joint J3 move. The other is spin mode, which occurs when one of the racks is separated from its leg pair, resulting in the corresponding foot spinning while the other leg and rack pair sliding. In spin mode, two rotational joints (J1, J5), one spin joint (J2 or J4) and the sliding joint (J3) are involved in the motion. Since the robot is symmetric, we only analyze the kinematics in LFS phase. The kinematic model is the same for both RFS and LFS phases. The assignment of coordinate frames based on Denavit-Hartenberg (D-H) method [18] for LFS_translation mode is shown in Figure 2.7.

LFS_translation mode: In LFS_translation mode, the reference frame is attached to the left foot which is fixed on the ground surface. The base coordinate frame is at joint 1 with the Z0 axis aligned with J1 rotation axis. The Z1 and Z2 axes are aligned with the sliding motion axis of J3 and the rotary motion axis of J5

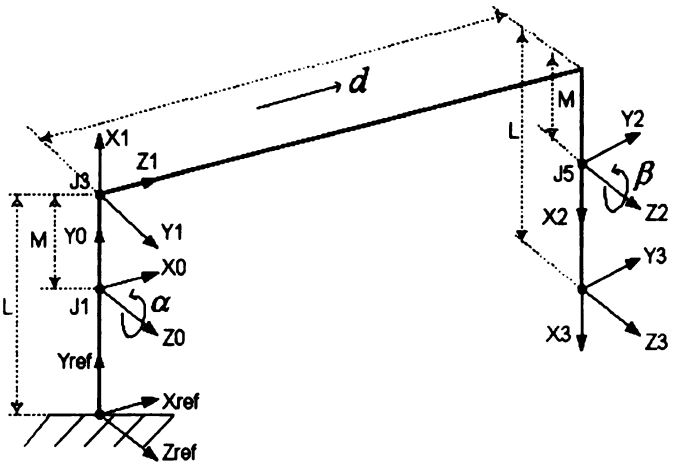

Figure 2.7: Coordinate frames: LFS_translation mode

respectively. The right foot can move freely with the "end-effector frame" attached at the center of suction cup. The L is the robot height and M is the distance between robot ankle to the leg. The prismatic distance between the centers of the robot feet is denoted as d.

The link coordinate parameters in LFS_translation mode is shown in Table 2.2, where α and β and d are joint variables. The four geometric parameters associated with each link in Table 2.2 are defined as follows:

- θ_i is the joint angle from the X_{i-1} axis to the X_i axis about the Z_{i-1} axis (using the right-hand rule).

- d_i is the distance from the origin of the $(i-1)$th coordinate frame to the intersection of the Z_{i-1} axis with the X_i axis along the Z_{i-1} axis.

- a_i is the offset distance from the intersection of the Z_{i-1} axis with the X_i axis to the origin of the ith frame along the X_i axis (or the shortest distance between the Z_{i-1} and Z_i axes).

- α_i is the offset angle from the Z_{i-1} axis to the Z_i axis about the X_i axis (using the right-hand rule).

Table 2.2: Link coordinate parameters: LFS_translation mode

Motion Joint	θ_i	α_i	a_i	d_i
J1 rotate	α	90^o	M	0
J3 translate	180^o	90^o	M	d
J5 rotate	β	0	L−M	0

LFS_spin mode: The assignment of coordinate frames for LFS_spin mode is shown in Figure 2.8.

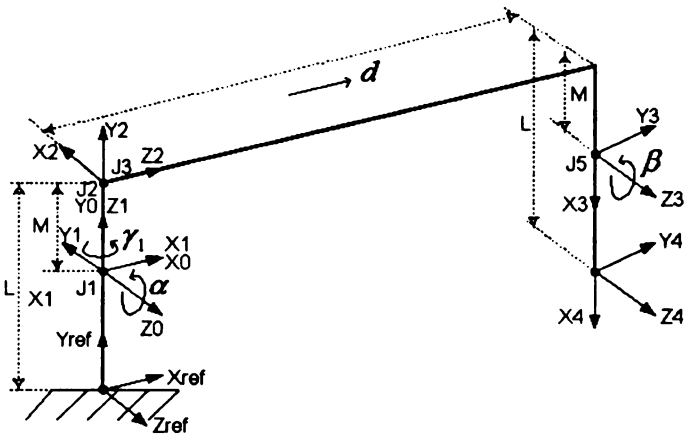

Figure 2.8: Coordinate frames: LFS_spin mode

In LFS_spin mode, four joints involved in the motion. Those are the rotational joint J1 and J5, prismatic slide motion of J3 and spin motion of J2. Note that Z1 is aligned with the spin motion axis of J2. The link coordinate parameters in LFS_spin mode is shown in Table 2.3, where α, β, γ_1 and d are joint variables.

Table 2.3: Link coordinate parameters: LFS_spin mode

Motion Joint	θ_i	α_i	a_i	d_i
J1 rotate	α	-90^o	0	0
J2 spin	γ_1	90^o	0	M
J3 translate	-90^o	90^o	M	d
J5 rotate	β	0	L-M	0

In LFS_spin mode, the slide motion of joint 3 and the spin motion of joint 2 are coupled, both are driven by the motor 2. The relation is expressed as $d = k\gamma_1$, where k is a constant.

2.2.2 Forward Kinematics

Once the D-H coordinate system has been established for each robot link, a homogeneous transformation matrix ${}^{i-1}\!A_i$ can easily be developed relating the ith coordinate frame to the $(i-1)$th coordinate frame as follows:

$$ {}^{i-1}\!A_i = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.1} $$

The homogeneous matrix ${}^{ref}T_i$ which specifies the location of the ith coordinate frame with respect to the reference coordinate system is the chain product of successive coordinate transformation matrices of ${}^{i-1}\!A_i$, and is expressed as

$$ {}^{ref}T_i = {}^{ref}A_0 {}^{0}A_1 {}^{1}A_2 \cdots {}^{i-1}\!A_i \tag{2.2} $$

The robot kinematic model is expressed by the 4×4 homogeneous transformation matrix $T = {}^{ref}T_n$, n is the number of robot moving joints. The T matrix has the combined effect of rotation, translation, perspective, and global scaling, and can be

expressed as:

$$T = \begin{bmatrix} \vec{n} & \vec{s} & \vec{a} & \vec{p} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \left[\begin{array}{cccc|c} n_x & s_x & a_x & & p_x \\ n_y & s_y & a_y & & p_y \\ n_z & s_z & a_z & & p_z \\ \hline 0 & 0 & 0 & & 1 \end{array}\right] = \left[\begin{array}{c|c} R_{3\times3} & P_{3\times1} \\ \hline f_{1\times3} & 1\times1 \end{array}\right], \tag{2.3}$$

where the upper right submatrix $P_{3\times1}$ represents the end-effector position with respect to the reference frame; the upper left submatrix $R_{3\times3}$ is the rotation matrix, representing the orientation of the end-effector frame; the lower left submatrix $f_{1\times3}$ represents perspective transformation which is useful for computer vision. The vector $\vec{n}$ is the normal vector which is aligned with the direction of X axis of the end-effector frame and is orthogonal to the suction cup surface. The sliding vector $\vec{s}$ is pointing in the direction of the Y axis of the end-effector frame which aligns with the robot slide motion direction. The approach vector $\vec{a}$ is aligned with the direction of Z of the end-effector.

LFS_translation mode: By plugging in the parameter value in Table 2.2 to Equation 2.1, the transformation matrices for adjacent coordinate frames can be derived as follows:

$${}^{ref}A_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & L-M \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

$$
{}^{0}A_1 = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha & M\cos\alpha \\ \sin\alpha & 0 & -\cos\alpha & M\sin\alpha \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.5}
$$

$$
{}^{1}A_2 = \begin{bmatrix} -1 & 0 & 0 & -M \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.6}
$$

$$
{}^{2}A_3 = \begin{bmatrix} \cos\beta & -\sin\beta & 0 & (L-M)\cos\beta \\ \sin\beta & \cos\beta & 0 & (L-M)\sin\beta \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.7}
$$

According to Equation 2.2, the robot kinematics matrix is derived as:

$$
T_{translation} = {}^{ref}T_3 = {}^{ref}A_0\,{}^{0}A_1\,{}^{1}A_2\,{}^{2}A_3 \tag{2.8}
$$

$$
= \begin{bmatrix} -\cos(\alpha+\beta) & \sin(\alpha+\beta) & 0 & -(L-M)\cos(\alpha+\beta)+d\sin\alpha \\ -\sin(\alpha+\beta) & -\cos(\alpha+\beta) & 0 & -(L-M)[\sin(\alpha+\beta)-1]-d\cos\alpha \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

The rotation submatrix $R_{3\times3}$ needs nine elements to completely describe the end-effector orientation of robot. It is necessary to find a convenient expression. A set of Euler angles (ψ), (θ), (ϕ), are used to describe the orientation with respect to a

fixed reference frame. There are many different types of Euler angle representation [18]. Here the roll($R_{z,\phi}$), pitch($R_{y,\theta}$), yaw ($R_{x,\psi}$) were used to represent the robot orientation. They correspond to the following rotations in sequence:

1. A rotation of ψ about the X_{ref} axis ($R_{x,\psi}$).
2. A rotation of θ about the Y_{ref} axis ($R_{y,\theta}$).
3. A rotation of ϕ about the Z_{ref} axis ($R_{z,\phi}$).

The resultant composite rotation matrix is:

$$\begin{aligned} R_{\psi,\theta,\phi} &= R_{z,\phi}R_{y,\theta}R_{x,\psi} \\ &= \begin{bmatrix} \cos\phi\cos\theta & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \sin\phi\cos\theta & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi \\ -\sin\theta & \cos\theta\sin\psi & \cos\theta\cos\psi \end{bmatrix} \end{aligned} \tag{2.9}$$

Comparing this matrix with the rotation submatrix $R_{3\times3}$ in Equation 2.8, we have the following relationship between the joint angle and Euler angle.

$$\begin{cases} \psi = 0 \\ \theta = 0 \\ \phi = \alpha + \beta - \pi \end{cases}$$

LFS_spin mode: By plugging in the parameter value in Table 2.3 to Equation 2.1, the transformation matrices for adjacent coordinate frames in LFS_spin mode can be derived as follows:

$$^{ref}A_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & L-M \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.10}$$

$$
{}^{0}A_1 = \begin{bmatrix} \cos\alpha & 0 & -\sin\alpha & 0 \\ \sin\alpha & 0 & \cos\alpha & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.11}
$$

$$
{}^{1}A_2 = \begin{bmatrix} \cos\gamma_1 & 0 & \sin\gamma_1 & 0 \\ \sin\gamma_1 & 0 & -\cos\gamma_1 & 0 \\ 0 & 1 & 0 & M \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.12}
$$

$$
{}^{2}A_3 = \begin{bmatrix} 0 & 0 & -1 & 0 \\ -1 & 0 & 0 & -M \\ 0 & 1 & 0 & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.13}
$$

$$
{}^{3}A_4 = \begin{bmatrix} \cos\beta & -\sin\beta & 0 & (L-M)\cos\beta \\ \sin\beta & \cos\beta & 0 & (L-M)\sin\beta \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.14}
$$

According to Equation 2.2, the robot kinematics matrix in LFS_spin mode is derived as:

$$
T_{spin} = {}^{ref}T_4 = {}^{ref}A_0\,{}^{0}A_1\,{}^{1}A_2\,{}^{2}A_3\,{}^{3}A_4 \tag{2.15}
$$

$$= \begin{bmatrix} \sin\alpha\cos\beta + \cos\alpha\sin\beta\sin\gamma_1 & -\sin\alpha\sin\beta + \cos\alpha\cos\beta\sin\gamma_1 & -\cos\alpha\cos\gamma_1 & p_x \\ -\cos\alpha\cos\beta + \sin\alpha\sin\beta\sin\gamma_1 & \cos\alpha\sin\beta + \sin\alpha\cos\beta\sin\gamma_1 & -\sin\alpha\cos\gamma_1 & p_y \\ \sin\beta\cos\gamma_1 & \cos\beta\cos\gamma_1 & \sin\gamma_1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where

$$\begin{aligned} p_x &= (L-M)(\sin\alpha\cos\beta + \cos\alpha\sin\beta\sin\gamma_1) + d\cos\alpha\sin\gamma_1 \\ p_y &= (L-M)(1 - \cos\alpha\cos\beta + \sin\alpha\sin\beta\sin\gamma_1) + d\sin\alpha\sin\gamma_1 \\ p_z &= (L-M)\sin\beta\cos\gamma_1 + d\cos\gamma_1 \\ d &= k\gamma_1 \end{aligned}$$

It is easy to use Euler angles system I [18] to represent the robot orientation. They correspond to the following sequence of rotations:

1. A rotation of ϕ angle about the Z_{ref} axis $(R_{z,\phi})$.
2. A rotation of θ about the rotated X'_{ref} axis $(R_{x',\theta})$.
3. Finally, a rotation of ψ about the rotated Z'_{ref} axis $(R_{z',\psi})$.

The resultant composite rotation matrix is:

$$R_{\psi,\theta,\phi} = R_{z,\phi}R_{y,\theta}R_{x,\psi} \tag{2.16}$$

$$= \begin{bmatrix} \cos\phi\cos\psi - \sin\phi\cos\theta\sin\psi & -\cos\phi\sin\psi - \sin\phi\cos\theta\cos\psi & \sin\phi\sin\theta \\ \sin\phi\cos\psi + \cos\phi\cos\theta\sin\psi & -\sin\phi\sin\psi + \cos\phi\cos\theta\cos\psi & -\cos\phi\sin\theta \\ \sin\theta\sin\psi & \sin\theta\cos\psi & \cos\theta \end{bmatrix}$$

Comparing this matrix with the rotation submatrix $R_{3\times3}$ in Equation 2.15, we

have the following relationship between the joint angle and Euler angle.

$$\begin{cases} \psi = \alpha - \pi/2 \\ \theta = \pi/2 - \gamma_1 \\ \phi = \beta \end{cases}$$

Thus far, the robot transformation matrix has been deduced for both motion modes. The solution of forward kinematic problem is obtained by evaluating each element in T matrix for a given set of joint variables. The robot orientation can also be represented by the Euler angles.

2.2.3 Inverse Kinematics

In this section, the inverse kinematics are studied, *i.e.*, for a given robot position and orientation, derive the corresponding joint variables which can make the robot reach the desired configuration.

The solution of inverse kinematics can be obtained in many forms. In the motion planning and control of the robot, the orientation vectors play a critical role. In order to make the suction foot grip the surface firmly, we must ensure that the normal vector $\vec{n}$ be perpendicular to the contact surface. By specifying the position vector $\vec{p}$ and the foot normal vector $\vec{n}$, the relation between the suction foot and the contact surface is determined. Therefore, it will bring convenience in the implementation if the solution of inverse kinematics only contains the two vectors $\vec{n}$ and $\vec{p}$.

Among the possible solutions of robot inverse kinematics, some representation of the solution may be inconsistent and ill-conditioned. For example, the arc cosine function does not behave well since its accuracy in determining the angle is dependent on the angle, *i.e.*, $\cos\theta = \cos(-\theta)$. In order to evaluate θ for $-\pi \leq \theta \leq \pi$, an arc tangent function, *atan*2(y,x), which returns $\tan^{-1}(y/x)$ adjusted to the proper

quadrant is defined and will be used in the solution of inverse kinematics.

$$\theta = atan2(y,x) = \begin{cases} 0^o \leq \theta \leq 90^o & for \ +x \ \ and \ \ +y \\ 90^o \leq \theta \leq 180^o & for \ -x \ \ and \ \ +y \\ -180^o \leq \theta \leq -90^o & for \ -x \ \ and \ \ -y \\ -90^o \leq \theta \leq 0^o & for \ +x \ \ and \ \ -y \end{cases} \tag{2.17}$$

Inverse kinematics analysis is considered in LFS_translation and LFS_spin mode respectively.

LFS_translation mode: The transformation matrix is rewritten as in the following.

$$T = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.18}$$

$$= \begin{bmatrix} -\cos(\alpha+\beta) & \sin(\alpha+\beta) & 0 & -(L-M)\cos(\alpha+\beta)+d\sin\alpha \\ -\sin(\alpha+\beta) & -\cos(\alpha+\beta) & 0 & -(L-M)[\sin(\alpha+\beta)-1]-d\cos\alpha \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

By comparing both sides of this Equation, we have:

$$\frac{n_y}{n_x} = \frac{\sin(\alpha+\beta)}{\cos(\alpha+\beta)} \quad \Longrightarrow \quad \alpha+\beta = \arctan(n_y, n_x) \tag{2.19}$$

$$\begin{cases} d\sin\alpha = p_x - (L-M)n_x \\ d\cos\alpha = -p_y + (L-M)(n_y+1) \end{cases} \Longrightarrow \begin{cases} \tan\alpha = \frac{p_x-(L-M)n_x}{-p_y+(L-M)(n_y+1)} \\ d = \frac{p_x-(L-M)n_x}{\sin\alpha} \end{cases} \tag{2.20}$$

From Equation 2.19 and 2.20, the joint variables are solved using position vector $\vec{p}$ and the foot normal vector $\vec{n}$:

$$\begin{cases} \alpha &= atan2\ [p_x - (L-M)n_x,\ -p_y + (L-M)(n_y+1)] \\ \beta &= atan2\ [n_y,\ n_x] - \alpha \\ d &= \frac{p_x-(L-M)n_x}{\sin\alpha} \end{cases} \tag{2.21}$$

LFS_spin mode: By analyzing the Equation 2.15, we have:

$$\begin{cases} p_x = (L-M)n_x + d\cos\alpha\sin\gamma_1 \\ p_y = (L-M)(1+n_y) + d\sin\alpha\sin\gamma_1 \end{cases} \Longrightarrow \tan\alpha = \frac{p_y-(L-M)(n_y+1)}{p_x-(L-M)n_x} \tag{2.22}$$

$$\begin{cases} p_x = (L-M)n_x + d\cos\alpha\sin\gamma_1 \\ p_z = (L-M)n_z + d\cos\gamma_1 \end{cases} \Longrightarrow \tan\gamma_1 = \frac{p_x-(L-M)n_x}{(p_z-(L-M)n_z)\cos\alpha} \tag{2.23}$$

$$\begin{cases} n_z = \sin\beta\cos\gamma_1 \\ s_z = \cos\beta\cos\gamma_1 \end{cases} \Longrightarrow \tan\beta = \frac{n_z}{s_z} \tag{2.24}$$

From Equations 2.22, 2.23 and 2.24, the joint variables are solved as follows:

$$\begin{cases} \alpha &= atan2[p_y-(L-M)(n_y+1),\ p_x-(L-M)n_x] \\ \gamma_1 &= atan2[p_x-(L-M)n_x,\ (p_z-(L-M)n_z)\cos\alpha] \\ d &= k\gamma_1 \\ \beta &= atan2(n_z,\ s_z) \end{cases} \tag{2.25}$$

So far, the robot inverse kinematics has been derived for both motion modes. The solutions of the inverse kinematics will be used in the implementation of the robot motion controller.

2.2.4 Eulerian Jacobian

This section derives the Eulerian Jacobian, which relates the linear velocity and the Euler angular velocities to the joint velocities. By differentiating the Eulerian angle and the position vector $\vec{p}$ with respect to time t, the Jacobian matrix is derived as follows.

LFS_translation mode:

$$\begin{cases} \frac{dp_x}{dt} = (L-M)\sin(\alpha+\beta)\dot{\alpha} + 2d\cos\alpha\dot{\alpha} + (L-M)\sin(\alpha+\beta)\dot{\beta} + 2\sin\alpha\dot{d} \\ \frac{dp_y}{dt} = -(L-M)\cos(\alpha+\beta)\dot{\alpha} + 2d\sin\alpha\dot{\alpha} - (L-M)\cos(\alpha+\beta)\dot{\beta} - 2\cos\alpha\dot{d} \\ \omega_z = \dot{\alpha} + \dot{\beta} \end{cases} \quad (2.26)$$

In matrix form, we have

$$\begin{bmatrix} \dot{p}_x \\ \dot{p}_y \\ \omega_z \end{bmatrix} = J_{3\times3} \begin{bmatrix} \dot{\alpha} \\ \dot{d} \\ \dot{\beta} \end{bmatrix} \quad (2.27)$$

$$J_{3\times3} = \begin{bmatrix} (L-M)\sin(\alpha+\beta) + 2d\cos\alpha & 2\sin\alpha & (L-M)\sin(\alpha+\beta) \\ -(L-M)\cos(\alpha+\beta) + 2d\sin\alpha & -2\cos\alpha & -(L-M)\cos(\alpha+\beta) \\ 1 & 0 & 1 \end{bmatrix}$$

and $p_z = 0$, $\omega_x = 0$, $\omega_y = 0$.

LFS_spin mode: The Jacobian matrix is derived as

$$\begin{bmatrix} \dot{p}_x \\ \dot{p}_y \\ \dot{p}_z \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = J_{6\times 4} \begin{bmatrix} \dot{\alpha} \\ \dot{\gamma}_1 \\ \dot{d} \\ \dot{\beta} \end{bmatrix} \tag{2.28}$$

where

$$\begin{aligned}
J[1,1] &= (L-M)(\cos\alpha\cos\beta - \sin\alpha\sin\beta\sin\gamma_1) - d\sin\alpha\sin\gamma_1 \\
J[1,2] &= (L-M)\cos\alpha\sin\beta\cos\gamma_1 + d\cos\alpha\cos\gamma_1 \\
J[1,3] &= \cos\alpha\sin\gamma_1 \\
J[1,4] &= (L-M)(-\sin\alpha\sin\beta + \cos\alpha\cos\beta\sin\gamma_1) \\
J[2,1] &= (L-M)(\sin\alpha\cos\beta + \cos\alpha\sin\beta\sin\gamma_1) + d\cos\alpha\sin\gamma_1 \\
J[2,2] &= (L-M)\sin\alpha\sin\beta\cos\gamma_1 + d\sin\alpha\cos\gamma_1 \\
J[2,3] &= \sin\alpha\sin\gamma_1 \\
J[2,4] &= (L-M)(\cos\alpha\sin\beta + \sin\alpha\cos\beta\sin\gamma_1) \\
J[3,1] &= 0 \\
J[3,2] &= -(L-M)\sin\beta\sin\gamma_1 - d\sin\gamma_1 \\
J[3,3] &= \cos\gamma_1 \\
J[3,4] &= (L-M)\cos\beta\cos\gamma_1) \\
J[4] &= [\ 1 \quad 0 \quad 0 \quad 0\ ] \quad J[5] = [\ 0 \quad -1 \quad 0 \quad 0\ ] \quad J[6] = [\ 0 \quad 0 \quad 0 \quad 1\ ]
\end{aligned}$$

2.2.5 *Simulation*

Forward kinematics deals with the following question: given the joint variables (rotary or translation) and the geometric link parameters, what is the position and orientation of the robot end-effector. On the other hand, the inverse kinematics addresses the problem: given a desired position and orientation of the end-effector and the robot link parameter, what is the corresponding joint variables which make the robot reach the desired prescribed configuration.

Computer simulation is conducted to verify the validity of the robot kinematic model as shown in Figure 2.9. The software initially generates locations in the joint space within the joint variable limits. These joint variables are inputed into the forward kinematic model to obtain the transformation matrix T. The T matrix is then fed into the inverse kinematic model to obtain the joint angle solution which should agree to the desired joints angles previously fed into the direct kinematics routine.

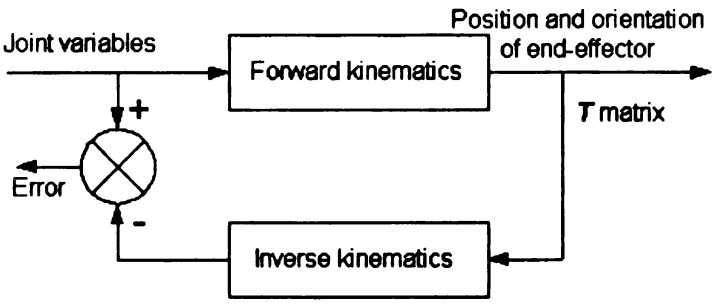

Figure 2.9: Block diagram of kinematic model simulation

The Tables in Figure 2.10 and 2.11 show some list of the simulation results. Based on the desired joint variables, the forward kinematics calculate the robot position and orientation of its free foot related to the reference frame located at the center of its standing foot. Then for each position and orientation, the inverse kinematics is employed to obtain the corresponding joint variables which lead to that configuration. Due to the mechanical stop, each joint variable has its motion limits. The simulation

is conducted within the following joint variable ranges: $0 \leq \alpha \leq 60^0$; $0 \leq \gamma_1 \leq 60^o$, $-30^o \leq \beta \leq 90^o$. The coupling between spin angle and robot length in LFS_spin mode is expressed as $(94 - 0.2 \times \gamma_1) \leq d \leq 94$ mm. The limit of translation distance in LFS_translation mode is $94 \leq d \leq 140$ mm.

Simulation results of robot kinematic model in LFS_spin mode															
Desired joint variables				Position			Orientation			Inverse kinematics			
(degree)			(mm)	(mm)			X	Y	Z	(degree)			(mm)
α	γ_1	β	d	x	y	z	[x,y,z]	[x,y,z]	[x,y,z]	α'	γ_1'	β'	d'
0	0	0	94.0	94.0	0	0	[0,-1,0]	[1,0,0]	[0,0,1]	0	0	0	94.0
0	0	$\pi/3$	94.0	131.24	21.5	0	[0.866,-0.5,0]	[0.5,0.866,0]	[0,0,1]	0	0	$\pi/3$	94.0
0	0	$\pi/2$	94.0	137.0	43.9	0	[1,0,0]	[0,1,0]	[0,0,1]	0	0	$\pi/2$	94.0
0	$\pi/4$	0	93.84	66.36	0	-66.36	[0,-1,0]	[.707,0,-.707]	[.707,0,.707]	0	$\pi/4$	0	93.84
0	$\pi/4$	$\pi/6$	93.84	81.56	5.76	-81.56	[.3536,-.866,-.3536]	[.612,.5,-.612]	[.707,0,.707]	0	$\pi/4$	$\pi/6$	93.84
$\pi/6$	0	0	94.0	102.91	52.76	0	[0.5,-0.866,0]	[0.866,0.5,0]	[0,0,1]	$\pi/6$	0	0	94.0
$\pi/6$	0	$-\pi/6$	94.0	81.41	47.0	0	[0,-1,0]	[1,0,0]	[0,0,1]	$\pi/6$	0	$-\pi/6$	94.0
$\pi/6$	0	$\pi/3$	94.0	124.41	90.0	0	[1,0,0]	[0,1,0]	[0,0,1]	$\pi/6$	0	$\pi/3$	94.0
$\pi/6$	$\pi/3$	$\pi/3$	93.79	67.49	57.14	-113.5	[.625,-.217,-.75]	[-.217,.875,-.433]	[.75,.433,.5]	$\pi/6$	$\pi/3$	$\pi/3$	93.79
$\pi/3$	$\pi/3$	$\pi/2$	93.79	34.2	102.23	-118.46	[.65,.125,-.75]	[-.625,.65,-.433]	[.433,.75,.5]	$\pi/3$	$\pi/3$	$\pi/2$	93.79

Figure 2.10: Simulation results of robot kinematic model in LFS_spin

Simulation results of robot kinematic model in LFS_translation mode												
Desired joint variables			Position			Orientation			Inverse kinematics		
(degree)		(mm)	(mm)			X	Y	Z	(degree)		(mm)
α	β	d	x	y	z	[x,y,z]	[x,y,z]	[x,y,z]	α'	β'	d'
0	0	94.0	94.0	0	0	[0,-1,0]	[1,0,0]	[0,0,1]	0	0	94.0
0	0	100.0	100.0	0	0	[0,-1,0]	[1,0,0]	[0,0,1]	0	0	100.0
$\pi/6$	0	100.0	108.1	55.76	0	[0.5,-0.866,0]	[0.866,0.5,0]	[0,0,1]	$\pi/6$	0	100.0
$\pi/6$	$-\pi/6$	100.0	86.60	50.0	0	[0,-1,0]	[1,0,0]	[0,0,1]	$\pi/6$	$-\pi/6$	100.0
$\pi/4$	$\pi/6$	120.0	126.39	116.7	0	[0.966,-0.259,0]	[0.259,0.966,0]	[0,0,1]	$\pi/4$	$\pi/6$	120.0
$\pi/3$	$\pi/2$	140.0	91.5	201.48	0	[0.5,0.866,0]	[-0.866,0.5,0]	[0,01]	$\pi/3$	$\pi/2$	140.0

Figure 2.11: Simulation results of robot kinematic model in LFS_translation

The simulation results show that the desired joint variables and the calculated joint variables matches well. This verifies the correctness of the robot kinematic model.

2.3 Locomotion Analysis

In order to generate smooth locomotion, the robot is required to solve the robot kinematics in real-time and generate synchronized motion control of multiple joints. The key point in the motion control is to determine which motion modes should be applied to and combined together to accomplish the task. In order to distinguish between different motion modes, a contact switch is installed on each leg to determine whether the leg and rack are locked together. The contact switches give us the information on motion mode switching so that different kinematic models are used to plan the robot trajectory.

For example, in order to accomplish "move forward" walking task, LFS_translation and RFS_translation motion modes get involved. Assuming foot 1 is initially the standing foot, and foot 2 is the front foot, the sequence of the joint motions to accomplish one step of moving forward is illustrated in Figure 2.12.

1. Rotate J1 to lift up the front foot off the ground, and slide J3 in LFS_translation mode to extend the robot by a stride of d.

2. Rotate J1 again in opposite direction to make the front foot touch the ground.

3. After switching the standing foot, rotate J5 to lift up the rear foot off the ground, and slide J3 in RFS_translation mode to contract the robot and bring the rear foot forward by a distance of d.

4. Rotate J5 again to make the rear foot touch the surface.

With the completion of this sequence, the standing foot 1 of the robot has moved forward d distance in straight line. Stages 1 is governed by LFS_translation motion mode and stage 3 by RFS_translation motion mode, thus, Equations 2.8 and 2.21 should be used in the trajectory planning and motion control.

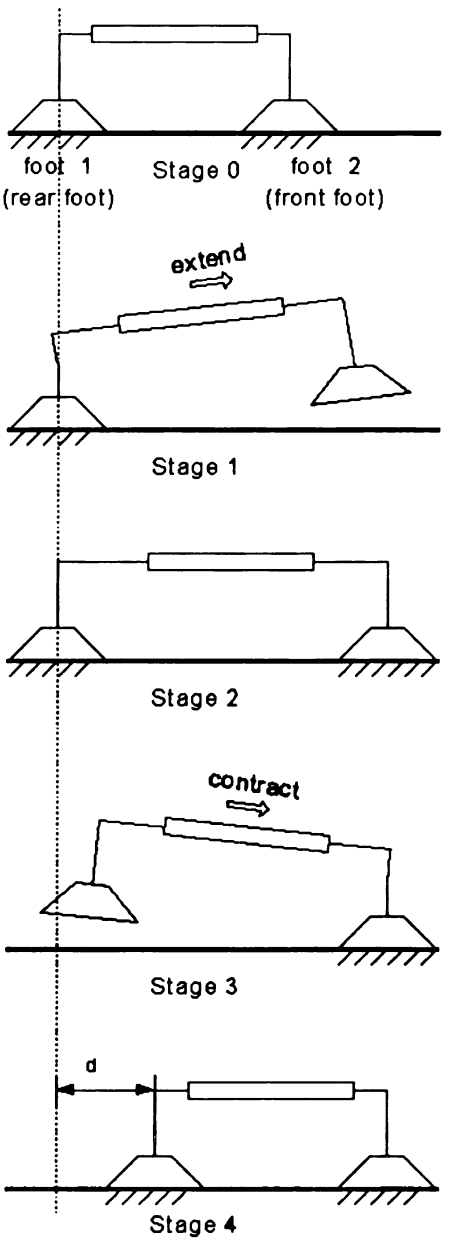

Figure 2.12: Locomotion sequence to accomplish "Move forward"

In order to change the direction of its walk, the robot needs to steer one of its feet. Steering is achieved by performing the following sequence and is illustrated in Figure 2.13.

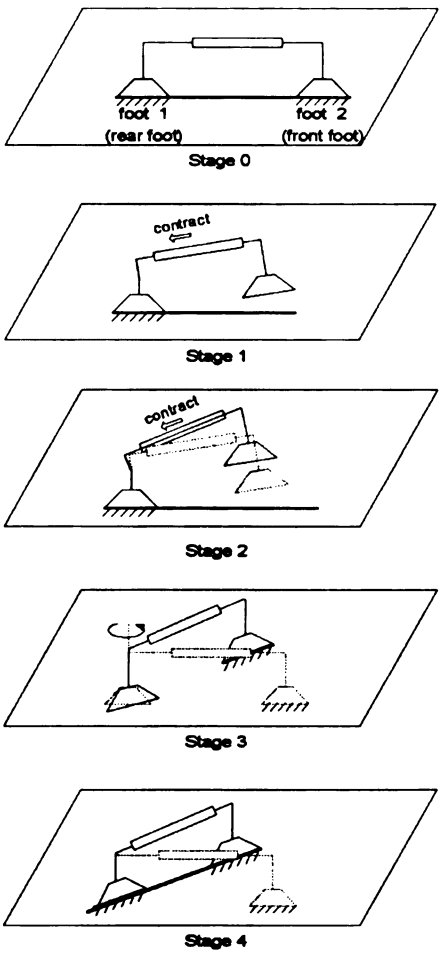

Figure 2.13: Locomotion sequence to accomplish "Make a left turn"

1. Rotate J1 to lift up the front foot off the ground, and slide J3 in LFS_translation mode to contract the robot body until the leg1/rack1 pair is separated.

2. Continue to contract in LFS_spin mode causing the robot body to spin around J2 axis. In this case, foot 1 is wound up.

3. After switching the standing foot, rotate J5 to lift up foot 1 off the surface, and slide J3 in opposite direction in RFS_spin mode to unwind foot 1 until the leg

1 and rack 1 are locked.

4. Rotate J5 again in opposing direction to bring the rear foot in contact with the surface.

Step 1 is controlled in LFS_translation motion mode, which is governed by Equation 2.8. Step 2 is controlled in LFS_spin motion mode, which is governed by Equation 2.15. Step 3 is controlled in RFS_spin motion mode. After this motion sequence, the robot has made a left turn and is ready to move along the new direction.

In one step, the robot cannot reach arbitrarily desired position with the desired orientation. When the robot cannot post the foot to the desired position, it needs to take several steps and adjusts its standing foot properly. The increased complexity of motion planning is the penalty paid for the under-actuation structure.

The robot also has the ability to transit between differently inclined surfaces. Figure 2.14 shows the locomotion sequence when the robot move from a floor to a wall surface. The most important thing the robot needs to do is to locate itself to a proper position in front of the wall and then start to transit.

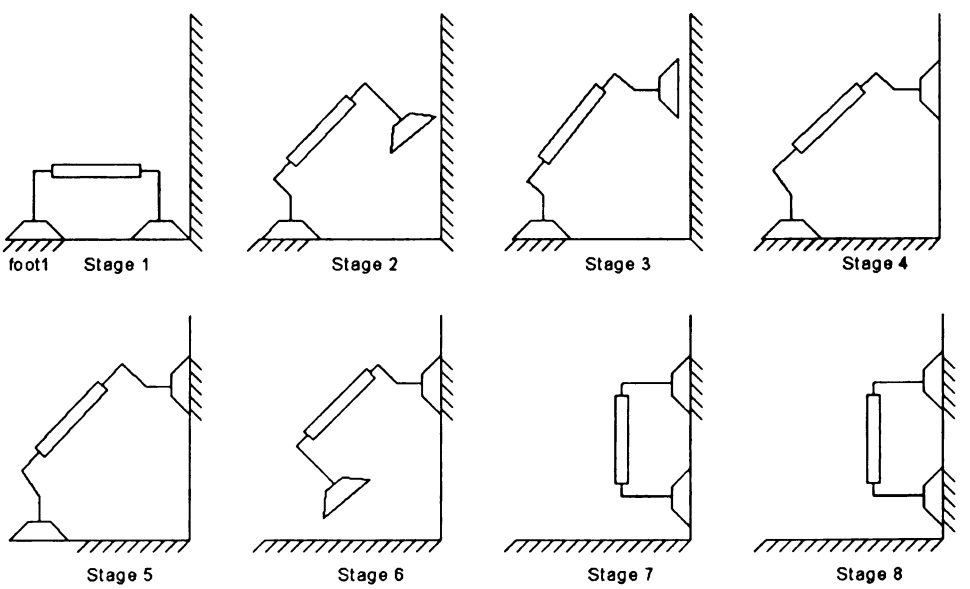

Figure 2.14: Locomotion sequence to transit from ground to a wall surface

1. The robot walks to the proper location close to a wall with the robot main axis perpendicular to the wall surface.

2. Rotate J1 to lift up the front foot to its maximum allowed value.

3. Slide J3 in LFS_translation mode to extend the robot to its maximum length and rotate J5 to align the front foot with the wall surface.

4. Rotate J1 in opposite direction to put the robot down to the wall surface and adjust J5 until the front foot touch the wall seamlessly.

5. Release the suction on the rear foot, and make the front foot the standing foot.

6. Slide J3 in RFS_translation mode until the robot contract to its minimum length, and rotate J1 to adjust the rear foot.

7. Rotate J5 to bring the rear foot in contact with the wall surface.

8. Execute the suction of rear foot and then the robot is ready to move around on the wall.

2.4 Dynamic Model

The robot dynamic model is a set of mathematical equations describing the dynamic behavior of the robot, i.e., the relationship between the robot joint forces/torques and the robot joint positions, velocities and accelerations. Such equations of motion are useful for the design of suitable control laws or strategies to achieve a desired system performance. The dynamic model of a robot can be obtained from known physical laws such as the laws of Newtonian mechanics and Lagrangian mechanics. The derivation of the robot dynamic model based on the Lagrange-Euler (L-E) formulation is straight forward and systematic. This section derives the dynamic model of the CRAWLER robot based on the L-E method.

2.4.1 *Lagrange-Euler Method*

In the L-E formulation, the system's dynamic behavior is described in terms of work and energy using generalized coordinates. All the workless forces and constraint forces are automatically eliminated in this method. By directly using the L-E formulation and the D-H link coordinate representation of a robot arm, the resultant robot dynamic model is a compact matrix description of the robot equations of motion.

Assuming a robot has n links, the L-E formulation is expressed in the following form

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = F_i \quad i = 1, 2, ..., n \tag{2.29}$$

where

- L= Lagrangian function= kinetic energy K − potential energy P
- K= total kinetic energy of the robot
- P= total potential energy of the robot
- q_i= generalized coordinates of the robot
- $\dot{q}_i$=first time derivative of the generalized coordinate, q_i
- F_i = generalized force(or torque) applied to the robot at joint i to drive link i.

The L-E formulation requires knowledge of the kinetic energy of the physical robot system, which in turn requires knowledge of the velocity of each joint. Let ${}^i\mathbf{r}_i$ be a point fixed in a link i and expressed in homogeneous coordinates with respect to ith link coordinate frame,

$$ {}^{i}r_i = \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix} \tag{2.30} $$

Let ${}^{0}r_i$ be the same point ${}^{i}r_i$ with respect to the base coordinate frame, ${}^{i-1}A_i$ the homogeneous coordinate transformation matrix which relates the spatial displacement of the ith link coordinate frame to the $(i-1)$th link coordinate frame, and ${}^{0}A_i$ the coordinate frame matrix which relates the ith coordinate frame to the base coordinate frame; then ${}^{0}r_i$ is related to the point ${}^{i}r_i$ by

$$ {}^{0}r_i = {}^{0}A_i {}^{i}r_i = {}^{0}A_1 {}^{1}A_2 \cdots {}^{i-1}A_i {}^{i}r_i \tag{2.31} $$

The velocity of ${}^{i}r_i$ expressed in the base coordinate frame can be written as

$$ \begin{aligned} v_i &\triangleq {}^{0}v_i = \frac{d}{dt}({}^{0}r_i) = \frac{d}{dt}({}^{0}A_i {}^{i}r_i) \\ &= {}^{0}\dot{A}_i {}^{1}A_2 \cdots {}^{i-1}A_i {}^{i}r_i + {}^{0}A_i {}^{1}\dot{A}_2 \cdots {}^{i-1}A_i {}^{i}r_i + \cdots + {}^{0}A_1 \cdots {}^{i-1}\dot{A}_i \ {}^{i}r_i + {}^{0}A_i {}^{i}\dot{r}_i \\ &= \left(\sum_{j=1}^{i} \frac{\partial {}^{0}A_i}{\partial q_j} \dot{q}_j \right) {}^{i}r_i \end{aligned} \tag{2.32} $$

The partial derivative of ${}^{0}A_i$ with respect to q_j can be easily calculated with the help of a matrix Q_i which, for a revolute joint, is defined as

$$Q_i = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.33}$$

and, for a prismatic joint, as

$$Q_i = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.34}$$

It then follows that

$$\frac{\partial {}^{i-1}A_i}{\partial q_i} = Q_i {}^{i-1}A_i \tag{2.35}$$

Let us define

$$U_{ij} \triangleq \frac{\partial {}^0A_i}{\partial q_j} = \begin{cases} {}^0A_i Q_j {}^{j-1}A_i & for \quad j \leq i \\ 0 & for \quad j > i \end{cases} \tag{2.36}$$

Equation 2.36 can be interpreted as the effect of the motion of joint j on all the points on link i. Using this notation, v_i can be expressed as

$$
\begin{aligned}
v_i &= \left(\sum_{j=1}^{i} \frac{\partial {}^0A_i}{\partial q_j} \dot{q}_j \right) {}^i r_i \\
&= \left(\sum_{j=1}^{i} U_{ij} \dot{q}_j \right) {}^i r_i
\end{aligned}
\tag{2.37}
$$

The interaction effects of the motion of joint j and joint k on all the points on link i can be expressed as

$$
U_{ijk} \triangleq \frac{\partial U_{ij}}{\partial q_k} = \begin{cases} {}^0A_{j-1} Q_j {}^{j-1}A_{k-1} Q_k {}^{k-1}A_i & for \quad i \geq k \geq j \\ {}^0A_{k-1} Q_k {}^{k-1}A_{j-1} Q_j {}^{j-1}A_i & for \quad i \geq j \geq k \\ 0 & for \quad j < j \ or \ i < k \end{cases}
\tag{2.38}
$$

After obtaining the joint velocity of each link, we need to find the kinetic energy of link i. Let dK_i be the kinetic energy of a particle with differential mass dm in link i, then

$$
\begin{aligned}
dK_i &= \frac{1}{2} Tr(v_i v_i^T) dm \\
&= \frac{1}{2} Tr \left[\sum_{p=1}^{i} U_{ip} \dot{q}_p {}^i r_i \left(\sum_{r=1}^{i} U_{ir} \dot{q}_r {}^i r_i \right)^T \right] dm \\
&= \frac{1}{2} Tr \left[\sum_{p=1}^{i} \sum_{r=1}^{i} U_{ip} ({}^i r_i dm {}^i r_i{}^T) U_{ir}{}^T \dot{q}_p \dot{q}_r \right]
\end{aligned}
\tag{2.39}
$$

where the trace operator is the sum of diagonal elements of a square matrix, *i.e.*, $Tr(A) \triangleq \sum_{i=1}^{n} a_{ii}$, where A is any square matrix with elements of a_{ij}.

Then the kinetic energy of link i is

$$K_i = \int dK_i = \frac{1}{2} Tr \left[\sum_{p=1}^{i} \sum_{r=1}^{i} U_{ip} \left(\int {}^i r_i {}^i r_i^T dm \right) U_{ir}^T \dot{q}_p \dot{q}_r \right] \tag{2.40}$$
$$= \frac{1}{2} Tr \left[\sum_{p=1}^{i} \sum_{r=1}^{i} U_{ip} J_i U_{ir}^T \dot{q}_p \dot{q}_r \right]$$

where J_i is the inertia of all the points on link i. J_i is dependent on the mass distribution of link i and is expressed with respect to the ith coordinate frame. Hence, J_i is a coefficient term needs be computed only once and can be expressed in inertia tensor

$$J_i = \int {}^i r_i {}^i r_i^T dm = \begin{bmatrix} \int x_i^2 dm & \int x_i y_i dm & \int x_i z_i dm & \int x_i dm \\ \int x_i y_i dm & \int y_i^2 dm & \int y_i z_i dm & \int y_i dm \\ \int x_i z_i dm & \int y_i z_i dm & \int z_i^2 dm & \int z_i dm \\ \int x_i dm & \int y_i dm & \int z_i dm & \int dm \end{bmatrix} \tag{2.41}$$
$$= \begin{bmatrix} \frac{-I_{xx}+I_{yy}+I_{zz}}{2} & I_{xy} & I_{xz} & m_i \bar{x}_i \\ I_{xy} & \frac{I_{xx}-I_{yy}+I_{zz}}{2} & I_{yz} & m_i \bar{y}_i \\ I_{xz} & I_{yz} & \frac{I_{xx}+I_{yy}-I_{zz}}{2} & m_i \bar{z}_i \\ m_i \bar{x}_i & m_i \bar{y}_i & m_i \bar{z}_i & m_i \end{bmatrix}$$

where I_{xx}, I_{yy}, and I_{zz} are known as the moments of inertia of the link about the axes x, y and z, respectively, which are defined as

$$\left. \begin{aligned} I_{xx} &= \int (y^2 + z^2) dm \\ I_{yy} &= \int (x^2 + z^2) dm \\ I_{zz} &= \int (x^2 + y^2) dm \end{aligned} \right\} \tag{2.42}$$

I_{xy}, I_{xz}, and I_{yz}, are known as the products of inertia which are defined as

$$\left.\begin{aligned} I_{xy} &= I_{yx} = \int xydm \\ I_{xz} &= I_{zx} = \int xzdm \\ I_{yz} &= I_{zy} = \int yzdm \end{aligned}\right\} \tag{2.43}$$

and where ${}^i\bar{r}_i = [\bar{x}_i, \bar{y}_i, \bar{z}_i, 1]^T$ is the center of mass vector of link i expressed in the ith link coordinate frame.

Hence, the total kinetic energy K of the robot is

$$K = \sum_{i=1}^{n} K_i = \frac{1}{2} \sum_{i=1}^{n} \sum_{p=1}^{i} \sum_{r=1}^{i} \left[Tr(U_{ip} J_i {U_{ir}}^T) \dot{q}_p \dot{q}_r \right] \tag{2.44}$$

The total potential energy of the robot can be obtained by summing all the potential energies in each link, P_i

$$P = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n} -m_i \mathbf{g} ({}^0\mathbf{A}_i {}^i\bar{\mathbf{r}}_i) \tag{2.45}$$

where $\mathbf{g} = [\mathbf{g_x}, \mathbf{g_y}, \mathbf{g_z}, 0]$ is a gravity row vector expressed in the base coordinate system, and $|\mathbf{g}| = g = 9.8062 m/sec^2$ is the gravitational constant.

The lagrangian function $L = K - P$ is given by

$$L = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{i} \sum_{k=1}^{i} \left[Tr(U_{ij} J_i {U_{ik}}^T) \dot{q}_j \dot{q}_k \right] + \sum_{i=1}^{n} m_i \mathbf{g} ({}^0\mathbf{A}_i {}^i\bar{\mathbf{r}}_i) \tag{2.46}$$

Applying the L-E formulation to this Lagrangian function yields the generalized

force required to drive the ith link of the robot, for $i = 1, 2, \cdots, n$

$$
\begin{aligned}
F_i &= \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} \qquad (2.47)\\
&= \sum_{j=i}^{n}\sum_{k=1}^{j} \mathrm{Tr}(U_{jk} J_j U_{ji}^{T})\ddot{q}_k + \sum_{j=i}^{n}\sum_{k=1}^{j}\sum_{m=1}^{j} \mathrm{Tr}(U_{jkm} J_j U_{ji}^{T})\dot{q}_k \dot{q}_m - \sum_{j=i}^{n} m_j g U_{ji}\bar{r}_j \\
&= \sum_{k=1}^{n} D_{ik}\ddot{q}_k + \sum_{k=1}^{n}\sum_{m=1}^{n} h_{ikm}\dot{q}_k\dot{q}_m + c_i
\end{aligned}
$$

In a matrix form, the robot dynamics can be expressed as

$$F(t) = D(q(t))\ddot{q}(t) + h(q(t), \dot{q}(t)) + G(q(t)) \qquad (2.48)$$

where $q(t), \dot{q}(t), \ddot{q}(t)$ are $n \times 1$ vectors of generalized joint variable, velocity, and acceleration, respectively, and

- $F(t)$ is an $n \times 1$ generalized torque vector applied at joints $i = 1, 2, \cdots, n$.
- $D(q)$ is an $n \times n$ inertial acceleration-related symmetric matrix.
- $h(q, \dot{q})$ is an $n \times 1$ nonlinear Coriolis and centrifugal force vector.
- $G(q)$ is an $n \times 1$ gravitational force vector.

2.4.2 Derivation of the Dynamic Model

This section derives the dynamic model of the CRAWLER robot. Since the CRAWLER robot is symmetric, with identical leg/rack pairs and foot assemblies, the robot dynamics is analyzed in LFS phase only. The dynamic model is the same for both RFS and LFS phases.

Figure 2.15 shows the assembly of the CRAWLER robot. Assuming foot 1 supports the robot, the coordinate frame is set in the same way as in Figure 2.7, with

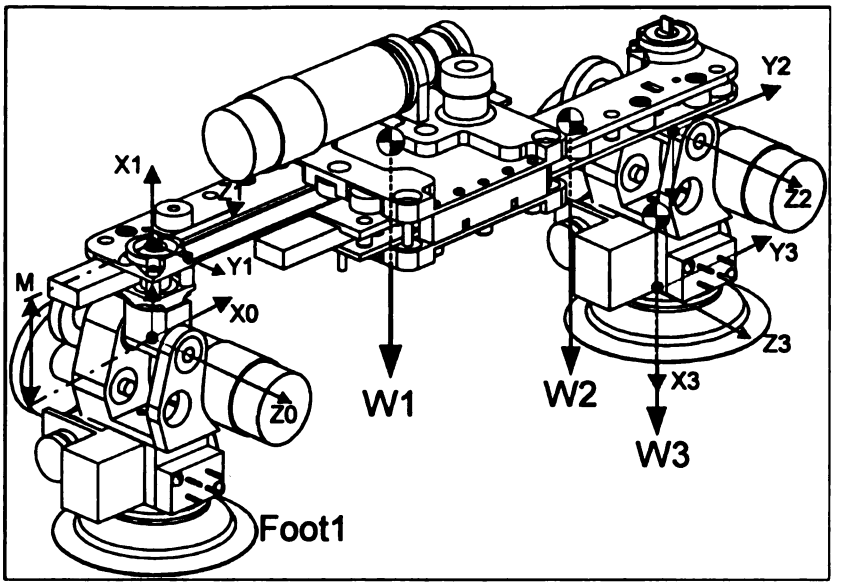

Figure 2.15: CRAWLER robot assembly

$X_0Y_0Z_0$ as the base coordinate frame. The robot consists of three links driven by three motors. Link 1 consists of leg1/rack1 pair and the body plate, this combination rotates around axis Z_0; link 2 is leg2/rack2 assembly, which slides along Z_1 axis; link 3 is foot 2 assembly, which rotates around axis Z_2. The location of the center of mass of each link, and the setup of coordinate frames are illustrated in Figure 2.15. When the robot moves in LFS_translation mode, the generalized joint variables are α, d and β.

As discussed in section 2.2.2, the homogeneous coordinate transformation matrices, ${}^{i-1}A_i$, $i = 1, 2, 3$, are shown in Equations 2.5 to 2.7. By plugging in these matrices to Equation 2.36, we have

$$U_{11} = \frac{\partial^0 A_1}{\partial q_1} = \frac{\partial^0 A_1}{\partial \alpha} \tag{2.49}$$

$$= \begin{bmatrix} -\sin\alpha & 0 & \cos\alpha & -M\sin\alpha \\ \cos\alpha & 0 & \sin\alpha & M\cos\alpha \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$U_{21} = \frac{\partial^0 A_2}{\partial q_1} = \frac{\partial^0 A_2}{\partial \alpha} \tag{2.50}$$

$$= \begin{bmatrix} \sin\alpha & \cos\alpha & 0 & d\cos\alpha \\ -\cos\alpha & \sin\alpha & 0 & d\sin\alpha \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$U_{31} = \frac{\partial^0 A_3}{\partial q_1} = \frac{\partial^0 A_3}{\partial \alpha} \tag{2.51}$$

$$= \begin{bmatrix} \sin(\alpha+\beta) & \cos(\alpha+\beta) & 0 & (L-M)\sin(\alpha+\beta)+d\cos\alpha \\ -\cos(\alpha+\beta) & \sin(\alpha+\beta) & 0 & -(L-M)\cos(\alpha+\beta)+d\sin\alpha \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$U_{22} = \frac{\partial^0 A_2}{\partial q_2} = \frac{\partial^0 A_2}{\partial d} \tag{2.52}$$

$$= \begin{bmatrix} 0 & 0 & 0 & \sin\alpha \\ 0 & 0 & 0 & -\cos\alpha \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$U_{32} = \frac{\partial^0 A_3}{\partial q_2} = \frac{\partial^0 A_3}{\partial d} \tag{2.53}$$

$$= \begin{bmatrix} 0 & 0 & 0 & \sin\alpha \\ 0 & 0 & 0 & -\cos\alpha \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$U_{33} = \frac{\partial^0 A_3}{\partial q_3} = \frac{\partial^0 A_3}{\partial \beta} \tag{2.54}$$

$$= \begin{bmatrix} \sin(\alpha+\beta) & \cos(\alpha+\beta) & 0 & (L-M)\sin(\alpha+\beta) \\ -\cos(\alpha+\beta) & \sin(\alpha+\beta) & 0 & -(L-M)\cos(\alpha+\beta) \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

By using

$$D_{ik} = \sum_{j=max(i,k)}^{n} U_{jk} J_j {U_{ji}}^T \tag{2.55}$$

the elements of the 3×3 inertia matrix $D(q)$ are

$$
\begin{aligned}
D_{11} &= Tr(U_{11}J_1{U_{11}}^T) + Tr(U_{21}J_2{U_{21}}^T) + Tr(U_{31}J_3{U_{31}}^T) \qquad (2.56)\\
D_{12} &= D_{21} = Tr(U_{22}J_2{U_{21}}^T) + Tr(U_{32}J_3{U_{31}}^T)\\
D_{13} &= D_{31} = Tr(U_{33}J_3{U_{31}}^T)\\
D_{22} &= Tr(U_{22}J_2{U_{22}}^T) + Tr(U_{32}J_3{U_{32}}^T)\\
D_{23} &= D_{32} = Tr(U_{33}J_3{U_{32}}^T)\\
D_{33} &= Tr(U_{33}J_3{U_{33}}^T)
\end{aligned}
$$

To derive the 3×1 Coriolis and centrifugal force vector, $h(q, \dot{q}) = [h_1, h_2, h_3]^T$, we us the following equation

$$
h_i = \sum_{k=1}^{3}\sum_{m=1}^{3} h_{ikm}\dot{q}_k\dot{q}_m = \sum_{k=1}^{3}\sum_{m=1}^{3}[\sum_{j=max(i,k,m)}^{3} Tr(U_{jkm}J_j{U_{ji}}^T)]\dot{q}_k\dot{q}_m \qquad (2.57)
$$

It is worth noting that the inertial coefficient in J_i can be obtained by using the AutoCAD design parameters and Equation 2.41. Some of the coefficient may be zero, therefore simplifies the expression of $D(q)$ and $h(q, \dot{q})$ greatly.

Since the gravity force plays a dominant role in robot dynamics when the robot moves at slow speeds, we derive the gravity-related terms in detail as follows.

The elements of the gravity term, $G(q) = [c_1, c_2, c_3]^T$, are

$$
c_i = \sum_{j=1}^{3} -(m_j \mathbf{g} \mathbf{U_{ji}} \bar{\mathbf{r}}_j) \qquad (2.58)
$$

Noticing the coordinate system in Figure 2.15, the center of mass of each link j

expressed in the jth link coordinate frame can be simplified as

$$ {}^{1}\bar{r}_1 = \begin{bmatrix} 0 \\ 0 \\ |\bar{z}_1| \\ 1 \end{bmatrix} \quad {}^{2}\bar{r}_2 = \begin{bmatrix} -M \\ -|\bar{y}_2| \\ 0 \\ 1 \end{bmatrix} \quad {}^{3}\bar{r}_3 = \begin{bmatrix} -|\bar{x}_3| \\ 0 \\ 0 \\ 1 \end{bmatrix} $$

The gravity representations in base coordinate frame are different when the CRAWLER robot walks in different places, which are illustrated in Figures 2.16 to 2.19. When the robot walks on a floor, we have $\mathbf{g}= [0, -g, 0, 0]^T$. Similarly, $\mathbf{g}= [0, g, 0, 0]^T$ when the robot sticks on a ceiling, $\mathbf{g}= [-g, 0, 0, 0]^T$ when the robot climbs up along a wall, and $\mathbf{g}= [g, 0, 0, 0]^T$, when the robot climbs down along a wall.

By plugging the necessary parameters in Equation 2.58, the corresponding gravity terms in those cases is derived.

When the CRAWLER robot moves on a ceiling, we have

$$ \begin{aligned} c_1 &= -(m_1\mathbf{g}\mathbf{U}_{11}{}^{1}\bar{\mathbf{r}}_1 + \mathbf{m_2}\mathbf{g}\mathbf{U}_{21}{}^{2}\bar{\mathbf{r}}_2 + \mathbf{m_3}\mathbf{g}\mathbf{U}_{31}{}^{3}\bar{\mathbf{r}}_3) \\ &= [m_1 g|\bar{z}_1| \sin\alpha + m_1 g M \cos\alpha] \\ &\quad +[m_2 g M \cos\alpha + m_2 g(d - |\bar{y}_2|)\sin\alpha] \\ &\quad +[m_3 g(|\bar{x}_3| - (L - M))\cos(\alpha + \beta) + m_3 g d \sin\alpha] \end{aligned} \tag{2.59} $$

$$ \begin{aligned} c_2 &= -(m_2\mathbf{g}\mathbf{U}_{22}{}^{2}\bar{\mathbf{r}}_2 + \mathbf{m_3}\mathbf{g}\mathbf{U}_{32}{}^{3}\bar{\mathbf{r}}_3) \\ &= -(m_2 + m_3) g \cos\alpha \end{aligned} \tag{2.60} $$

$$
\begin{aligned}
c_3 &= -m_3 \mathbf{g} \mathbf{U_{33}}^{\mathbf{3}}\bar{\mathbf{r}}_{\mathbf{3}} \\
&= m_3 g[|\bar{x}_3| - (L - M)] \cos(\alpha + \beta)
\end{aligned} \tag{2.61}
$$

When the robot moves on a floor, we have

$$
\begin{aligned}
c_1 &= -(m_1 \mathbf{g} \mathbf{U_{11}}^{\mathbf{1}}\bar{\mathbf{r}}_{\mathbf{1}} + \mathbf{m_2 g U_{21}}^{\mathbf{2}}\bar{\mathbf{r}}_{\mathbf{2}} + \mathbf{m_3 g U_{31}}^{\mathbf{3}}\bar{\mathbf{r}}_{\mathbf{3}}) \\
&= -[m_1 g|\bar{z}_1| \sin\alpha + m_1 g M \cos\alpha] \\
&\quad -[m_2 g M \cos\alpha + m_2 g(d - |\bar{y}_2|) \sin\alpha] \\
&\quad -[m_3 g(|\bar{x}_3| - (L - M)) \cos(\alpha + \beta) + m_3 g d \sin\alpha]
\end{aligned} \tag{2.62}
$$

$$
\begin{aligned}
c_2 &= -(m_2 \mathbf{g} \mathbf{U_{22}}^{\mathbf{2}}\bar{\mathbf{r}}_{\mathbf{2}} + \mathbf{m_3 g U_{32}}^{\mathbf{3}}\bar{\mathbf{r}}_{\mathbf{3}}) \\
&= (m_2 + m_3) g \cos\alpha
\end{aligned} \tag{2.63}
$$

$$
\begin{aligned}
c_3 &= -m_3 \mathbf{g} \mathbf{U_{33}}^{\mathbf{3}}\bar{\mathbf{r}}_{\mathbf{3}} \\
&= -m_3 g[|\bar{x}_3| - (L - M)] \cos(\alpha + \beta)
\end{aligned} \tag{2.64}
$$

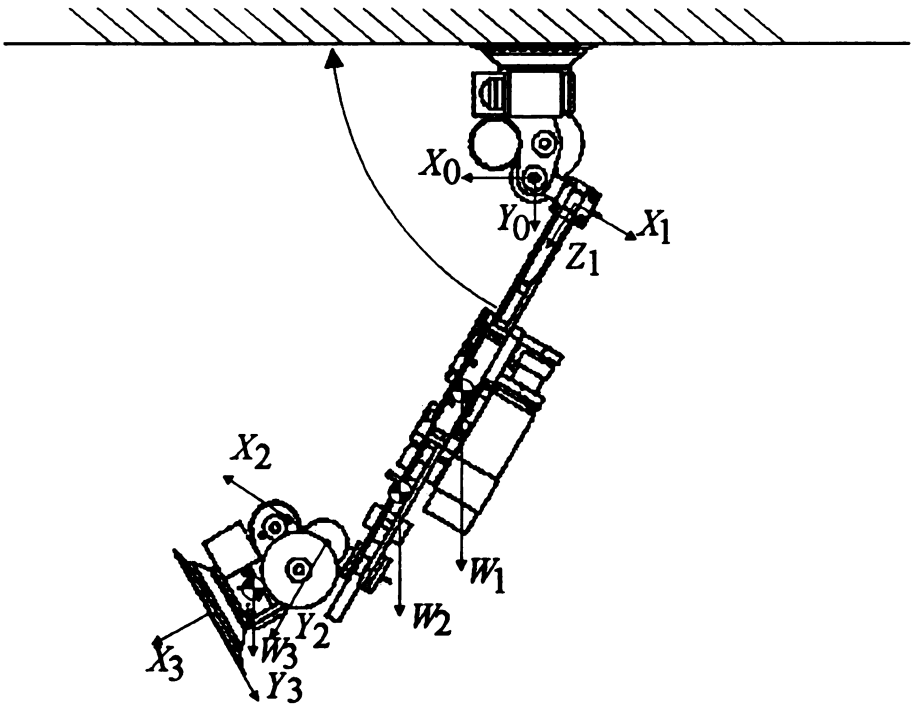

Figure 2.16: Gravity analysis when the CRAWLER robot sticks on a ceiling

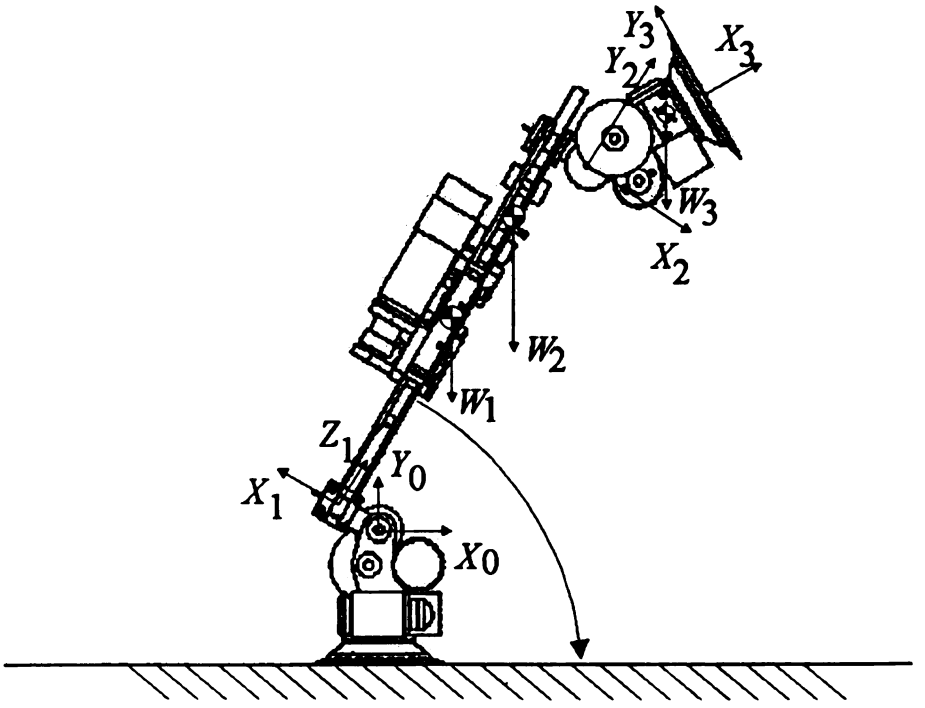

Figure 2.17: Gravity analysis when the CRAWLER robot walks on a floor

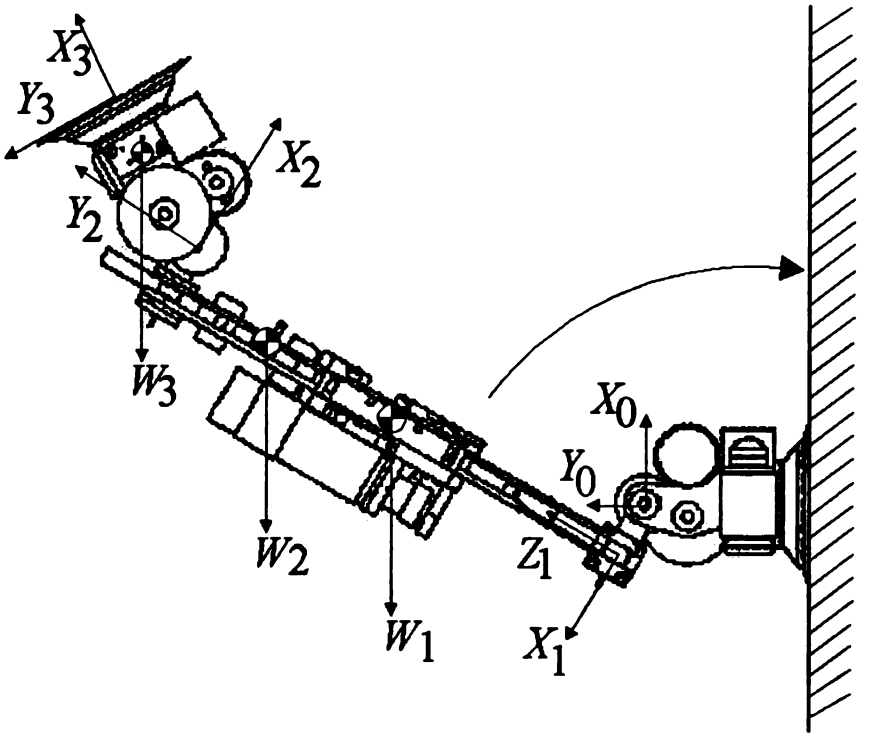

Figure 2.18: Gravity analysis when the CRAWLER robot climbs up a wall

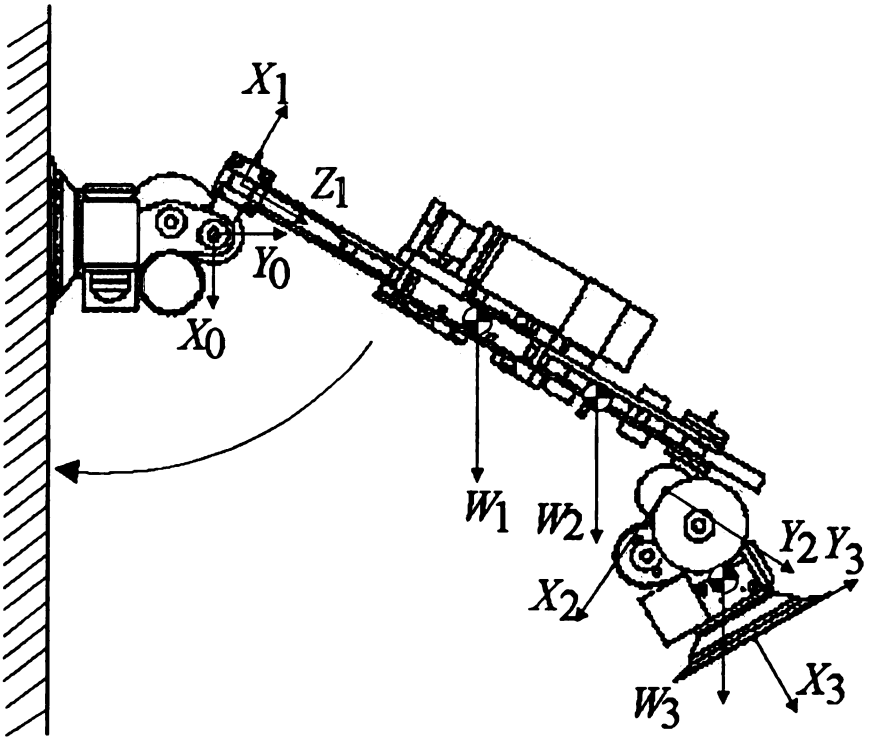

Figure 2.19: Gravity analysis when the CRAWLER robot climbs down a wall

When the robot climbs up on a wall, we have

$$\begin{aligned} c_1 &= -(m_1\mathbf{gU_{11}}^1\bar{\mathbf{r}}_1 + \mathbf{m_2gU_{21}}^2\bar{\mathbf{r}}_2 + \mathbf{m_3gU_{31}}^3\bar{\mathbf{r}}_3) \\ &= [m_1 g|\bar{z}_1|\cos\alpha - m_1 gM\sin\alpha] \\ &\quad +[-m_2 gM\sin\alpha + m_2 g(d-|\bar{y}_2|)\cos\alpha] \\ &\quad +[-m_3 g(|\bar{x}_3|+(L-M))\sin(\alpha+\beta) - m_3 gd\cos\alpha] \end{aligned} \tag{2.65}$$

$$\begin{aligned} c_2 &= -(m_2\mathbf{gU_{22}}^2\bar{\mathbf{r}}_2 + \mathbf{m_3gU_{32}}^3\bar{\mathbf{r}}_3) \\ &= (m_2+m_3)g\sin\alpha \end{aligned} \tag{2.66}$$

$$\begin{aligned} c_3 &= -m_3\mathbf{gU_{33}}^3\bar{\mathbf{r}}_3 \\ &= -m_3 g[|\bar{x}_3| - (L-M)]\sin(\alpha+\beta) \end{aligned} \tag{2.67}$$

When the robot climbs down on a wall, we have

$$\begin{aligned} c_1 &= -(m_1\mathbf{gU_{11}}^1\bar{\mathbf{r}}_1 + \mathbf{m_2gU_{21}}^2\bar{\mathbf{r}}_2 + \mathbf{m_3gU_{31}}^3\bar{\mathbf{r}}_3) \\ &= -[m_1 g|\bar{z}_1|\cos\alpha - m_1 gM\sin\alpha] \\ &\quad -[-m_2 gM\sin\alpha + m_2 g(d-|\bar{y}_2|)\cos\alpha] \\ &\quad -[-m_3 g(|\bar{x}_3|+(L-M))\sin(\alpha+\beta) - m_3 gd\cos\alpha] \end{aligned} \tag{2.68}$$

$$c_2 = -(m_2 \mathbf{g} \mathbf{U_{22}}^{\mathbf{2}}\bar{\mathbf{r}}_{\mathbf{2}} + \mathbf{m_3 g U_{32}}^{\mathbf{3}}\bar{\mathbf{r}}_{\mathbf{3}}) \tag{2.69}$$
$$= -(m_2 + m_3) g \sin\alpha$$

$$c_3 = -m_3 \mathbf{g} \mathbf{U_{33}}^{\mathbf{3}}\bar{\mathbf{r}}_{\mathbf{3}} \tag{2.70}$$
$$= m_3 g[|\bar{x}_3| - (L - M)] \sin(\alpha + \beta)$$

The generalized force can be identified by considering the virtual work done by non-conservative forces acting on the robot. For the CRAWLER robot case, $F = Tor - f$, where $Tor = [tor_1, tor_2, tor_3]^T$ is a 3×1 vector, and tor_i is the torque executed on joint i by motor i. $f = [f_1, f_2, f_3]^T$ is a vector representing friction terms, with

$$f_i = K_{f_i} sgn(\dot{q}_i) \tag{2.71}$$

where K_{f_i} is a constant friction coefficient. $sgn(x)$ is a signum function defined as

$$sgn(x) = \begin{cases} +1 & if \quad x \geq 0 \\ -1 & if \quad x < 0 \end{cases}$$

Hence, the dynamic equation of the CRAWLER robot is given in matrix form by

$$Tor(t) = D(q)\ddot{q}(t) + h(q, \dot{q}) + G(q) + f(\dot{q}) \tag{2.72}$$

where the generalized joint variables are $q_1 \equiv \alpha$, $q_2 \equiv d$, and $q_3 \equiv \beta$.

This equation and its corresponding terms shown in Equations 2.55 to 2.71 describe the dynamic behavior of the CRAWLER robot in terms of the relationship between the input joint torques and the output motion. In practical situation, the CRAWLER robot is unable to move at fast speed. Therefore, the acceleration-related inertia matrix $D(q)$ and the velocity-related Coriolis and centrifugal force vector $h(q, \dot{q})$ are neglectable. The gravitational effects of the links play a dominant role in robot dynamics. A control algorithm which compensate the gravitational effects will be discussed in chapter 3.

2.5 Summary

This chapter presents the modeling of the CRAWLER robot. The mechanical structure and locomotion modes of the robot are introduced. The kinematic model that describes the relation between the joint variables and the position/orientation of the robot's free foot with respect to its standing foot is derived. The robot kinematic model is governed by different equations in different motion modes. Therefore, it is important to use the corresponding kinematic equations when plan the robot motion. In order to study the dynamic behavior of the CRAWLER robot, the dynamic model is derived based on the L-E formulation. The dynamic model describes the relation between the input joint torques and the output motion. The gravitational effects play a dominant role in robot dynamics.

In next chapter, an embedded control system for the CRAWLER robot will be introduced. The control system will use the derived kinematic model and compensate the gravitational effects to achieve a good performance.

CHAPTER 3

EMBEDDED CONTROL SYSTEM DESIGN

As a self-contained embedded control system, the climbing robot needs to carry its own power source, sensors, control system, and associated hardware. Thus minimization of weight and power consumption is critical to prolonged operation. The TMS320LF2407 digital signal processor (DSP) from Texas Instruments (TI) Inc. is an ideal candidate for an embedded controller because of its high-speed performance, its support for multi-motor control, and its low power consumption. This chapter describes the control system design and implementation based on a TI DSP.

3.1 Control System Overview

3.1.1 Actuators and Sensors

To minimize the weight and complexity, the climbing robot has a limited number of actuators and primary sensors. The actuators include three DC servo motors with encoder feedback, two suction pump motors, and two micro valves. The primary sensor components include two pressure sensors and six touch sensors located at the suction feet. In order to distinguish between different motion modes, two contact switches are installed on robot legs to determine whether the leg and rack are locked. For remote control operation the robot has a wireless receiver module, which communicates with the transmitter module in a remote controller. All the signals from those components and sensors need to be processed and integrated into an on-board control system.

Apart from the primary sensors which are critical for operation, additional application sensors can be installed on the robot as payloads when requested by specific tasks. The application sensors include, but are not limited to, wireless cameras, sonar sensors, inclinometers, microphones and infrared sensors. For reconnaissance

purpose, a wireless pin-hole camera is installed and the video images are transmitted to and processed at a host computer.

Figure 3.1 is the close-up of the robot foot where the sensors are shown. The touch sensors are attached to the suction cup in different radial directions. They consist of an outer tube and an inner super-elastic wire isolated by a silicon tube. When the inner wire touches the outer tube the switch closed. In such a way, the touch sensors provide a set of digital inputs to DSP to facilitate the robot in adjusting the suction foot orientation.

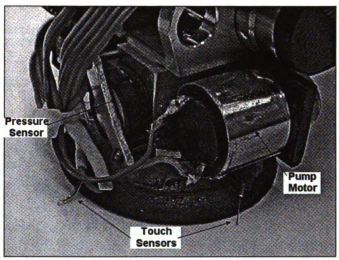

Figure 3.1: Sensors at robot foot

The major components of the robot and their functions are listed in Table 3.1.

To enable teleoperation, a remote controller is built around an RF (radio frequency) transmitter module which is operated at 418 Mhz. The picture of the 16-key remote controller is shown in Figure 3.2. The key sequences encode a set of operator commands.

3.1.2 Control System Structure

The control system structure is illustrated in the block diagram as shown in Figure 3.3. The physical actuators and sensors are represented in the right block. Other

Table 3.1: Actuators and sensors used in the CRAWLER robot

Actuators & sensors	Functions
DC Servo Motor 1	Drives joint 1; adjusts tilt angle of foot 1
DC Servo Motor 2	Drives joint 2, 3, 4; but not all of them simultaneously
DC Servo Motor 3	Drives Joint 5; adjusts tilt angle of foot 2
Encoder 1, 2, 3	Provide joint position information
Suction pump 1, 2	Create suction force, support foot on surfaces
Micro-valve 1, 2	Release the suction foot
Pressure sensor 1, 2	Decide whether the foot is firmly attached to a surface
Touch sensors	Attach to the suction cup in different radial directions; facilitate the adjustment of the suction foot orientation
Contact switch 1, 2	Determine whether the leg/rack pair is engaged; distinguish between different motion modes
Transmitter/Receiver	Wireless communication between the robot and its 16-key remote controller
Wireless camera	Get images and transmit them to a host computer

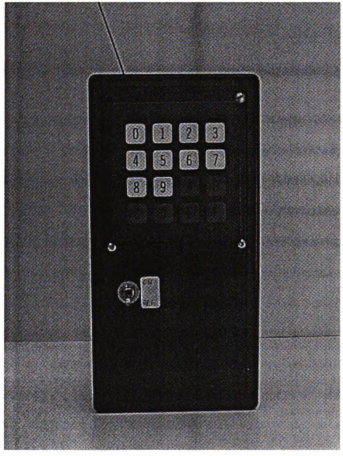

Figure 3.2: Picture of the remote controller

blocks represent the on-board software modules including command interpreter, task level scheduler, trajectory planner, joint level controller and motion planner.

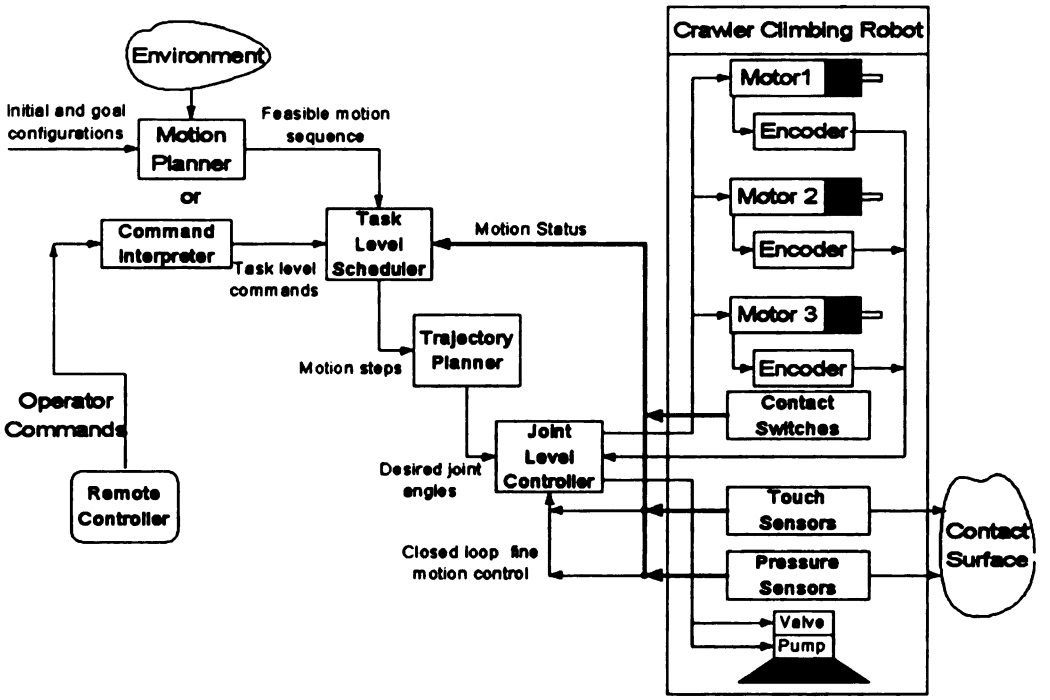

Figure 3.3: Control system block diagram of the climbing robot

The operator commands, such as "move forward", "make left turn", are transmitted from the remote controller held by a human operator and decoded by the on-board command interpreter. The generated task level commands are then fed into the task level scheduler. The task level scheduler uses a finite state machine to keep track of robot motion status and decompose the command into several motion steps. The trajectory planner solves the inverse kinematic model and interpolates the path to generate a set of desired joint angles. The digital motor controller then drives each joint to the desired set points so that the foot is placed to the desired location.

Placing the foot on a surface consists of gross and fine motion controls. The gross motion control is based on the desired solution of the inverse kinematics and it brings the free foot to the neighborhood of the desired position and orientation. However, the results of the actual foot placement may not be sufficiently accurate due to the

existence of uncertainties and disturbances caused by backlash, gear friction, sensor error, and varying gravitational effects. Thus, after the gross motion control, the joint level controller conducts the fine motion control. The fine motion control utilizes the signal from the touch sensors and pressure sensors as a feedback to adjust the robot foot to ensure that it grips the contact surface reliably. Adjustment of the robot foot is a closed loop control process and only requires varying the tilt angle in small increments to align the foot with the contact surface.

When the robot has motion planning ability, which is described in the next chapter, the motion planner generates a feasible motion sequence and transmits it to the task level scheduler. After the motion sequence has been executed, the robot is able to travel from its initial configuration to its goal configuration, while avoiding the obstacles in the environment.

3.2 Hardware Design

This section describes the hardware implementation of a self-contained embedded robot control system based on a TI DSP chip — TMS320LF2407.

3.2.1 TMS320LF2407 DSP Overview

The TMS320LF2407 device [31, 32, 78] is a new member of the C2000 family of TI DSP controllers. It is targeted to meet the needs of control system applications. By integrating the high performance of a DSP core and the on-chip peripherals into a single-chip solution, the LF2407 device is a low-cost alternative to traditional microcontroller units (MCUs) and expensive multichip designs. This chip provides all the resources needed to build a self-contained embedded control system. The functional block diagram of the LF2407 DSP controller is shown in Figure 3.4.

The LF2407 DSP controller is based on the 16-bit, fixed-point, low-power C2xx

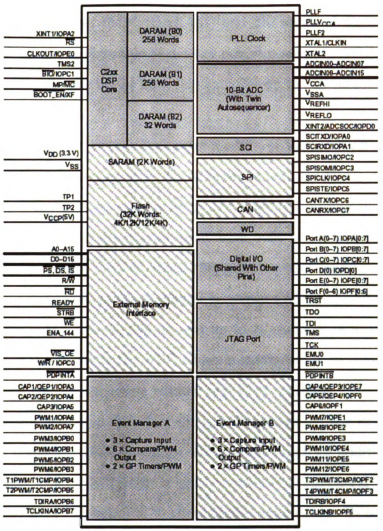

Figure 3.4: Functional block diagram of the LF2407 DSP controller

DSP core. The parallel architecture of the central processing unit (CPU) enables the DSP to perform in high speed at 30 million instructions per second (MIPS). Thus it allows the DSP to often process algorithm in real time rather than approximate results with look-up tables.

By integrating memory and peripherals onto a single chip, the DSP devices reduce system cost and save circuit board space. LF2407 DSP provides 32K words on-chip flash memory, which offers a reprogrammable solution useful for the initial prototyping of applications. The 2.5K words on-chip random access memory (RAM) includes

544 words of Dual-Access RAM (DARAM) and 2K words of Single-Access RAM (SARAM), which are configurable as data or program memory. With the help of an external memory interface the LF2407 chip can offer up to 64K program memory and 64K data memory space.

The LF2407 DSP has two event manager modules (EVA and EVB) that have been optimized for digital motor control. Capabilities of each module include two 16-bit general-purpose timers, three capture units, and eight 16-bit pulse-width modulation (PWM) channels. Two of the capture units have built-in quadrature encoder pulse (QEP) circuits, which are used to obtain the motor encoder readings easily.

The high performance analog-to-digital converter (ADC) has 10-bits resolution, a minimum conversion time of 500 ns, and up to 16 channels of analog input. The auto-sequencing capability of the ADC allows a maximum of 16 conversions to take place in a single conversion session without any CPU overhead.

A serial communications interface (SCI) is integrated on the device to provide asynchronous communication capability. For systems requiring additional communication interfaces, the LF2407 offers a synchronous serial peripheral interface (SPI) and a controller area network (CAN) communications module. To maximize device flexibility, functional pins are configurable as general purpose inputs/outputs (GPIO). Thus, up to 40 multiplexed digital I/O pins are available. JTAG (Joint Test Action Group) module provides non-intrusive real-time debugging capability, which is helpful to reduce the system development time.

The LF2407 DSP is based on low-power 3.3V CMOS technology and it integrates many power management features. It has three power down modes and the ability to power-down each peripheral module independently. These features make the LF2407 DSP a desirable device to be used in an embedded control system with power constraints.

3.2.2 *DSP Implementation of the Controller*

Due to the limitations of size, weight, and power consumption, one of the challenging tasks in designing the controller for the miniature robot is to use as few components as possible. Figure 3.5 illustrates the controller block diagram based on a TMS320LF2407 DSP chip.

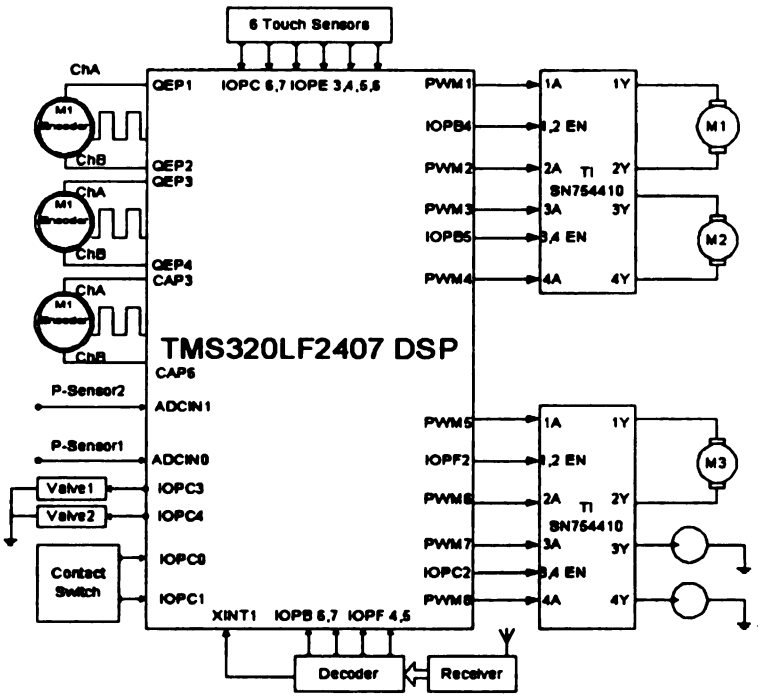

Figure 3.5: Block diagram of the DSP-based controller

Two quadruple half-H driver SN754410 from TI are used to drive the three servo motors and the two pump motors. The servo motors are driven by the PWM outputs PWM 1, 2, PWM 3, 4, and PWM 5, 6 of EVA via the H-bridge driver. Timer 1 is used as the time base to generate the PWM signal with a frequency of 20 KHz. Timer 3 is used to generate the servo-control sampling rate of 1 KHz. During each servo sampling period (1 ms), the compare registers, CMPR 1, CMPR 2, and CMPR 3 are updated for Motor 1, Motor 2, and Motor 3, respectively according to the calculated joint control value. The LF2407 DSP has two built-in quadrature encoder pulse (QEP) circuits. The encoder readings of servo Motor 1 and Motor 2 are easily obtained

using the QEP1/QEP2 and QEP3/QEP4 of the event manager modules with Timer 2 and Timer 4 as the time base, respectively. However, for Motor 3, an alternative method needs to be employed to get the encoder reading. The encoder channel A of servo Motor 3 is connected to capture unit pin 3 (CAP3), and the capture unit 3 is enabled to detect the rising edge of the encoder pulse. Channel B is connected to the CAP6/IOPF1 pin and this shared function pin is configured as digital input IOPF1. Whenever the capture interrupt is triggered in CAP 3, an interrupt service routine detects the digital level of the encoder channel B to determine the rotation direction and adjusts the encoder pulse count value. Apart from serving as the time base for PWM waveform, Timer 1 is also used as the time base for capture 3 operation. These two functions do not affect each other. A software solution for getting the encoder reading of Motor 3 using the capture unit is described in section 3.4.2.

The two pump motors require only binary switches to drive them. So the shared function pins PWM7/IOPE1 and PWM8/IOPE2 are configured as digital output pins. By using the H-bridge driver the pump can be turned on by setting the output high. The micro valves are also controlled by digital outputs from the DSP by using two transistors as drivers.

The touch sensors and contact switches produce high-level or low-level digital signals that are connected directly to the I/O pins of the DSP. The analog inputs from the pressure sensors are converted by the ADC to determine whether the suction foot is securely attached to a flat surface. A receiver module and a decoder chip are used for remote control operation. The impulse signal from the decoder triggers the external interrupt pin of the DSP. The interrupt service routine processes the four digital I/O outputs from the decoder and translates the remote control signal into proper commands.

It is clear that the resources of the DSP chip are efficiently utilized. As a result, the components count of the embedded control system is very small and the circuit

is very compact.

3.3 Control Strategy

It is convenient to control the CRAWLER robot by using joint level PD (proportional plus derivative) controllers, where each joint of the robot is treated as a servomechanism. However, this method is ineffective because the joint level controller doesn't take into consideration the dynamic effects of the whole robot links. Our control strategy is to combine a PD controller with a feedforward nonlinear compensation term to minimize the gravity loading effects on each joint. This control strategy can achieve better performance than the conventional PD control method because it not only achieves the joint level control but also compensates for the nonlinear gravitational effects which are determined by the configuration of the whole robot links.

3.3.1 Motor Dynamics

Each joint of the CRAWLER robot is driven by a DC servo motor. This section introduces the motor transfer function from which control algorithms will be obtained.

The equivalent circuit of a permanent magnet DC servo motor [18]is shown in Figure 3.6 based on the following variables:

- V_a Applied voltage
- L Armature inductance
- R Armature resistance
- V_b Back EMF (electromotive force)
- J_m Moment of inertia of the motor

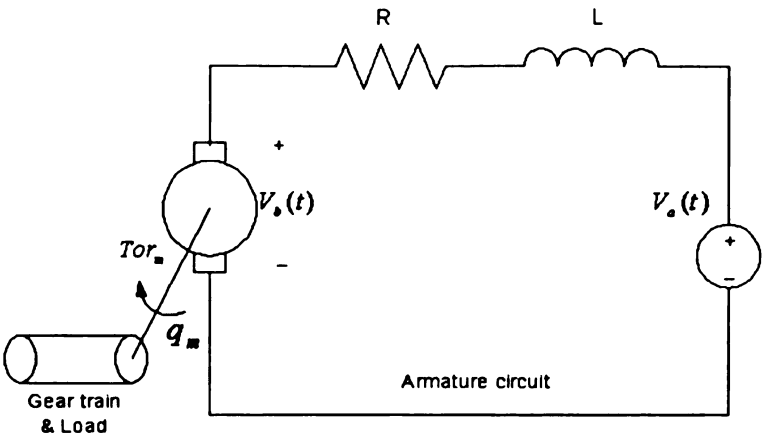

Figure 3.6: Equivalent circuit of DC servo motor

- f_m Viscous friction coefficient of the motor
- q_m Angular displacement of the motor shaft
- Tor_m Torque de lived by the motor
- K_b Back EMF constant
- K_a Torque constant of the motor

The following equations describe the motor characteristics.

$$\begin{aligned} Tor_m(t) &= K_a i_a(t) \\ V_b(t) &= K_b \dot{q}_m(t) \\ V_a(t) &= R i_a(t) + L\frac{di_a(t)}{dt} + V_b(t) \\ Tor_m(t) &= J_m \ddot{q}_m(t) + f_m \dot{q}_m(t) \end{aligned} \tag{3.1}$$

Taking the Laplace transform of these equations we have:

$$Tor_m(s) = K_a I_a(s) \tag{3.2}$$

$$I_a(s) = \frac{V_a(s) - sK_b q_m(s)}{R + sL} \tag{3.3}$$

$$Tor_m(s) = s^2 J_m q_m(s) + s f_m q_m(s) \tag{3.4}$$

By manipulating terms, we obtain the transfer function from the applied voltage to the angular displacement of the motor shaft

$$\frac{q_m(s)}{V_a(s)} = \frac{K_a}{s[s^2 J_m L + (L f_m + R J_m)s + R f_m + K_a K_b]} \tag{3.5}$$

Since the electrical time constant of the motor is much smaller than the mechanical time constant, we can neglect the armature inductance effect, L. Also assume that the friction is zero,*i.e.*, $f_m = 0$, the above equation is simplified as

$$\frac{q_m(s)}{V_a(s)} = \frac{Ka}{s(sRJ_m + K_a K_b)} = \frac{K}{s(T_m s + 1)} \tag{3.6}$$

where

$$K \triangleq 1/K_b$$

is called the motor gain constant, and

$$T_m \triangleq \frac{RJ_m}{K_a K_b}$$

the motor time constant.

Table 3.2 lists the actual parameters of the motors [47] used in the CRAWLER robot which are selected from API Motion company.

Table 3.2: Motor parameters: escap 17S78-208P from API-Portescap

Parameters	variable	unit	value
Back EMF constant	K_b	V/1000rpm (Vs/rad)	0.56 (0.0053)
Torque constant	K_a	mNm/A	5.3
Terminal resistance	R	ohm	6.9
Rotor inductance	L	mH	0.15
Rotor inertia	J_m	$kgm^2.10^{-7}$	0.50
Mechanical time constant	T_m	ms	12

3.3.2 Control Algorithm

When the motors are used to drive the robot joints, gear trains are installed to increase the motor load capability and to couple the motor shaft with the joint shaft. If the gear ratio $n = \frac{q_L}{q_m} < 1$, where q_L is the angular displacement of the joint shaft, then we have

$$\frac{q_L(s)}{V_a(s)} = \frac{nKa}{s(sRJ_m + K_aK_b)} \tag{3.7}$$

The purpose of the joint level controller is to servo the motor so that the actual angular displacement of the joint will track a desired angular displacement specified by a preplanned trajectory. The PD feedback controller is designed as follows

$$
\begin{aligned}
V_a(t) &= \frac{K_p[q^d{}_L(t) - q_L(t)] + K_v[\dot{q}^d{}_L(t) - \dot{q}_L(t)]}{n} \\
&= \frac{K_p e(t) + K_v \dot{e}(t)}{n}
\end{aligned} \tag{3.8}
$$

where K_p, K_v are the positional feedback gain and error derivative feedback gain, respectively. The gear ratio n is included to compute the applied voltage referred to the motor shaft. $q^d{}_L(t)$, $\dot{q}^d{}_L(t)$ are the desired position and velocity, respectively, which are precomputed from a trajectory planner.

Taking the Laplace transform of Equation 3.8 and substituting into Equation 3.7 yield the closed-loop transfer function relating the actual angular displacement to the desired angular displacement as:

$$
\frac{q_L(s)}{q^d{}_L(s)} = \frac{K_a K_v s + K_a K_p}{s^2 R J_m + s(K_a K_b + K_a K_v) + K_a K_p} \tag{3.9}
$$

Figure 3.7 shows the block diagram of this closed-loop PD control system.

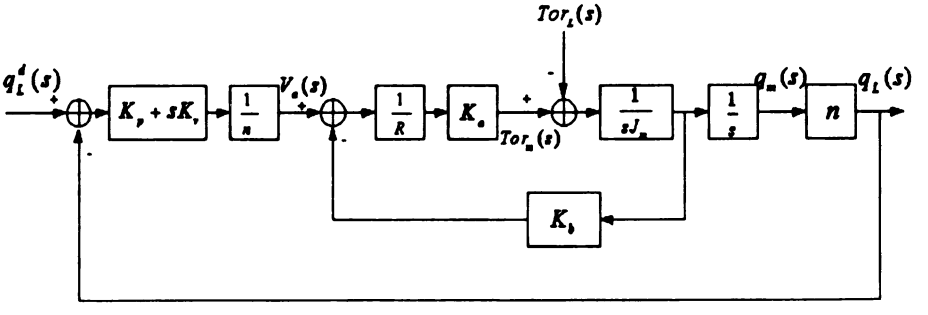

Figure 3.7: PD control block diagram

Notice that the torque generated at the motor shaft has to compensate for the load torque, thus from Equation 3.4 we have

$$Tor_m(s) = s^2 J_m q_m(s) + s f_m q_m(s) + Tor_L(s) \tag{3.10}$$

where $Tor_L(s)$ is the Laplace transform of the load torque referred to the motor shaft.

The transfer function relating the load torque to the actual joint displacement is given by

$$\frac{q_L(s)}{Tor_L(s)}|_{q^d{}_L(s)=0} = \frac{-nR}{s^2 R J_m + s(K_a K_b + K_a K_v) + K_a K_p} \tag{3.11}$$

Using the superposition principle and Equations 3.9 and 3.11, we can obtain the actual displacement of the joint from these two inputs as follows:

$$q_L(s) = \frac{(K_a K_v s + K_a K_p) q^d{}_L(s) - nR Tor_L(s)}{s^2 R J_m + s(K_a K_b + K_a K_v) + K_a K_p} \tag{3.12}$$

The robot dynamic model has been derived in section 2.4 and we restate it as below in Equation 3.13.

$$Tor(t) \;=\; D(q)\ddot{q}(t) + h(q, \dot{q}) + G(q) + f(\dot{q}) \tag{3.13}$$

It reveals that four dynamic terms contribute to the torques applied at robot joints. In practical implementation, the CRAWLER robot is only able to move at slow speeds. Therefore, the dynamic effects related to the joint acceleration and velocity are very small and neglectable. The dynamic effects other than the gravity torques can be assumed to be a very small constant value. Thus, Equation 3.13 is simplified as

$$Tor(t) = G(t) + Te \tag{3.14}$$

where $G(t) = G(q(t))$ and Te is a small constant.

Considering the gear train, the load torque applied at the shaft of the driving motor is

$$Tor_L(t) = nTor(t) = nG(t) + nTe \tag{3.15}$$

where $n < 1$ is the gear ratio.

The corresponding Laplace transform of Equation 3.15 is

$$Tor_L(s) = nG(s) + \frac{nTe}{s} \tag{3.16}$$

The gravity effects are determined by the configuration of all the robot links and plays a dominant role in robot dynamic model. In order to compensate for gravity effects, we can precompute these torque values based on the desired trajectory and feed the computed torques into the controller to minimize the effects as shown in Figure 3.8. This is called the PD+gravity compensation method.

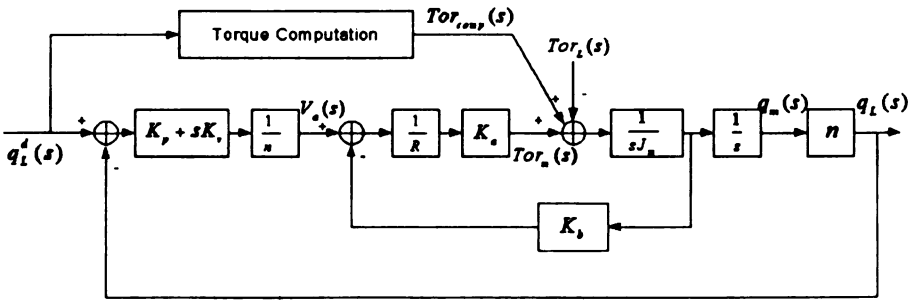

Figure 3.8: PD+gravity compensation control block diagram

The computation of Tor_{comp} depends on the dynamic model. Since the gravity is the dominant term in robot dynamics, Tor_{comp} only computes the gravity term as

$$Tor_{comp}(t) = n\hat{G}(t) \tag{3.17}$$

The element of $\hat{G}(t)$ is c_i. It represents the gravity effects exerted at joint i and can be calculated using Equations 2.59 to 2.70 in section 2.4.

Now we have the following proposition

Proposition 3.3.1 *When a motor rotates in a direction to overcome the gravity effects of its load, the motor control system using PD+gravity compensation method achieves a smaller steady-state error in response to a step input than the system using the conventional PD control method.*

Proof. We prove this proposition by comparing the steady-state errors of the conventional PD controller and our proposed PD+gravity compensation controller.

Using Equation 3.12, the system error $e(t) = {q^d}_L(t) - q_L(t)$ in the Laplace transform domain can be expressed as

$$\begin{aligned} E(s) &= {q^d}_L(s) - q_L(s) \\ &= \frac{(s^2 J_m R + sK_a K_b)q_L^d(s) + nRTor_L(s)}{s^2 RJ_m + s(K_a K_b + K_a K_v) + K_a K_p} \end{aligned} \tag{3.18}$$

For a step input of magnitude A, ${q^d}_L(t) = A$, its Laplace transform is ${q^d}_L(s) = A/s$. Using the final value theorem, the steady-state error of the PD control system due to the step input can be expressed as

$$
\begin{aligned}
e_{ss} &= \lim_{t\to\infty} e(t) = \lim_{s\to 0} sE(s) \qquad (3.19)\\
&= s\lim_{s\to 0} \frac{(s^2 J_m R + sK_a K_b)A/s + nRTor_L(s)}{s^2 RJ_m + s(K_a K_b + K_a K_v) + K_a K_p}\\
&= s\lim_{s\to 0} \frac{nRTor_L(s)}{s^2 RJ_m + s(K_a K_b + K_a K_v) + K_a K_p}\\
&= \lim_{s\to 0} \frac{n^2 R[sG(s) + T_e]}{s^2 RJ_m + s(K_a K_b + K_a K_v) + K_a K_p}\\
&= \lim_{s\to 0} \frac{sn^2 RG(s)}{s^2 RJ_m + s(K_a K_b + K_a K_v) + K_a K_p} + \frac{n^2 RT_e}{K_a K_p}
\end{aligned}
$$

By investigating the gravity terms in Equations 2.59 to 2.70, and their corresponding Laplace transfer functions, we know that the term

$$\lim_{s\to 0} \frac{sn^2 RG(s)}{s^2 RJ_m + s(K_a K_b + K_a K_v) + K_a K_p}$$

is non-zero. Therefore, the steady-state error of the PD control system is a function of the gravity effects. In the case when the motor rotates to overcome the gravity force, this term is positive resulting in large steady-state errors.

On the other hand, the error equation of the control system using PD+gravity compensation method is

$$
\begin{aligned}
E(s) &= q^d{}_L(s) - q_L(s) \qquad (3.20)\\
&= \frac{(s^2 J_m R + sK_a K_b)q_L^d(s) + nR[Tor_L(s) - Tor_{comp}(s)]}{s^2 RJ_m + s(K_a K_b + K_a K_v) + K_a K_p}
\end{aligned}
$$

The steady-state position error corresponding to the step input becomes

$$\begin{aligned} e_{ss} &= s\lim_{s\to 0} \frac{nR[Tor_L(s) - Tor_{comp}(s)]}{s^2RJ_m + s(K_aK_b + K_aK_v) + K_aK_p} \\ &= \lim_{s\to 0} \frac{snR[nG(s) + nT_e/s - n\hat{G}(s)]}{s^2RJ_m + s(K_aK_b + K_aK_v) + K_aK_p} \end{aligned} \tag{3.21}$$

If the computed torque Tor_{comp} is equivalent to the gravity loading of the links, i.e., $\hat{G}(s) = G(s)$, then the steady-state position error reduces to

$$e_{ss} = \frac{n^2RT_e}{K_aK_p} \tag{3.22}$$

which is small because T_e is assumed to be very small and other parameters have bounded positive values. □

In real digital control implementation, the computed torque Tor_{comp} is converted to the applied voltage through a voltage-torque characteristic coefficient K_{vt}. A typical voltage-torque curve can be found in [18]. The coefficient K_{vt} is obtained from experiments. Then the PD+gravity compensation method is expressed as

$$V_a(t) = \frac{K_pe(t) + K_v\dot{e}(t)}{n} + \frac{Tor_{comp}}{K_{vt}} \tag{3.23}$$

3.4 Software Development

3.4.1 Software Modules

Software modules have been developed in C and TI DSP assembly language to control the climbing robots. They work well for both the FLIPPER and the CRAWLER robots. The main software modules running on the embedded controller are:

- Task level scheduler
- Trajectory planner
- Joint level servo controller
- Remote command interpreter
- Communication module

Task level scheduler detects the motion status and decomposes the task level commands into several motion steps. A finite state machine is developed to keep track of the robot motion status, such as the switching between different motion modes, the changing of standing foot, and the states of the suction pumps.

Trajectory planner activates different kinematic models according to different motion modes and solves the inverse kinematics to plan the joint level trajectory. The interpolation is conducted to generate desired joint angles.

Joint level servo controller implements the PD+gravity compensation control algorithm to drive the motors to desired angles.

Command interpreter is used to decode the commands sent by a human operator through a remote transmitter. The receiver chip on the embedded controller board triggers an external interrupt whenever a button in the remote controller panel is pushed. The interrupt service routine processes four digital I/O outputs from the receiver and translates the remote control signal into proper commands.

Communication module handles the RS-232 serial communication between embedded controller board and a host computer. A command interface is implemented to facilitate the testing of the robot control system.

3.4.2 Encoder Feedback

Accurate digital motor control is the basic requirement for the robot to accomplish certain tasks. To achieve this, the servo controller requires position information from the encoder as feedback. As mentioned before, it is easy to obtain encoder reading for motor 1 and motor 2 using the two built-in QEP circuit of DSP. However, for motor 3, software solution is adopted to get the encoder reading because adding additional circuits is not desirable for a miniature robot limited in size and weight.

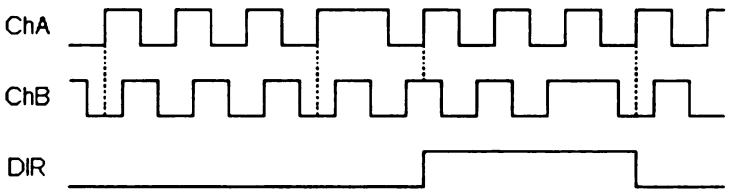

Figure 3.9: Encoder pulse and motor direction

The analysis of the magnetic encoder waveform of the servomotor shown in Figure 3.9 indicates that:

1. The phase shift of the encoder pulse sequences between two channels (A and B) is 90 degrees.

2. The logic levels of channel B corresponding to the rising edges of channel A alternate when the motor changes direction.

These properties are utilized to obtain the encoder reading. The encoder channel A of servo Motor 3 is connected to CAP3 pin and the capture unit 3 is enabled to detect the rising edge of the encoder pulse. Channel B is connected to input digital I/O pin IOPF1. Whenever a rising edge is detected, a capture interrupt is triggered. In the interrupt service routine, the logical level of channel B is detected through IOPF1. When channel B is logic low, it means that the motor rotates in a positive direction, then the encoder pulse count variable CAP3_cnt increments by one. When

channel B is logic high, it means the motor has changed the rotation direction, then CAP3_cnt decrements by one. In such a way, CAP3_cnt updates its value by +1 or -1 according to the rotation direction every time the encoder pulse triggers. Thus, the value of CAP3_cnt indicates the motor angle expressed in encoder counts. CAP3_cnt is set as a global variable and its value can be accessed by servo control routine to accomplish feedback control.

3.4.3 Servo Control Module

Servo control module is the fundamental and most important part for a robot control system. A specific interrupt service routine (ISR) processes the servo control and motion profile calculation. Timer 3 is selected to generate servo control sampling period of 1 ms by setting the associated period register, T3PR, with proper value. Figure 3.10 shows the flowchart of the servo ISR.

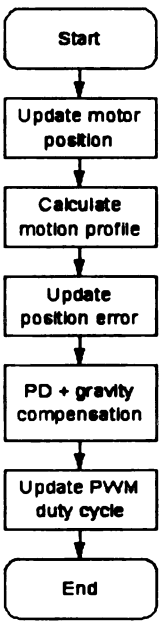

Figure 3.10: Servo control interrupt service routine flowchart

For smooth motion control, a motion profile is necessary to control the motor

acceleration and deceleration. When a motor is commanded to rotate a certain angle, the desired position for each servo step must be calculated by the motion profile routine. Without the motion profile, motion will be abrupt, causing excessive wear on the mechanical components and degrading the performance of the control algorithm. For our application, a linear piecewise trapezoidal/triangular velocity trajectory (Figure 3.11) is adopted to implement the motion profile. It consists of acceleration, constant motion, and deceleration sections for the motor rotating a large angle. When the motor rotates a small angle, it only consists of acceleration and deceleration motion.

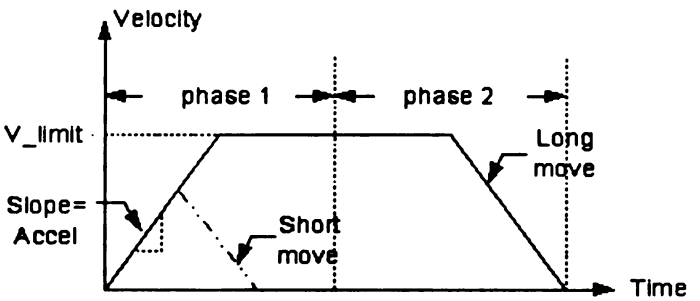

Figure 3.11: Velocity trajectory of motion profile

The flowchart for implementing the motion profile is shown in Figure 3.12. The value of the commanded move distance is divided by two and placed in variable ph1_dist. The motion destination (or final desired position) is the initial position of the motor plus the commanded move distance. The desired distance, desired velocity and desired position for each servo step is placed in variable, d_dist, d_vel and d_pos respectively. The flat_cnt variable is used to determine when the constant velocity section starts or stops. The first_half is the flag indicating whether the motion is in phase 1 or phase 2. When the desired position reaches the motion destination, the move_in_progress flag is cleared ignoring further motion profile calculation until the move is completed and this flag is set. The output of the motion profile routine is the desired position for each servo step. Those desired positions guide the motor rotating from acceleration stage through constant motion stage to deceleration stage

until settling down at the destination position.

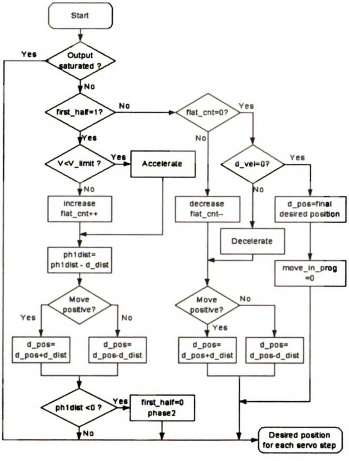

Figure 3.12: Motion profile flowchart

3.5 Experiments

The CRAWLER robot and its embedded control system based on TMS320LF2407 DSP chip has been successfully designed and implemented. Experiments were conducted to evaluate the performance of the robot.

Figure 3.13 shows a sequence of snapshots of the CRAWLER moving one step forward on a wall surface.

Figure 3.14 shows the CRAWLER robot walking around a corner. Figure 3.14 (1-4) show that the CRAWLER makes a 45° left turn. After the CRAWLER moves

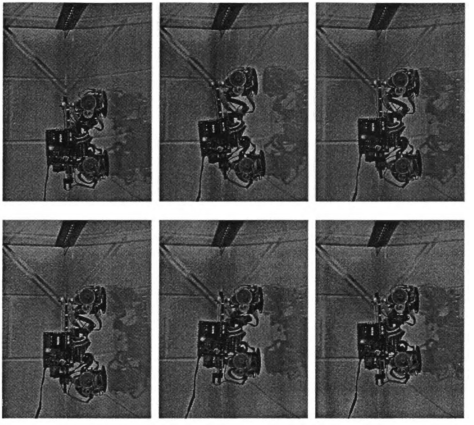

Figure 3.13: The CRAWLER robot moving forward on a wall

forward by two steps (5-8), it makes a second 45° left turn (9-10), and then continues moving forward (11-12) along the new direction. The 4th snapshot shows that the robot unwinds the rear foot after a turn to make the robot ready to move in translation mode.

Figure 3.15 shows that the robot is capable of moving between surfaces.

3.6 Summary

This chapter describes the embedded control system design of the climbing robot. By using TI DSP, the number of electrical components is minimized. A efficient circuit board is designed, and installed on the robot body, which makes the control system self-contained and tetherless. A joint level PD+gravity compensation method

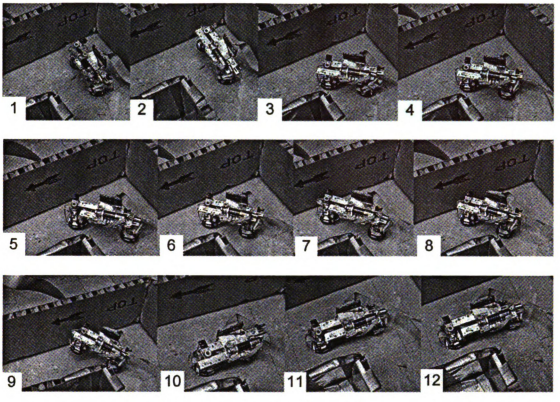

Figure 3.14: The CRAWLER robot making turns around a corner

is proposed. This control method outperforms the conventional PD controller because it not only achieves the joint level control but also compensates for the gravity effects which is determined by the configuration of all the robot links. Software modules of the control system are also presented in this chapter. Experimental results are presented with snapshots showing the capabilities of the robot.

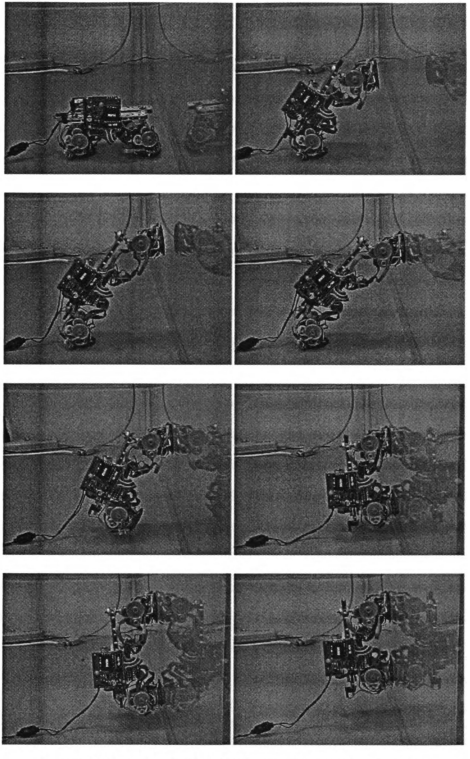

Figure 3.15: The CRAWLER robot moving between surfaces

CHAPTER 4

MOTION PLANNING OF THE CRAWLER ROBOT

With the installation of the robot controller, the climbing robot now can be tele-operated by human operators. However, in order to make the robot smart, we need to provide the robot with motion planning ability. A robot with motion planning ability is capable of determining autonomously a motion path to accomplish a particular task without collision with obstacles. This automatic determination of a path relieves the user from the burden of specifying tediously detailed instructions for the robot. Therefore, the user can specify tasks more declaratively, by stating *what* he wants done rather than *how* to do it.

There are two different situations when the robot planes its motion. In the first situation, the robot has exact knowledge of the environmental settings and the location and shape of obstacles. The purpose of the motion planing in this situation is to find a collision-free path connecting the initial and goal position. It is called the basic motion planning. In the second situation, the robot has no knowledge about the environment. The robot uses its own sensors to detect and navigate through the environment. It is called sensor-based motion planning or robot navigation. In this chapter, we only study the basic motion planning problem by assuming that the environment is known.

The goal of basic motion planning is to compute a motion path, which brings a robot from an initial configuration to a goal configuration while avoiding obstacles in the robot's environment. However, finding a motion path is not enough for the CRAWLER robot. Motion planner of the CRAWLER robot must be able to generate detailed motion sequences to guide the robot travel through the path. Due to the under-actuated mechanism and mechanical constraints, the motion planning of the CRAWLER robot is extremely difficult. The robot can perform only restricted motions even in the absence of obstacles. Since the standing foot alternates when

the robot moves, the motion planner must be able to schedule correct foot placement for each motion step. For the CRAWLER robot, motor 2 drives the robot forward/backward and makes turn. Since only one control input is available, the translation motion and rotation motion are highly coupled. This makes the motion planning of the CRAWLER robot more complicated than that of the car-like robot, where two separate control inputs are available to control the translation motion and rotation motion, respectively. The switching between different motion modes further complicated the motion planning of the CRAWLER robot.

In this chapter, motion planning problems and their solutions of our bipedal CRAWLER robot are presented. To deal with the constraints imposed by the kinematic structure, the configuration space is broadened by introducing the robot motion status to form a **hybrid configuration space**. The discrete motion status — standing foot and motion mode, is used to decompose the continuous configuration space into several submanifolds. Based on the study of the robot motion pattern, a motion planning method which consists of a global planner and a local planner is developed. The motion status information enables us to define a cost function which is used in the motion planning algorithm to generate a feasible path and an optimal motion sequence. Simulation and experimental results have verified the theoretical development.

4.1 Preliminaries

The primary goal of this section is to provide the necessary background for investigating the robot motion planning problem. Some preliminary mathematical concepts are introduced first, and then the exact cell decomposition motion planning method and the A^* searching algorithm are described which will be used in our own motion planning approach in following sections.

4.1.1 Configuration Space

The motion planning problem has received a lot of attention over the years. The configuration space concept proposed by Lozano-Perez [59, 60, 58] and J.C. Latombe [39], etc, served as a powerful representational tool for both the development and the analysis of motion planning algorithms. Robot position and orientation are characterized by just one point in configuration space. With the configuration space framework, it becomes possible to develop algorithms that are generalizable and adaptable to a wide variety of applications.

Definition 4.1.1 (Topological Space) *[48]: A* **topology** *on a set X is a collection $\mathcal{T}$ of subsets of X having the following properties:*

1. *Both X and the empty set $\emptyset$ are in $\mathcal{T}$.*
2. *The union of elements of any subcollection of $\mathcal{T}$ is in $\mathcal{T}$.*
3. *The intersection of the elements of any finite subcollection of $\mathcal{T}$ is in $\mathcal{T}$.*

A set X for which a topology $\mathcal{T}$ has been specified is called a **topological space.**

If X is a topological space with topology $\mathcal{T}$, we say that a subset U of X is an **open set** of X if U belongs to the collection $\mathcal{T}$.

Definition 4.1.2 (Homeomorphism) *[39]: Let $f : X \rightarrow Y$ be a bijective function between topological spaces X and Y. Since f is bijective, the inverse f^{-1} exists. If both f and f^{-1} are continuous, then f is a* **homeomorphism.**

Definition 4.1.3 (Manifolds) *[48]: A topological space M is a* **manifold** *if for every $x \in M$, an open set $O \subset M$ exists such that:*

1. *$x \in O$.*
2. *O is (locally) homeomorphic to R^n.*

3. n is fixed for all $x \in M$.

The dimension n is referred to as the dimension of the manifold, M.

If X and Y are topological spaces, the Cartesian product $X \times Y = \{(x, y) : x \in X \text{and } y \in Y\}$ defines a new topological space.

As defined by Latombe [39], the configuration space concept is presented as follows. Let $\mathcal{R}$ be a rigid object — the robot — moving in a Euclidean space $\mathcal{W}$, called *workspace*, represented as R^N, with N=2 or 3. $\mathcal{W}$ is equipped with a fixed Cartesian coordinate frame, denoted by $\mathcal{F}_W$. A moving reference frame, $\mathcal{F}_R$, is attached to $\mathcal{R}$.

Definition 4.1.4 (Configuration Space) *[39]: A configuration* $\mathbf{q}$ *of* $\mathcal{R}$ *is a specification of the position and orientation of* $\mathcal{F}_R$ *with respect to* $\mathcal{F}_W$. *The* **configuration space** *of* $\mathcal{R}$ *is the space* C *of all the possible configurations of* $\mathcal{R}$.

The configuration space C is intrinsically independent of the choice of the frames $\mathcal{F}_W$ and $\mathcal{F}_R$. Only the representation of C depends on these frames. A configuration can be described by a list of real parameters. For 2-D rigid objects, where N=2, a configuration $\mathbf{q}$ of $\mathcal{R}$ can be bijectively represented by three parameters, x, y and θ, where (x, y) determines the position of the origin of $\mathcal{F}_R$ in $\mathcal{F}_W$ and $\theta \in [0, 2\pi)$ is the rotation parameter. For 3-D rigid objects, a configuration can be represented by position parameters (x, y, z) and 3 Euler angles ϕ, θ, and ψ, which determine the orientation of $\mathcal{F}_R$'s axes with respect to $\mathcal{F}_W$.

The subset of $\mathcal{W}$ occupied by $\mathcal{R}$ at configuration $\mathbf{q}$ is denoted by $\mathcal{R}(\mathbf{q})$. A point a of $\mathcal{R}$ is denoted by $a(q)$ in $\mathcal{W}$ when $\mathcal{R}$ is at configuration $\mathbf{q}$. Thus, for any two configurations $\mathbf{q}$ and $\mathbf{q}$', $a(q)$ and $a(q')$ are the same point in $\mathcal{R}$, but in general do not coincide in $\mathcal{W}$.

By defining the topology in terms of a metric, the configuration space is a topological space generated by the set of all possible configurations. The **robot displace-**

ment metric is defined as:

$$d(q, q') = \max_{a \in R} \| a(q) - a(q') \|$$

where $\| x - x' \|$ denotes the Euclidean distance between any two points x and x' in R^N. This metric represents the distance between two configurations **q** and **q'** and it tends toward zero when the regions $\mathcal{R}(q)$ and $\mathcal{R}$(q') tend to coincide, and conversely.

Definition 4.1.5 (Path in configuration space $\mathcal{C}$) *[39]: A* **path** *of the robot $\mathcal{R}$ from the initial configuration $\mathbf{q}_{init}$ to the goal configuration $\mathbf{q}_{goal}$ is a continuous map $\tau : [0, 1] \longrightarrow \mathcal{C}$ with $\tau(0) = \mathbf{q}_{init}$ and $\tau(1) = \mathbf{q}_{goal}$.*

The continuity of τ entails that for all $s_0 \in [0, 1]$, $\lim_{s \to s_0} max_{r \in \mathcal{R}} \| r(\tau(s)) - r(\tau(s_0)) \| = 0$, with s taking its values in [0,1].

If no kinematic or dynamic constraints limit the motion of $\mathcal{R}$, we say that $\mathcal{R}$ is a **free-flying object**.

Definition 4.1.6 (Minkowski Operator) *[39]: Let X be an affine space whose origin is O, and A and B be any two subsets of X. The Minkowski operator $\oplus$ (addition) and $\ominus$ (subtraction) are defined by:*

$$\begin{aligned} A \oplus B &= \{x | x = a + b, a \in A, b \in B\}, \\ \ominus B &= \{-b | b \in B\}, \\ A \ominus B &= A \oplus (\ominus B). \end{aligned}$$

Let obstacles $\mathcal{B}_1, \cdots \mathcal{B}_p$ be fixed rigid objects distributed in $\mathcal{W}$. Every obstacle $\mathcal{B}_i$, $i = 1$ to p, in the workspace $\mathcal{W}$ maps in configuration space $\mathcal{C}$ to a region:

$$\mathcal{CB}_i = \{\mathrm{q} \in \mathcal{C} | \mathcal{R}(\mathrm{q}) \cap \mathcal{B}_i \neq \emptyset\}$$

which is called a $\mathcal{C}$-**obstacle**. The union of all the $\mathcal{C}$-obstacles:

$$\cup_{i=1}^{p} \mathcal{CB}_i$$

is called the $\mathcal{C}$-**obstacle region**, and the set:

$$\mathcal{C}_{free} = \mathcal{C} \setminus \cup_{i=1}^{p} \mathcal{CB}_i = \{q \in \mathcal{C} | \mathcal{R}(q) \cap (\cup_{i=1}^{p} \mathcal{B}_i) = \emptyset\}$$

is called the **free space**. Any configuration in $\mathcal{C}_{free}$ is called a **free configuration**.

Assume that both the geometry of $\mathcal{R}$ and $\mathcal{B}_1 \cdots \mathcal{B}_p$ and the location of the $\mathcal{B}_i$'s in $\mathcal{W}$ are accurately known. The **basic motion planning problem** is: Given an initial and goal configuration q_{init}, q_{goal}, generate a free path τ between the two configurations, specifying a continuous sequence of position and orientations of $\mathcal{R}$ avoiding contact with the $\mathcal{B}_i$'s if they belong to the same connected component of $\mathcal{C}_{free}$, and to report failure otherwise.

For a *point robot*, the workspace and configuration space are the same. The $\mathcal{C}$-obstacles are identical to the obstacles in the workspace. When robot $\mathcal{R}$ is a planar object allowed to translate freely without rotation, the configuration space is R^2, but the $\mathcal{C}$-obstacles are obtained by "growing" the obstacles by the shape of the robot as illustrated in Figure 4.1, which corresponding to the Minkowski sum of robot and obstacle.

When the robot can translate and rotate in a planar work space, the configuration space is 3-dimensional, with $(x, y, \theta) \in R^2 \times [0, 360^o)$. The $\mathcal{C}$-obstacle looks like a twisted pillar as shown in Figure 4.2. Now imagine sweeping a horizontal plane upwards through the θ direction of configuration space. Each slice is a Minkowski sum whose shape changes continuously from $\theta = 0$ to $\theta = 360^o$. The stack of slices constitutes the $\mathcal{C}$-obstacle.

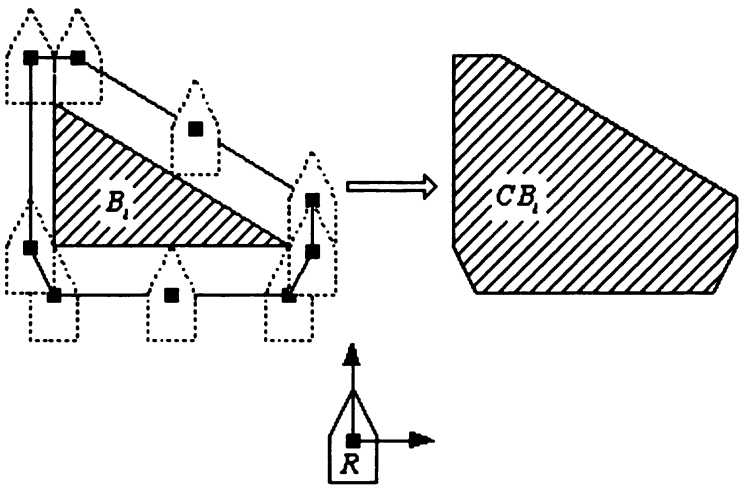

Figure 4.1: $\mathcal{C}$-obstacle in a translational case; $\mathcal{B}_i$ is an obstacle, by superimpose the shape of a robot ($\mathcal{R}$) all through the peripheral of the obstacle, the $\mathcal{C}$-obstacle $\mathcal{CB}_i$ is constructed.

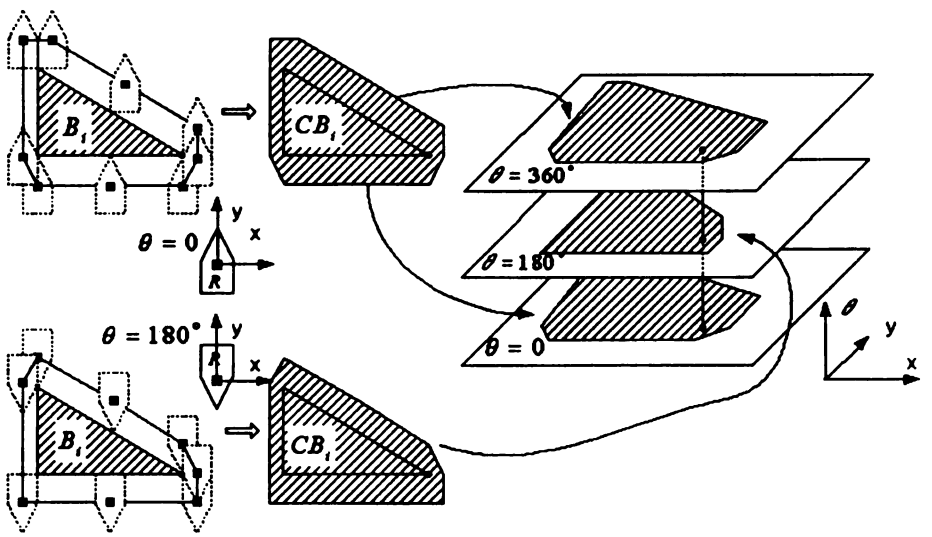

Figure 4.2: $\mathcal{C}$-obstacle in translational and rotational case. $\mathcal{B}_i$ is an obstacle, $\mathcal{CB}_i$ is a slice corresponding to the Minkowski sum of the obstacle and the robot $\mathcal{R}$ whose shape rotates continuously from $\theta = 0$ to $\theta = 360°$. The $\mathcal{C}$-obstacle is a 3-D stack of the slices.

Now consider a more complex articulated manipulator robot. This robot $\mathcal{R}$ is made of k moving rigid objects $\mathcal{R}_1, \cdots, \mathcal{R}_k$ connected by mechanical joints (revolute or prismatic joints) which constrain their relative motion. Let $\mathcal{F}_{R_i}$ be the frame attached to $\mathcal{R}_i, i \in [1, k]$. The configuration of $\mathcal{R}$ is a specification of the position and orientation of every frame $\mathcal{F}_{R_i}$, $i = 1$ to k, with respect to $\mathcal{F}_W$. Therefore, the configuration space of a fixed-base articulated robot with k revolute and prismatic joints and no kinematic loop is a k-dimensional smooth manifold.

4.1.2 Exact Cell Decomposition

There are three general approaches to the basic motion planning problem [39]: cell decomposition methods, roadmap methods, and artificial potential field methods. In this section, an exact cell decomposition method is presented in the simple case where $\mathcal{C} = R^2$ and the C-obstacle region forms a polygonal region in $\mathcal{C}$.

The decomposition of $\mathcal{C}_{free}$ and the associated connectivity graph are defined as follows:

Definition 4.1.7 convex polygonal decomposition*[39]: A convex polygonal decomposition $\mathcal{K}$ of $\mathcal{C}_{free}$ is a finite collection of convex polygons, called* **cells**, *such that the interiors of any two cells do not intersect and the union of all the cells is equal to the closure of $\mathcal{C}_{free}$. Two cells k and k' in $\mathcal{K}$ are* **adjacent** *if and only if $k \cap k'$ is a line segment of non-zero length.*

Definition 4.1.8 connectivity graph *[39]: The connectivity graph $\mathcal{G}$ associated with a convex polygonal decomposition $\mathcal{K}$ of $\mathcal{C}_{free}$ is the non-directed graph specified as follows: 1) $\mathcal{G}$'s nodes are the cells in $\mathcal{K}$. 2) Two nodes is $\mathcal{G}$ are connected by a link if and only if the corresponding cells are adjacent.*

The exact cell decomposition algorithm for planning a free path connecting $\mathbf{q}_{init}$ and $\mathbf{q}_{goal}$ is the following:

1. Generate a convex polygonal decomposition $\mathcal{K}$ of $\mathcal{C}_{free}$.

2. Construct the connectivity graph $\mathcal{G}$ associated with $\mathcal{K}$.

3. Search $\mathcal{G}$ for a sequence of adjacent cells between $\mathbf{q}_{init}$ and $\mathbf{q}_{goal}$.

4. If the search terminates successfully, return the generated sequence of cells; otherwise, return failure.

The output of the algorithm is a sequence $k_1, \cdots, k_p$ of cells called a *channel*, such that $\mathbf{q}_{init} \in k_1$, $\mathbf{q}_{goal} \in k_p$ and for every $j \in [1, p-1]$, k_j and k_{j+1} are adjacent. A continuous free path can be computed from the channel, for example by connecting the initial configuration to the goal configuration through the midpoints of the intersections of every two successive cells.

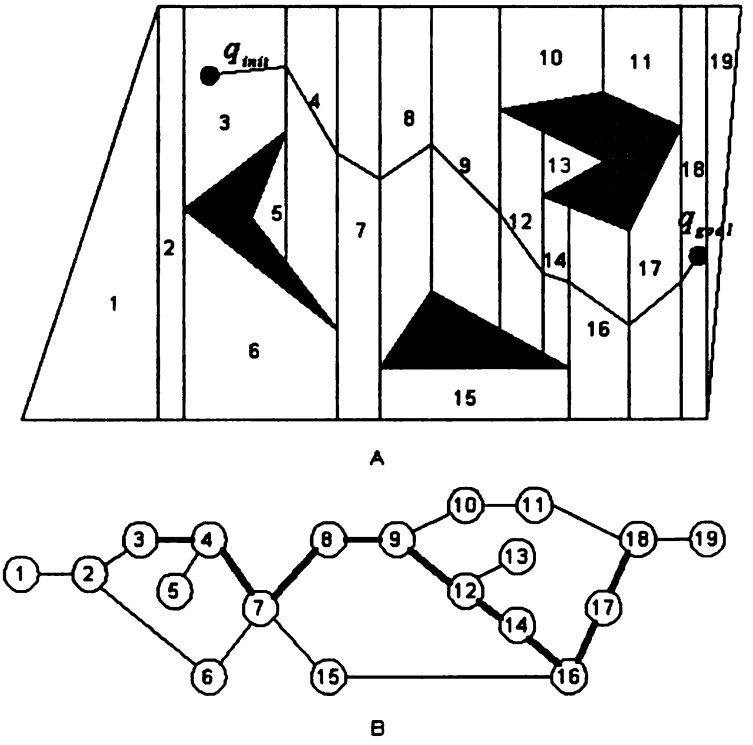

Figure 4.3: Trapezoidal decomposition method

An efficient exact cell decomposition method called **trapezoidal decomposition** is illustrated in Figure 4.3. The free space is externally bounded by a polygon and

internally bounded by three polygonal obstacles. The first step of the method consists of exactly decomposing the free space into trapezoidal and triangular cells. These cells are built by drawing vertical rays from the C-obstacle vertices. Two cells are adjacent if they share a portion of an edge of non-zero length. The second step is to construct the connectivity graph representing the adjacency relation between the cells and search this graph for a path (thick lines in Figure 4.3 B). A path in the connectivity graph determines a channel in free space. The channel is the sequence of cells labeled as 3, 4, 7, 8, 9, 12, 14, 16, 17 and 18. The third step of the method consists of transforming this channel into a free path, by connecting the initial configuration to the goal configuration through the midpoints of the intersections of every two successive cells.

4.1.3 A Algorithm*

In many cases, motion planning involves searching a graph to find a minimum-cost path connecting the initial node and the goal node. A^* Algorithm [25, 39, 33]is the most popular method for graph searching, guaranteed to return a path of minimum cost whenever the path exists.

A^* explores a weighted graph G iteratively by following paths originating at N_{init}. The cost of a path is defined as the sum of the costs of its arcs. A^* employs an additive evaluation function $f(N) = g(N) + h(N)$, where $g(N)$ is the cost of the currently evaluated path from N_{init} to N, and $h(N)$ is a heuristic estimate of the cost of the path remaining between N and goal node N_{goal}. A^* constructs a tree T of selected paths of G using the elementary operation of **node-expansion**, i.e., generating all successors of a given node. T is represented by associating to each visited node N a pointer to its parent node in the current T. Starting with N_{init}, A^* selects for expansion the leaf node of T that has the lowest f value, and only maintains the lowest f path to any given node. It is known that if $f(N)$ is a lower bound to the

cost of any continuation path from N to N_{goal}, then A^* is **admissible**, that is, it is guaranteed to find the optimal path. A^* algorithm is described as follows:

1. Put the start node, N_{init}, on a list called OPEN of unexpanded nodes.
2. If OPEN is empty, exit with failure; no solution exists.
3. Remove from OPEN a node, N, at which f(N)=g(N)+h(N) is minimum (break ties arbitrarily, but in favor of a goal node) and place it on a list called CLOSED to be used for expanded nodes.
4. If N is a goal node, exit successfully with the solution obtained by tracing back the path along the pointers from N to N_{init}, (pointers are assigned in steps 5 and 6).
5. Expand node N, generating all its successors with pointers back to N.
6. For every successor N' of N:
   - Calculate $f(N')$
   - If N' was neither in OPEN nor in CLOSED, then add it to OPEN. Assign the newly computed $f(N')$ to node N'.
   - If N' already resided in OPEN or CLOSED, compare the newly computed $f(N')$ with that previously assigned to N'. If the new value is lower, substitute it for the old (N' now points back to N instead of to its predecessor). If the matching node N' resided in CLOSED, move it back to OPEN.
7. GO to (2).

4.2 Motion Planning Analysis

For a given robot and set of obstacles, the basic motion planning techniques under configuration space framework only consider geometric issues to determine a collision-

free path. For more complicated problems, it is beneficial to broaden configuration space framework to handle sensory information, kinematic and dynamic constraints, etc. Sharma and Sutanto [67] proposed a vision-based motion planning framework in which they unified the configuration space and an image feature space and considered sensors as an integral part of the motion goal. In [39], the dynamic motion planning problem — where the obstacles in the workspace are moving, was studied in **configuration-time space**. The approach is to add a dimension — time — to the robot's configuration space and take the specificity of the time dimension into account. LaValle [42, 40] introduced a game-theory-based mathematical foundation upon which analyzes and algorithms can be developed for a broad class of motion planning problems. He advocated the expansion from **configuration space** to **state space** and to broader space — **information space** to cover general motion strategy problems, involving uncertainty in sensing and control, environment uncertainties, and the coordination of multiple robots. The framework is general enough to encompass every motion planning issue; however, more work needs to be done to tailor the idea to specific applications.

In this section, "hybrid configuration space" concept is proposed to incorporate the continuous configuration space with discrete motion status space. The broadened space is a more effective tool to analyze the motion planning of the climbing robot.

4.2.1 Motion Status Space

The robot motion status has two fields as shown in Table 4.1. It represents precisely which foot is supporting the robot and in what motion mode the robot is moving.

Table 4.1: Motion status fields

Standing Foot	Motion Mode

Notice that there exist only two choices for the first field, either LFS or RFS. Neither foot grips the floor is an invalid motion status, because in this case the robot cannot climb a wall and the motion direction is undetermined. Both feet grip the floor is also a invalid status because the robot cannot move except to release one of the foot. We assign three values $(-1, 0, +1)$ to represent each of the motion modes, *i.e.* 0 represents translation mode, -1 represents spin mode 1, and $+1$ represents spin mode 2. The combination of the fields makes six motion statuses, namely LFS-1, LFS0, LFS+1, RFS-1, RFS0 and RFS+1. For example, LFS-1 means the left foot supports the robot while the right root is free to move, and the robot is under spin-1 mode. The motion status space is defined as follows:

Definition 4.2.1 (Motion Status Space) *Let's define a motion status set S whose elements are the six motion statuses, i.e. S =\{LFS-1, LFS0, LFS+1, RFS-1, RFS0, RFS+1\}. The collection of all subsets of S is a topology on S and is called the discrete topology. We say the topological space S the* **motion status space.**

4.2.2 Motion Pattern Analysis

When the CRAWLER robot walks on a 2-D plane, motor 2 controls the robot to move forward/backward or make turns. Motors 1 and 3, which control the robot tilt angles α and β, only help the robot to lift the free foot up from or down to the contact surface to reduce the friction. For the ease of explanation, we assume that the friction between the free foot and the contact surface is zero when the foot slide along the surface. Thus motors 1 and 3 can be ignored when studying the motion planning on a 2-D plane, and motor 2 is the only driving motor. Throughout the paper, foot 1 is assumed to be the left foot while foot 2 is the right foot.

Fixed-base Case

We first study the motion when the robot is standing on one of its feet. Since the base is fixed at the center of the standing foot, the configuration space of the robot is one-dimensional, represented by the driving angle of motor 2, ψ. The motion pattern of the robot and the mapping between the configuration space and robot workspace in LFS phase and RFS phase is shown in Figure 4.4 and Figure 4.5, respectively.

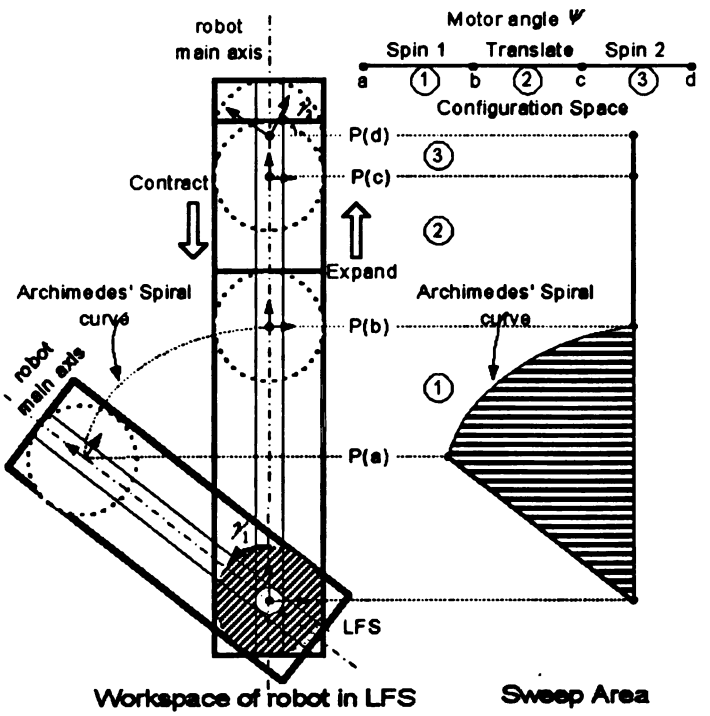

Figure 4.4: Motion pattern of the CRAWLER robot in LFS phase

In Figure 4.4, the shaded circle represents the left foot which supports the robot, while the dotted circles represent the right foot which moves freely. In the one-dimensional configuration space, $\psi = a$ represents the motor angle corresponding to the mechanical extreme when the robot contracts; $\psi = d$ represents the motor angle corresponding to the mechanical extreme when the robot expands; $\psi = b$ and $\psi = c$ are the switching points from the translation mode to the spin-1 mode and spin-2 mode respectively. $P(a), P(b), P(c)$, and $P(d)$ represent the positions of the center

of foot 2 in the robot workspace corresponding to the different values of motor angle ψ. In range [b, c], the robot moves in the translation mode, *i.e.*, both lock-pins are outside the cam and the legs contract or extend. When motor 2 rotates CW and ψ falls in $[a, b)$, the robot moves in spin-1 mode. In this range, leg 2 contracts and foot 1 rotates, making the free foot move along an Archimede's spiral curve. When motor 2 rotates in CCW direction and ψ falls in $(c, d]$, the robot moves in spin-2 mode. In this range, leg 1 extends and foot 2 winds up.

Similarly, Figure 4.5 shows the motion pattern of the robot in RFS phase. When the motor angle ψ is in the range $[b, c]$, it moves in the translation mode. The CCW rotation of motor 2 puts ψ within $(c, d]$, and the robot moves in spin-2 mode. In this mode, since foot 2 supports the robot, the rotation of foot 2 and the extending of leg 1 causes the robot to move along a spiral curve. The CW rotation of motor 2 makes the robot move in spin-1 mode where leg 2 contracts and foot 1 winds up.

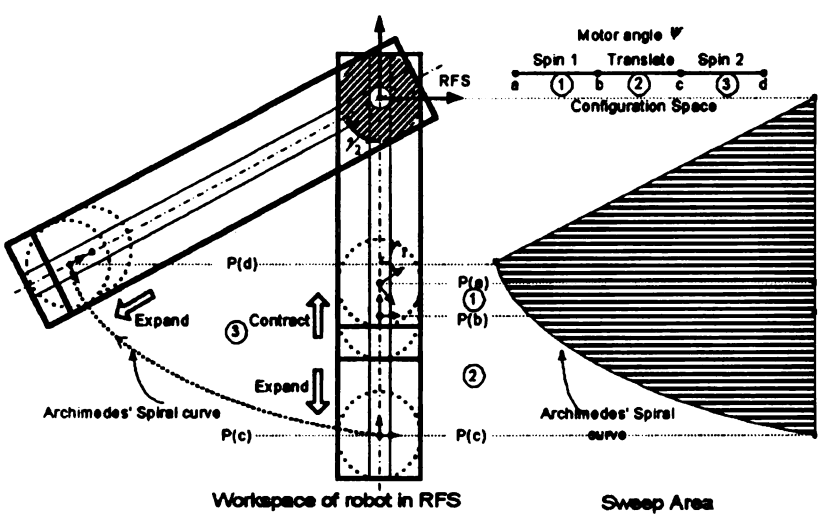

Figure 4.5: Motion pattern of the CRAWLER robot in RFS phase

Let ℓ be the distance between the centers of the two feet, then the robot motion

in LFS and RFS can be modeled as follows:

$$\begin{cases} \dot{\gamma}_1 &= h\dot{\psi} \\ \dot{\ell} &= k\dot{\psi} \quad \text{if } a \leq \psi < b \\ \gamma_2 &= 0 \end{cases} \tag{4.1}$$

$$\begin{cases} \gamma_1 &= 0 \\ \dot{\ell} &= 2k\dot{\psi} \quad \text{if } b \leq \psi \leq c \\ \gamma_2 &= 0 \end{cases} \tag{4.2}$$

$$\begin{cases} \gamma_1 &= 0 \\ \dot{\ell} &= k\dot{\psi} \quad \text{if } c < \psi \leq d \\ \dot{\gamma}_2 &= h\dot{\psi} \end{cases} \tag{4.3}$$

where $a < b < c < d$, k, and h are constants. γ_1 and γ_2 are the rotation angles of foot 1 and foot 2, respectively. When foot 1 (or foot 2) is the standing foot, γ_1 (or γ_2) represents the robot **swing angle** relative to the **robot main axis**.

Definition 4.2.2 (Robot Main axis) *The main axis of the CRAWLER robot is the straight line connecting the center of the two feet when both $\gamma_1 = 0$ and $\gamma_2 = 0$. It is the motion direction when the robot moves in translation mode.*

The following property is observed which will be used in the robot motion planning:

Property 4.2.1 *The sweep area of the robot moving in RFS phase is much larger than that in LFS phase.*

Due to the under-actuated mechanism, the CRAWLER robot needs to switch the standing foot to make a right turn. Figure 4.6 shows the motion pattern of the robot

to accomplish a right turn when the left foot is the standing foot. In the first step, motor 2 rotates CW causing the robot to contract until it reaches P(b). At this point the robot motion switches from the translation mode to spin-1 mode. The difference between making a left turn or right turn is in step 2. To make a right turn, the robot needs to switch the standing foot to the right foot (foot 2). The continuous CW rotation of motor 2 in RFS phase will wind up foot 1 to the desired angle γ_1, and make the robot contract. In step 3, the standing foot switches back to the left foot (foot 1), and motor 2 rotates in CCW direction. The robot still moves in spin-1 mode, but the CCW rotation of motor 2 makes the robot extend and swing to the right by γ_1 until motor 2 angle reaches b.

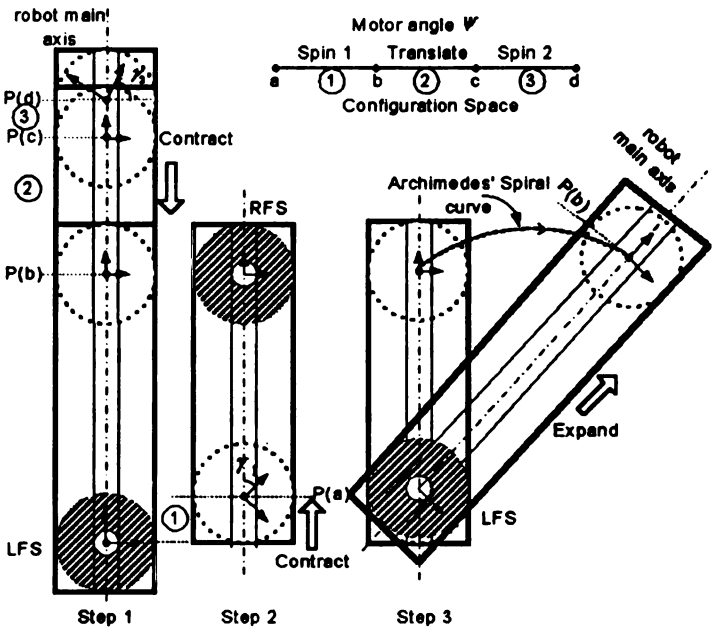

Figure 4.6: Motion pattern of the CRAWLER robot making right turn in LFS phase

Similarly, Figure 4.7 shows the motion pattern of the robot to accomplish a right turn when the right foot is the standing foot. The CCW rotation of motor 2 makes the robot extend until it reaches P(c). In step 2, the robot switches standing foot to left foot and moves in spin-2 mode. The further CCW rotation of motor 2 makes the robot extend, and foot 2 wind up to a desired angle γ_2. After switching the standing

foot to the right foot, motor 2 rotates in CW direction, making the robot contract and swing to the right by γ_2.

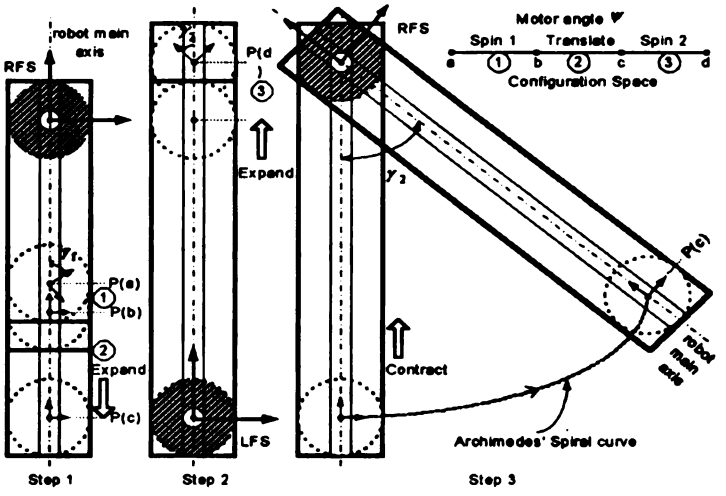

Figure 4.7: Motion pattern of the CRAWLER robot making right turn in RFS phase

In practical implementation, the switching of the standing foot is a procedure that cost a lot of battery power. The robot needs to turn on the pump motor of its free foot; adjust tilt angle to ensure that the suction cup grips the flat surface firmly; and then turn off the pump motor of the original standing foot. The robot needs to minimize this switching to decrease the power consumption.

Free Motion Case

At the second stage, we study the robot motion when it walks on a plane. Let $\mathcal{W}$ be a 2-D Euclidean space R^2, representing the robot physical workspace, equipped with a fixed frame X_0OY_0 as reference. The robot reference coordinate frame is attached at the center of foot 1 and is moving when the robot walks in $\mathcal{W}$. At each specific location of $\mathcal{W}$, the robot may swing and its length may change when the robot contracts or extends. The term *robot posture* is used to describe the robot length, and the swing angle related to the robot main axis. Now we have the following proposition:

Proposition 4.2.1 *The configuration space of the CRAWLER robot walking on a 2-D plane is 4-dimensional and can be parameterized by* (x, y, θ, ψ). (x, y) *and* θ *represent the position and orientation of the moving robot reference frame with respect to* X_0OY_0. ψ *is the motor 2 angle which determines the robot posture. The configuration of the CRAWLER robot is a specification of position, orientation and posture.*

4.2.3 Hybrid Configuration Space

The configuration of the robot cannot uniquely determine the robot motion. For example, in Figure 4.8 (A) and (B), although the robot has the same position and orientation in the workspace X_0OY_0, the current standing foot will determine which obstacle the robot will hit when it extends. In LFS phase, the extension of the robot will cause it to collide with obstacle 1. On the other hand, if the right foot is the standing foot, the robot may hit obstacle 2. Moreover, even when the robot has the same position, orientation, and standing foot, the motion mode will determine which obstacle it will hit. In Figure 4.8 (C), the contraction of the robot makes it move in spin-1 mode, and therefore it will collide with obstacle 2. On the other hand, the extension of the robot will make it hit obstacle 1.

Therefore we define the **hybrid configuration space** concept to integrate the robot configuration with the motion status, and analyze the motion planning under this framework.

Definition 4.2.3 (Hybrid Configuration Space) *Let two topological spaces* C *and* S *be the robot configuration space and the motion status space, respectively. The product of these two topology spaces* $(C \times S)$ *is defined as the* **hybrid configuration space**.

By adding the robot motion status dimension to the configuration space description, we can explicitly control the robot standing foot and motion mode; thereby

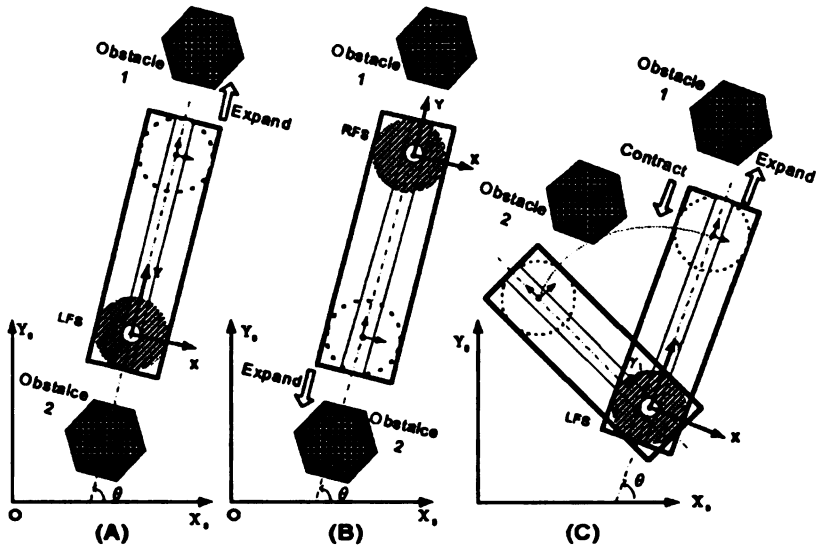

Figure 4.8: The CRAWLER robot collides with obstacles

uniquely determine the robot motion. The configuration space is decomposed into two submanifolds, one for each standing foot. Each submanifold is further partitioned into three subspaces associated with the motion modes. Each of the three subspaces has different characteristics as described by Equation 4.1 to Equation 4.3. Unlike the stratified configuration space proposed in [22], where the configuration space is decomposed into submanifolds upon which the system has different sets of equations of motion, the hybrid configuration space enables us to more selectively choose among the six submanifolds associate with the motion status. The introduction of the motion status variable indicating the standing foot and motion modes allows us to describe the motion planning approach for the climbing robot more efficiently.

4.3 Motion Planning Algorithm

4.3.1 Problem formulation

The motion planning of the CRAWLER robot can be described as follows: given the initial and final configurations (position (x, y), orientation θ, posture ψ) and the

status (standing foot, motion mode) of the robot, find a collision- free path P and a feasible motion sequence to allow the robot to navigate through P.

A motion planning approach consisting of a global planner and a local planner is developed and shown in Figure 4.9.

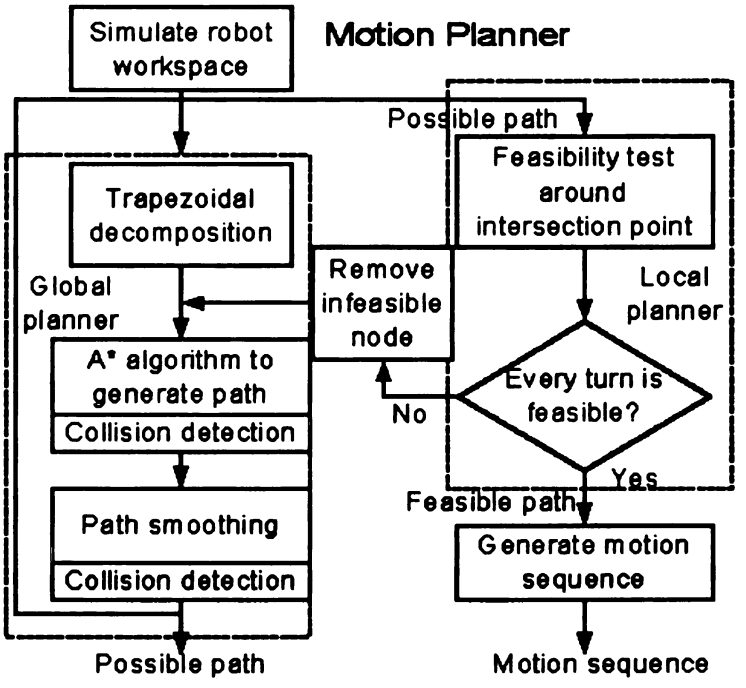

Figure 4.9: Block diagram of the motion planner

4.3.2 Global Planner

The motion planning approach adopts a two-level refinement. The global planner simplifies the robot as a rectangular body and searches the path in a 3-D sub-space (x, y, θ) of the hybrid configuration space without taking the ψ and motion status dimensions into account. The robot is regarded as a free-flying rigid object, *i.e.* the motion of the robot is not limited by any kinematic constraints except the obstacles. The shape of the robot is assumed to be enclosed in a rectangular boundary and the representation of the robot is reduced to just the location (x, y) and orientation θ. The width of the rectangle has the same physical dimension as the robot's width,

while the length is $P(b)$, *i.e.*, the smallest length value in translation mode. The local frame of the robot is fixed at the axis of foot 1. The purpose of the global planner is to find a collision-free path by initially ignoring the details of the robot constraints. The global planner uses the trapezoidal decomposition method and the A^* algorithm described in section 4.1.3 to generate a collision-free minimal length path. The collision detection is achieved by "growing" the obstacle boundary with the robot width. After that, the global planner smooths out the path by eliminating sharp turns and combining the line segments to form a straight line whenever possible. The constructed collision-free path after smoothing is named the *possible path*.

The trapezoidal decomposition method generates paths that can be smoothed relatively easily, but this is not necessarily the case for many current randomized algorithms (e.g., the probabilistic roadmap algorithm or Rapidly-Exploring Random Tree algorithm). For randomized algorithms, the time required to smooth paths to the degree required by the local planner can be large. This is the reason we have selected the trapezoidal decomposition method rather than randomized methods.

4.3.3 Local Planner

The *possible path* describes the global motion of the simplified robot but may not be feasible for the real robot to travel. Any straight line in a *possible path* is always feasible. However, intersections of the line segments where the robot makes turns may be non-traversable. The purpose of the local planner is to make a feasibility test in the region around each intersection point by searching for a motion sequence which can accomplish the turn. If a motion sequence exists, the turn is feasible but the path may contain deviations from the *possible path*. The new path which passed the test is named the *feasible path*. If a motion sequence cannot be found due to the obstacles crowding around the intersection point, the path passing though this point is infeasible and ruled out. In this case, after removing the infeasible points, the global

planner is called again to generate another smooth *possible path*. The feasibility test of the new *possible path* by the local planner is repeated. This interactive process between the global planner and the local planner continues until a *feasible path* is constructed or the motion planner returns a failure.

The feasibility test is conducted locally and considers the robot constraints. The process is basically a search of a decision graph guided by a cost function. The initial and goal nodes of the graph are the starting and final configurations of the turn. At each node, a collision detection is performed and a decision is made among four possible actions the robot can take from the current configuration. Those are (1) switch foot, (2) translate, (3) turn left, and (4) turn right.

Since the standing foot and robot configuration at each motion step is pre-determined by the prior node, the collision detection of the current node is conducted in the one dimensional configuration space ψ. The projection of the local obstacles into the 1-D configuration space is made by taking advantage of the sweep area and the motion status. For example, in Figure 4.4, if an obstacle maps outside the range of $\psi \in [a, d]$ of the configuration space, then it will not collide with the robot. If an obstacle is located in the sweep area, it will be mapped to somewhere within range $\psi \in [a, b]$. The collision will possibly occur when the robot moves in spin-1 mode. Thus, the decision will be make to either avoid turn left or to turn left just with a limited angle avoiding the collision. When the required turning angle is larger than the maximum angle the robot can spin, it becomes necessary for the robot to make several small intermediate turns.

The time taken by the local planner to examine the feasibility of the possible path in the vicinity of the turning intersections is minimized because the global planner has smoothed the path and reduced the number of turns. Another reason for the motion planner to smooth the path is that the CRAWLER robot is designed to prefer straight line motion.

The last task of the motion planner is to generate a motion sequence to travel along the feasible path. The prefered motion sequence to travel in the straight line is moving in translation mode with full stride. Since the motion sequence around every intersection has already been produced when the local planner made its feasibility test, the combination of the straight line and intersection motion sequence composes the whole motion sequence. The connection between the straight line motion sequence and the intersection motion sequence is a noticeable issue. The motion at full stride may reach the intersection on a wrong foot or at a wrong step size. There are two solutions to this problem. The first one is to adjust the last foot step only and the second one is to share the step size difference among all the translation strides. Either solution is easy to implement.

4.3.4 Cost Function

The complete feasibility test of the local planner requires an exhaustive search and is practically infeasible. So some restrictions and heuristics are introduced to guide the search for a feasible path and motion sequence:

1. The robot is designed to prefer translation motion. Hence, the motion planner should generate straight line paths and minimize the number of turns whenever possible.

2. The sweep area of the robot in RFS is much larger than that in LFS. This leads to the selection of left foot as the standing foot because the robot is prone to collide with obstacles in RFS.

3. The right turn of the robot involves the switching of the standing foot which consumes a large amount of power. Therefore, the motion planner should avoid right turns whenever possible.

4. The robot should use maximum stride length in translation mode, *i.e.*, travel the maximum slide distance from P(b) to P(c) to avoid unnecessary foot switching.

However, the construction of a decision graph still leaves a large number of possible motion sequences even if the graph is not fully expanded. A criteria to determine the "optimal motion sequence" among those possibilities is required. Taking into consideration the above heuristic rules, the optimal motion sequence is the one that minimizes the foot switching, mode switching, and renders the shortest feasible path.

In order to mathematically describe the optimal motion sequence and the heuristic rule, we first quantify the motion status and then define a cost function to guide the search. As mentioned before, the motion status has been quantified by assigning three values $(-1, 0, +1)$ to represent each of the motion modes, *i.e.*, 0 represents the translation mode, -1 represents the spin-1 mode, and $+1$ represents the spin-2 mode. Further, we define LFS $\triangleq -1$, and RFS $\triangleq 1$.

The cost function of each motion step is defined as follows:

$$\begin{aligned} \text{Cost}(k) = W_0 \cdot \text{dist}(k) + W_1 \cdot \text{foot}(k) + W_2 \cdot |\text{foot}(k) - \text{foot}(k-1)| \\ + W_3 \cdot |\text{mode}(k) - \text{mode}(k-1)| \end{aligned} \tag{4.4}$$

where, $\text{dist}(k) = \sqrt{(x(k) - x(k-1))^2 + (y(k) - y(k-1))^2}$, and where W_0, W_1, W_2, and W_3 are weights.

At each step k, the motion planner keeps track of the robot location, standing foot and motion mode, and uses the corresponding value to calculate the cost function. The motion sequence which minimizes the total cost is selected as the optimal one. By changing the weights of the cost function, we can change the definition of an optimal path.

The first term of the right hand side of the Equation 4.4 shows that the cost of a short path is smaller than that of a long path. It is noted that the actual path traveled by the robot using the motion sequence has some variations from the original possible path. The second term reflects that the motion planner prefers LFS because it reduces the cost function value when foot(k) = LFS = -1. The third term reveals that the robot favors minimal foot switching. The cost of one foot switching is $2W_2$. If the robot travels the same distance with more foot switchings, then the cost will be higher. The fourth term discourages mode switching. For example, the cost of the robot remaining in the same motion mode is 0, while the cost of changing from the translation mode to the spin-1 mode is 1, and switching from spin-1 to spin-2 is 2. The minimization of the third and fourth terms helps to reduce the power usage.

The smoothing of the *possible path* by the global planner also implicitly penalizes the sharp turns and reflects the preference to linear motion over rotation motion.

4.4 Performance Evaluation

To observe the behavior of our motion planning approach, we performed simulations of the robot to navigate among polygonal obstacles in 2-D space as well as an actual experiment in which the robot moves in a maze.

4.4.1 Simulation

A motion planning software is developed under Linux on a host computer. The simulated robot workspace is easily produced using **xfig** drawing utility and saved in a .fig file. The global motion planner then reads the file and uses the polygon trapezoidalization method to generate a robot path. An $O(n \log n)$ sweeping-line based algorithm is used to horizontally decompose the free space in the robot workspace. A set of edges is produced by connecting the midpoints of adjacent parallel edges

of trapezoid free-space. The A^* algorithm is used to generate a distance-optimal path. During the process, a simple collision detection is performed by considering the width and length of the robot boundary. For a better performance of the local planner, a smoothing algorithm is employed to eliminate unnecessary turns and check if the removal of a turn will cause the new resulting path to collide with any of the obstacles. Figures 4.10 and 4.11 show the path generated by the A^* algorithm and the smoothed possible path, respectively.

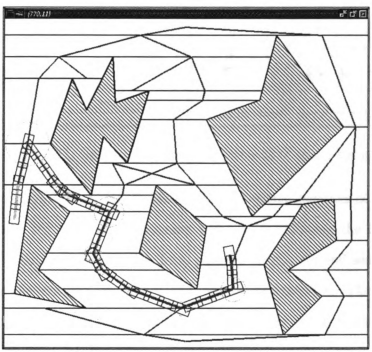

Figure 4.10: Generated path by the A^* algorithm

The local planner then makes a feasibility test and refines the robot motion within the region surrounded by a circle centered at each intersection point. The radius of the circle is determined by the robot turning angle. If the two line segments form an acute angle where the robot will take a sharp turn, then the corresponding radius is set to a large value. If the two line segments form an obtuse angle, the local planner selects a small radius. The two intersection points between the circle and the two line segments are the starting and final points of a turn. The local planner determines

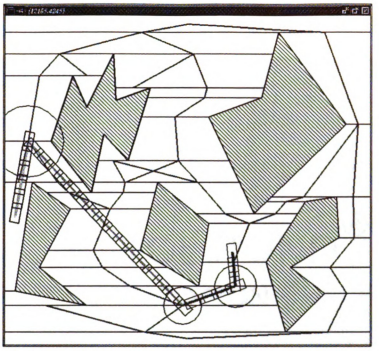

Figure 4.11: Smoothed path after removal of sharp turns

the turning direction (left or right) based on the relative position of the incoming and outgoing line segments of the turning point. By following the heuristics and complying with some restrictions, the local planner constructs and searches a graph which describes the motion sequences used to accomplish the turn. The output of the motion planner is a feasible path and a motion sequence which minimizes the cost function.

4.4.2 Experiment

In a real experiment, a CRAWLER robot is required to move from the initial configuration at the right lower part of a maze to the goal configuration at the upper left part of the maze as shown in Figure 4.12. The environment of the maze is simulated using the drawing utility. Once a user selects a destination via the user interface, the motion planner running on the host PC reads the description of the environment from a .fig file and generates a motion sequence. The sequence of commands are

imported to the physical robot to navigate in the maze. A snapshots showing the robot traveling through the maze is presented in Figure 4.12.

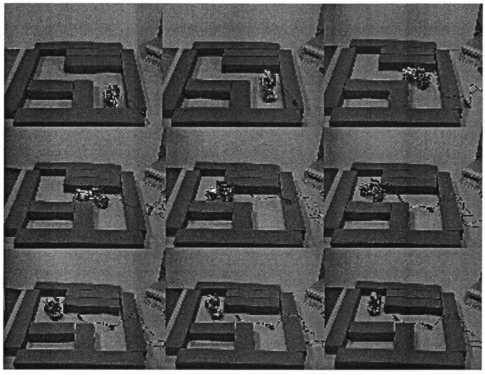

Figure 4.12: Robot navigating in a maze

4.5 Summary

This chapter investigates the motion planning of the CRAWLER robot. The under-actuated mechanism makes the motion planning extremely difficult because the robot can only perform restricted motions even in the absence of obstacles. Based on a detailed analysis of the robot motion pattern, a hybrid configuration space concept is proposed to deal with the constraints imposed by the kinematic structure. Under hybrid configuration space framework, the robot motion can be uniquely determined by incorporating the continuous configuration space with discrete motion status. A motion planning algorithm is developed which consists of a global planner and a local planner. A cost function is defined based on the motion status information to guide

the search for an optimal path. With the motion planning capability, the CRAWLER robot is able to autonomously determine a path and a set of motion sequence to travel through an environment without collision with obstacles. Simulation and experimental results verified the efficency of the motion planning method.

CHAPTER 5

CONCLUDING REMARKS

5.1 Conclusion

This dissertation describes the development of the CRAWLER climbing robot. The climbing robot uses two suction feet to support itself on surfaces. It adopts a bipedal structure and an under-actuated mechanism to achieve a good balance between compactness and mobility. This bipedal structure provides the robot with the capability to transit between two inclined surfaces. The under-actuated mechanism contributes to the small size and light weight of the robot because it uses less motors to drive more joints. A small penalty paid for under actuation is increased complexity in modeling, control and motion planning.

The mechanical structure and locomotion modes of the robot are analyzed. The kinematic model that describes the relation between the joint variables and the position/orientation of the robot's free foot with respect to its standing foot is derived. The robot kinematic model is governed by different equations in different motion modes. Therefore, it is important to use the corresponding kinematic equations when planning the robot motion. In order to study the dynamic behavior of the climbing robot, the dynamic model is derived based on the Lagrange-Euler formulation. The dynamic model describes the relation between the input joint torques and the output motion. It reveals that the gravitational effects play a dominant role in robot dynamics when the robot moves at slow speeds.

An embedded control system is designed and implemented based on a TI DSP chip. The self-contained controller hardware is built, which efficiently utilizes the DSP resources and minimizes the component count. A joint level PD+gravity compensation method is proposed. This control method outperforms the conventional PD controller because it not only achieves joint level control but also compensates for

the gravity loading, which is determined by the configuration of all the robot links. Software modules are developed to effectively control the robot. Experimental results demonstrate that the robot has the capability to climb walls, walk on ceilings, crawl through pipes, and transit between two inclined surfaces.

In motion planning analysis, a hybrid configuration space concept is proposed. The hybrid configuration space incorporates the continuous configuration space with discrete motion status (*i.e.*, standing foot, motion mode), and it is a more effective tool to analyze the motion planning of the climbing robot. Based on the motion pattern analysis, a motion planning method is developed to generate a collision-free path and feasible motion sequence to allow the robot to travel along the path. The motion planning approach consists of a global planner and a local planner. This approach of two levels of refinement reduces the overall complexity and simplifies implementation. A cost function is defined based on the motion status information and a number of heuristics to help the search for an optimal motion sequence. Experiments and simulations demonstrate the effective performance of the CRAWLER robot system.

5.2 Future Work

In this section, some recommendations for the improvement of the CRAWLER robot and for future research directions of climbing robots are made.

As a prototype robot, the CRAWLER robot has some mechanical malfunctions and hence needs improvement. It is observed that the shafts of joint 1 and joint 5 slip (see Figure 1.3), resulting in degradation of performance. Sometimes the joints are not able to lift up the robot body due to the slipping of the joint shafts. We suggest that the joints 1 and 5 be fixed to their respective ankles with screws passing through their shafts (Figure 2.1). Another problem we experienced is the difficulty in getting accurate information on when the leg/rack pairs are disengaged. This information

determines the motion mode switching. Currently, we have installed two contact switches on the robot legs to determine when the lock-pins slide inside the lock-pin cams. But this indirect scheme is not sufficiently accurate to determine the motion mode switching. The suggested solution is to install a contact switch inside each cam to get more accurate information on motion mode switching.

When a CRAWLER robot climbs in different places, the gravity effects change with the varying inclinations. It would be beneficial to install a miniature inclinometer to facilitate the compensation of gravity effects.

Basic motion planning assumes that the robot knows the environment *a prior*. This assumption is reasonable but more work needs to be done to obtain the environment settings by processing video images or using a global laser scanner. One application scenario is of a FLIPPER robot that climbs on a office ceiling to detect and transmit video images of the targeted office environment to a host computer. The host computer processes the video images to obtain the environment settings and then executes the motion planning algorithm. The motion planner on the host computer generates a feasible motion sequence and commands the actual CRAWLER robot to travel through the environment.

Climbing robots using suction feet have some constraints. The first constraint is that they are unable to move at fast speeds. This limitation is due to the fact that it takes time to vacuum the air out of the suction cup until a proper pressure is established or to release the foot by opening the valve until the suction cup can be peeled off from the surface. The second constraint is that the robots with suction feet can only walk on flat and non-porous surfaces. In order to overcome those problems, great efforts should be made to exploit novel adhesive mechanisms which are adaptable to different kinds of surfaces without sacrificing the robot motion speed. Inspired by the exceptional ability of gecko lizards who are able to climb rapidly on nearly any vertical surfaces, some scientists are now investigating the structure of the

gecko foot [4] in hopes of providing the key to robotic climbing.

Another promising adhesive mechanism is based on aerodynamic attraction. This technique has been demonstrated at the 2000 DARPA Distributed Robotics Program PI (Principal Investigator) Meeting and is illustrated in Figure 5.1. A rotating column of air is created by a rotating disk containing miniature turbo plates located on the inside rim of the disk. This cylindrical column of air has properties similar to a vortex in that the interior air pressure is much lower than the ambient air pressure, and as a consequence, a resultant force is generated which attracts the disk towards the ceiling. The climbing robot based on this adhesive technique is driven by wheels, leading to fast speed when moving on a wall. However, it is difficult for the robot to transit between different inclined surfaces, such as from a floor to a wall, or from a wall to a ceiling. A future research focus is to combine the two adhesive mechanisms (suction foot and aerodynamic attraction) and exploit the advantages of both the wheeled locomotion and articulate structure to achieve the quick motion and smooth transition.

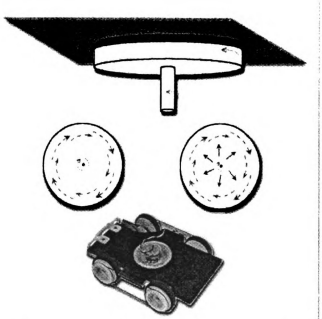

Figure 5.1: Aerodynamic attraction and a prototype wheeled climbing robot

BIBLIOGRAPHY

[1] M. Abderrahim, C. Balaguer, A. Gimenez, J.M. Pastor, and V.M. Padron. Roma: a climbing robot for inspection operations. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, number 3, pages 2303–2308, Detroit, Michigan, USA, 1999.

[2] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo. Obprm: An obstacle-based prm for 3d workspaces. *Robotics: The Algorithmic Perspective (1998 Workshop on the Algorithmic Foundations of Robotics)*, pages 155–168, 1998.

[3] N. M. Amato and Y. Wu. A randomized roadmap method for path and manipulation planning. *IEEE Trans. on Robotics and Automation*, pages 113–120, 1996.

[4] Kellar Autumn, Yiching A. Liang, S. Tonia Hsieh, Wolfgang Zesch, Wal Pang Chan, Thomas W. Kenny, Ronald Fearing, and Robert J. Full. Adhesive force of a single gecko foot-hair. *Nature*, 405:681–684, June 2000.

[5] Fr.-W. Bach, H. Haferkamp, J. Lindemaier, and M. Rachkov. Underwater climbing robot for contact arc metal drilling and cutting. In *Proceedings of the 1996 IEEE IECON 22nd International Conference on Industrial Electronics, Control, and Instrumentation*, volume 3, pages 1560–1565, 1994.

[6] Paul G. Backes, Yoseph Bar-Cohen, and Benjamin Joffe. The multifunction automated crawling system (macs). In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, pages 335–340, Albuquerque, New Mexico, USA, 1997.

[7] B. Bahr, Y. Li, and M. Najafi. Design and suction cup analysis of a wall climbing robot. *Computers and Electrical Engineering*, 22(3):193–209, 1996.

[8] M. Barbehenn and S. Hutchinson. Efficient search and hierarchical motion planning by dynamically maintaining single-source shortest paths trees. *IEEE Trans. On Robotics and Automation*, 11(2):198–214, 1995.

[9] Barraquand, B. Langlois, and J. C. Latombe. Numerical potential field techniques for robot path planning. *IEEE Transactions on System, Man and Cybernetics*, 22(2):224–241, 1992.

[10] Barraquand and J. C. Latombe. A monte-carlo algorithm for path planning with many degrees of freedom. In *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, pages 1712–1717, 1990.

[11] R. Bohlin and L. E. Kavraki. Path planning using lazy prm. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, pages 521–528, 2000.

[12] L. Briones, P. Bustamante, and M.A. Serna. Wall-climbing robot for inspection in nuclear power plants. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, volume 2, pages 1409–1414, 1994.

[13] R. A. Brooks. Solving the find-path problem by good representation of free space. *IEEE Transactions on System, Man and Cybernetics*, 13(3):190–197, 1983.

[14] John F. Canny. *The complexity of Robot Motion Planning.* The MIT Press, 1988.

[15] H. Choset and J. Burdick. Sensor based planning, part I: The generalized voronoi graph. In *Proceedings of the 1995 IEEE international Conference on Robotics and Automation*, pages 1649–1655, 1995.

[16] G. Dangi, J. Stam, and D. Aslam. Design, fabrication and testing of a smart robotic foot for microrobotic system. In *Proceeding of the 2000 International Symposium on Robotics*, 2000.

[17] H. Edelsbrunner. *Algorithms in Combinatorial Geometry.* Springer-Verlag, 1987.

[18] K.S. Fu, R.C. Gonzalez, and C.S.G Lee. *Robotics: Control, Sensing, Vision and Intelligence.* Mcgraw-Hill, 1987.

[19] S. Galt and B.L. Luk. Smooth and efficient walking motions and control for an 8-legged robot. In *Proceedings of the UKACC International Conference on Control*, volume 2, pages 1652–1657, 1997.

[20] S. Galt, B.L. Luk, D.S. Cooke, and A.A. Collie. A tele-operated semi-intelligent climbing robot for nuclear applications. In *Proceedings of the Fourth Annual Conference on Mechatronics and Machine Vision in Practice*, pages 118–123, 1997.

[21] S. K. Ghosh and D. M. Mount. An output sensitive algorithm for computing visibility graphs. In *Proceedings of the IEEE Symp. on Foundations of Computer Science*, pages 11–19, 1987.

[22] Bill Goodwine and Joel W. Burdick. Controllability of kinematic control systems on stratified configuration spaces. *IEEE Transactions On Automatic and Control*, 46(3):358–368, 2001.

[23] Juan Carlos Grieco, Manuel Prieto, Manuel Armada, and Pablo Gonzalez de Santos. A six-legged climbing robot for hight payloads. In *Proceedings of the 1998 IEEE International Conference on Control Applications*, pages 446–450, Trieste, Italy, 1998.

[24] Lin Guo, K. Rogers, and R. Kirkham. A climbing robot with continuous motion. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 2495–2500, Albuquerque, New Mexico, USA, 1997.

[25] P.E. Hart, N.J. Nilsson, and B Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems, Science and Cybernetics*, 4(2):100–107, 1968.

[26] S. Hirose, A. Nagakubo, and R. Toyama. Machine that can walk and climb on floors, walls and ceilings. In *Proceedings of the Fifth International Conference on Advanced Robotics*, number 1, pages 753–758, 1991.

[27] Shigeo Hirose and Hiroshi Tsutsumitake. Disk rover: A wall-climbing robot using permanent magnet disks. In *Proceedings of the 1992 IEEE/RSJ International Conference on Itelligent Robots and Systems*, pages 2074–2079, Raleigh, NC, 1992.

[28] T. Horsch, F. Schwarz, and H. Tolle. Motion planning for many degrees of freedom — random reflections at c-space obstacles. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 3318–3323, San Diego, CA, 1994.

[29] D. Hsu, L. Kavraki, J. C. Latombe, R. Motwani, and S. Sorkin. On finding narrow passages with probabilistic roadmap planners. *Robotics: The Algorithmic Perspective (1998 Workshop on the Algorithmic Foundations of Robotics)*, pages 141–154, 1998.

[30] Y. K. Hwang and N. Ahuja. A potential field approach to path planning. *IEEE Trans. On Robotics and Automation*, 8(1):23–32, 1992.

[31] Spectrum Digital Inc. *TMS320LF2407 evaluation module technical reference*, January 2000.

[32] Texas Instruments Inc. *TMS320LF/LC240x DSP controllers reference guide*, January 2000.

[33] Laveen Kanal and Vipin Kumar. *Search in Artificial Intelligence.* Springer-Verlag, Boston, MA, USA, 1988.

[34] L. E. Kavraki and J. C. Latombe. Randomized preprocessing of configurations space for fast path planning. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 2138–2139, San Diego, CA, 1994.

[35] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. On Robotics and Automation*, 12(4):566–580, 1996.

[36] Y. Kawaguchi, I. Yoshida, H. Kurumatani, T. Kikuta, and Y. Yamada. Internal pipe inspection robot. In *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, pages 857–862, 1995.

[37] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *Journal of Robot Research*, 5(1):90–98, 1986.

[38] J. J. Kuffner and S. M. LaValle. Rrt-connect: an efficient approach to single-query path planning. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, pages 995–1001, 2000.

[39] J. C. Latombe. *Robot motion planning.* Kluwer Academic Publishers, Boston, MA, USA, 1991.

[40] S. M. Lavalle. Robot motion planning: A game-theoretic foundation. *Algorithmica*, 26:430–465, 2000.

[41] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, pages 473–479, 1999.

[42] Steven Michael Lavalle. *A Game-theoretic Framework for Robot Motion Planning.* PhD thesis, University of Illinois at Urbana-Champaign, 1995.

[43] B. Luk, A. Collie, and Billingsley. Robug II: An intelligent wall climbing robot. In *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, pages 2342–2347, Sacramento, Ca, USA, 1991.

[44] B. Luk, A. Collie, V. Piefort, and G. S. Virk. Robug III: A tele-operated climbing and walking robot. In *UKACC International Conference on Control*, pages 347–352, 1996.

[45] Mark Minor, Hans Dulimarta, Girish Danghi, Ranjan Mukherjee, R. Lal Tummala, and Dean Aslam. Design, implementation and evaluation of an underactuated miniature biped climbing robot. In *Proceeding of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1999–2005, Takamatsu, Japan, 2000.

[46] Mark A. Minor. *Design and Control of Constrained Robotic Systems for Enhanced Dexterity and Mobility.* PhD thesis, Michigan State University, 2000.

[47] API Motion. *Miniature high performance motors and peripheral components for motion solution.* West Chester, PA, USA, 1998.

[48] James R. Munkres. *Topology, second edition.* Prentice Hall, 2000.

[49] A. Nagakubo and S. Hirose. Walking and running of the quadruped wall-climbing robot. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, volume 2, pages 1005–1012, 1994.

[50] Werner Neubauer. A spider-like robot that climbs vertically in ducts or pipes. In *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*, volume 2, pages 1178–1185, 1994.

[51] A. Nishi. A biped walking robot capable of moving on a vertical wall. *Mechatronics*, 2(5):543–554, 1992.

[52] A. Nishi. Development of wall-climbing robots. *Computers and Electrical Engineering*, 22(2):123–149, 1996.

[53] A. Nishi and H. Miyagi. Control of a wall-climbing robot using propulsive force of propeller. In *IEEE/RSJ International Workshop on Intelligent Robots and Systems*, number 3, pages 1561–1567, 1991.

[54] A. Nishi and H. Miyagi. A wall climbing robot using propulsive force of propeller. In *Proceedings of the Fifth International Conference on Advanced Robotics*, number 1, pages 320–325, 1991.

[55] C. Nissoux, T. Simeon, and J. P. Laumond. Visibility based probabilistic roadmaps. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1316–1321, 1999.

[56] C. O'Dunlaing and C. K. Yap. A retraction method for planning the motion of a disc. *Journal of Algorithms*, 6:104–111, 1982.

[57] R. T. Pack, J. L. Christopher, and K. Kawamura. A rubbertuator-based structure climbing inspection robot. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, pages 1869–1874, Albuquerque, New Mexico, USA, 1997.

[58] T. Lozano Perez. Automatic planning of manipulator transfer movements. *IEEE Transactions on System, Man and Cybernetics*, SMC-11(10):681–698, 1981.

[59] T. Lozano Perez. Spatial planning: A configuration space approach. *IEEE Transactions on Computers*, C-32(2):108–120, 1983.

[60] T. Lozano Perez. A simple motion-planning algorithm for general robot manipulators. *IEEE Journal of Robotics and Automation*, RA-3(3):224–238, 1987.

[61] T. Lozano Perez and M. A. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10):560–570, 1979.

[62] E. Rimon and D. E. Koditschek. Exact robot navigation using artificial potential fields. *IEEE Trans. On Robotics and Automation*, 8(5):501–518, 1992.

[63] H. Rohnert. Shortest paths in the plane with convex polygonal obstacles. *Information Processing Letters*, 23:71–76, 1986.

[64] S. W. Ryu, J. J. Park, S. M. Ryew, and H. R. Choi. Self-constained wall-climbing robot with closed link mechanism. In *Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 839–844, Maui, Hawaii, USA, 2001.

[65] J. Savall, A. Avello, and L. Briones. Two compact robots for remote inspection of hazardous areas in nuclear power plants. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, pages 1993–1998, 1999.

[66] J. T. Schwartz and M. Sharir. On the piano movers' problem: I. the case of a two-dimensional rigid polygonal body moving amids polygonal barriers. *Communication on pure and applied mathematics*, 36:345–398, 1983.

[67] Rajeev Sharma and Herry Sutanto. Unifying configuration space and sensor space for vision-based motion planning. In *Proceedings of the 1996 IEEE international Conference on Robotics and Automation*, number 2, pages 3572–3577, Minncapolis, Minnesota, USA, 1996.

[68] Linzhi Sun, Ping Sun, and Xinjie Qin. Study on micro-robot in small pipe. In *Proceedings of the UKACC International Conference on Control*, pages 1212–1217, 1998.

[69] Linzhi Sun, Ping Sun, Xinjie Qin, and Cunmin Wang. Micro robot in small pipe with electromagnetic actuator. In *Proceedings of the International Symposium on Micromechatronics and Human Science*, pages 243–248, 1998.

[70] P. Svestka and M. Overmars. A probabilistic learning approach to motion planning. *Algorithmic Foundations of Robotics (1994 Workshop on the Algorithmic Foundations of Robotics)*, pages 19–37, 1995.

[71] R. L. Tummala, R. Mukherjee, D. Aslam, N. Xi, S. Mahadevan, and J. Weng. Reconfigurable and adaptable micro-robot. In *Proceedings of 1999 IEEE International Conference on Systems, Man and Cybernetics*, pages 687–691, 1999.

[72] T. S. White, N. Hewer, B. L. Luk, and J. Hazel. Design and operational performance of a climbing robot used for weld inspection in hazardous environments. In *Proceedings of the 1998 IEEE International Conference on Control Applications*, pages 451–455, Trieste, Italy, 1998.

[73] S. A. Wilmarth, N. M. Amato, and P. F. Stiller. Maprm: A probabilistic roadmap planner with sampling on the medial axis of the free space. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, pages 1024–1031, 1999.

[74] Jizhong Xiao, Hans Dulimarta, Zhenyu Yu, Ning Xi, and R. Lal Tummala. Dsp solution for wall-climber micro-robot control using tms320lf2407 chip. In *Proceedings of the 43rd IEEE Midwest Symposium on Circuits and Systems*, Lansing, Michigan, USA, 2000.

[75] Jizhong Xiao, Mark Minor, Hans Dulimarta, Ning Xi, R. Mukherjee, and R. Lal Tummala. Modeling and control of an under-actuated miniature crawler robot. In *Proceedings of 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1546–1551, Maui, Hawaii, USA, 2001.

[76] Wang Yan, Liu Shuliang, Xu Dianguo, Zhao Yanzheng, Shao Hao, and Gao Xueshan. Development and application of wall-climbing robots. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, number 2, pages 1207–1212, Detroit, Michigan, USA, 1999.

[77] T. Yano, S. Numao, and Y. Kitamura. Development of a self-contained wall climbing robot with scanning type suction cups. In *Proceedings of the 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems*, number 1, pages 249–254, Victoria, Canada, 1998.

[78] Zhenyu Yu. 3.3v DSP for digital motor control. Technical Report Application Report SPRA550, Texas Instruments Inc., 1999.

[79] Meng Yue, Mark Minor, Ning Xi, and Ranjan Mukherjee. Kinematics workspace analyses of a miniature walking robot. In *Proceedings of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1798–1803, Kyongju, Korea, 1999.