11-28-1998

Integration of fault tolerance and hardware redundancy techniques into the design of mobile platforms

Miguel Angel Espina
Florida International University

DOI: 10.25148/etd.FI15101314

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

INTEGRATION OF FAULT TOLERANCE AND HARDWARE REDUNDANCY TECHNIQUES INTO THE DESIGN OF MOBILE PLATFORMS

A thesis submitted in partial fulfillment of the

requirements for the degree of

MASTER OF SCIENCE

in

MECHANICAL ENGINEERING

by

Miguel Angel Espina

1999

To: Dean Gordon Hopkins
College of Engineering

This thesis, written by Miguel Angel Espina, and entitled Integration of Fault Tolerance and Hardware Redundancy Techniques Into the Design of Mobile Platforms, having been approved in respect to style and intellectual content, is referred to you for judgement.

We have read this thesis and recommended that it be approved.

T.C. Yih, Ph.D.

Ibrahim Tansel, Ph.D.

Sabri Tosunoglu, Ph.D., Major Professor

Date of Defense: November 20, 1998

The thesis of Miguel Angel Espina is approved.

Dean Gordon Hopkins
College of Engineering

Dean Richard L. Campbell
Division of Graduate Studies

Florida International University, 1999

DEDICATION

I dedicate this thesis to my wife. Without her patience, understanding, support, and most of all unconditional love, the completion of this work would not have been possible.

ACKNOWLEDGMENTS

I wish to thank the members of my committee for their support in the completion of this thesis work. Their gentle but firm direction has been most appreciated. Major Professor Sabri Tosunoglu was particularly helpful in guiding me toward a qualitative methodology. I also like to thank my friends Pat Batsomboon, Victor Monterverde, Ernesto Romero and Ray Reyes for their contribution to my thesis work in different ways. Finally, I wish to thank my wife, my parents and relatives for their support. Without them, this work would not have been successfully completed.

ABSTRACT OF THE THESIS

INTEGRATION OF FAULT TOLERANCE AND HARDWARE REDUNDANCY TECHNIQUES INTO THE DESIGN OF MOBILE PLATFORMS

by

Miguel Angel Espina

Florida International University, 1999

Miami, Florida

Professor Sabri Tosunoglu, Major Professor

This work addresses the development of a fault-tolerant mobile platform. Fault-tolerant mechanical system design is an emerging technology that attempts to build highly reliable systems by incorporating hardware and software architectures. For this purpose, previous work in fault-tolerant were reviewed. Alternate architectures were evaluated to maximize the fault tolerance capabilities of the driving and steering systems of a mobile platform.

The literature review showed that most of the research work on fault tolerance has been done in the area of kinematics and control systems of robotic arms. Therefore, hardware redundancy and fault tolerance in mobile robots is an area to be researched.

The prototype constructed as part of this work demonstrated basic principles and uses of a fault-tolerant mechanism, and is believed to be the first such system in its class. It is recommended that different driving and steering architectures, and the fault-tolerant controllers' performance be tested on this prototype.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE PAGE

Chapter 1. Fault Tolerance and Mobile Robotics

1.1 Introduction

Manipulators and mobile platforms are often used to perform operations in hazardous environments such as nuclear reactors, outer space, deep sea, waste management operations and other sites that may represent a potential danger for human beings. The utilization of robotic systems in these sites eliminates the need for human labor in such hazardous environments. In the past, the justification for using field robots was to avoid placing human workers in hazardous situations such as those in the accidents at Three Mile Island and Chernobyl. As mentioned by Jamshide and Eicker in [7], the potential of robotic systems in the waste management field as well as emergency and remedial response operations is untapped. However, a failure of any robotic system while operating in a hazardous environment may result in the abortion of the mission if the system has not been designed to tolerate and deal with failures.

1.1.1 Literature Survey on Fault Tolerance

Fault-tolerant (FT) design is an important factor in achieving the high degree of safety and reliability required from robotic systems operating in such environments. In general, a fault-tolerant system is designed to provide the mechanical architecture with a backup subsystem that enables the entire system to undergo a failure and continue to work with minimum or no adverse consequences on its performance. Fault tolerance in mechanical systems represents an emerging technology that promises new possibilities in the design of next-generation systems. These systems will be smarter than the existing machines. They will also be more reliable, less expensive and easy to operate.

1.1.1.1 Robot Kinematics and Fault Tolerance

In order to tolerate failures, a mechanical system should be provided with a modular architecture that gives locality and autonomy to its components. Tesar, Sreevijayan, Tosunoglu and Monteverde have studied and demonstrated how hardware redundancy (i.e. fault tolerance-FT) can be implemented in robotic systems in [24], [25], [26] and [27]. This is achieved by introducing redundancies at different levels. These are listed as

- Joint Level
- Link Level
- Subsystem Level
- System Level

The first category represents the simplest means of driving individual joints. FT is introduced by incorporating two actuators that can either work together to increase the torque output of the joint or use one to produce torque and the other as backup. In a third mode of operation, one actuator will produce torque for the required motion while the other produces a counter-torque to compensate for potential backlash.

At link level, FT is applied using parallel link mechanisms. Four-bar mechanisms are more stable than the most commonly implemented serial links. Upon link failure, the stability of the parallel mechanism provides a faster recovery and limits the error propagation. In most cases, these mechanisms provide room for the introduction of FT at joint level by actuating two independent ground joints.

At subsystem level, FT is provided through kinematic redundancy. This is providing the robot with more than three and six degrees of freedom (DOF) for planar and three-dimensional robots, respectively. For instance, a three-dimensional robotic arm requires at least six DOF. The addition of another DOF would allow the system to maneuver in the workspace and perform the task even after a failure. In such a case, the brake is applied to the failed joint. Then, the controller will reconfigure the system as a 6 DOF one. However, the full range of motion of the robot after failure may be greatly affected if the robot redundancy is not well designed.

FT at system level is achieved by the cooperation of more than two robots in performing one task. If one system fails, the others will be reconfigured to take over the task left unfinished by the failed robot.

In [27], the problem of introducing fault tolerance at system level is extensively addressed. FT at this level is desirable for a number of reasons. First of all, many robotic operations are distributed in space, time, and/or functionality. Therefore, they require a distributed solution. In addition, chances are that a task can be more quickly performed if it is divided into small tasks distributed among a number of robots operating in parallel. Finally, less capable robots, working towards a common goal, could be cheaper and more reliable than one robot performing the entire mission.

Introducing FT at system level, however, is a very difficult task. Several issues have to be addressed before the system operates with the adequate level of reliability. These robotic systems often operate in dynamic environments in which unexpected obstacles can be run into. Here, excellent robot communications, conflict resolution and coherent actions represent some of the features to be present in this type of system.

1.1.1.2 Fault-Tolerant Control System Design

Through the design process, mechanical systems can be provided with the hardware required to handle faults. Then, the FT controller takes advantage of the hardware capabilities to reconfigure and recover the system from failures.

As clearly explained by Tosunoglu, Tesar and Sreevijayan in [27], Fault-Responsive Control Systems (FRS) constitute the operational part on a FT system design. FRSs perform the following tasks.

- Fault Detection and Identification (FDI)
- System Reconfiguration and Replanning (SRR)
- Task Recovery (TR)

In general, the FTS has to coordinate the fault detection and identification (FDI) process to identify and confirm the occurrence of a fault. Furthermore, FRS is in charge of the system reconfiguration and task replanning (SRR). This is done by allocating system resources such that the fault is isolated. In addition, FRS should adapt its control algorithm (ACA) to a more suitable one for the new system architecture. Finally, the FRS has to perform task recovery (TR) operations and continue the job.

FDI may be implemented by means of two schemes: model-based or model-free schemes. Model-free schemes operate off-line and require operator assistance. Model-based schemes are more suitable for real-time autonomous systems. They require a mathematical model of the system being monitored. These methods generate an error, called residual, that is compared against previously established, nominal values. Then, the comparison unit generates a report on the status of the monitored system. Finally, the new system output is generated based on this report.

SRR takes action to isolate the fault as soon as the FDI unit detects it. Then, the reconfiguration stage may require steps such as sensor isolation in case of reading problems, locking one half of a dually actuated joint if clutching is defective, brake application to a failed joint, and so on. The actions taken in the reconfiguration stage result in a mechanical system with a different architecture. Therefore, the original task has to be remapped.

At the ACA stage, the most suitable control system is selected for the resulting mechanical configuration. In order to do this, information on the isolated components, joint space, and errors accumulated in the system is gathered by the FRS from the SRR stage. Many methods such as the computed torque, sliding controller and adaptive controllers can be successfully integrated for this task. Before failure, the simple and very popular computed torque method can be used to run the system. When a failure is identified and the task is remapped, system parameters have to be newly estimated. The estimation has to be performed for a system in which the number of uncertainties has increased. Adaptive controllers are a more robust type of controller. This is a better choice for parameter identification after failure and reconfiguration.

Finally, in the TR phase, all system parameters are estimated and fine tuned. At this point the system will be ready to switch to a simpler type of controller to complete the task.

1.1.1.3 Fault Tolerance Capacity Measures

As it is frequently seen in most engineering problems, cost, size and weight are the design constraints to be minimized when designing robotic systems. Due to the fact that FT is achieved by introducing redundant components in the system, methods for configuring, measuring and optimizing the amount of these components introduced in the system design are required.

Tosunoglu and Monteverde developed an interesting method to measure FT capacity (FTCM). Based on the contribution of system FT from each of the levels explained above, a weighting system was developed. This weighting system agrees with a hierarchical list of different system architecture. Using this framework, the mathematical formulation for FT measurement is introduced.

FT measurement (f) was defined as:

$$f = 1 - \left[\frac{\sum_{i=1}^{4} w_i a_{i,base}}{\sum_{i=1}^{4} w_i a_{i,actual}} \right] \qquad \textbf{(1.1)}$$

where

f = fault tolerance measure of the robot in %.

w_i = weighting of the i'th component for the base and actual system.

a_i = total number of i'th components included in the base or actual system.

A base system is defined as the simplest configuration that is required to perform the same task assigned to the robot being evaluated. For instance, a 6 DOF serial robot with no redundant components yields a fault tolerance capacity of f = 0%. As the number of redundant components in the system increases, the equation denominator tends to infinity. On the other hand, the numerator, which represents the basic system configuration for the type of robot (i.e., Planar or spatial robot) remains constant. Consequently, the term in parenthesis tends to zero and the fault tolerance capacity of the system f tends to 100%. This method is more deeply explained by Tosunoglu and Monteverde in [16] and [23].

1.1.2 Literature Survey on Mobile Platforms

The common denominator for the introduction of robotic applications mentioned before is the need to substitute a human operator in these operations. However, eliminating human intervention in these operations also eliminates the analytical and critical thinking capacity of the work force. The decision-making capacity of humans should then be replaced with some sort of machine intelligence. In mobile robotics, these intelligent mechanical systems are called Autonomous Mobile Robots (AMR). As a type of robotic system, AMRs are an important part in the evolution of artificial intelligence. At this level of technology, the mechanical system is provided with a certain ability to move using legs, wheels or tracks. This motion is performed with a little intervention from human operators.

1.1.2.1 Autonomous Mobile Robots (AMR)

Among the first AMRs mentioned in the literature is ELSI (Electro-light-sensitive Internal-External) by W. Grey Walter (1910-1977). Based on this robot, several toys appeared on the market. They were developed based on the principle of following a light source and deciding where to navigate or avoid. Their fancy motion gave the perception of "free will" and independence. Surprisingly, the devices were just reacting in no well-determined way to their environment. Active effort to build, control and analyze intelligent mobile devices started after World War II. The first serious AMR, called Shakey, was developed and demonstrated at Stanford Research Institute in the early 60s. The robot had a camera and a tactile detector. The on-board processor was connected to an off-board large time-shearing computer in charge of computations. Actuator control was performed off-board. Image processing and planning was done on-board.

At that time, this was the most sophisticated work in applying artificial intelligence to robots. Its main sensor was a scanning camera with a broad view angle. With this sense of vision and the different levels of software, Shakey was able to navigate, explore and learn the world around it. The robot's position in the world was determined by the so-called "dead reckoning" (i.e., keeping track of the actual motion). However, the actual displacement was measured from the wheel rotation. Therefore, errors due to slippage were introduced. Consequently, the vision system was unable to correctly locate obstacles in the navigation grid. At this point it was realized that more robust machine vision systems and better reorientation methods would be required for further research.

In the early 70s, NASA in cooperation with the Jet Propulsion Laboratory (JPL) started a program to reduce the ground support requirements, provide real-time control and give more opportunities for AMR to actively participate in new space missions. The program's goals also included improvement in the reliability and performance of robots as support in space exploration, space assembly, automation of manufacturing facilities, launch and earth orbital and man-controlled remote operations of systems in hostile environments. This robot, the Mars-Rover, was able to a simple scene, plan its own path and follow it towards the target.

Stanford University Artificial Intelligence Laboratory was the base for further experiments from 1973 to 1981. A remotely-controlled and TV-equipped mobile robot was developed by Hans Moravec. The Cart located objects and deduced its motion using stereo imaging methods. The system, while reliable, turned out to be very slow. The motion was in lurches of one meter for every ten to fifteen minutes. After moving one meter, the robot would stop, take pictures of the environment and think about them for a long time. Then, it calculated its new path. It moved along it and stopped again.

This system was modernized and upgraded at the Robotic Institute of Carnegie-Mellon University. This new version of the robot had a cylindrical shape of about one meter tall and thirty centimeters in diameter. It had three individual steering wheels that gave it three DOF to move in a plane.

The Hexapode, developed at Ohio State University (1976-1977), is known as the first operative multilegged robot in the world. Each DOF of this robot is driven by electric motors. The main purpose of this machine was to study the body stability while the legs are performing a complex multi-DOF motion. The robot weigted about 20 kg and was able to carry loads up to 15 kg. The travel speed was 20 cm/s.

Another successful six-legged robot was Functionoid. This walking robot, developed by Odetics, Inc., was able to lift the back of a 2200-pound truck. The robot could operate in full load for one hour using its system of motor drives before recharging the batteries.

By 1978, the Institut National des Sciences Apliques in France developed a semi-autonomous remote-controlled robot called Hilare. The vehicle was able to move at a constant speed along a straight line. Also, it could turn on its axis though a given angle. In addition the robot was able to move in a straight line for a distance less than a predetermined limit. Hilare was equipped with a 3-D vision system together with video camera. In order to detect closing-in obstacles and perform wall following, the robot utilized ultrasonic devices. Infrared beacons mounted in the room corners gave the absolute coordinates of the robot environment.

One of the first totally autonomous robots to exhibit a high level of sophistication was Robart-I. The robot, built at the Naval Post-graduate School, was supposed to randomly patrol the site and sense for fire, smoke, flooding, toxic gas, intrusion, etc. If any of the conditions was found, the robot had to issue the appropriate warning. Its motion routine was randomly chosen from a set of sixteen programmed routines that filled the gaps. Some of these routines would move the robot to a new vantage location, where it could elect to stop and resume in surveillance mode.

SCIMR (Self-Contained Independent Mobile Robot) was actually the first, completely autonomous robot. Built by University of Pennsylvania Moore School of

Electrical Engineering, it would travel down a hallway at a certain distance from the wall. This way, it could tell when it reached an intersection. SCIMR would then build a map of the hallways. Back then, bit map representations were hard to generate, a slow process and memory-inefficient. Therefore, the maps were graphed in a grid where the nodes were the interceptions and the lines the hallways.

In the mean time Australia National University constructed a robot for research operations called AMR. The robot was provided with an electronic system to perform real-time image acquisition, speech recognition and synthesis. Stepper motors individually controlled the TV camera's zoom, the aperture and focus. Ultrasonic sensors were mounted in the camera to complete the 3-D environment information for the computer. The results obtained from research with this robot led to the appearance of industrial transportation devices. The AMR architecture has been greatly implemented on industrial applications. Material handling in flexible manufacturing systems, mine finding, bomb detection and nuclear station surveying are only some examples of its current application. For more information on the evolution of mobile robotics, the reader may review [15, 20, 21].

1.1.2.2 Mobile Robot Navigation

The main objective for a mobile robot is to be able to navigate from one point to another and establish its new position with relative accuracy and a certain level of repeatability. In mobile robotics the area that studies this problem is known as robot positioning.

Despite many years of research in this area, there has been no complete solution method that is able to determine robot position. Instead, mobile robot designers have developed partial solutions to the different problems evolved from the robot operational environment. These position measurement problems may be divided in two main groups: relative position measurements (RPM) and absolute position measurements (APM).

The first group, RPM, represents those methods that derive the current robot position based on information of a previous position and the velocity profile through a known trajectory during a given period of time. This is a simple mathematical method known as dead reckoning.

Regardless of its complexity and constraints, a predefined robot path can always be divided into smaller, achievable trajectories i $(0< i <\propto)$ from point $A(x_i,y_i)$ to point $B(x_i,y_i)$.

Dead reckoning is used for navigation in most of the land-based mobile robotic systems [2]. One of the simplest implementation of dead reckoning is called odometry. This method measures the wheel rotation and steering angle using motion and velocity tracking sensors. Odometry is an inexpensive way to generate accurate position information during short periods of time. However, accumulation of very small errors over time, from consecutive measurements generates highly inaccurate position information. This is particularly important for steering control where the position error will increase proportionally with the distance traveled.

One obvious source for the accumulation of these errors is the wheel slippage. In addition, several other indirect reasons can cause reading errors. Physical problems such as unequal wheel diameter, inaccuracies in average wheel diameter, wheel misalignment and many others can systematically introduce position errors. Furthermore, traveling over uneven terrain, uneven floor texture, running over unexpected obstacles and a wheel rotating without having a point of contact with the floor are only a few other sources for odometry errors.

Many different alternatives can be implemented in order to reduce odometry errors and improve dead reckoning. One of them is to use an encoder trailer. The University of Michigan has implemented this approach by attaching a correcting mechanism to a Remotech Andros V tracked vehicle as shown in [2]. Due to track slippage, motor encoders accumulate large position errors. The encoder trailer will only measure the robot's actual linear motion.

Optical encoders are the most commonly used sensors in odometry-based navigation systems. In principle, an optical encoder consists of a focused light beam that aims at a matched photo-detector. The light ray is periodically interrupted by a coded pattern of dark/clear areas on the surface of an intermediate disk attached to the shaft of interest. The simple functioning principle and the generation of digital output make this sensor a very reliable, yet cost-effective, noise-free device.

There are two types of optical encoders. Incremental encoders measure rotational velocity from which relative position can be derived. The simplest encoder of this type is the single-channel tachometer encoder. This device is well suited for velocity feedback in medium- to high-speed control applications. A typical application for these devices is the driving system of mobile platforms. These sensors have two major drawbacks. First of all, they run into stability problems at low operational speed. Also, since they can not detect direction of rotation, they can not be used for position control applications.

These problems are overcome by adding a second, 90°-out-of-phase channel to the single-channel tachometer/encoder architecture. Then, this device becomes a phase-quadrature incremental encoder. This architecture allows for the rotational direction detection while increasing the resolution. In order to keep track of the complete revolutions for continuous rotation applications, most incremental encoders incorporate a third index channel.

The other type of optical encoder is called an absolute encoder. These devices are often used in slower-speed rotational applications as well as those applications where loss of reference position is not acceptable. A typical application for these devices is the steering system of mobile platforms where slow and infrequent rotations are required. Interface complexity and the associated high cost are potential disadvantages of absolute encoders. Each additional encoder track doubles its resolution, but it also increases the number of leads of the parallel data output and quadruples the encoder cost.

More demanding dead reckoning methods involve the use of two robots for reestablishing their position using each other as a relative landmark. These methods are known as Mutual Referencing, and Internal Position Error Correction. For detailed information on these methods can be found in [2].

The benefits of using Odometry and its different approaches outweigh its inherent problems. Odometry, combined with other techniques, can improve the system positioning reliability. For instance, this method can be applied in land-marked sites where the distance between them is short enough for the robot to find them with acceptable accuracy while navigating using Odometry. Then, the robot can recalibrate its global position using the landmark.

Often, information on landmarks or the possibility of placing them is not an option in hazardous environments. Here, many sensor systems fail to provide useful data. Therefore, a self-contained navigation system based on odometry that always provides an estimate of the robot's position may be the only suitable choice.

The term landmark introduces one of the methods grouped in the APM category. Landmarks are well-defined features in the environment that are easily recognized by the robot sensors. Therefore, landmark-based positioning methods require previous knowledge of the position and orientation of the marks. In general, the process of landmark-based positioning requires the acquisition of a vast amount of sensory information using special beacons. Then, the navigation system should search, detect and segment the landmarks contained in the collected information. Detecting and establishing a correspondence with the landmark is the most demanding task in the positioning process. This may be influenced by the accuracy of the initial estimate and the true position and orientation of the robot. Correspondence is not established until the previously stored map of the environment matches the sensed data. The stored map may be obtained from a CAD representation of the environment, sensor data collected by another robot or information about the environment collected by the robot through its own sensors. This technique is called map matching. Finally, the robot position is calculated using different methods such as triangulation and geometric shape.

Landmarks can be artificial or natural. Natural landmark-based positioning is best suited for highly structured sites such as hallways, manufacturing facilities, and hospital buildings. Here, reference points are environmental features that are meant for functions other than robot navigation. On the other hand, artificial landmark-based positioning methods use objects specially designed for robot navigation that need to be placed in the environment. For this purpose, special marks or bar codes may be placed at various locations for the platform to recognize and determine its position.

1.1.2.3 Path Planning

One of the most exciting topics in mobile robotics research is the study, development and implementation of robot path planning systems. These systems should handle a variety of operations performed under different environments, speeds, conditions and levels of reliability.

Sometimes, robots have to operate in known environments with a very limited number of static obstacles whose shape and some other important properties are known. In other cases, mobile robots have to work in closed, remote sites with a variety of dynamic and/or static obstacles (for instance, nuclear waste sites). Most of the time they are required to operate in real-time response mode. For such operations, speed and maneuverability become critical.

Many researchers have developed methods to address this issue for different robot architectures. One example of previous work in this area is the one by Vasseur and Pin in [30]. This paper presents a new navigation and path planning algorithm for car-like robots which allow for accurate and efficient maneuvering while remaining very fast and suitable for real-time applications.

When robots are not circular, precise maneuvering always implies working in the configuration space of the vehicle. Due to the complexity of this space, the path planning typically involves computer-intensive models and rarely allows for real-time applications.

Path planning involves analysis of the admissible configuration space. This means the calculation of all possible collision-free configurations of the robot. In very constrained areas, approach which take into consideration the shape and the kinematic constraints of the robot are the only feasible approaches for precise maneuver planning.

Previous works in this area have been based in a discrete robot orientation and a computed space configuration for each robot orientation. This way, once the configuration space has been built, it must still be searched for to find the possible path. Other approaches to the path planning problem have suggested to discretize the entire configuration space of the robot, including orientation and position, and perform a dynamic search with the number of maneuvers as the parameter to minimize. Both approaches have the same drawback: they have to trade off accuracy and computation time. These disadvantages make the real-time operation very difficult to achieve.

For path planning purposes, not only the kinematic models, but also the shape and size of the robot are important factors. A very common approach to solving the shape and size constraints is to grow the obstacles by a radius of the platform and assuming a point mobile platform to design the robot trajectory. This method is highly accurate for circular platforms. However, only a few platforms are designed with a circular shape as designers try to maximize carrying area and stability. For non-circular platforms, this technique can be accurately used if the platform can be modeled as a polygon.

The technique presented in [30] by Vasseura, Pina and Taylor is called convex cells. This work describes a new method to design the maneuvers required from a car-like robot to achieve two separate configurations in the work space while traveling in a collision-free trajectory and respecting the platform kinematic constraints. The method takes advantage of predefined trajectories to avoid time-consuming computer operations such as optimal path searching. The major difference between this method and the previous one is that partitioning the configuration space into convex cells allows for the selection of large arcs and straight-line trajectories as opposed to minute trajectories calculated one by one. The convex cell method assumes that data from the environment surrounding the robot is known from existing maps or data collected from scanning sensors. It also assumes that the work site can be modeled as a convex cell (i.e., a region in a plane bounded by line segments that form an angle no greater than 180°, measured form the inner side of the polygon). However, this initial assumption may become a drawback for its implementation in unknown environments.

Despite the wide research work available for this topic, navigation and path planning continues to be a field in which there exists many questions to be answered by further research. This is the case of the technique introduced by Khatib in [8] and known as the Potential Field Method (PFM). In the last few years, PFM has become very popular in obstacle avoidance algorithms. The reason for such popularity is its simplicity and elegance.

PFM is based on a differential equation that describes the robot and its surroundings in a combined system. The PFM method assumes imaginary forces acting on the robot. These imaginary forces influence the robot trajectory to the target. Obstacles, when "seen" by the sensors, are considered to exert repulsive forces on the mobile robot. On the other hand, the target exerts attractive forces on the vehicle. The vector sum of these forces determines the magnitude resultant force. The resultant force determines the direction and travel speed of the robot.

This method is very simple and effective in real-time applications. It can be quickly implemented and results can be obtained initially without requiring any refinement. However, four significant problems have been identified to be inherent to PFM regardless of the particular implementation. As explained by Koren and Borestain in [9], the occurrence of cyclic behavior is a very common problem with this technique. This is produced when the robot runs into a dead end such as a U-shaped corner. Here, a problem of local minimum has been encountered. The next problem with the PFM is that the robot is unable to pass between closely placed obstacles. Generally, this problem is found when the resultant force vector of the virtual repulsive forces from the obstacles points out to a direction away from the opening between the obstacles the robot is supposed to go through. The third problem commonly encountered when using this technique is the occurrence of robot trajectory oscillation in the presence of obstacles. This causes the robot to follow an unstable path and deviate form the optimum one. Finally, oscillation in narrow passages is a more severe problem. This occurs when the robot travels in narrow corridors receiving virtual repulsive forces from opposite sides. These opposite forces are the cause of the oscillations that may result in a collision. The problems mentioned before only manifest in high-speed mobile robots and therefore, low-speed applications are more suitable for the PFM path planning method.

Koren and Borenstein present a different approach to solving the problems encountered in the PFM method. Their method is called Virtual Force Field (VFF). This method can be reviewed in [10].

1.2 Motivation/Problem Description

The topic of fault tolerance was first introduced in the 1950s, mainly by the computer industry. An immeasurable amount of work in this area has been done since then. This previous work has been almost entirely dedicated to the design of fault-tolerant control systems and software development for robotic platforms and manipulators. The sections above are only a brief review of the vast research results developed in fault tolerance, mobile platforms and navigation strategies. However, it should be noted that only a few researchers have studied the kinematic and dynamic effects of the design and implementation of hardware redundancy at different levels in robotic manipulator systems. In fact, there is a lack of research work in the mechanical design of fault-tolerant mobile platforms. That is, mobile platforms provided with the necessary kinematic and hardware redundancy to undergo a failure of their main hardware devices. The lack of previous work in this area makes this research work significant and unique.

The main objective of this thesis is to introduce and analyze general schemes toward the implementation of fault-tolerant system architectures through hardware redundancy. In particular, some of these schemes are incorporated in the design of a general-purpose mobile platform. This work will be pursued in three main phases. First, a literature research is performed. This includes basic background information on principles, concepts and philosophy of the FT system design. Fault tolerance capacity measurement methods, and FT design schemes formerly developed and implemented are also included in this research. In addition, previous practical applications of FT design in robotics are reviewed in this phase. The previous work in mobile platform design and navigation techniques are also reviewed and documented since the present work addresses design and control of fault-tolerant mobile platforms.

The second phase of this work is dedicated to presenting different alternatives to the design of a Fault-Tolerant mobile platform. This includes the kinematic model of the system. Here, the mechanical design of the actuation and steering system of the platform are described in detail. Fault avoidance, fault masking and fault tolerance techniques described in the first phase are taken into account for this purpose.

The third phase includes the construction of an operational prototype able to perform basic mobile-robot operations while undergoing component failures. After a description of the design and construction of the prototype platform, the system response to an experimental test and results are presented. Final comments and further research work that need to be accomplished are also discussed.

1.3 Summary

In this chapter, a literature review on mobile platforms and fault tolerance has been presented. In the first section, previous work that addresses the issue of kinematic redundancy and its effect on the fault tolerance capacity of robotic systems is introduced. Furthermore, a fault tolerance capacity measure method, developed by Tosunoglu and Monteverde is briefly explained. Previous work on fault-tolerant robot design has also been addressed in this section.

Section 1.2 begins with a report on the evolution of the autonomous mobile robots. Mobile robots, developed over time by the most prestigious robotic research laboratories around the world, are introduced. The rest of this section is dedicated to the issues of mobile robot navigation and path planning, which is an important area as it relates to the control of mobile platforms.

Finally, section 1.3 explains the problem being addressed in this thesis. A great amount of work has been done on the design of control systems capable of dealing with system faults and errors. In contrast, there is a lack of research work on the design of fault-tolerant mobile systems, which makes this thesis work interesting and unique.

Chapter 2. Fault-Tolerant System Design

2.1 Principles, Concepts and Definitions

In this chapter, several definitions will be reviewed to have a better understanding of the concept of the FT system design. First of all, a mechanical system is said to be *fault-tolerant* if it has been designed to continue its predefined operation after the occurrence of a failure with a minimum degradation in its performance. The term *Failure* represents another important definition. In short, *Failure* is the indirect consequence of the occurrence of a *fault.*

Faulty conditions are produced by physical defects or flaws within the system hardware and/or software. *Faults* can be generated by unmet design specs during the design process. A poor judgement while selecting system components could also lead to system failures. Other factors that may contribute to the occurrence of *faulty conditions* are the external conditions the system operates. High levels of radiation, extremely high or low temperatures and electromagnetic fields as well as operator mistakes are among these adverse external factors. Finally, component defects produced during the manufacturing process and wear may also generate faulty conditions.

In general, faulty condition of a component produces an *error.* In turn, this error produces *failure* of one or more of the components or even the entire system.

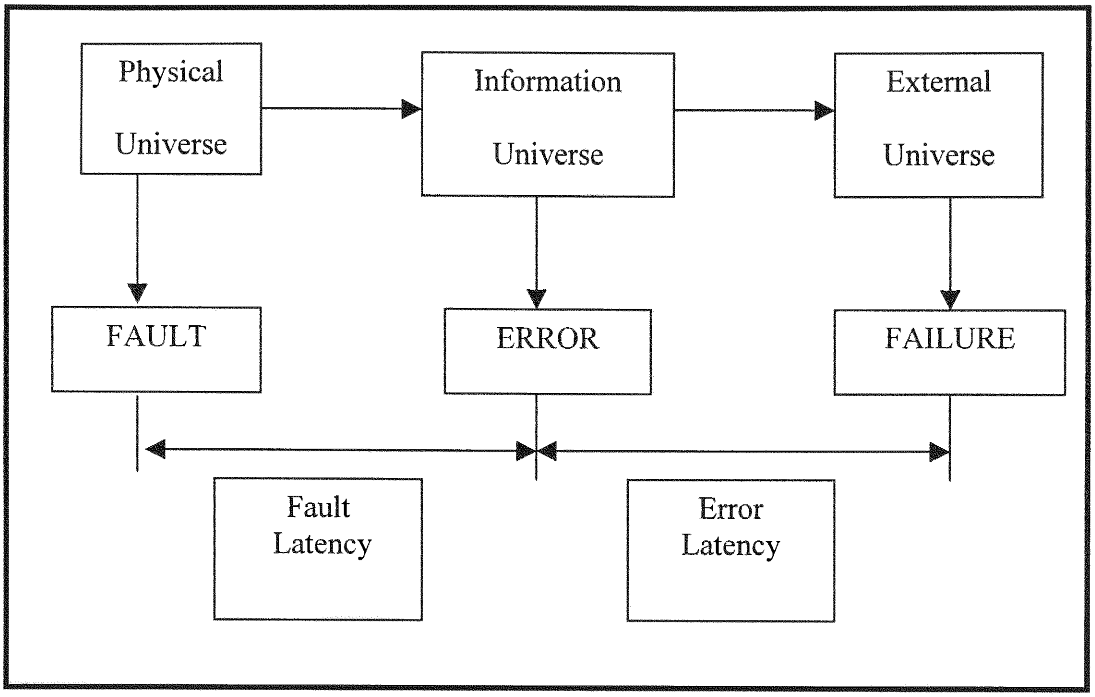

Figure 2.1. Chronological Order of Fault and Failure.

A system, after one of its components become defective, undergoes a period of operation until an *error* is produced. This time period is called "*Fault Latency*". Then, the *error* propagates throughout the system for a time period until the failure occurs. This time period is called "*Error Latency*". Figure 2.1.1 below places these definitions in chronological order.

The main reason for introducing *FT* to the system architecture is the need for higher levels of reliability. That is, the aim is to increase the probability of the system to be fully operational during a desired period of time $[t_o, t_i]$ provided that the system is fully operational at the beginning of the time interval (t_o). *Reliability* $R(t_o, t_i)$ is an obligated definition to take into account when time intervals are long and even the smallest period of incorrect system performance is unacceptable.

Availability $A(t)$ and $R(t_o, t_i)$ definitions are very closely related. $A(t)$ is defined as the probability of a system of being available for operation at a desired instant t. Obviously, the higher the parameter $A(t)$ value, the greater the chances for the system to have high reliability.

Finally, after a system failure, only part of its functions will emerge unharmed. Those operational functions should be evaluated to measure their *level of performance* (L). In doing so, the system capacity to operate at a level L or above at time t can be measured. The parameter in charge of this evaluation is called *performability* $P(L, t)$.

2.2 Design Philosophy

In general, the design process is a sequence of dilemmas ranging from confusion to despair to discovery. This process may be tedious and iterative. One of the traditional interpretation of the design process is often presented in six steps (see Figure 2.2).

- Recognition of a need
- Definition of the problem
- Synthesis
- Analysis and optimization
- Presentation.

2.2.1 System Design Overview

After the need for a design and the problem have been defined, all ideas, requirements and concepts are put together in a process known as synthesis. In this phase, a mechanical system that will perform the desired task takes its initial shape and mechanical architecture. Then, in the analysis and optimization step, the designer uses mathematical models to determine the materials, sizes and other parameters required to satisfy the defined need. This step may involve highly iterative processes that require good mathematical skills. The evaluation phase deals with the evaluation of the entire system against the original system specs given in the problem statement. This process includes prototype building and testing. In the past, testing often meant destructive testing. Prototypes were constructed just to be destroyed in experiments such as drop tests. Consequently, a lot of time, effort and money were spent in such a design phase. The current development of computer systems and the Computer Aided Design, Manufacturing and Engineering (CAD/CAM/CAE) technology has made the synthesis and analysis phases faster and more efficient. The use of rapid prototyping, finite element analysis and 3-D parametric solid modeling technology play a very important role at this stage as well as for the analysis and optimization phase. These tools greatly reduce the time-to-market period of a product.

Finally, the design presentation stage involves communicating and often "selling" it through different means. Traditionally, drawings and prototypes were the most common way of presenting the product. Today, computer technology and the internet have become in very fast and economical ways of communicating the design features.

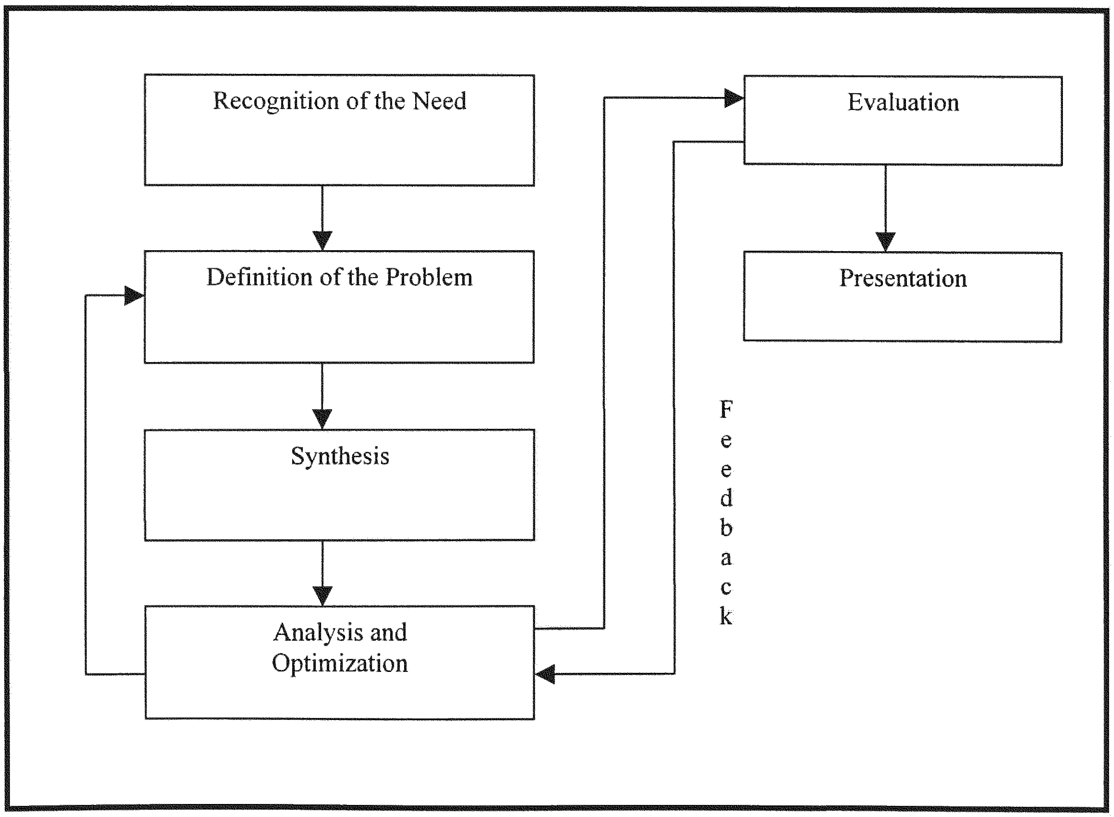

Figure 2.2. Traditional Design Process

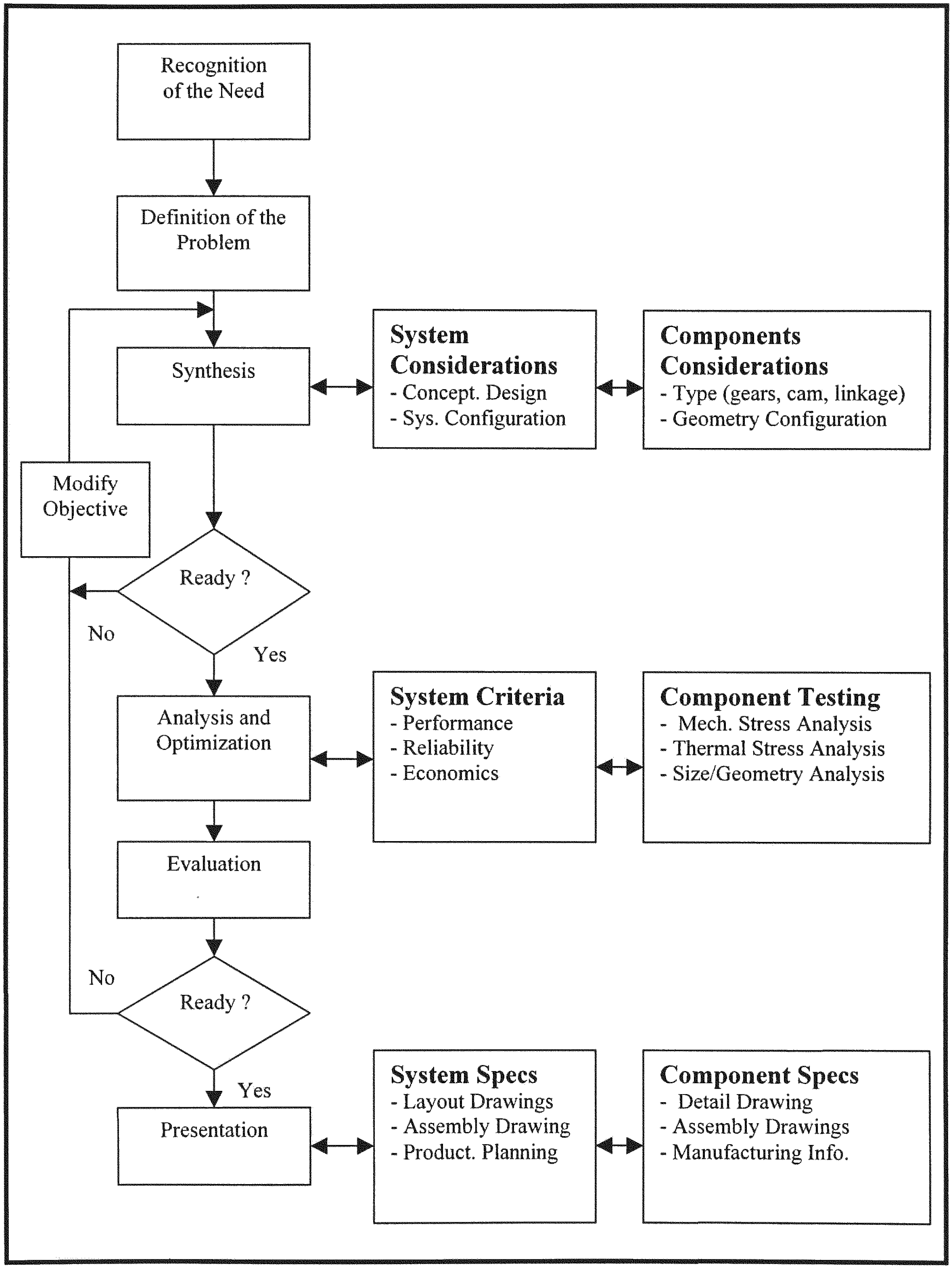

Figure 2.3. Current Design Process

The figure above shows the design phases and the effects of the current CAD/CAM/CAE development in this process.

2.2.2 System Design Techniques

A mechanical system can be provided with fault-tolerant capabilities by introducing hardware redundancy at different levels (i.e., component, module, subsystem and system levels) [16, 24]. In addition, this hardware redundancy may be introduced at each level in four different domains. These domains are listed as:

- Software Domain
- Information Domain
- Time Domain
- Hardware Domain

Software is crucial for the operation of modern, robust automated systems. Indeed, software constitutes the major proportion of complexity of most robotic systems. Software techniques are largely implemented in robotics to control the system, simulate its motion and environment. Fault detection and recovery in the hardware domain is also controlled in the software domain. Two main methods have been used to structure Fault-Tolerant software systems. These methods are known as *Recovery Block Scheme* (*RBS*) and *N-Version Programming Schemes* (*NVPS*) (see [1]). Both methods are developed under the assumption that despite the use of several fault prevention techniques, a complex software system will always contain residual faults when it is run. These methods, however, can not assure a total protection against faults.

The *RBS* for providing *FT* to sequential programs was introduced by Horning et al. in 1974. The *RBS* method is based on the addition of decision making software block (also called modules). This addition may be seen as an increase of the software complexity and therefore, a decrease rather than an increase of the overall reliability of the software. However, it is not the case because the modules are independent from each other. Consequently, each module is able to start its execution at the same state due to the backward error recovery provided. Thus, the strength of this scheme derives from use of modules that do not contain common faults.

NVPS technique is based on the development of different versions of the program each independently designed to satisfy a common task. When the program is executed, the individual outputs are compared to each other. The system output will be that one provided by the majority of agreeing outputs. Control over the *NVPS* is provided by the driver program. This program is responsible for running each of the versions, waiting for the different versions to finish the execution and perform the comparison of the *N* versions. This technique requires synchronization of actions between the driver and the versions as well as automatic execution and shared input space. More detailed information the reader may review [1].

Redundancy in the time domain is basically used to detect transient as well as permanent error in the system. Here, redundant information is added to the data stream to be used as double check for faults. These techniques are closely related to those of software domain and computer hardware reliability. The use of software and time domain redundancies is often a more economical alternative than hardware redundancy. They can reduce the hardware redundancy required for the system to achieve predicted levels of reliability.

Introduction of redundancy in the hardware domain is the most suited for implementation in mechanical design of Fault-Tolerant system. It generally includes three common approaches. The *fault avoidance* is regarded as preliminary step. Here, designers make sure that more reliable parts for the task are selected. In doing so, several reliability tests may be performed to determine the extent to which a critical component may influence
the overall reliability *R*(*t*) of the system. For example, as explained in [1], after a "burn in" period, electronic components usually fail at a constant rate that is modeled by

$$R(t) = e^{-\lambda_t} \tag{2.1}$$

where

λ = Failure rate sum of the system components

t = Desired failure-free time period

Design and methodology review as well as quality control inspections are also performed at this design stage.

The second approach is the introduction of *passive hardware redundancy techniques* such as fault detection and fault masking methods to prevent errors from propagating throughout the entire system. These techniques are called passive because a scheme is developed such that the faulty unit continues operational as its output is isolated and therefore it causes no effect on the system output. Passive hardware techniques do not require fault detection, location and system reconfiguration phases to provide *FT* capabilities.

Several fault detection and fault masking schemes have been developed in the past. Voting systems are good examples of this approach. They are useful in systems with high degree of parallel hardware redundancy. In general, the system would have an odd number (at least 3) of the same devices (force sensors, encoders, etc.). Decisions are made based on simple logic: if two inputs agree and the third is substantially different above a threshold value, the latter input is disregarded. The layout of a single voter system is shown in the figure below.

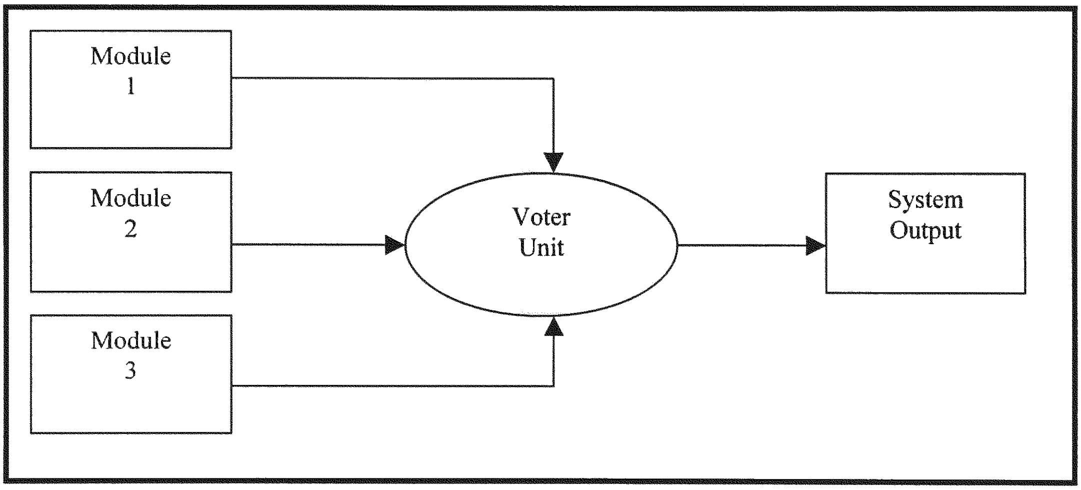

Figure 2.4. Simple Voter System

The voter scheme shown above has a single point of failure in the voter unit. This means that if the voter unit fails, the system output will be a faulty one. This is the reason why voting schemes are integrated in other passive schemes such as *Triple Modular* (*TMR*) and *N-Modular* redundancy schemes. *TMR* overcomes the susceptibility of a single voter to failure.

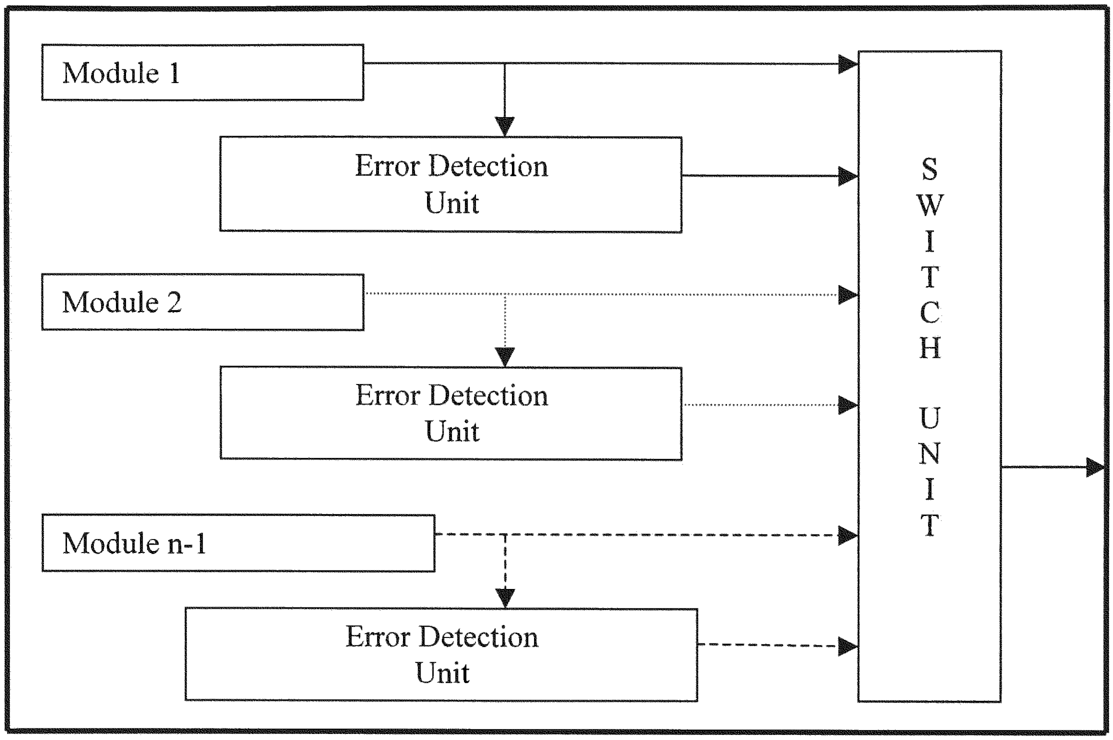

Figure 2.5. Standby with Spearing Scheme

The third design approach is to introduce *error containment/recovery* capabilities to the system. This is achieved by applying active *hardware redundancy techniques.* Unlike *passive hardware redundancy schemes*, this approach gives the system the ability to detect error, identify the faulty component(s) and recover the system's normal functioning through reconfiguration. However, this technique does not attempt to prevent faults from producing errors.

Standby with sparing is one of the active hardware redundancy techniques. It uses one of the *n* modules to produce system output. The remaining *n-1* modules are used as spare units. When an error is detected, the faulty unit is identified by the error detection module. Then, it is disabled by the switch mechanism. One of the spare units is finally put to work as the system goes through a recovery phase. This scheme is found in two modes: *hot standby* and *cold standby* modes. In the first mode, all modules are powered at once including the spares. This provides the system with a faster response and reconfiguration period. However, this operation mode is not suitable for applications in which power source is limited due to the high power consumption. This scheme is suitable for systems that require high precision such as robots and aircraft. Figure 2.5 shows the standby with spearing fault-tolerant scheme.

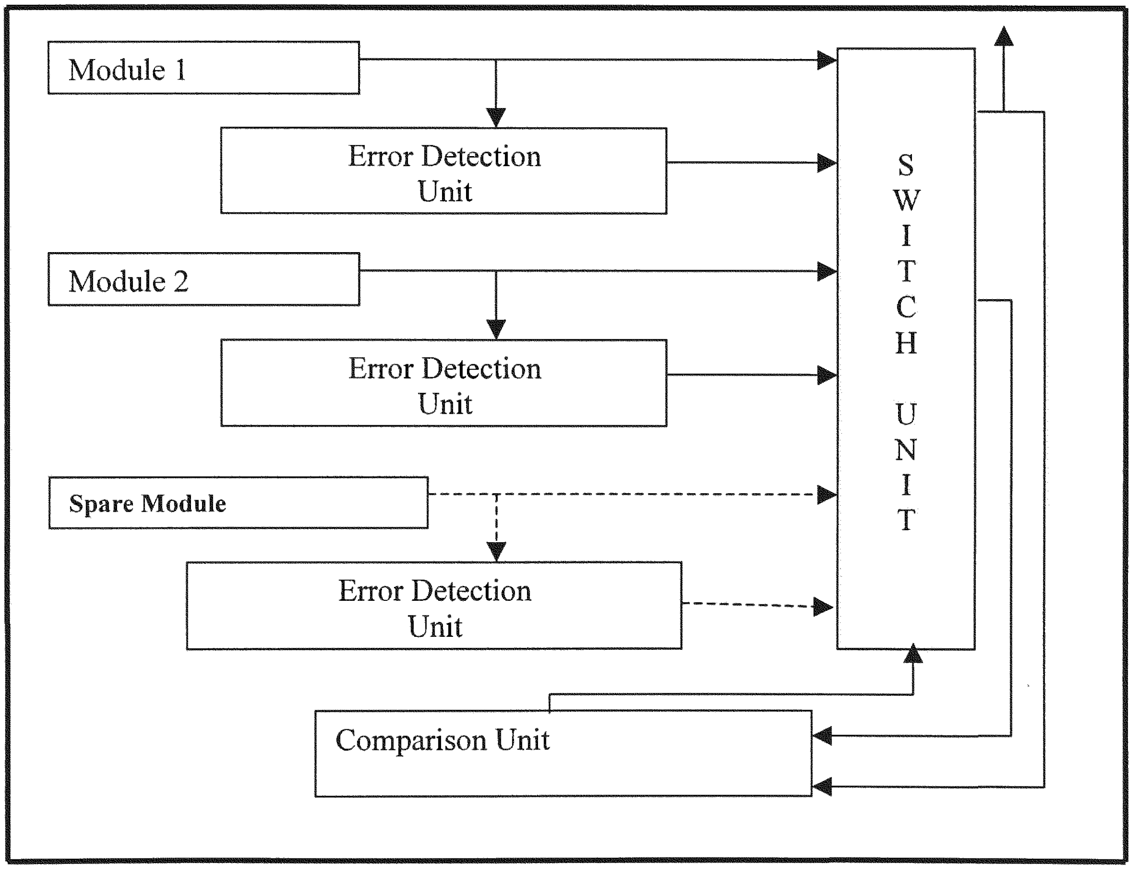

Figure 2.6. Spare-and-a-Pair Scheme

Unlike this mode, in the cold standby sparing mode, the spare modules are not powered until an error is detected. This provides slower system response. Here, the power consumption is lower than the former mode. Typically, space applications are most suitable for cold standby technique due to scarcity of power.

The *Spare-and-a-pare* scheme shown in Figure 2.6 above is another active hardware redundancy technique. Here, always two modules are required to work at a time. The system output is generated from one of the two module outputs. Then, the outputs are compared in a comparison unit. If an agreement is reached, the system remains unchanged. Otherwise, the module initially assumed to be fully functional is disabled and the spare module is enabled.

Hybrid hardware redundancy techniques are a combination of the attractive features of the techniques previously explained. Therefore, they are another way of providing fault tolerance to the system design. Here, fault-masking schemes are utilized to prevent errors from propagating throughout the system. Fault detection, location and recovery techniques are also used to reconfigure the system in case a fault occurs.

One of these techniques is the *N-Modular redundancy with spares*. Here, n active modules and m spare modules are used. The technique incorporates a voting system that decides upon the system output. The system output is then used to detect the faulty module in the disagreement detection unit. Then, this unit will disable the faulty unit and enable one of the spares.

Self-Purging (*S-P*) is another hybrid scheme that similar to the *N-Modular* redundancy technique explained above. The only difference is that in *S-P* schemes all modules are used at once. Then, the voted output is used to compare the different module outputs. The disagreeing module is finally disabled.

2.3 Applications

Generally, robotic systems used in operations under extreme environmental conditions are provided with a high degree of autonomy. Often, they are required to travel on unknown surfaces and around obstacles of unknown shape. These uncertainties would be easily overcome by a human being. However, for a robot, operations like this would mean the integration of quite a complicated system. For example, it would have to have some kind of a mobile platform on which one or more manipulators as well as tools and materials can be transported. The platform should be provided with a highly maneuverable steering system. This is done through actuation of several degrees of freedom for each wheel.

Another complicated task is the design of a navigation system capable of detecting and avoiding the obstacles encountered. Here, the use of position, velocity and vision sensors becomes absolutely indispensable. In order for a robotic system to perform the fairly simple task of avoiding an obstacle, a great number of components and complicated system designs are needed. However, despite the complexity of mobile robotic systems and due to the broad application field for these robotic systems, researchers are willing to become involved in this type of work.

One of the most suitable fields for the use of mobile platform is the hazardous waste management. These operations are performed in compliance with the Department of Energy (DOE) and the Environmental Protection Agency's (EPA) regulations. Under these regulations, several tasks may be automated through the use of robotic systems. For instance, at the start of a hazardous waste clean up operation, the first step a robotic system is suitable for is to evaluate the degree of contamination. This information is gathered through sensors and Data Acquisition Systems (DAQ). The information is analyzed to determine the level of personal safety required based on EPA regulations. After these requirements have been established, the robot/mobile platform may continue to monitor the environment to rapidly detect any emission of harmful gases.

Mobile robots, provided with sensor systems of different kinds can inspect the sites and collect information on the nature of the waste, the media affected and the zones for potential exposure. In addition, information on the site topography, structure and storage is collected.

Usually, site conditions are an indication of soil contamination. In order to confirm and measure this contamination, either several real time instruments or rapid turnaround analytical techniques are used. In this case, a mobile platform could perfectly do the job. Other applications also suitable for mobile robots are soil, groundwater and water surface characterization and drum removal.

Robotic platform can also provide electric and hydraulic power as well as logistic support to the clean up operation. Platform can carry a variety of tools, instruments and DAQ to perform basic, utilitarian tasks. These basic tasks help reduce the labor required, increase the job efficiency and reduce the probability for accident and exposure (i.e. conservation of human resources). In addition, mobile platforms can be used to provide the cleanup crew with enhanced equipment for first aid.

Currently, all mobile platforms at DOE sites have tethers so that the platform can be fished back in case of a component failure. These tethers however, limit the platform maneuverability of the platforms. In cluttered environments, when the distance traveled increases, the tethers run into the problem of getting caught with equipment or parts. By removing the tethers and simultaneously adding fault tolerance capabilities (hence, improving reliability), overall system performance and utility is enhanced with the proposed work on fault-tolerant design of mobile platform.

Chapter 3. Mechanical Design

3.1 Introduction

Generally, robotic systems are used in operations under extreme environmental conditions. In such cases, they are provided with a high degree of autonomy. Often, they are required to travel on unknown surfaces and around obstacles of an unknown shape. These uncertainties would be easily overcome by a human being. However, for a robot, operations like this would mean the integration of a quite complicated system. For example, it would have to have some kind of a mobile platform on which one or more manipulators as well as tools and materials can be transported. The platform should be provided with a highly maneuverable steering system.

Another complicated task is the design of a navigation system capable of detecting and avoiding the obstacles encountered. Here, the use of position, velocity and vision sensors is absolutely indispensable. Consequently, in order for a robotic system to perform the simple task of avoiding an obstacle, a great number of components and complicated design is needed.

When operating in hazardous environments, a partial or total failure of mechanical components of a robotic system can be disastrous for the end result of the task. This may represent an unaffordable loss of time and money. Furthermore, trying to retrieve a robot from such an environment can be dangerous or even impossible for human beings. Therefore, safety and reliability have to be primary concerns for designers. Introducing fault tolerance in to the platform design would greatly increase its reliability.

In general, this chapter introduces different fault-tolerant mobile platform design alternatives. In addition, different techniques, design philosophy, and concepts commonly used in the field of robotics and fault-tolerant system design will be integrated into a final, modular design. The theoretical background for this work is deeply explained in former chapters. Specifically, the attention will be focused on providing the actuation of the driving and steering systems of a mobile platform with some degree of fault tolerance capabilities. The last sections of this chapter are dedicated to providing important information about the design process of key components of the mobile platform such as actuators, sensors and power supply.

3.2 Conceptual Design and Alternatives

The different design alternatives that have been considered in this work were developed around the idea of a *general-purpose mobile platform.* This general-purpose equipment could be seen as a car-like vehicle with four wheels and a base. The size, shape, and capability of the platform may vary according to the task to be performed. In order to provide the required hardware to give fault tolerance capability, the basic architecture is divided into three major systems: the drive system, the steering system and, the base or chassis.

3.2.1 Basic Platform

The initial design in Figure 3.1 shows a basic mechanical architecture of a car-like vehicle with four wheels. The base is designed to support the weight of the different equipment, materials, tools and/or manipulators to be carried. Also, it may carry most of the navigation sensors and feedback electronic devices. Since this work focuses on introducing fault tolerance in the actuation systems, the base structure is not shown in Figure 3.1.

The drive system powers the rear wheels of the platform. The *differential drive system* is composed of two brushless actuator-brake-clutch mechanisms that transmit torque from the actuator(s) to the rear wheels through a differential mechanism as demonstrated in Figure 3.2. The dual actuator-brake-clutch (clutch not shown in Figure 3.2) provides fault tolerant at joint level to increase the overall system reliability. This mechanism gives the driving system the capability to continue to operate after a failure on any of its components. For example, if any error is detected from one of the while

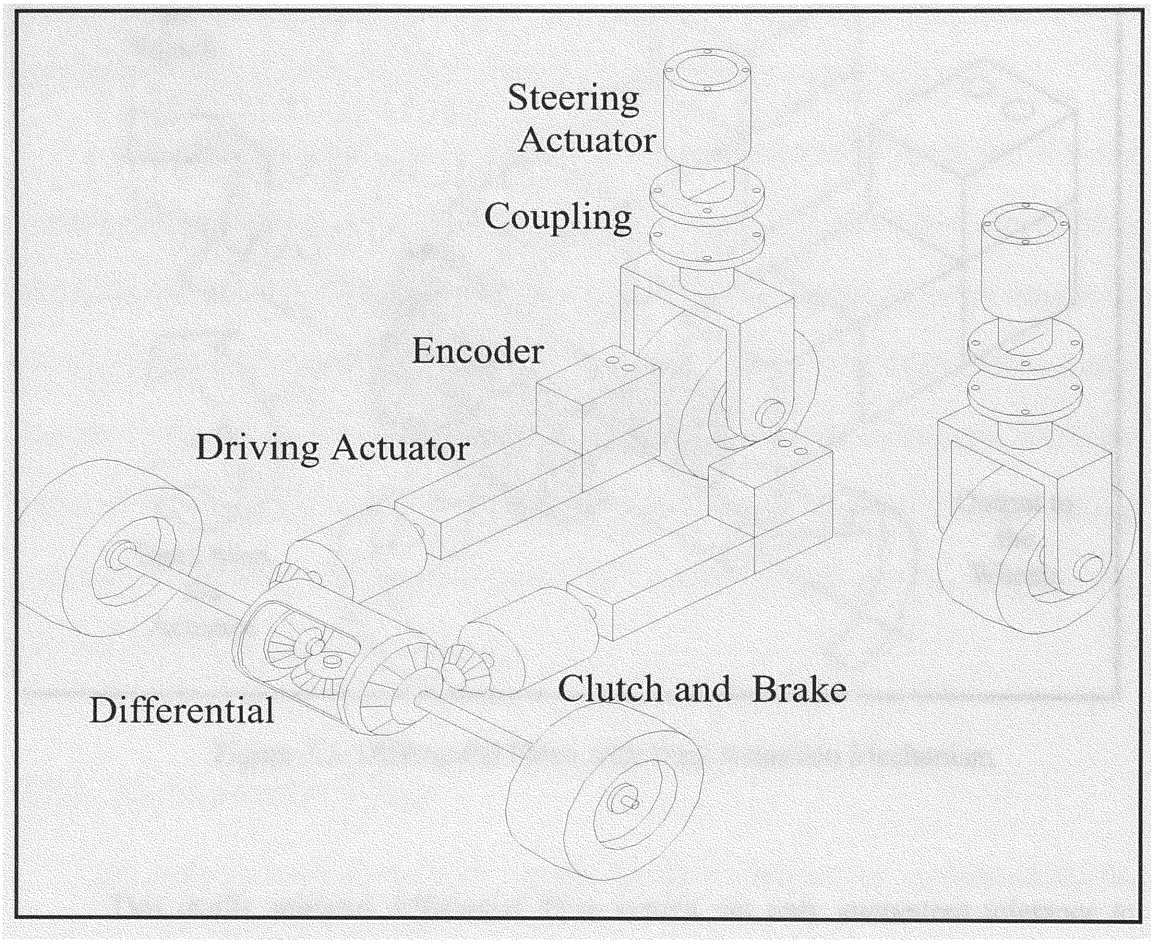

Figure 3.1. Basic Mechanical Architecture of a Car-Like Mobile Platform

generating torque, this subsystem can be disconnected from the entire system by releasing the clutch. Then, the operation may resume after actuating the other subsystem by engaging the clutch. In this case, the entire system will continue to perform its task under the same torque requirements. Previous work concerning this type of mechanism can be found in [16] and [23].

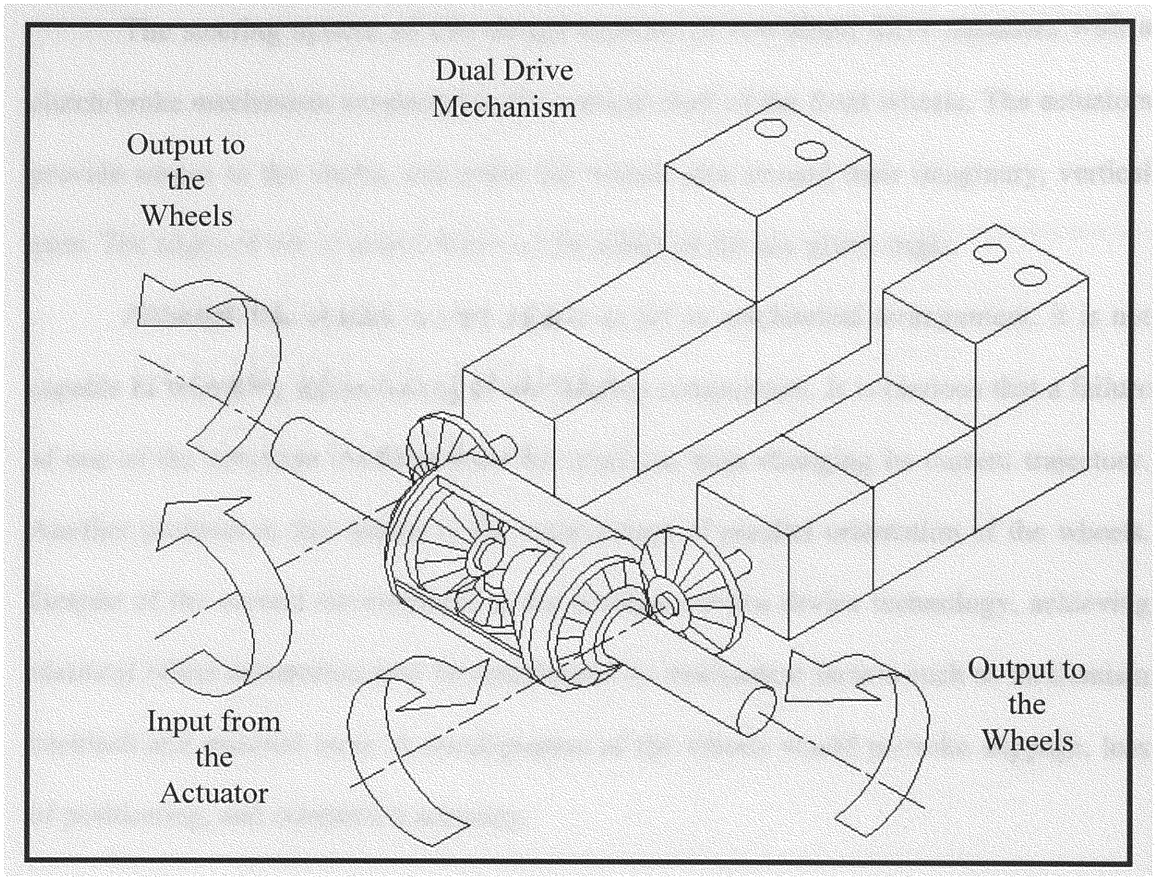

Figure 3.2. Differential Drive with Dual Actuation Mechanism

This dually actuated differential drive system not only guarantees tolerance to failure of any of its individual components but also increases the load-carrying capacity of the platform. In addition, backlash of the differential mechanism at starting motion can be corrected when accurate positioning is to be achieved. This can be done by simultaneously providing a large torque required to move the platform through one subsystem and a small torque for a very short period of time, in the reverse sense with the other subsystem.

The steering system of this design consists of two direct drive actuators with a clutch/brake mechanism connected to the vertical shaft of the front wheels. The actuators provide torque to the shafts, and make the wheels spin around their imaginary, vertical axes. The required wheel orientation may be achieved for any given angle.

Although this system is very simple as far as mechanical arrangement, it is not capable of tolerating failure in any of the steering components. It is obvious that a failure of one of the actuators would prevent the platform from changing its current trajectory. Another problem in this design is the achievement of parallel orientation of the wheels. Despite of the current development in the position control device technology, achieving identical wheel orientation may be constrained by mechanical factors such as mechanism backlash and material wear. A misalignment of the wheels would provoke slippage, loss of positioning, and orientation accuracy.

3.2.2 An Improved Design

In order to improve the initial design, the above issues were considered for the second design alternative. In this one, the dually actuated differential drive system was not modified because the original design had a simple, effective configuration.

The steering system, on the other hand, was redesigned as shown in Figure 3.3. Here, spur gears are placed at the end of the vertical shaft of the front wheels to form what is known as *synchro drive steering system*. The shaft is driven by a clutch/actuator mechanism similar to those used to drive the platform. Then, a third, idler spur gear interconnects the shafts such that the torque is simultaneously transmitted to the vertical shafts. In doing so, a parallel alignment of the wheels is guaranteed. Therefore, the platform can follow the desired trajectory smoothly, with minimum slippage. Now, up to three actuators can be used for steering.

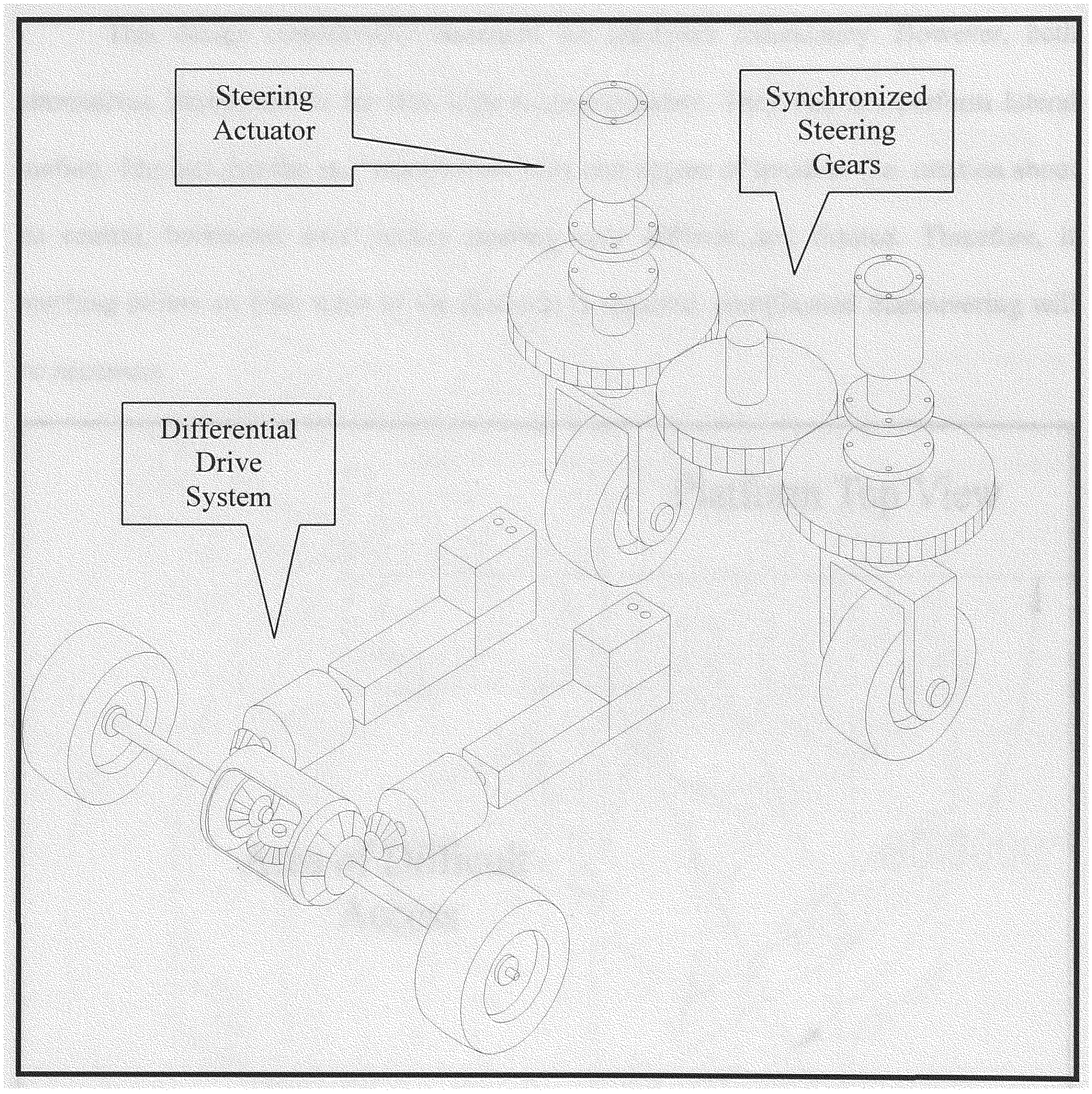

Figure 3.3. Dually Actuated Differential Drive and Synchronized Steering

This hardware redundancy allows for the implementation of a *Spare-and-a-Pair* Fault-Tolerant Scheme (see Chapter II). Steering can now be performed with one or two actuators while the other acts as a backup in case of failure.

This design considerably increased the hardware redundancy. However, both alternatives introduced so far lack high maneuverability. They can not perform lateral motion. The fact that the rear wheels have only one degree of freedom (i.e. rotation about its central, horizontal axis) makes steering very difficult and limited. Therefore, if reaching points on both sides of the platform is required, complicated maneuvering will be necessary.

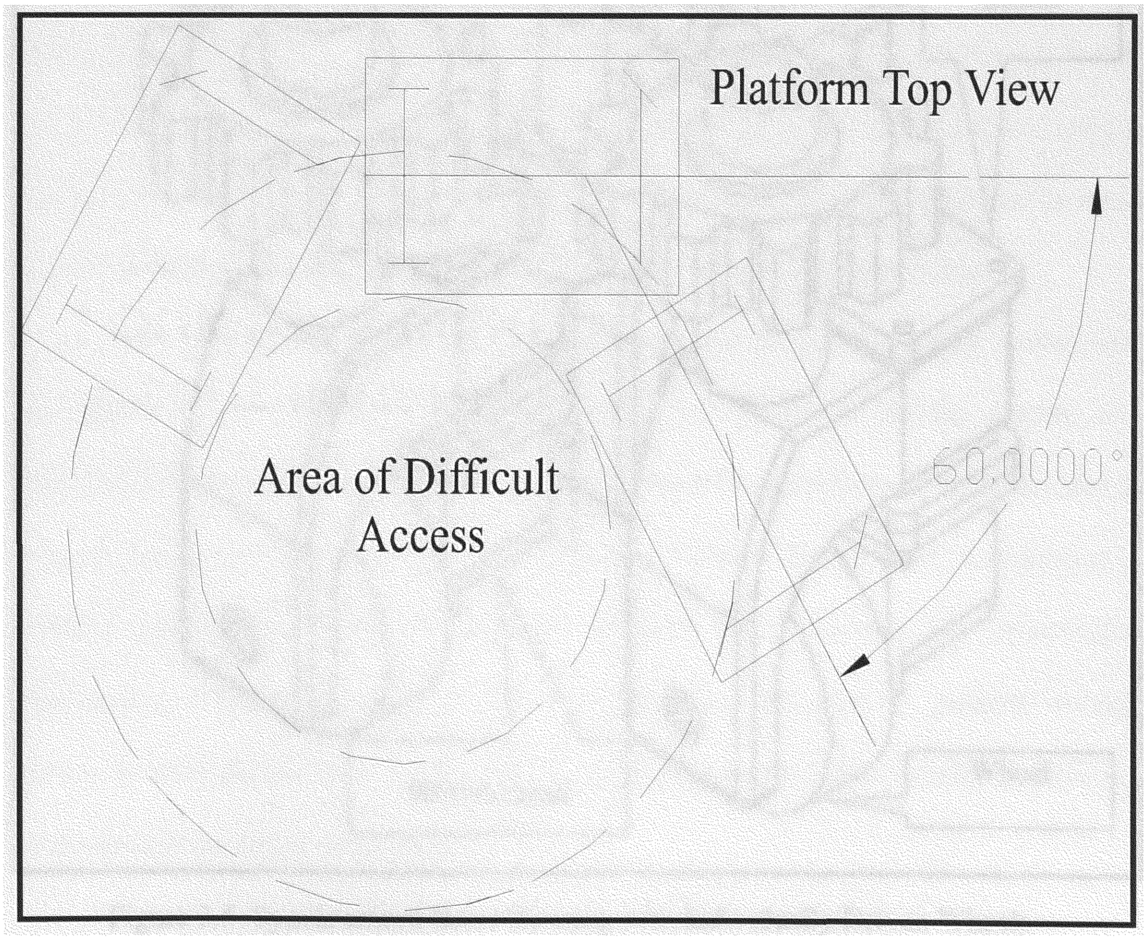

Figure 3.4. Differential Drive Provides Poor Maneuverability in Reduced Space

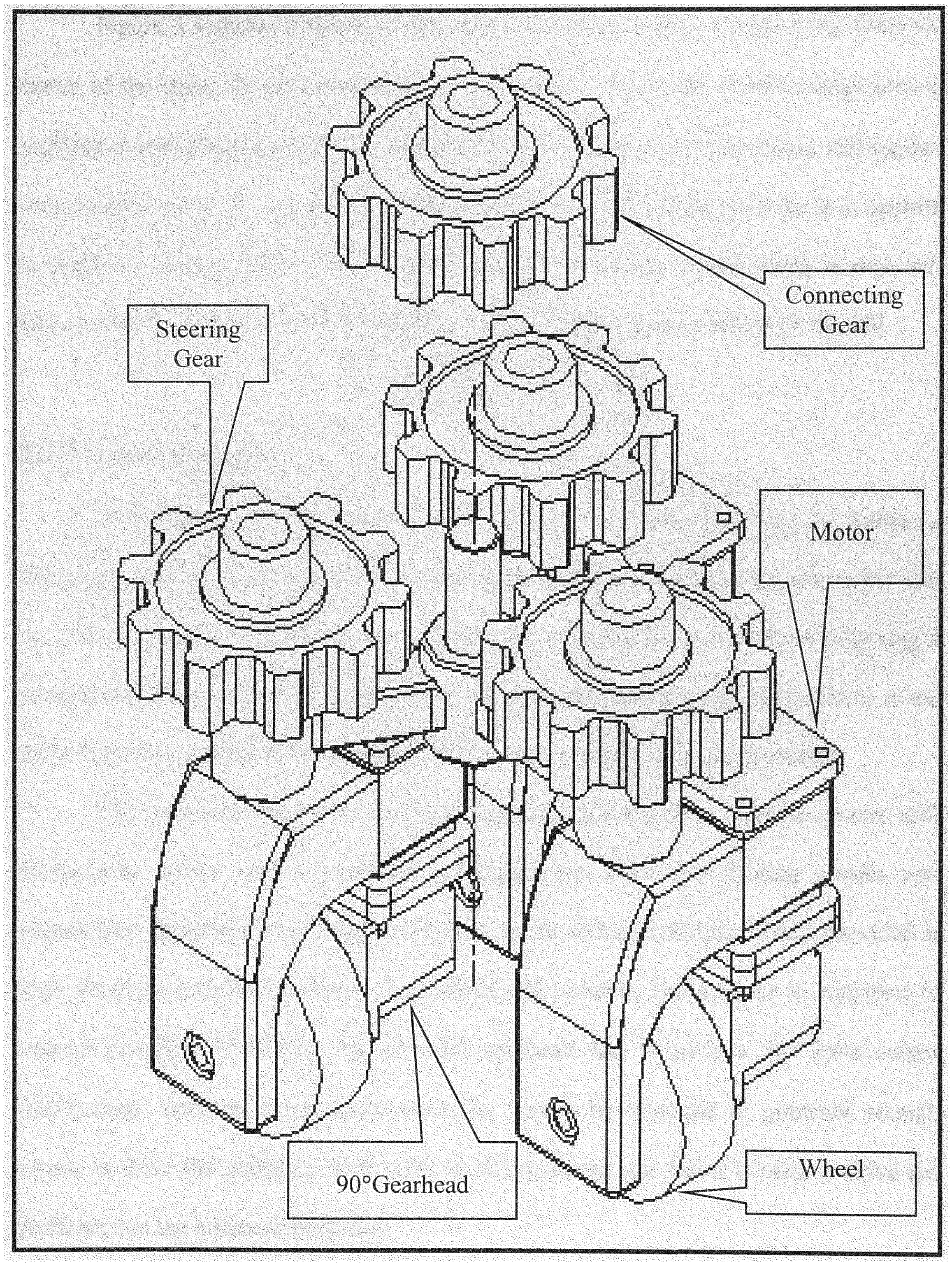

Figure 3.5. Synchronized Drive Steering with Individually Driven Wheels

Figure 3.4 shows a sketch of the platform turning around a point away from the center of the base. It can be noticed that even with a sharp turn of 60° a large area is required to turn about a particular point. Reaching interior points of the circle will require extra maneuvering. This may be the biggest design drawback if the platform is to operate in highly constrained sites where frequent obstacle avoidance maneuvering is required. Maneuverability issues have been considered in the past in works such as [9, 11, 30].

3.2.3 Final Design

After analyzing the maneuverability issue, it became necessary to follow a different approach to provide all the wheels with a steering degree of freedom such that the platform is able to move between any two points in the horizontal plane following a straight trajectory. If obstacles are found in the path, the platform should be able to avoid them following a chain of small, straight trajectories until the obstacle is cleared.

The modifications led to the implementation *synchro drive steering system with individually driven wheels* as shown in Figure 3.5. Here, the driving system was significantly modified. The torque transmitted by the differential drive is now provided at each wheel by brushless actuators, a gearhead and a clutch. The actuator is supported in vertical position. Therefore, the intended gearhead has to have a 90° input-output relationship. Each motor/gearhead assembly should be designed to generate enough torque to drive the platform. With such an arrangement, one motor is used to drive the platform and the others as back-ups.

Brushless actuators rather than brush dc servomotors are used in this design. Brushes are subject to wear due to the friction produced by their contact with the rotor surface. This problem is not present in brushless motors because they do not use brushes.

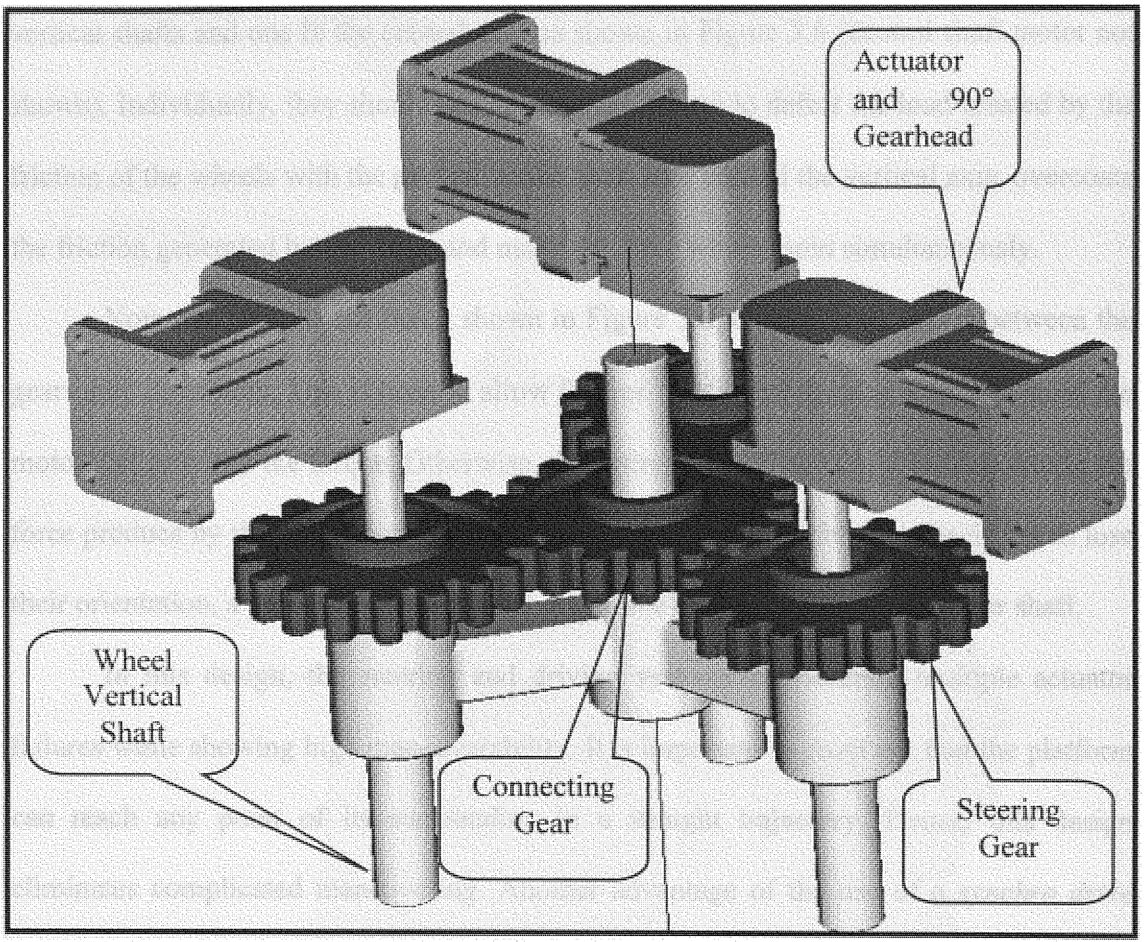

Figure 3.6. Synchronized Drive Steering System. Central Motor not Shown.

The selection of more reliable components (i.e., brushless servomotors) complies with the fault avoidance concept formerly mentioned in section "Design Philosophy". This design is more compact and eliminates the need for a differential mechanism.

In order to achieve a simultaneous, parallel orientation of the platform, the vertical shaft of the wheels are connected through a spur gear set (i.e. synchro drive steering mechanism). In a three-wheel platform, up to four actuator-clutch mechanisms may be in charge of the steering. One motor may be connected to each of the wheels vertical shafts and one to the central gear as shown in Figure 3.6 (central shaft motor not shown). Individually, they should produce enough torque to defeat the load created by the friction of the wheels with the ground, make them spin around the vertical axis, overcome the friction generated by the gears and make the three wheels spin simultaneously.

Note that, despite not being shown in Figure 3.6, clutches are placed between the gears and the motors. This is done to allow for free rotation of the shafts regardless of the motor that provides the torque. Otherwise, the entire mechanism would lock up due to the force produce by the magnetic field of the idle motors. In order for the wheels not to lose their orientation, a break should be placed at the free end of the connecting gear shaft.

In this design, the steering and driving systems can tolerate multiple actuator failures while showing high maneuverability. It is important to point out that the platform can reach any point of its surrounding in a straight trajectory. Hence, this design eliminates complicated maneuvering. Another advantage of the use of a *synchro drive steering mechanism* is that slippage is reduced. This improves the accuracy of the sensors information for *heading* and dead reckoning when it comes to navigation and position calculations.

Due to the high degree of redundancy, different hardware redundancy schemes presented in Chapter II can be implemented depending on the needs of a given application. For example, if the system is set up such that all actuators are used at once to

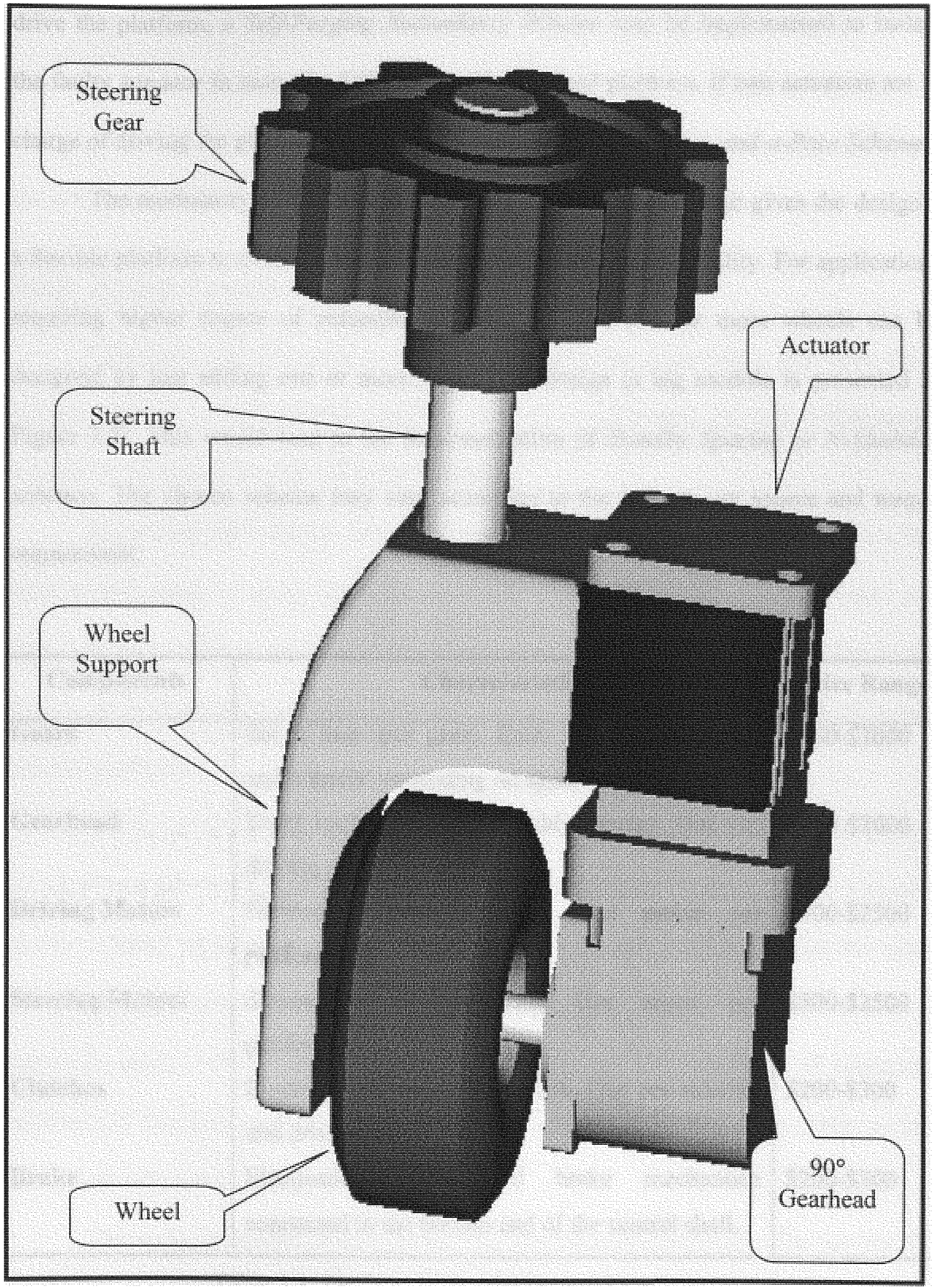

Figure 3.7. Leg Module. Clutch not Shown

drive the platform, a *Self-Purging Redundancy Scheme* may be implemented to isolate the faulty actuator in case of a failure. In a three-wheel platform, if two actuators are in charge of driving the platform, they can be controlled using a *Spare-and-a-Pare Scheme.*

The modularity of this design is another important feature that gives the designer a flexible platform to evaluate designs with different levels of reliability. For applications requiring higher degree of redundancy, platforms with four or more wheels can be designed by just adding one or more legs to the design (a leg module is presented in Figure 3.7). This would lead to the implementation of *Standby Sparing* or *N-Modular schemes*. The chosen scheme may vary according to the task, power source and torque requirement.

Components	Characteristics	Price Range
Gears	Set of four spur gears. Ratio 1:1. Pitch diameter and material depending on application.	$450-$3000
Gearhead	Right angle gear head. Highly precise. One per driving motor required.	$600-$3000
Driving Motors	Servomotor with encoder. One motor per platform leg.	$500-$2500
Steering Motors	Servomotor with encoder. One motor per platform leg.	$500-$2500
Clutches	Electronically controlled clutch. One per steering and driving motor.	$200-$300
Brake	Electronically controlled brake mechanism connected in the bottom end of the central shaft.	$200-$300

Table 3.1. Platform Main Mechanical Components

The recommended mechanical components that form the main structure of the three legs, mobile platform that is presented in this section are listed in the table below. In addition, the table gives some general information about the individual components and the price range.

3.2.4 Design Comparison

Chapter 2 was dedicated to review some hardware redundancy and basic fault-tolerant design schemes. These schemes have been developed for and implemented mainly by the electronic and computer industry.

In general, this chapter was dedicated to show different design alternatives that open a wide range of possibilities for the implementation of hardware redundancy and fault tolerance schemes on mechanical systems. Specifically, this work is intended for the design of fault-tolerant actuation systems (i.e. steering and driving systems) of mobile robotic platforms.

The initial design served as the starting point in the search for a basic architecture of a fault-tolerant mobile platform. The first design alternative proposes to use a simple, dually actuated differential mechanism. This is a simple and compact design that gives the platform the ability to undergo failure of one of the actuators and still be able to return to safety. Other features of this arrangement are the possibility of correcting mechanical backlash and increase in payload carrying capacity when both motors simultaneously actuate the platform.

The major drawback of this design is its limited maneuverability. The fact that the rear wheels have only one degree of freedom (rotation about their central axis) limits the steering operation to front-wheel steering only. Also, as mentioned before, it is clear that a failure of one of the actuators disables the steering mechanism.

The change in the arrangement of the driving system provides the platform wheels with two degrees of freedom. There are two main advantages to this wheel independence. First of all, maneuverability is greatly increased as any point in the horizontal plane can be reached in a straight trajectory provided that no obstacles are run into. Secondly, this driving system design can carry as many redundant actuators as the number of wheels allowed. This makes the design more flexible and modular. For example, a platform could be designed to carry six actuators (and six wheels). Then, one should be able to remove them up to a minimum of three according to the task's safety and reliability requirements. This makes the vehicle more maneuverable and fault tolerant than the other presented in this work. It also turns out to be a highly flexible test bed to experiment with different fault-tolerant schemes and alternatives.

3.3 Structural Design

The word robot can be defined as a mechanical device that is able to perform tasks and behave in a human-like fashion. Robots can substitute the human being in tasks requiring high accuracy and work pace. Such tasks are extremely difficult if not impossible to achieve by humans.

This interpretation of the word robot and the realization of effectiveness in the use of these machines may be enough for the end user to appreciate their availability. However, for the robot designer the definition is totally different. For them, a robot is a brew of gears, chains, sprockets, motors, encoders, wires, controllers and other electronic devices. In other words, a robot is a combination of mechanical and electronic components. In order for the design to meet its operational specs, a careful design and components selection research should take place.

The following sections provide basic information necessary for selecting and making optimal use of the components required to operate a fault tolerant mobile platform like the one presented in section 3.2.3 Final Design.

3.3.1 Actuators

Actuators are the muscles of the robots. Attach a motor to the wheels of a mobile platform and it will scoot around the floor. If a motor is attached to the joint of a manipulator, it will move.

There are different types of actuators. However, only few of them are suitable for autonomous, mobile robotics applications. Alternating current (AC) is very rarely used in robotics. Direct current (DC) dominates this field of application. DC current is used to run on-board-electronics, solenoids and actuators.

Reversible DC actuators are most suitable for robotics applications. Most DC motors are inherently bi-directional, but some design limitations may prevent reversibility. According to their way of rotation, DC motors can be classified as:

- Continuos DC Motors
- Stepper DC Motors.

The first type, Continues DC motors, are further classified as brush or brushless motors. For applications requiring high levels of reliability, brushless are selected over brush DC motors. The reason being is that the contact between the rotor surface and the brush while in operation causes the latter to wear and eventually produces misconnection. Therefore, brushless is the only type of DC motors to be considered in this work for robotics applications.

Brushless, DC servomotors are an important component for robotic, factory automation and numerically controlled applications. In general, they are lightweight, relatively small, easily integrated, efficient, controllable and maintenance-free devices. Their basic structure consists of a rotating part that contains permanent magnets and the stator. The stator is the attached to the motor case. It contains the winding poles to create the electromagnetic fields. The interaction between these electromagnetic fields and the one created by the magnets in the rotor provide the driving force of the motor.

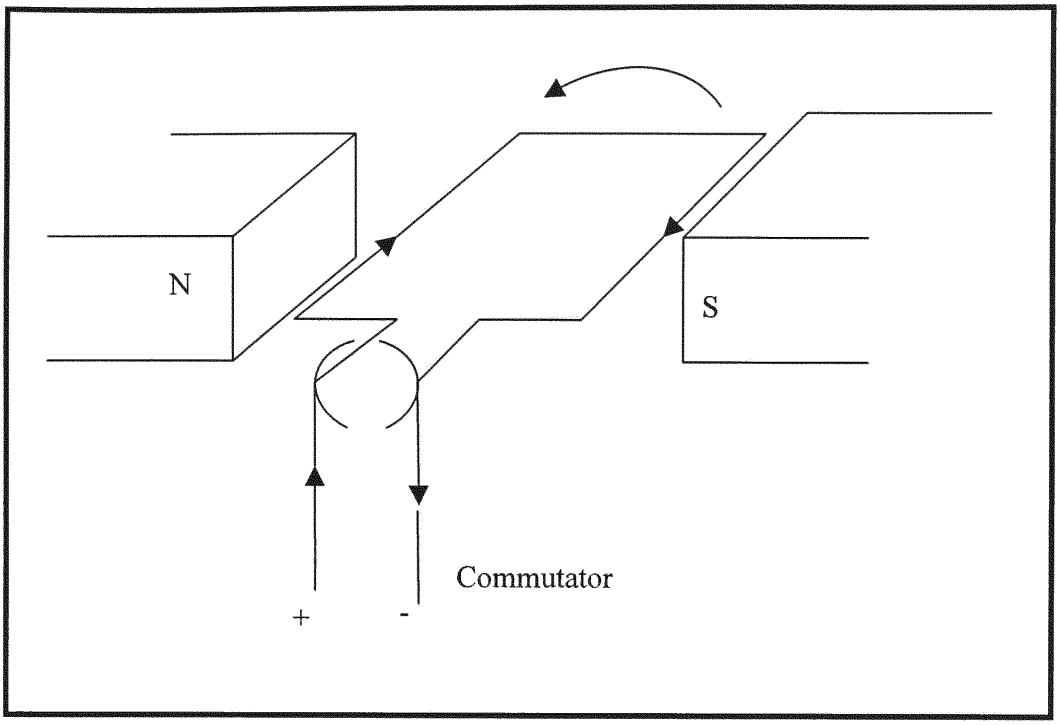

Figure 3.8. Conventional DC Brush Motor

In order to produce constant torque, these field should remain constant and in a relative orientation. This is done in a brush DC motor with the use of a commutator as shown in Figure 3.8. The windings are connected to the commutator in such a way that the current flow in the armature remains in a fixed position in space. This, in turn, produces shaft rotation.

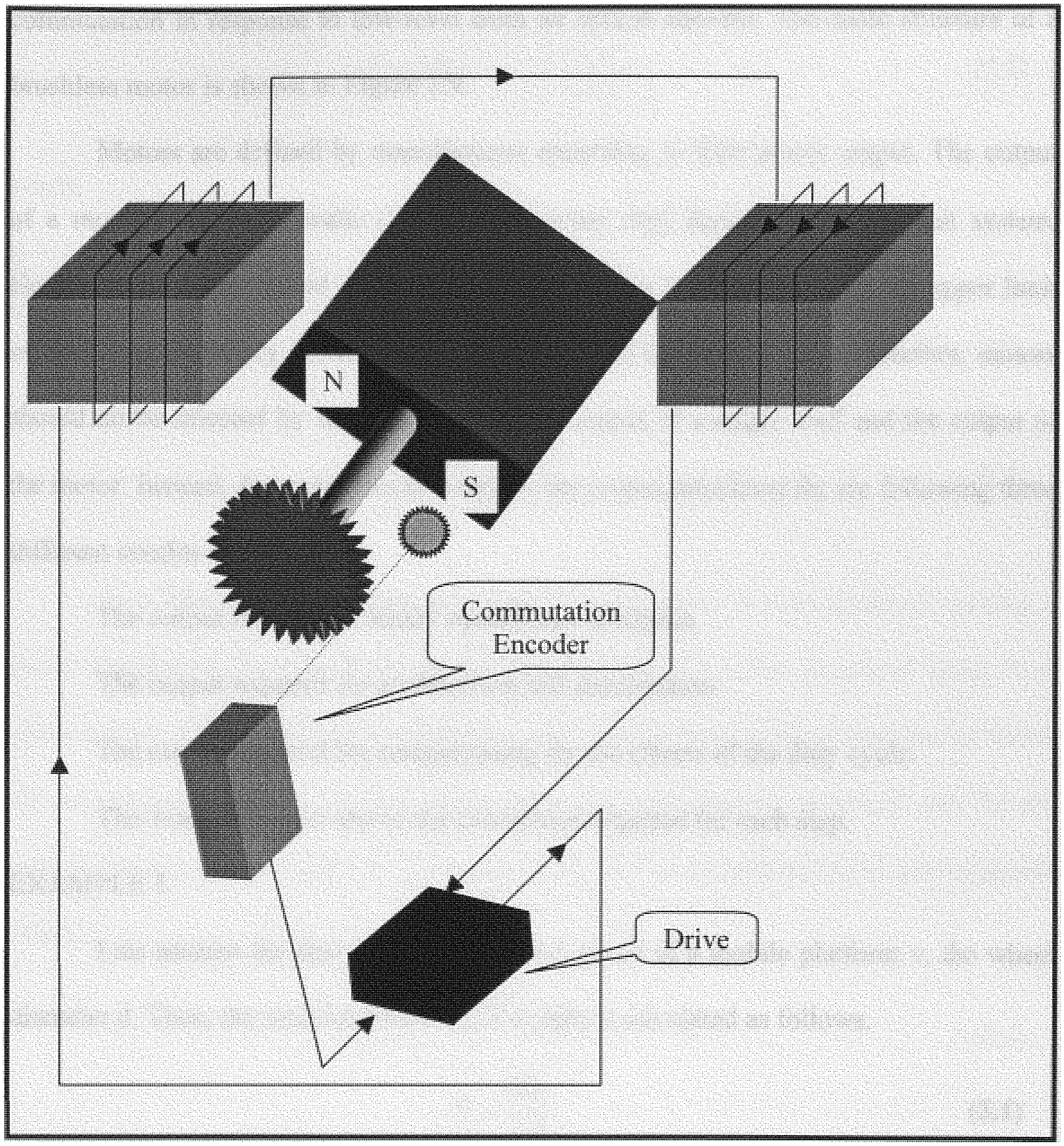

Figure 3.9. Brushless DC Motor

Unlike DC brush motors, brushless motors do not have commutator circuits. Therefore, they can not be driven by simply connecting them a DC power source. In brushless servomotor applications, an electronic amplifier is used to perform the commutation in response to low-level from an optical encoder. The basic structure of a brushless motor is shown in Figure 3.9.

Motors are defined by manufactures according to their power output. The output of a motor is given in watts (W) or horsepower (hp) depending on the unit system. However, the output stated in catalogs and nameplates is calculated from the upper limit of rotational frequency in stable conditions and from the rated torque. Therefore, motors should no be selected by solely comparing the output of a single load and the output of the motor. Instead, designers should compare the output calculated for the following three different conditions.

The output required in steady operational condition

The output required for acceleration and deceleration

The output required for compensating for the effects of the duty cycle.

The example below shows the calculation required for each step.

EXAMPLE 1.

Lets assume the desired, constant travel speed of a mobile platform v, the wheel diameter d. Then, the rotational velocity *N* in rpm is calculated as follows.

$$N = \frac{60v}{\pi d} \tag{3.1}$$

In order to calculate the output required in step1, the load torque *T* when the motor is operating at constant speed *N* should be determined. The load torque required for acceleration of the platform at rotational speed *N* in a time period t can be calculated as follows.

$$T = \left[W_t \left(\frac{d}{2} \right)^2 \right] \frac{2\pi N}{60gt}; \qquad \textbf{(3.2)}$$

where

g = Gravitational Constant.

Then, the rated power P in Watts can be calculated as:

$$P = \frac{N * T}{97.4} \qquad \textbf{(3.3)}$$

Now, from the rated torque vs. rotational speed curve in a manufacturer catalog, a motor equal or higher rated torque than the calculated T at the rotational speed N is chosen. This selection takes the designer to the step 2: check if the chosen motor meets the acceleration and deceleration specs. Using the manufacturer technical data for the motor, the acceleration time t_1 and deceleration time t_2 are calculated as per equations below.

$$t_1 = \frac{\left(J_M^2 + J_L^2\right)N}{375\left(T_M - T_L\right)} \qquad \textbf{(3.4)}$$

$$t_2 = \frac{\left(J_M^2 + J_L^2\right)N}{375\left(T_M + T_L\right)} \qquad \textbf{(3.5)}$$

where

J_M = Motor inertia

J_L = Load inertia

T_M = Maximum motor torque.

These times are compared with the acceleration and deceleration design specs to reach the constant travel speed *v*. Figure 3.10 shows a pyramidal speed profile that the platform should follow in any given displacement.

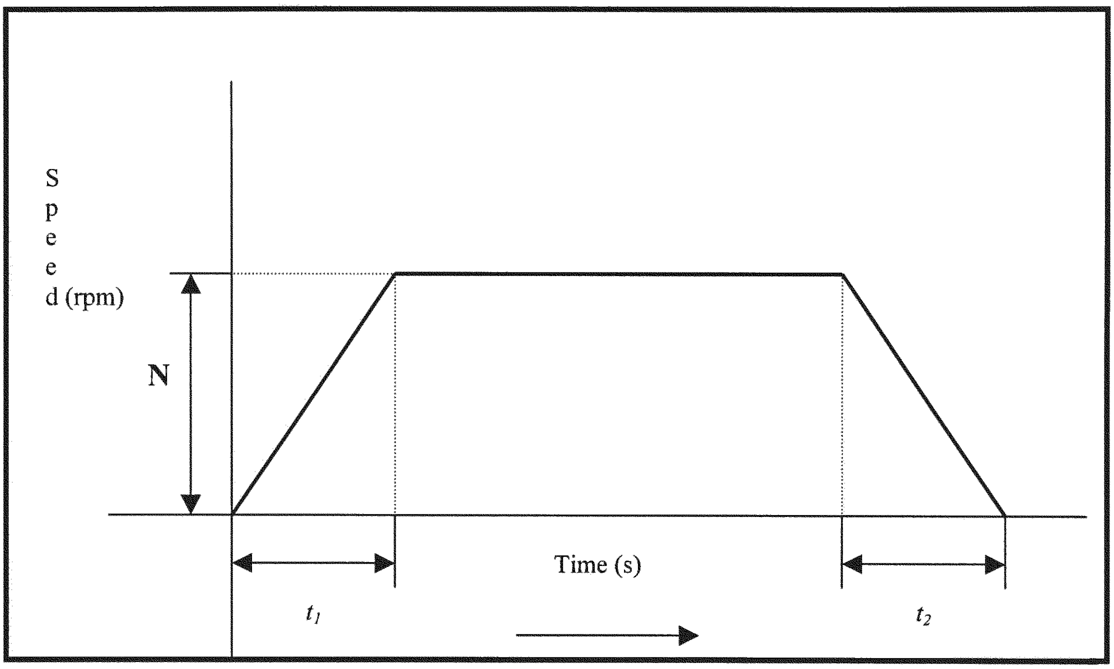

Figure 3.10. Pyramid Speed Profile

The results should not only be within the range shown in Figure 3.10 above but also, they should be at least five to ten milliseconds smaller than the specs. Since the calculations only take into account the motor data, this time addition is required to compensate for the delay time of the control circuit.

Next, calculation of the motor output considering the duty cycle is performed. Servomotors are designed to experience a temperature rise within certain range when they operate continuously at rated torque. Therefore, the influence of exceeding the rated current during acceleration and deceleration during long trajectories can not be overlooked.

The effective torque T_E should then be calculated based on the duty cycle of the particular application. Figure 3.11 shows a generic Speed-Torque vs. Time profile. Based on this profile a general equation to calculate the effective torque is derived in equation 3.6.

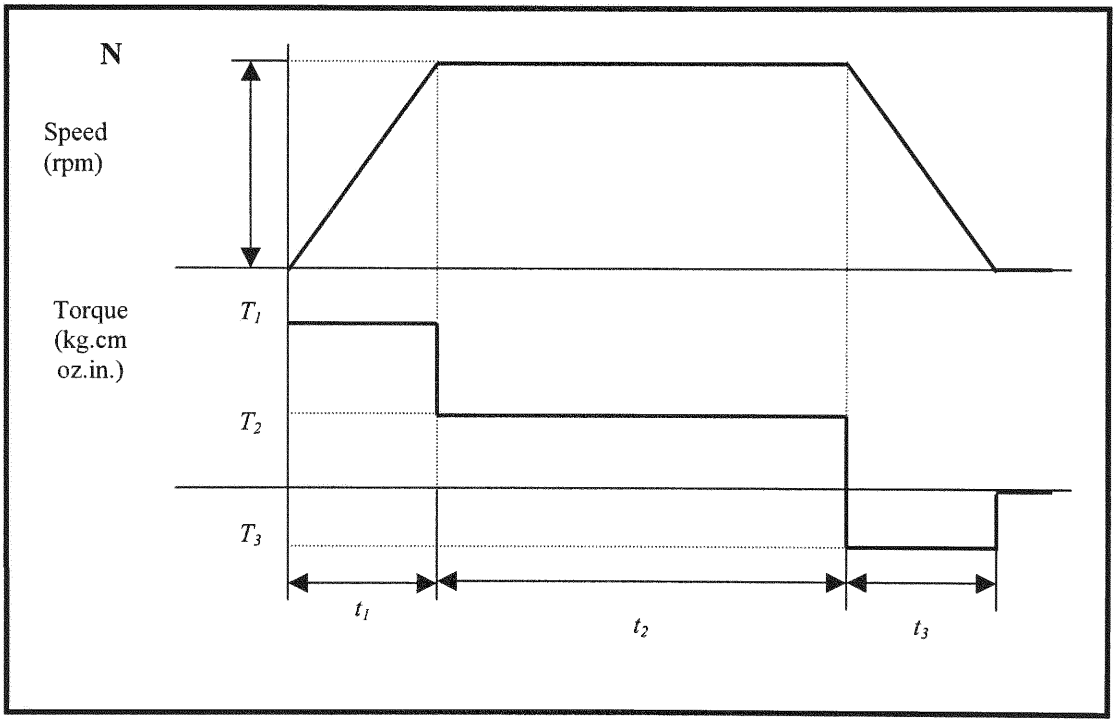

Figure 3.11. Speed-Torque vs. Time Curve

The value obtained from the equation 3.6 is compared with the one calculated in step one. For the final decision on the power output P of the selected motor is calculated using the equation from step one and the greater quantity between T_E and T.

$$T_E = \sqrt{\frac{T_1^2 t_1 + T_2^2 t_2 + T_3^2 t_3}{t_1 + t_2 + t_3}} \tag{3.6}$$

For the motor selection, the designer should plot T_E vs. N in the motor performance curve. The plotted point should be in the area below the curve. In addition, the chosen motor should allow for a torque excess of 50% over the plotted point at the operational speed N to give some protection margin against overload.

The main characteristic of servomotors is that they are able to generate large torque at high speed. Most design applications require high torque. However, this torque should be generated at reasonable rotational speed. For instance, a conveyor belt application is to move at a constant linear velocity of 12 in/sec and driven by 6 in-diameter pulleys. To meet these requirements, the motor connected directly to the pulley should drive the load at 0.64 rps or 38.2 rpm. This asks for a very large motor that consumes a lot of energy and is very expensive.

The common practice in this case is to use gearhead drives. Introduction of gearheads is highly beneficial. First of all, they reduce the motor rotational output velocity and increase the torque output by an amount equal to the gear ratio of the reduction mechanism. In turn, a smaller motor can be used to produce the same amount of torque.

Gearheads are capable of responsively accelerating or decelerating large loads if the gear ratio have been properly calculated. This calculation is known as inertia matching. A well-calculated gear ratio will effectively reduce the external load at the motor shaft by an inverse of the square of the gear ratio. The inertia matching value or effective inertia J_E is calculated as shown in equation 3.7 and Figure 3.12.

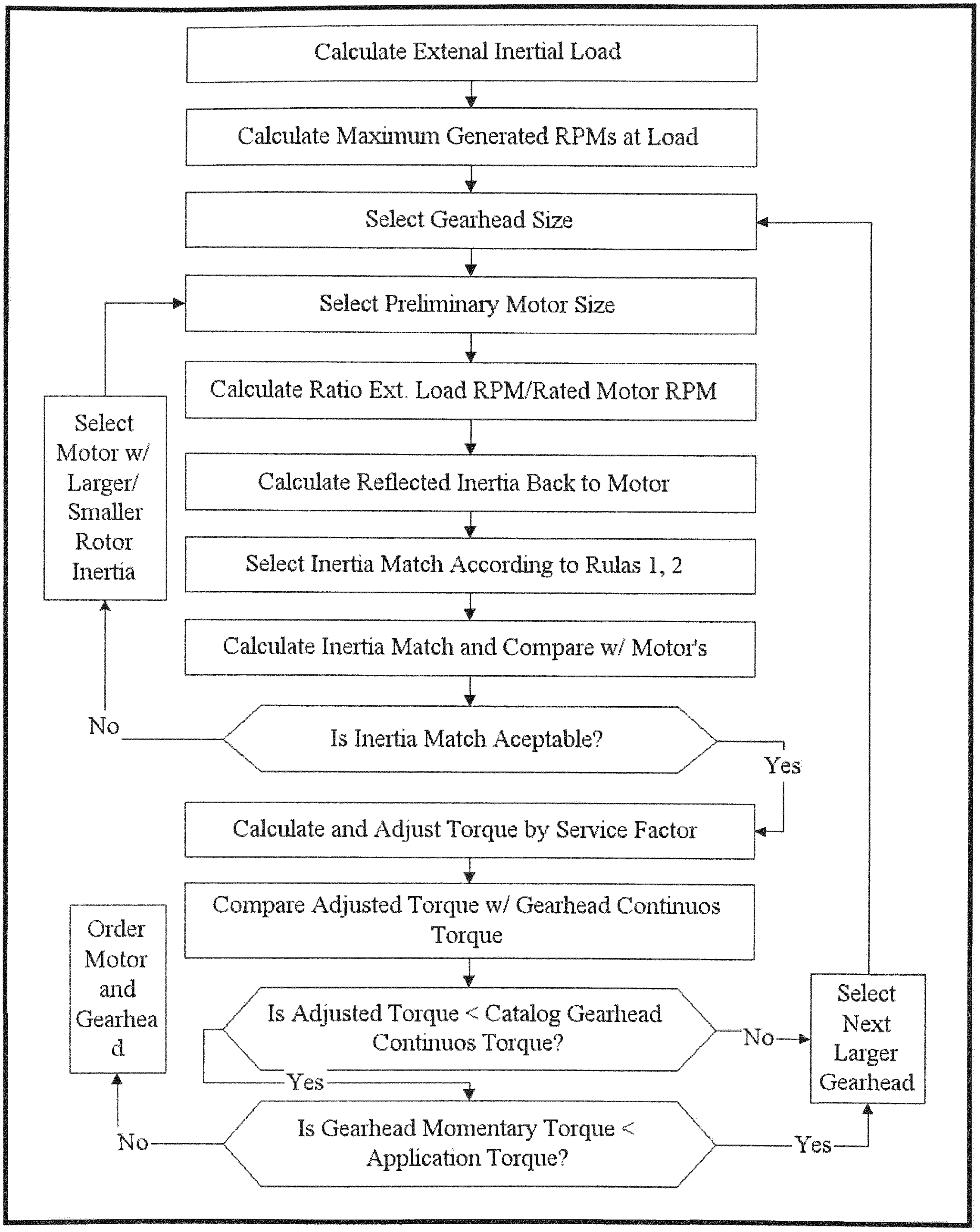
Calculate Extenal Inertial Load
Calculate Maximum Generated RPMs at Load
Select Gearhead Size
Select Preliminary Motor Size
Calculate Ratio Ext. Load RPM/Rated Motor RPM
Calculate Reflected Inertia Back to Motor
Select Inertia Match According to Rulas 1, 2
Calculate Inertia Match and Compare w/ Motor's
Is Inertia Match Aceptable?
No
Select Motor w/ Larger/ Smaller Rotor Inertia
Yes
Calculate and Adjust Torque by Service Factor
Compare Adjusted Torque w/ Gearhead Continuos Torque
Is Adjusted Torque < Catalog Gearhead Continuos Torque?
No
Select Next Larger Gearhead
Yes
Is Gearhead Momentary Torque < Application Torque?
No
Order Motor and Gearhea d
Yes

Figure 3.12. Gearhead Inertia Matching Process

$$J_E = \frac{J_L}{GR^2} + J_{GH} + J_{IP} \qquad (3.7)$$

where

J_E = Effective inertia

J_L = Load inertia

GR = Gear ratio

J_{GH} = Gearhead inertia

J_{IP} = Input pinion inertia.

The value of J_E is divided by the motor inertia J_P to obtain the ratio J_E/J_P. The parameter is J_P provided by the motor manufacturer in the catalog. According to the industry standards, if this ratio J_E/J_P is a number from 1 to 5, the motor will respond very fast to changes inn velocity. Otherwise, the system will respond acceptably fast for ratio values up to 20. If the ratio J_E/J_P is larger than 20 or smaller than 1, a motor with smaller or larger rotor inertia should be selected as required and the inertia matching process reinitiated.

Once an acceptable inertia match has been obtained, the effective torque T_E calculated using the equation above is adjusted by the service factor SF as shown in Figure 3.13. If the adjusted torque is smaller than the gearhead maximum continuos torque in the catalog and the momentary torque exceeds the torque required for the application, a perfect motor/gearhead match has been found. If any of these two necessary conditions are not meet, the next larger gearhead should be selected and the process of inertia matching is restarted.

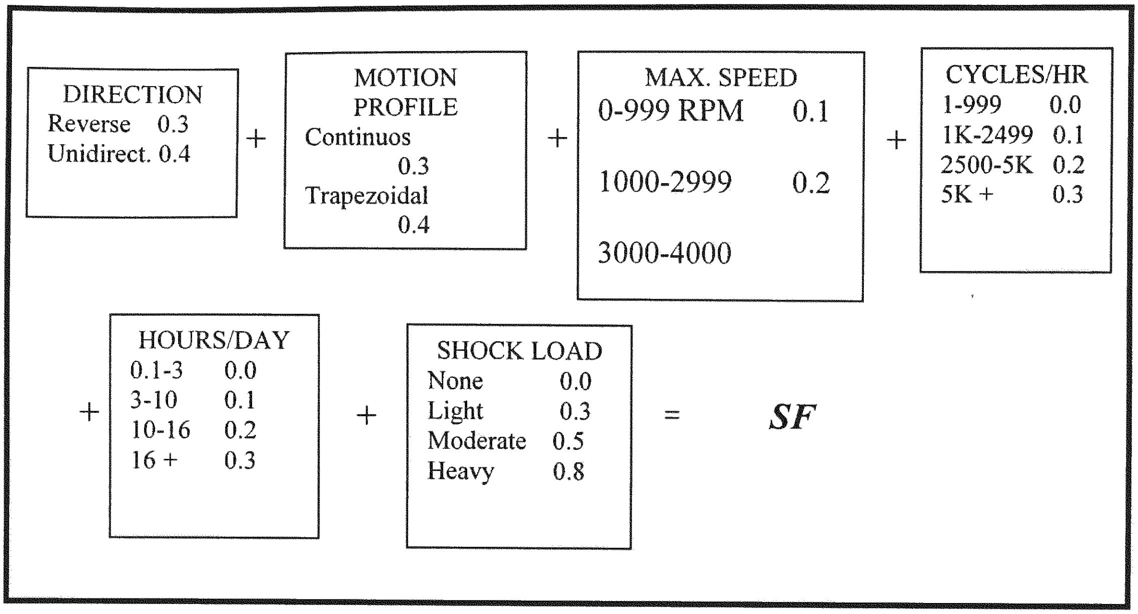

Figure 3.13. Service Factor Calculations

DC servomotors have become a relatively inexpensive source of mechanical power. They provide higher capacity at higher speed while operating silently and without maintenance. Their biggest advantage is in the close-loop operation mode. This is, they operate under the supervision of a control system with a position or velocity feedback information to the controller. This information is provided by sensors attached to the motor shaft. Sensors and control systems will be introduced in further sections.

Unlike DC servomotors, stepper motors are inherently open-loop actuators. In its normal operation mode, the stepwise rotation of the motor is synchronized with the command step train. This is the reason why the term "*digital synchronous motor*" is often used to denote a steeper motor. The basic components needed for the open-loop operation of a stepper motor are shown in figure 3.14 below.

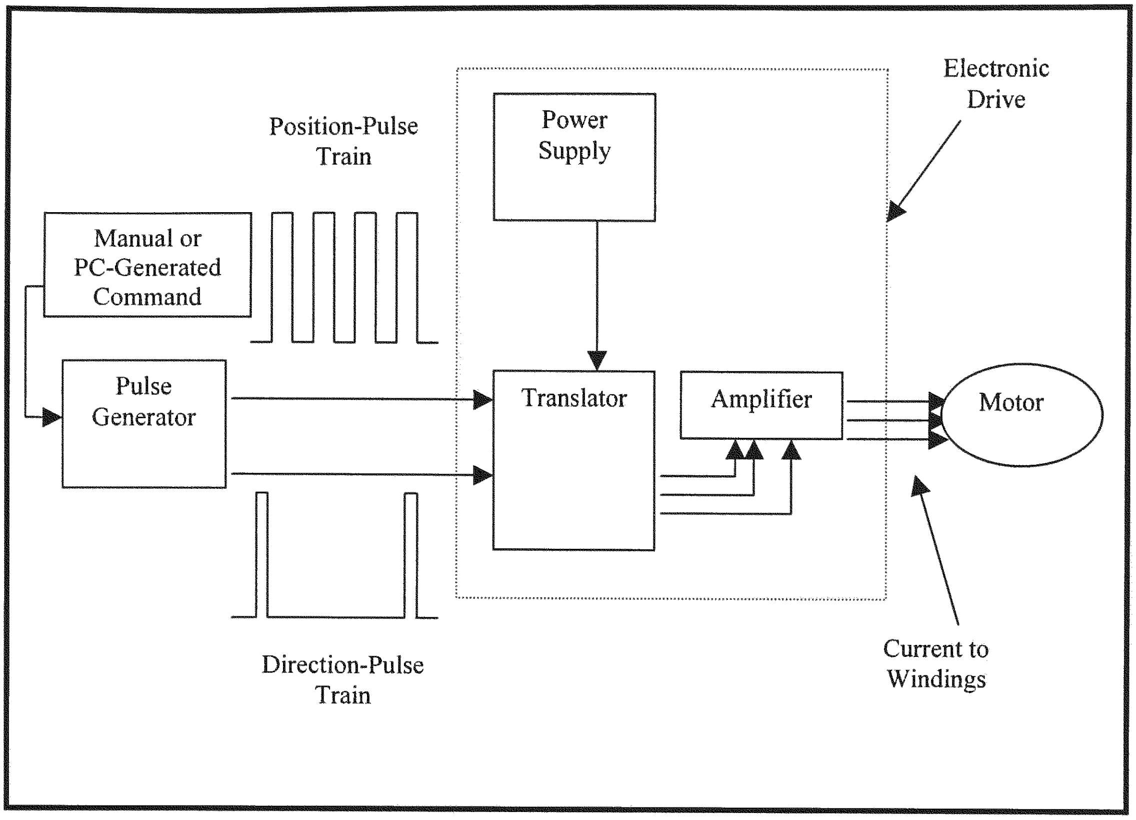

Figure 3.14. Stepper Motor Open-Loop Control System Diagram

The *pulse generator* is typically a variable frequency-oscillator. This oscillator will generate the required position-pulse and direction-pulse trains to produce a bi-directional rotation of the motor shaft. The position pulse determines the exact times at which angular steps are initiated. In steady-state operations at constant step rate, only one position command is necessary to produce the desired motion. For transient operation modes (i.e. following intricate trajectories) microprocessor-generated pulse trains may be required. The direction-pulse train indicates the times at which the direction of rotation should be reversed.

The *translator module* is in charge of interpreting the pulse train and translating it into the corresponding switching frequency for the stator field windings. In addition, the *translator module* directs the field currents to the corresponding phase winding using a solid-state circuitry of gates, latches, triggers and so forth. In order for the translator to keep track of the number of steps taken in a given direction, the *translator module* has to have a *preset indexer.* The required angle of rotation, stepping rate and direction are set manually or by computer generated commands. Translators usually operate from a 24 V. D.C. power supply.

When selecting the appropriate stepper motor for a particular application, required torque and speed profile must be clearly determined. For instance, if a fast motor response is desired, a stepper with large torque-to-inertia ratio should be selected. The tedious process of selecting a stepper motor can be simplified if it is done in orderly manner. The following steps provide some guidelines for this process.

First, list all the design specs. This may include operational parameters such as speed, acceleration, and repeatability. Load characteristics such as size, inertia and resistance torque should be listed as well.

The next step is to calculate the operating torque and the step rate for the application. Using Newton's second law, the required torque rating is calculated as shown in equation 3.8 below.

$$T = T_R + \frac{J_{eq}\,\omega_{Max}}{\Delta t} \qquad \textbf{(3.8)}$$

where,

T_R = net resistance torque

J_{eq} = moment of inertia of the entire system

ω_{Max} = maximum operating speed

Δ_t = time taken to accelerate the load to its maximum speed, starting from rest

The third step consists of selecting a suitable stepper motor using its torque versus stepping rate curve. If not stepper motor available meets the requirements, changes to the original design may be necessary. A solution to this problem may be the introduction of mechanical devices such as gear systems. Finally, a dive system that is compatible with the chosen stepper motor is selected.

In addition to the torque-versus-stepping-rate curve, other important parameters that define the stepper motor's performance may be mentioned. One of them is the *step angle*. The *step angle* determines how many steps the motor will provide in a motor shaft revolution. The *static holding torque* is the torque provided by the motor when it is powered with the rated voltage. Another important parameter is the *maximum slew rate,* which the highest stepping rate the motor can maintain in a steady-state operation at the rated load. It is also important to know the magnitude of the torque provided by the motor when the motor operates at maximum slew rate. The parameter that identifies this torque is called *pull-out torque*. Due to self-induction, the current does not build up instantaneously in the winding of the energized phase. The *electric time constant* is the parameter that defines the necessary time to build up the current in the winding of the energized phase.

Stepper motors are incremental actuators. Therefore, they are ideal for digital control applications. High precision, open-loop operations are also possible as long as the conditions are well within the motor capabilities. These motors are particularly well suited for printers. They are extensively used in magnetic tape and disk drives, joint actuators in robotic systems and indexing tables just to mention few applications.

Stepper motors present some advantageous features. First, the position error generated between steps is not cumulative. Consequently, high accuracy of motion can be obtained even in open-loop operations. Provided that no loses of step command occurs, accuracy is a function of the step angle only. In addition, open-loop systems are simpler and less expensive than close-loop systems and present no serious stability problems.

Despite the advantages open-loop-control stepper technology may provide, this type of actuator is not suitable in applications where lose of step might occur. In such case servomotors outperforms stepper technology. This is the case of robotic mobile platforms. Here, lose of steps may be produced by a variety of reasons. Odometry errors and wheel slippage are just two sources for introducing errors in the position control of the stepper motor. Therefore, this type actuator would not be used in the design of a fault-tolerant mobile platform.

3.3.2 Sensors

The information feedback circuit, which enables the system to correct a measured parameter, is what makes the difference between an open-loop control system and a close-loop one. It is not until a sensor is introduced as part of the feedback loop that a control circuit can detect and control positions. Optical encoders, resolvers are the most commonly used sensors in robotic odometry applications.

Optical encoders are generally found in servomotor control systems. This type of sensor is the recommended for highly reliable and precise applications. Therefore, they are recommended to perform odometry tasks for the fault-tolerant mobile platform design being presented in this thesis.

Optical encoders use an opaque disk that has one or more circular tracks that present certain arrangement of identical transparent windows. A parallel beam of light is projected to all the tracks. On the other side, the light beam is picked off by an array of photo-sensors (generally one for each track). Since the light beam is frequently interrupted by the opaque portions of the track, the input obtained from the photo-sensor is a train of series of voltage pulses. This signal can then be interpreted to determine the angular position and velocity of the encoder shaft. The internal hardware structure of an optical encoder is shown in figure 3.15.

Optical encoders are classified into incremental and absolute encoders according to the way they keep track of the shaft angular displacement. Incremental optical encoders are found in two different configurations. One of these configurations features a single circular track with identical and equally spaced slits. The slits have identical dimensions to those of the opaque areas. Two photodiodes are placed a quarter-pitch apart (half of the slit length) to pick off the light beam pulses.

In the other possible configuration, two identical tracks are used. One is offset a quarter-pitch from the other. One photodiode picks off the light beam of one of the tracks. In both configurations, an additional track with only one window and the respective photo-sensor is used to keep track of each revolution.

The digital data obtained from the encoder readings is used by a digital processor to compute displacement and velocities (i.e. odometry measurements).

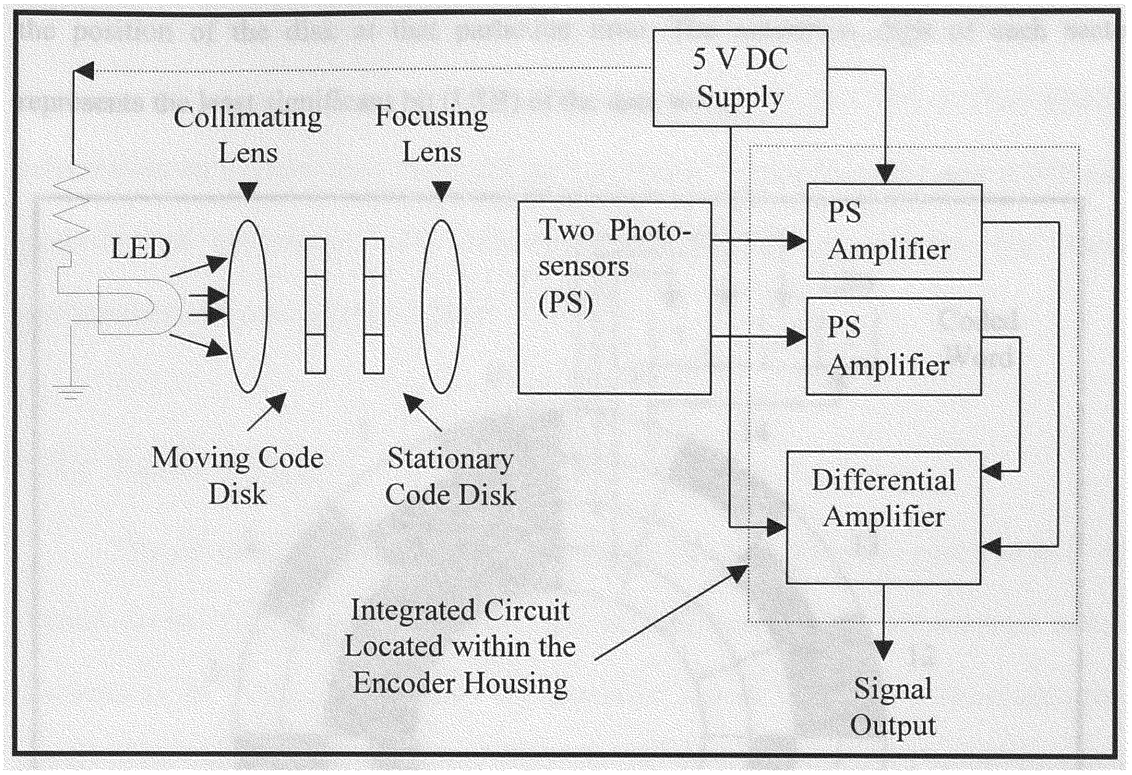

Figure 3.15. Hardware Structure of an Incremental Optical Encoder

Absolute encoders, on the other hand, directly generate coded data to represent the angular position. Unlike relative optical encoders, there is not a pulse count here. Just a simple code pattern on the encoder disk is used in this case. Figure 3.16 below shows the formation of a four digit binary code from an absolute encoder disk. In the figure, the number of tracks (*n*) is four. In practice however, this number may surpass fourteen. Each disk is divided into 2^n sectors. Each partition of the track generates one bit of data. An array of light sensors picks off the light beam from the other side of the disk. The set of pulse signal is captured by the light sensors as the disk rotates. At a given instant, the combination of signal levels will provide a coded data word that unmistakably identifies the position of the disk at that particular time. The outermost digit of each sector represents the least significant bit (LSB) of the data word.

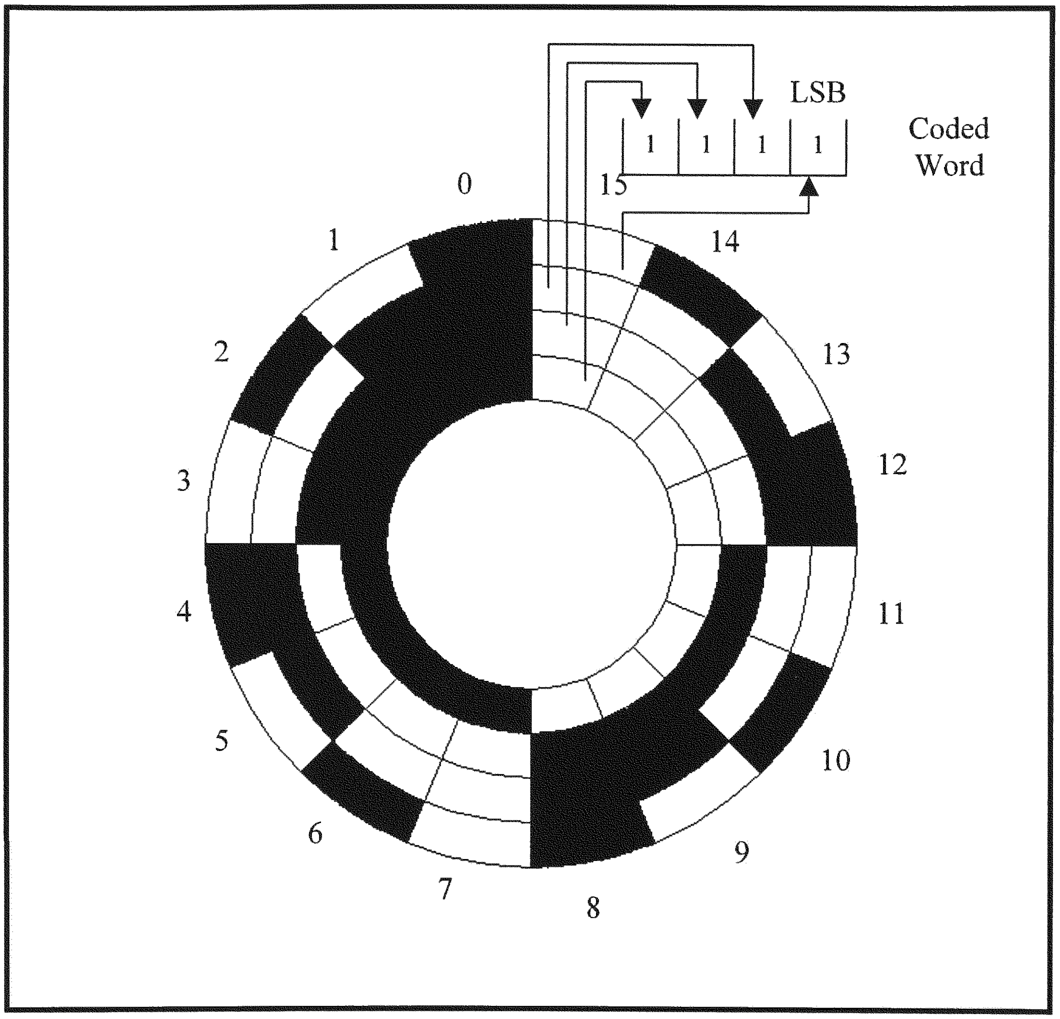

Figure 3.16. Absolute Optical Encoder Disk

Despite the accuracy of these devices, odometry is still affected by factors that are not related to electronic devices. Therefore, errors due to slippage for example, should be corrected using other methods. Otherwise, a constantly growing error will lead the mobile platform out of the desired path.

Heading sensors are of particular importance because they can compensate for the weakness of odometry measurements. The reader may consult [2] to find detailed information on devices such as gyroscopes, geomagnetic sensors and other positioning methods and technology. This information can be used to improve the overall performance of the navigation system.

3.3.3 Power Supply

The power supply for autonomous mobile vehicles is generally provided by a DC current source (i.e. batteries). Batteries are as important for the autonomous mobile platform as the frame, motors and the electronic circuit. Unlike the electronics they should power in robotic applications, batteries still do not have a good power-to-weight or power-to-size ratio.

There are mainly five types of commercially available batteries. These come in a great variety of sizes, shapes and configurations. For instance, Zinc batteries, also known as flashlight cells are found in two different types: carbon zinc or zinc chloride. These batteries are not suitable for most robotic applications due to the amount energy they can provide. Alkaline batteries are expected to perform four to five times better than zinc ones. They cost almost twice as much as zinc batteries. However, the increase in performance is worth the higher price. This type of battery however, is not rechargeable and therefore, it is not a good choice for robotic applications. Nickel-Cadmium batteries, on the other hand, are specially design to be recharged. However, they do not last as long as zinc or alkaline ones.

Lead-Acid batteries can supply the most amount of energy among the five types. They come in self-contained packs of 6, 12 and 24 V. Their only disadvantage is the weight. These heavy batteries are found in a wide range of power output. Therefore, this type of batteries is the best choice when the motors of the mobile platform are powerful enough to drive the battery weight.

Batteries carry all type of ratings and specifications. The most important ones from the designer point of view are the *voltage* and the *Amp-hour* current rates. The voltage rating is a straightforward concept. The battery will simply supply just above its rated voltage when new. It will not be able to provide power when the voltage has dropped eighty percent of it nominal value. It is important to keep close look at this rating because no enough voltage going through the controllers and electronics in the system may produce errors, lose of data and can damage the circuits.

The *Amp-hour* rating indicates the amount of current the battery can provide in a given time period. Specifically, the term *Amp-hour* states the amount of Amps the battery will steadily provide in an hour. For example, a battery rated at 2 Amp-h can provide 4 Amp for half an hour or I Amp for two hours. This all is theory however. In practice, due to the way ratings are obtained, it is very rare to find a battery capable of providing the rated power. They generally die long before they are supposed to. A good rule of thumb when selecting the system power supply is to chose a battery that provides from 20% to 40% more current than the amount required by the system. This will give a margin for components such as motors that draw more current that what they normally do in steady-state operations while accelerating and decelerating.

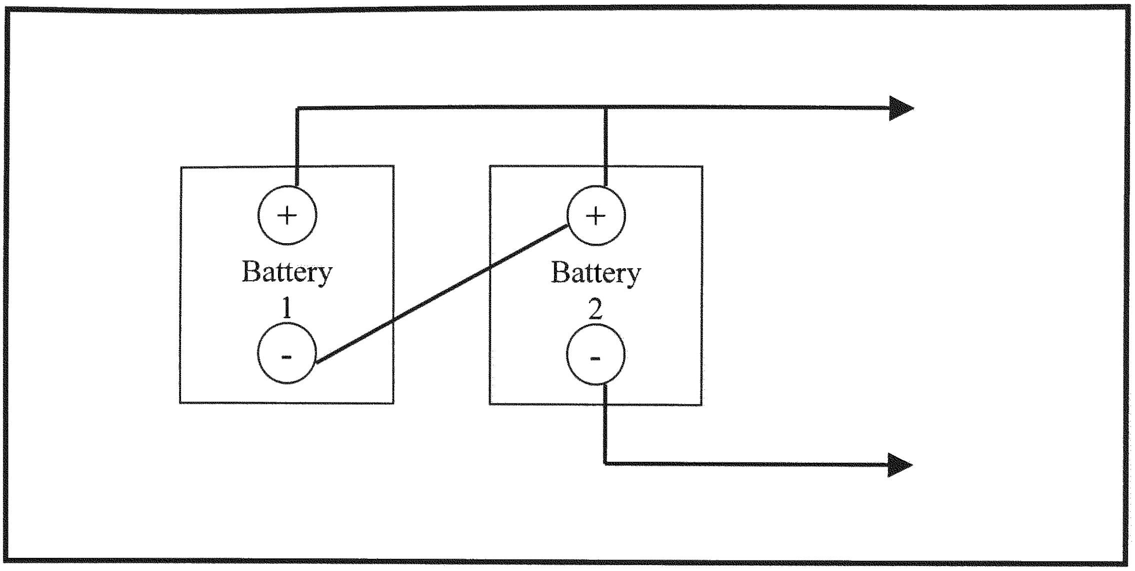

Figure 3.17. Battery Cell in Series Connection. Voltage Doubles

These ratings can be incremented by connecting several cells together. In order to increase the voltage supply of the source, the cells should be connected in series as shown in figure 3.17. The cell output voltage will be the sum of the voltage output of the individual cells. The current output of the pack of cells remains constant. On the other hand, if current output increase is necessary, the cells should be connected in parallel. This type of connection doubles the amount of current of the cell while keeps the voltage output unchanged. Figure 3.18 below shows the proper connection for parallel battery arrangement. Note that when connecting cells together, they may not discharge or charge at the same rate. Therefore, all cells in the pack should be recharged at once. If a cell is damaged and it can not deliver the same charge as the others, it should be replaced.

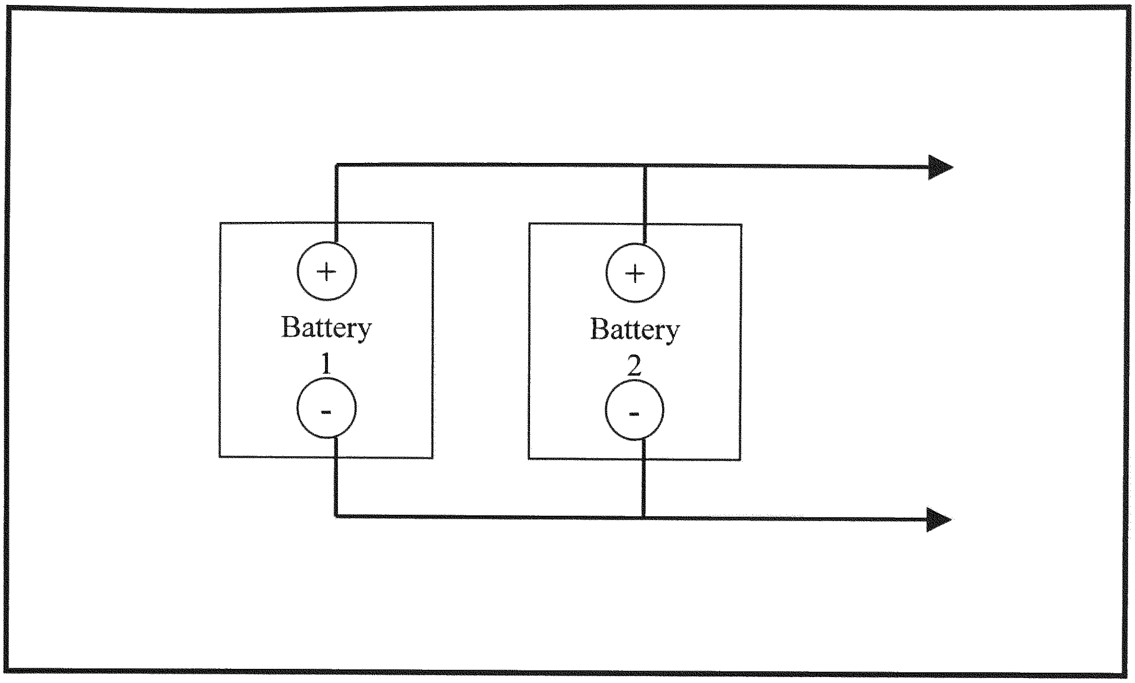

Figure 3.18. Battery Cell in Parallel Connection. Current Doubles

Chapter 4. Kinematic Model

4.1 Conventional Kinematic Models of Mobile Platforms

If a relationship between the animal kingdom and a mobile robotics can be established, it could safely be said that the animal brain is to the animal as robot control system is to the robot. Likewise, the skeleton is to the animal as the kinematic architecture to the robot. Depending on the skeleton structure and its characteristics, some animals are capable of performing tasks easier than others. Kinematics structure also influences the robot capability of accurately performing tasks such as maneuvering and odometry measurements for reckoning.

Due to the importance of selecting an adequate robot kinematic structure that best suits the task needs, this chapter will be entirely dedicated to present the most commonly implemented architectures for mobile robotics.

These are:

- Differential drive
- Ackerman Steering
- Tricycle Drive
- Synchro Drive
- Tracked Vehicles

4.1.1 Differential Drive Systems

In a typical differential drive architecture, encoders are mounted in the shaft of two driving motors to count the wheel revolutions. The Figure 4.1 shows this typical arrangement.

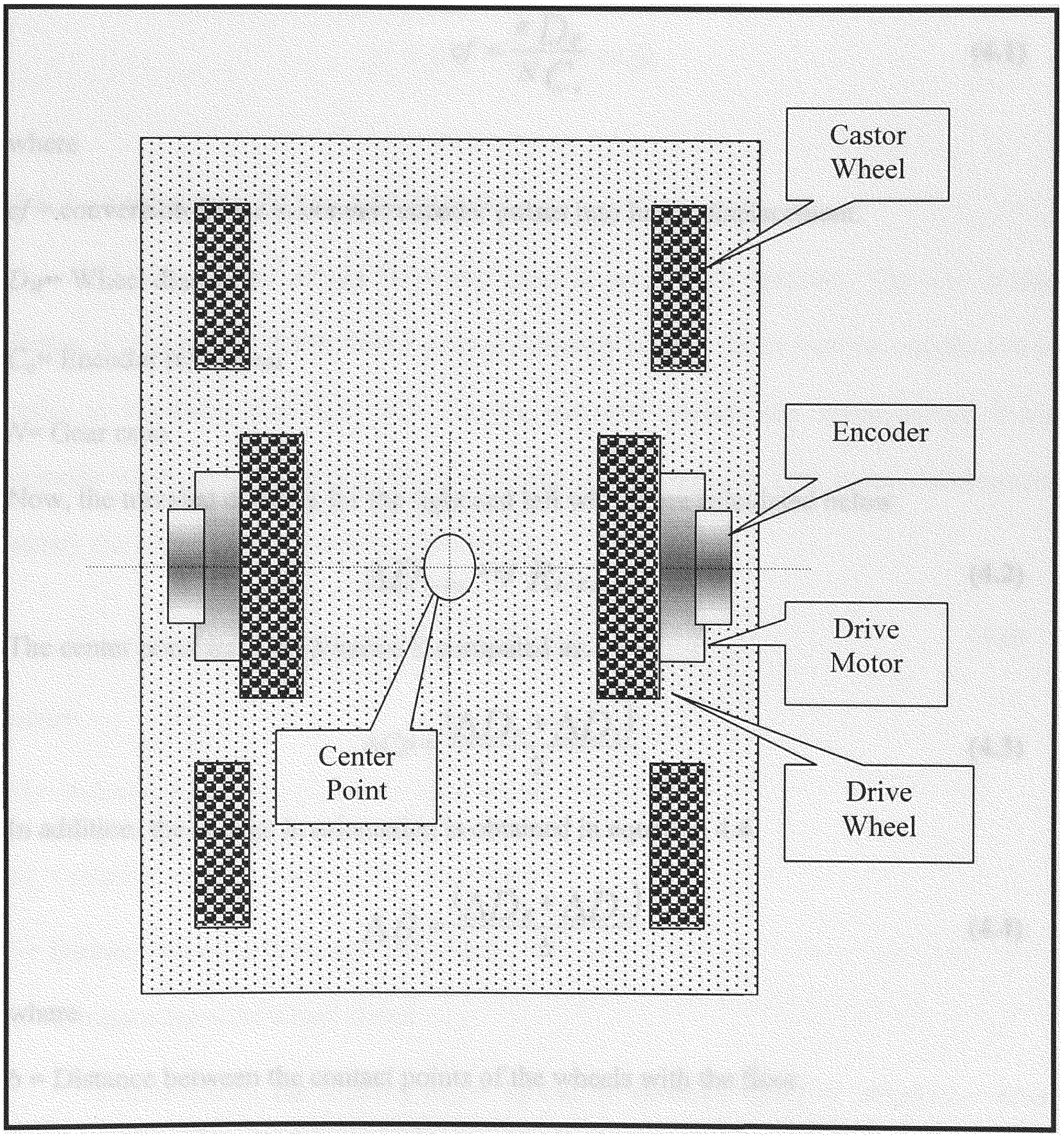

Figure 4.1. Differential Drive Platform Architecture

The robot can calculate its position using the odometry equations and the information on previous position. The odometry equations are as follows.

Lets assume that for a time period i, the left and right encoder provide readings R_L and R_R respectively. Also, let

$$cf = \frac{\pi D_W}{N C_e} \qquad \textbf{(4.1)}$$

where

cf = conversion factor to convert encoder pulses into linear displacement.

D_W= Wheel diameter

C_e= Encoder resolution

N= Gear ratio

Now, the traveled distance for the right and left wheels are calculated below.

$$\Delta D_{L/R,i} = cf\, R_{L/R,i} \qquad \textbf{(4.2)}$$

The center point traveled distance is computed as:

$$\Delta Cp = \frac{(\Delta D_R + \Delta D_L)}{2} \qquad \textbf{(4.3)}$$

In addition, the change in orientation is obtained in equation 4.4.

$$\Delta\theta_i = \frac{(\Delta D_R + \Delta D_L)}{b} \qquad \textbf{(4.4)}$$

where

b = Distance between the contact points of the wheels with the floor.

The current orientation of the robot θi may now be calculated as:

$$\theta_i = \theta_{i-1} + \Delta\theta_i \qquad \textbf{(4.5)}$$

Finally, the current position of the robot center point *Cp*(x_i;y_i) is obtained using the equations 4.6 and 4.7.

$$x_i = x_{i-1} + \Delta Cp_i \cos\theta_i \qquad \textbf{(4.6)}$$

$$y_i = y_{i=1} + \Delta Cp_i \sin\theta_i \qquad \textbf{(4.7)}$$

4.1.2 Ackerman Steering Systems

This kinematic design is mostly used in automobile industry. The key in this design is to make the inside wheel steer at a slightly sharper angle than the outside one in order to eliminate slippage. This steering geometry causes any instantaneous linear velocity vector to be tangent to an arc of center *A1* and radio *R*. This kinematic structure satisfy the Ackerman equations shown below.

$$\cot\theta_r - \cot\theta_L = \frac{d}{l} \qquad \textbf{(4.8)}$$

where

θ_R= Relative steering angle of the inner wheel

θ_L = relative steering angle of the outer wheel

l = Distance between front and rear wheel

d = Distance between wheel center

Conveniently, the steering angle θ_S can be measure using an imaginary wheel located at the reference point *A2* that is co-linearly oriented with the platform heading (see Figure 4.2). The steering angle θ_S may be expressed as a function of either θ_R or θ_L as shown below.

$$\cot\theta_S = \frac{d}{2i} + \cot\theta_R \qquad \textbf{(4.9)}$$

or,

$$\cot\theta_S = \cot\theta_L - \frac{d}{2l} \qquad \textbf{(4.9)}$$

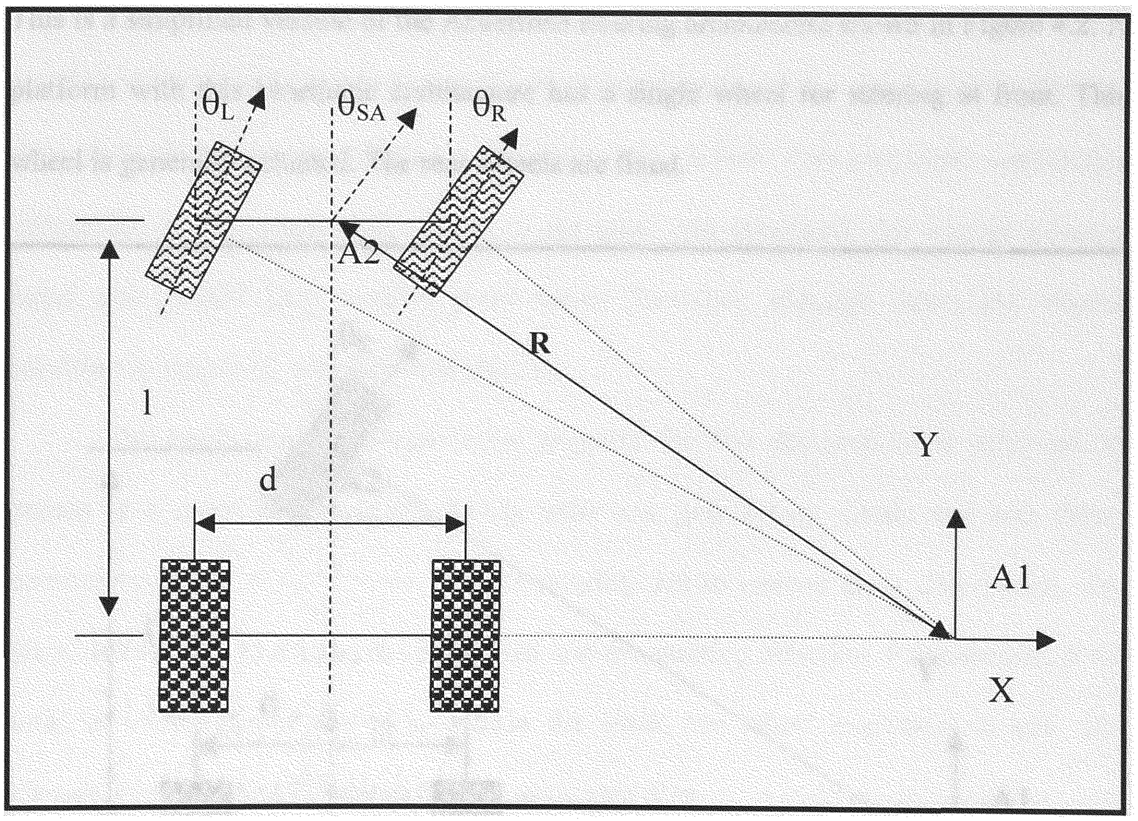

Figure 4.2. Ackerman Steering Kinematic Architecture

Ackerman steering provide a good kinematic structure to perform position odometry. In addition, the equations provide accurate results for all-terrain navigation where clearance and surface traction specs are demanding. This implies that Ackerman steering architecture is best suited for outdoor applications. However, this architecture is not recommended for applications in clustered environments where high maneuverability is required.

4.1.3 Tricycle Drive

The tricycle Drive architecture is very common in autonomously guided vehicles. This is a simplified version of the Ackerman steering architecture shown in Figure 4.2. A platform with this kinematic architecture has a single wheel for steering at front. This wheel is generally actuated. The rear wheels are fixed.

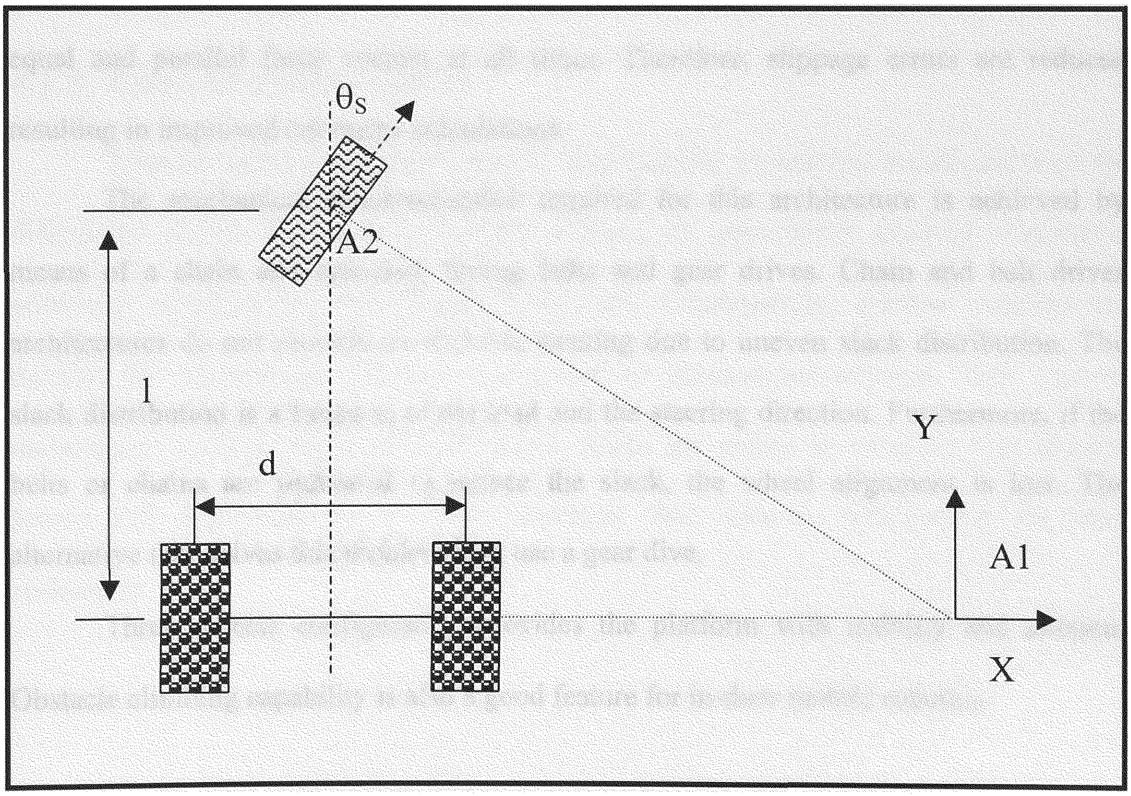

Figure 4.3. Tricycle Drive Kinematic Architecture

Alternatively, if the rear wheels are the driven ones, the solution to this kinematic configuration is given by the differential drive solution explained before. A stability problem is encountered due to that the change of position of the center of gravity away from the front wheel when traveling upward on an incline surface. The figure below shows the platform kinematic model.

4.1.4 Synchro Drive Kinematics

Synchro drive features an innovative kinematic structure of three or more wheel coupled in such a way that they rotate in the same direction at the same speed. Likewise, they pivot about their steering axes while executing a turn. This kinematic architecture introduces an advantage with respect to the other just presented. All wheels generate equal and parallel force vectors at all times. Therefore, slippage errors are reduced resulting in improved odometry calculations.

The mechanical synchronization required for this architecture is achieved by means of a chain and sprocket, timing belts and gear drives. Chain and belt drives architectures do not provide an accurate steering due to uneven slack distribution. The slack distribution is a function of the load and the steering direction. Furthermore, if the belts or chains are tightened to reduce the slack, the wheel alignment is lost. The alternative that solves this problem is to use a gear dive.

Three wheels configuration provides the platform with stability and traction. Obstacle climbing capability is also a good feature for in-door mobile robotics.

Odometry calculations are trivial. The vehicle direction is calculated directly form the steering encoder. The displacement in the direction of travel is calculated by the equation:

$$D = \frac{2\pi N_E}{C_E} R_{EW} \tag{4.10}$$

where

D = Vehicle displacement

N_E = Encoder measured pulses

C_E = Encoder pulses per revolution

R_{EW} = Effective wheel radius

4.1.5 Tracked Vehicle Drives

Tracked vehicles and their inherent skid steering are another modification of the differential drives. The steering process is based on pure wheel slippage. Therefore, odometry measurements are highly inaccurate. Consequently, tracked vehicles are suitable for teleoperated robotics as opposed to autonomous robotics. This type of architecture perform better in applications where the ability to navigate vast amount terrain and discontinuities is more important than accurate information for dead reckoning.

4.2 Model Comparison and Discussion

In section 4.2 above, the most commonly used mobile platform kinematic structures have been presented. In this section, the advantages and disadvantages of them will be explained. This explanation will lead to the selection of the most appropriate model for the design of a fault-tolerant mobile platform.

The first kinematic model, mentioned in section 4.2, is the *differential drive* model. The steering principle of this model is based on the driven wheel slippage produced when one driven wheel rotates counterclockwise and the other clockwise. The odometry errors produced by steering operations can easily be corrected. However, the introduction of hardware and/or software into the system to correct this problem only adds more ground for component failure. Therefore, this type of model should not be used for the application in discussion in this thesis work.

The next model is the *Ackerman Steering* kinematic model. This model is widely implemented in the automobile industry. It provides a good platform for the implementation of an odometry measurement system for outdoor navigation. The problem of this model resides in the difficult maneuvering in highly constrained environments.

The *tricycle* model with a front-driven-and-steered wheel is a simplified version of the Ackerman model. Therefore, it presents the same maneuvering limitations. If alternatively the fixed, rear wheels are powered then, the model becomes a differential drive with the same slippage problem.

Tracked vehicles also present odometry-measurement problems. Odometry measurement errors are the biggest in comparison with rest of the models. This platform is only a good choice for outdoors terrenes with very rough surfaces.

The *synchro drive* kinematic architecture has been left for the end of this section because this is the model suggested for the design of the fault-tolerant mobile platform being presented in this thesis work. As mentioned before, this model outperforms the others just presented. All wheels generate equal and parallel force vectors at all times. Therefore, slippage errors are reduced resulting in improved odometry calculations. In addition, this kinematic model enables the platform to head to any point in the horizontal plane without any previous maneuvering. Furthermore, the two degrees of freedom provided by the model to each wheel, increment the number of points at which redundant devices (i.e., more motors, encoders, potentiometers) can be attached to add alternative sources of mechanical power as well as system status information (i.e. odometry, errors, faults, etc).

Chapter 5. Experimental Setup and Test Results

5.1 Introduction

The initial chapters introduced fault tolerance and mobile platforms in a theoretical format. Work on the mobile robotics and fault-tolerant system design were summarized. In addition, important concepts, definitions and methodologies for the design of fault-tolerant systems as well as some applications were included here.

Subsequent chapters took in consideration the information acquired in the review stage to produce different conceptual alternatives for the structural design of a fault-tolerant mobile platform. Once a conceptual design was decided upon, procedures and important details on the design and selection of the main components and subsystems of the platform were explained in detail.

In this chapter, design and construction of a fault-tolerant mobile platform is presented. Experimental results are also reviewed and a discussion is included. In building the prototype, an attempt to replicate the suggested design has been made. However, since the focus of this thesis work is to find an efficient platform design in which to implement hardware redundancy techniques. Consequently, some changes have been made to the suggested design to comply with budget and building constrains.

5.2 Description of the Mobile Platform Testbed

The kinematic model of the platform prototype has been built following the suggestions made in Chapters 3 and 4. That is, the platform is built according to a synchronized drive architecture. It has a circular shape with three wheeled legs. In order to describe the structure of this prototype, its entire system is divided into different subsystems. These are listed as:

- Steering System
- Driving System
- Platform Base
- Control System
- Communication system
- Power Supply

5.2.1 Steering System

The synchronized steering system used in this prototype consists of a set of four 6"-spur gears. One gear is attached to the vertical shaft of each leg. A fourth spur gear interconnects the three legs providing simultaneous wheel orientation as shown in Figure 3.6. The gears are held onto a ½" steering shaft by means of an aluminum hob and two set screws. Free, concentric rotation of the steering shafts is assured through two ball bearings that are housed into the platform base. The torque to produce simultaneous steering of the wheel comes from two stepper motor/gearhead assemblies connected at the end of two of the vertical shafts. The steering angle is affected neither by the particular actuator location nor the motor performing the steering because all the gears used are identical (i.e., gear ratio 1:1).

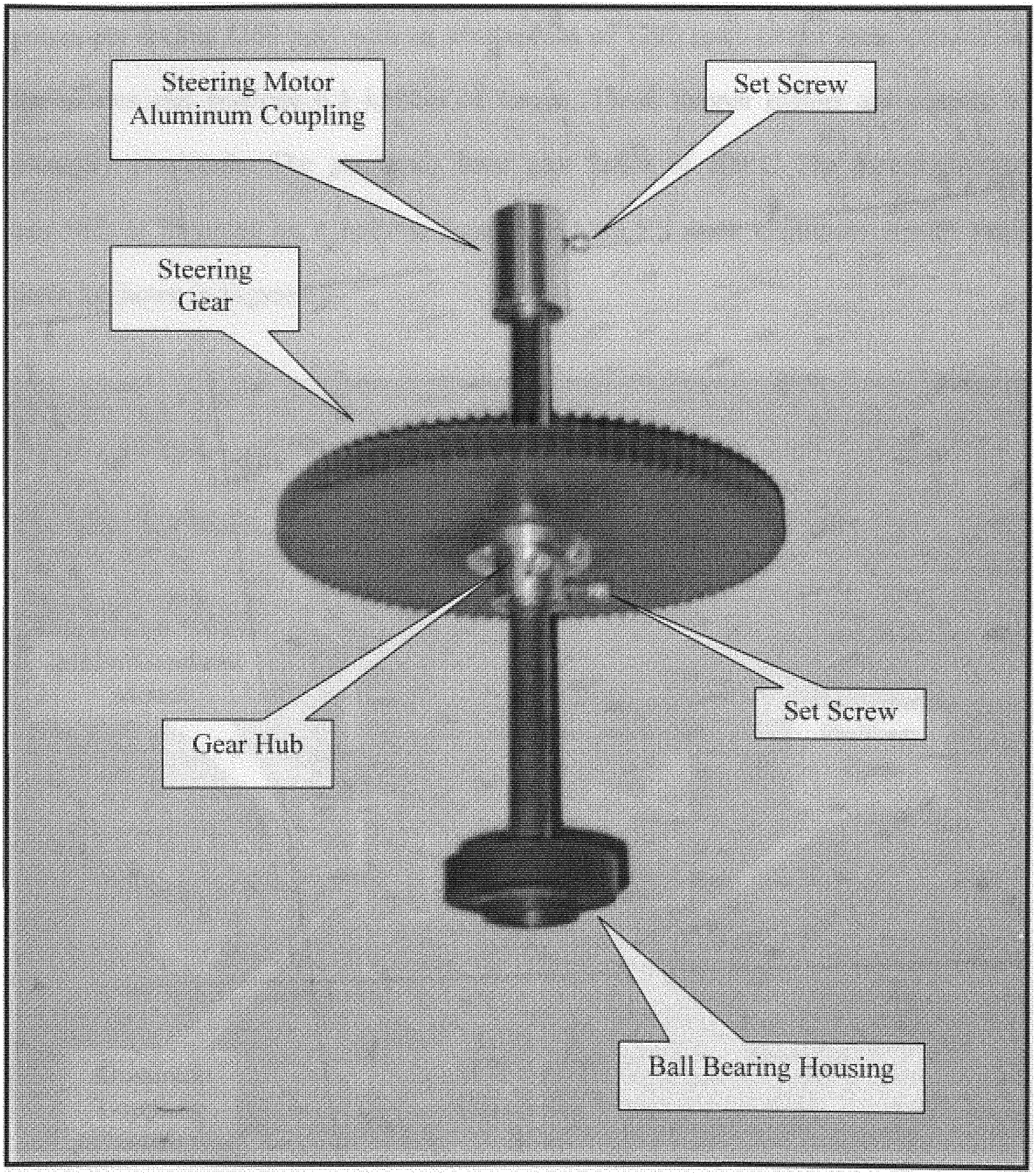

Figure 5.1 Steering Shaft Assembly

The actuators are uni-polar, four-phase stepper motors that can draw up to 1 Amp/phase. These NEMA-23-size motors perform best at a stepping rate of two hundred pulses per second (200 pps) with an acceleration of 1200 pps^2. Under these conditions, the motors can generate 6.5 kgf-cm of torque. These stepper motors are coupled to planetary gearheads. The gearheads have gear ratios of 100:1. The motor/gearhead assembly is mounted on a hollow, squared (3.5"x3.5") stock aluminum square beam piece as shown in Figure 5.2.

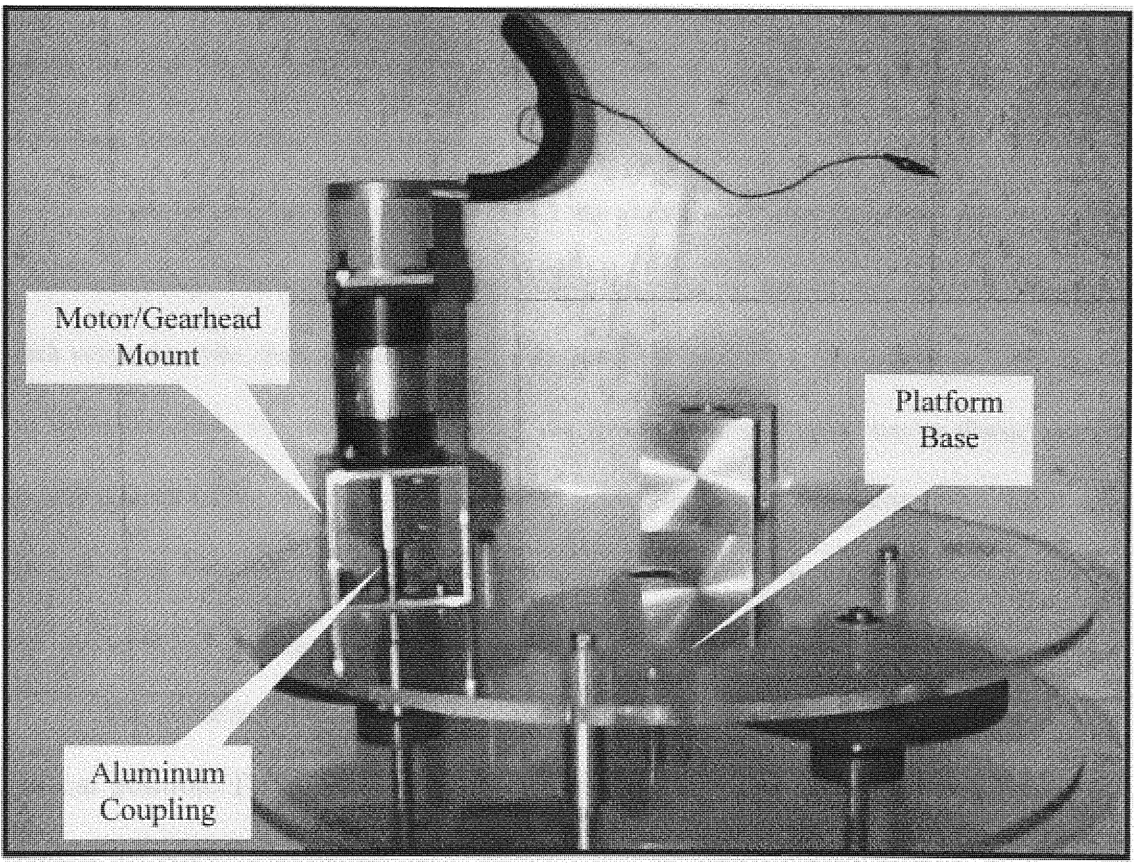

Figure 5.2. Motor/Gearhead Assembly Mounted on a Hollow Beam Piece

This aluminum piece is anchored to the platform base using four 3/16" set screws. The ½" output shafts of the gearheads are coupled the steering shaft of the wheel by means of and aluminum coupling and set screws (see Figure 5.1).

5.2.2 Driving System

The type of wheel available for this prototype was a caster, free-rotating industrial wheel. This type of wheel is mounted on a ¼"-diameter screw that acts as a shaft. The shaft passes through a hole on both sides of the fork stand and two ball bearings housed in the sides of the wheel. The shaft is secured with a lock nut that prevents the shaft from getting loose. These wheels are commonly used in shop floor tables to transport materials, tools and other equipment.

In order to provide driving power to the platform, it was necessary to modify the wheel assembly. The modification consisted in substituting the free rotation of the wheel with respect to the shaft with a press-fed shaft/wheel assembly. Thus, a new shaft has to be manufactured. The center portion of the new aluminum shaft has a diameter that tightly adjusts to the central hole of the wheel. The next step in the shaft firmly adjusts to the gear bore diameter. The shaft ends were machined such that they serve as one side of the ball bearing housing.

The platform driving power is provided by another stepper motor placed in a vertical position. The motor output shaft forms a 90° input-output relationship with the shaft of the wheel. The torque from the motor is transmitted through a set of bevel gears with a ratio of 4:1. In order to align the pinion and the gear, it was necessary to modify the fork stand. First of all, the horizontal upper part of the stand had to be enlarged. This was required to let the motor lay on that horizontal surface and let the pinion attached to its shaft properly mate with the gear attached to the wheel shaft. In addition, the sides of the fork stand had to be machined such that they can house the other side of the ball bearings. Finally, a new coupling to attach the wheel assembly to the vertical shaft was manufactured. The driven wheel assembly can be seen in Figure 5.3.

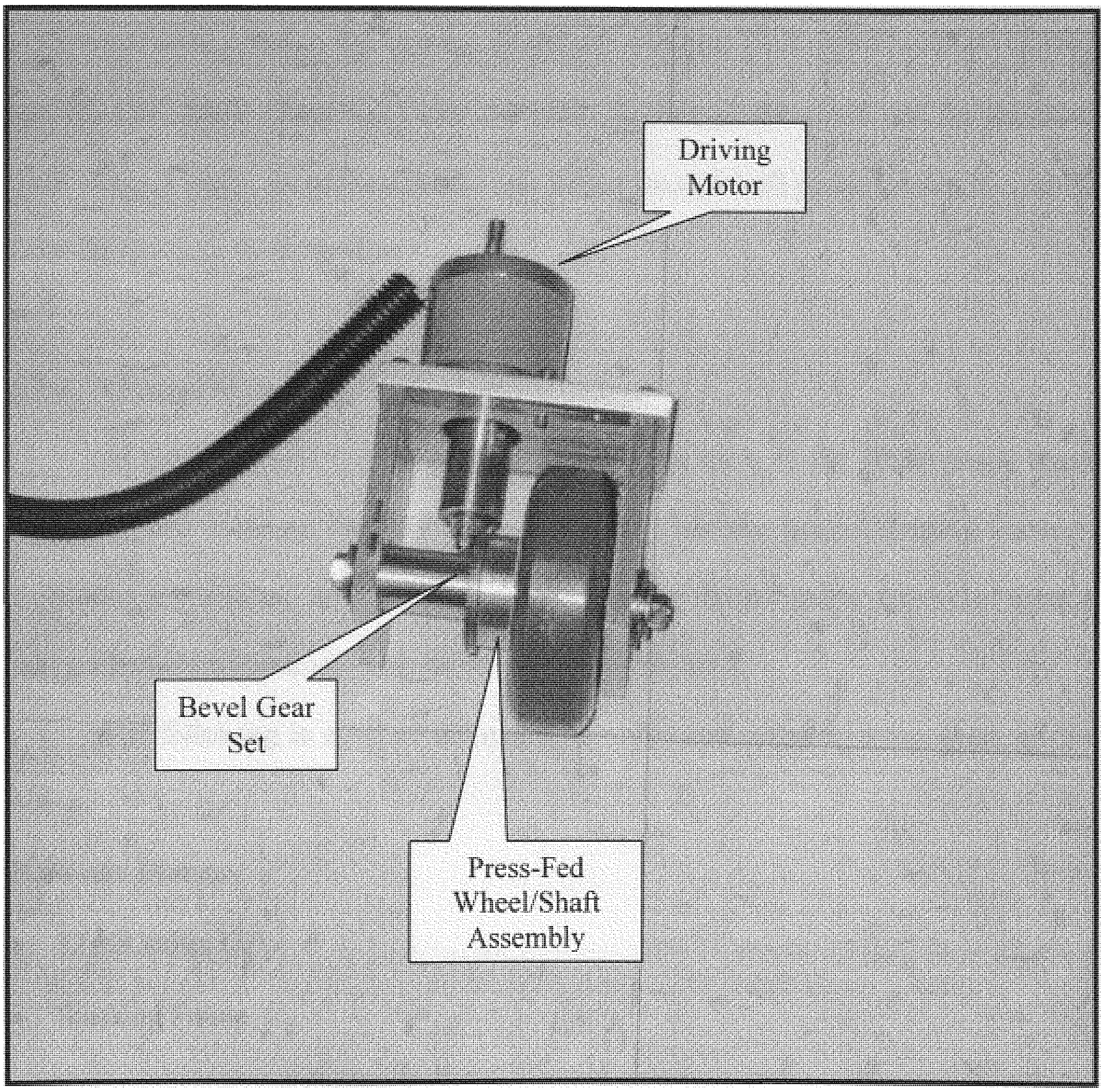

Figure 5.3. Driven Wheel Assembly

5.2.3 Platform Base

The base is simply conformed of two ½ in-thick plexiglass plates. At the center of these 24"-diameter plates a blind through hole was drilled. The purpose of this hole is to keep the housing for the ball bearings of the steering system connecting gear in place. Using this hole as the center, a bolt circle of one-pitch-diameter was drilled to assure a perfect matching between the center gear and any adjacent. These three holes are equally spaced, 120° apart. These holes play the same function as the center one for the other three shafts. Finally, an additional hole was drilled through both plates to route the cables from the motor to the controller.

5.2.4 Control System

At the core of the control system, the prototype has a 32-Bit 386Ex single board computer. This computer has all the capabilities and functions needed for most common embedded applications. Additional I/O modules can be stacked and/or connected using a telephone cable without affecting its networking capabilities.

The following is a list of the major hardware and software components of this board.

Hardware Features

- 32-Bit386Ex CPU
- Stacking Frame
- Automatic Network Connection
- 33 MHz CPU

- Real Time Clock
- Two Serial/Network Ports (RS232C/485)
- 512K FLASH EPROM
- 512K Battery Backed RAM
- 9-15 VDC/300 MAmp. Power Supply

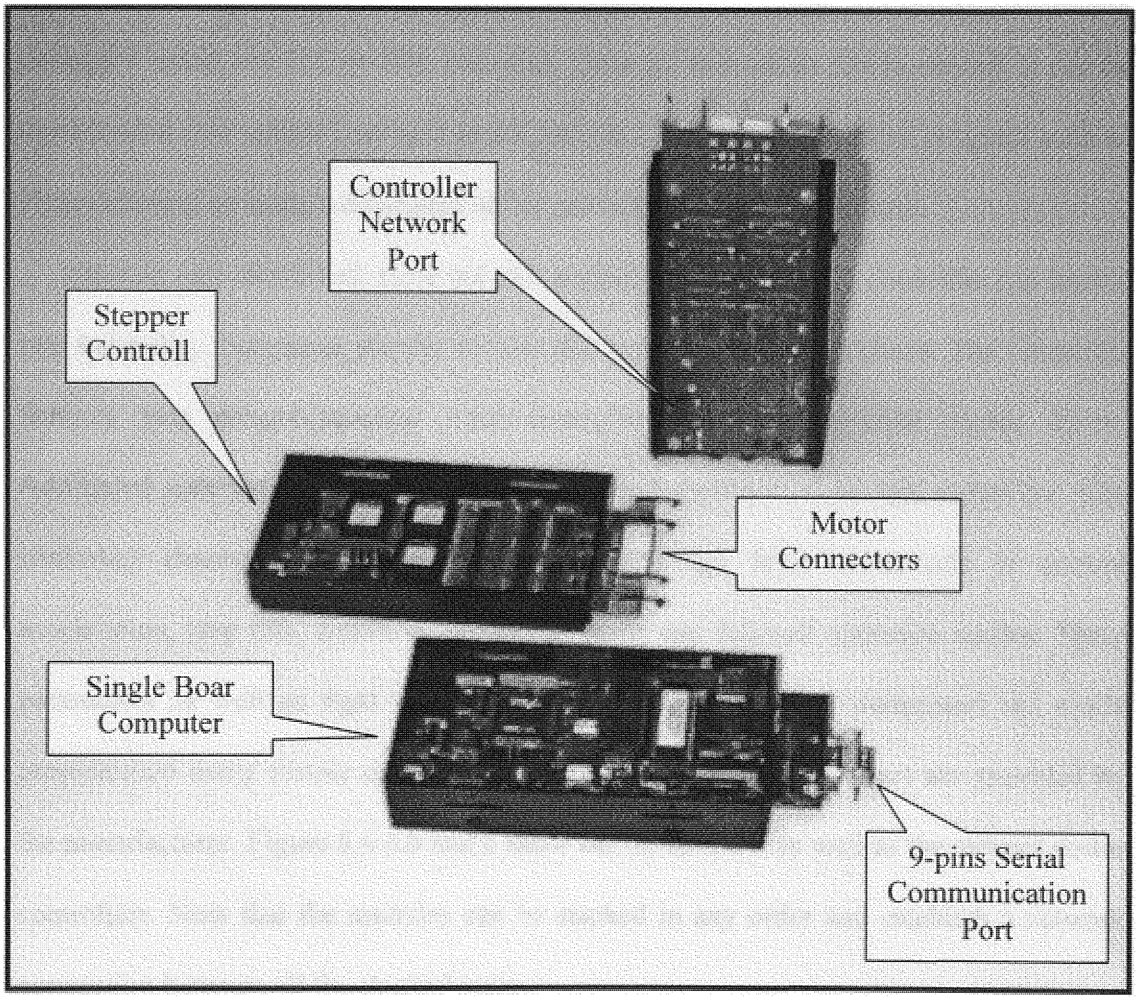

Figure 5.4.a. Single Board Computer and Stepper Motor Controller

Software Features

- On-Board FLASH Programming
- C and C++ Fully Supported
- I/O Driver Library Supplied
- FLASH-DOS

The board can store programs up to 450K of memory. Programs can be written and debugged using any C/C++ IDE compiler. Once an executable file is created, it can be downloaded into memory using a serial port on a PC and any terminal program such as Hyperterminal or using a download utility provided by the manufacturer.

Stacked onto this board, there are two stepper controllers. Each controller can handle two uni-polar, four-phase stepper motors that draw up to 1 Amp/phase. The controller features two pre-programmed micro-controller and two uni-polar motor drives. Additional external controllers and drivers can be connected to this controller. The controller channels can be programmed with high level commands to control acceleration, step rate, direction, number of steps and full/half operation modes. These parameters as well as right and left limits, indexing and stop parameters are easily programmed using simple calls in C/C++. DOS and Windows drivers are provided by the manufacturer. Figure 5.4.a,b and c show the on-bard PC as well as the stepper motor controllers. Note that the modules can be stacked in any order and establish a network connection between all the stacked units.

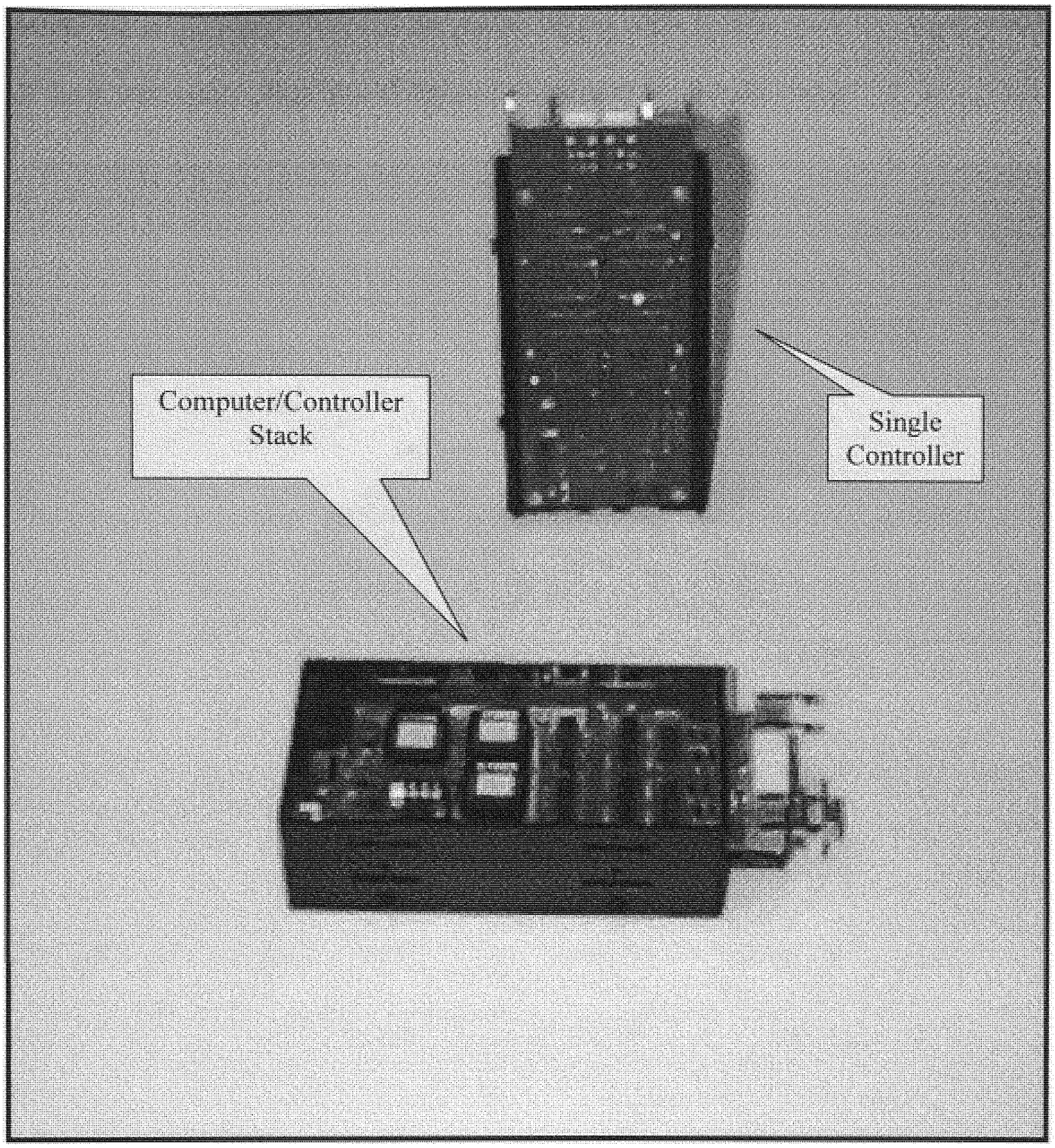

Figure 5.4.b. One Controller and Single Board Computer Network

Network set up steps and hook-up drawings are explained in detail in the literature provided by the manufacturer. The motor phases cables are plugged into the motor connector as shown in the literature. This is the two phase common wires are also connected to the positive pole of the power supply while a cable from the negative pole of the power supply plugs into its designated position to close the circuit.

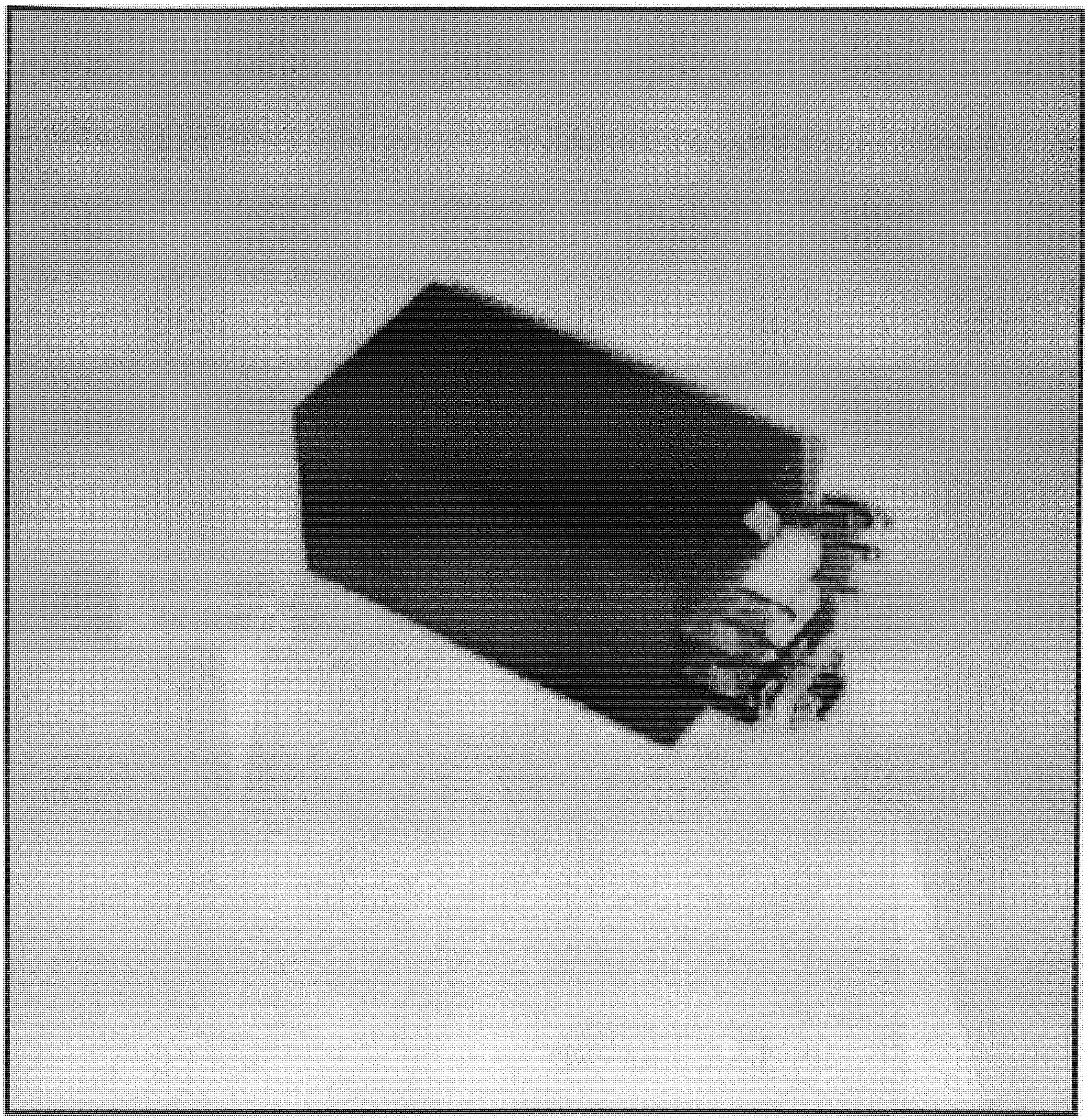

Figure 5.5.c. Complete Controller Assembly

5.2.5 Communication System

The communication between a remote, host personal computer (PC) and the on-board single board computer (SBC) is accomplished wirelessly. Wireless communication gives the mobile platform autonomy and great freedom of motion. In addition, not having to pull a long and heavy tether increases the load-carrying capacity of the prototype. This wireless data link uses a spread-spectrum data modem. The 900 MHz system can transmit data at 300-38400 bits per second.

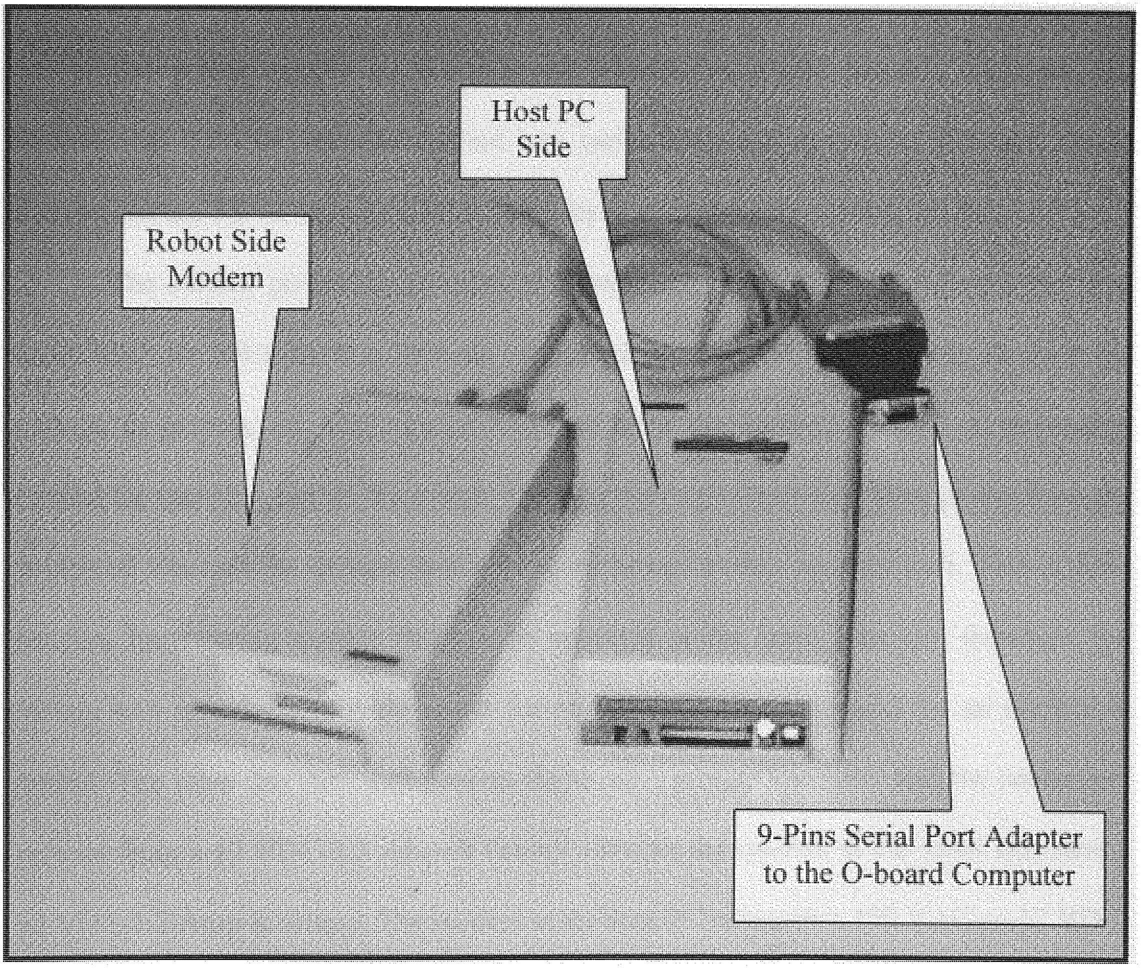

Figure 5.6. Wireless Data Link

The data transmission range is 300’ indoor and 3000’ outdoors. Figure 5.8 shows the modems for both host and on-board PC ends. The modem on the host PC side uses a 120 VAC-to-12 VDC 220 Mamps adapter. This modem connects directly through its 25-Pins serial port to any serial port (i.e., 25-pins or 9-pins port) in the host computer. The on-board modem connects to the serial port of the on-board PC using 25-pins to 9-pins serial as shown in Figure 5.6.

5.2.6 Power Supply

The power supply on the platform consists of sealed led-acid batteries. All batteries are 12 VDC and the current rate varies from 1.3 Amp-h to 4 Amp-h. Their dimensions are 3.5 in. long by 2.75 in. wide by 4 in. high and weight is 3.8 lb.

The motors’ power supply consists of two cells of two 12 VDC-4 Amp-h batteries each. The two batteries in a cell are connected in series. This connection doubles the voltage to provide 24 VDC that is required from the controller driver to accurately control the stepping rate of the motor. The two steering motors take their power from the same cell whereas the driving motor takes its power from a separate cell. This is done because the steering motors will not be operated at the same time and therefore, the battery will be able to provide power for any of them when required for an acceptable period. The other 24 VDC cell powers the driving motor. The SBC and the controllers operate from a 12 VDC-2.2 Amp.-h battery whereas the modem at the robot side uses a 12 VDC-1.2 Amp-h battery. The modem at the host PC side uses a 120 VAC-to-12 VDC-220 MAmp. AC/DC adapter provided by the manufacturer.

Table 5.1 summarizes technical information about the parts used to build the prototype platform. A picture of the prototype completely assembled is shown in Figure 5.7 below.

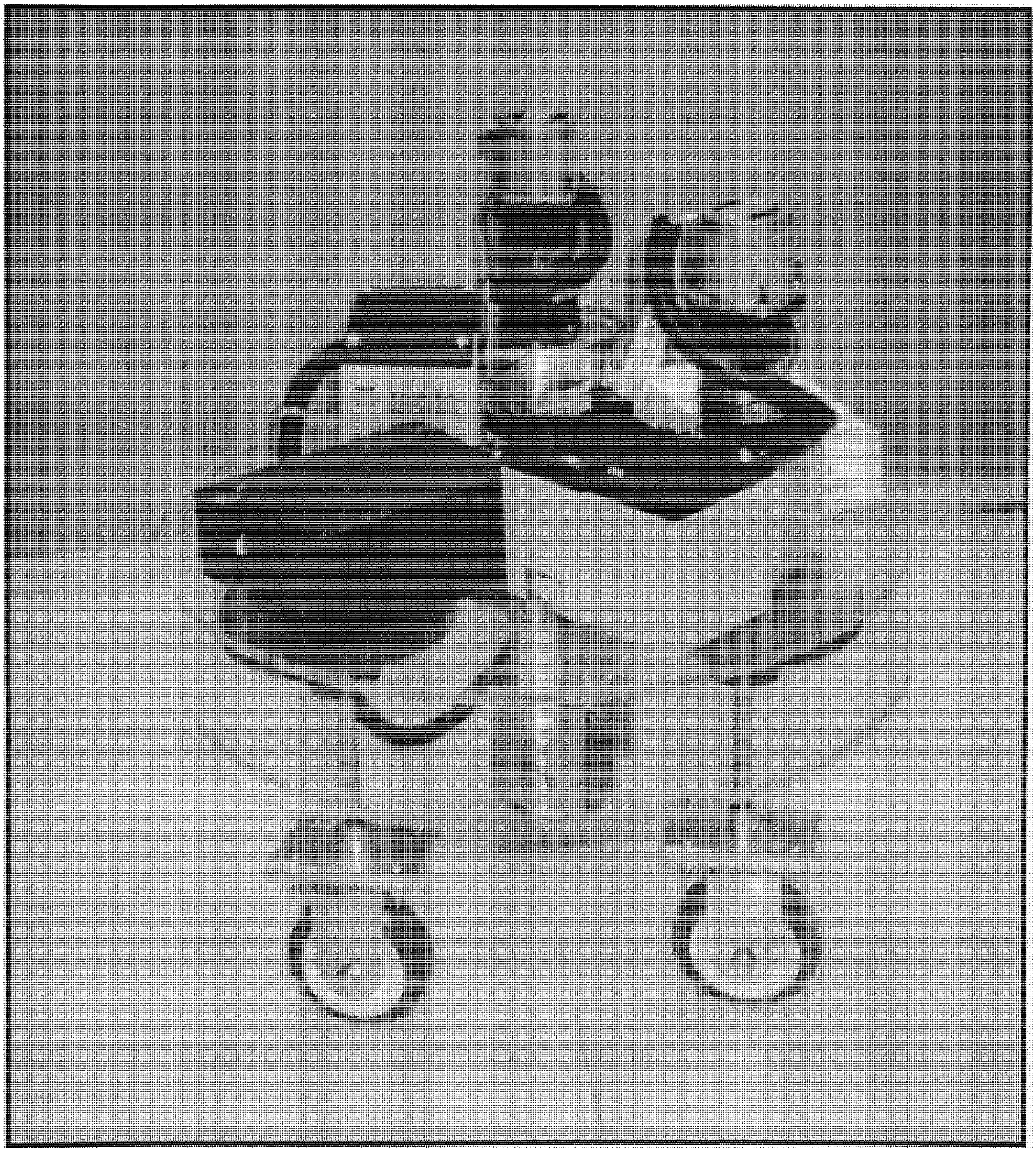

Figure 5.7 Prototype Entirely Assembled.

Part Name	Description	Vendor Information	Unit Price ($)	Qty Req	Total Cost ($)
Motors	Four phase, unipolar stepper,12 V DC, 0.6 Amp/phase, 1.8°/step, NEMA 23.	SKC	54.00	4	216.00
Computer	25 MHz SB 386Ex Computer	RLC	312.00	1	312.00
Memory	512 Flash and RAM memory upgrade	RLC	79.00	1	79.00
Controller	Stepper motor controller	RLC	267.00	2	514.00
Gearhead	Planetary gearheads, 100:1 ratio	CGI	190.00	2	380.00
Bevel Gears	Brass bevel gear set 4:1 ratio, 32 pitch	Small Parts Inc. 13980 N.W. 58th. Court. Miami Lakes, FL. 33014 1(800)220-4242	38.00	2	76.00
Data link	Wireless data link RF modem	Arrick Robotics	650.00	1	650.00
Ball Bearings	Sealed ball bearings. ½" bore diameter	Small Parts Inc.	3.75	8	30.00

Table 5.1. Mobile Platform Components Specs

5.3 Test Results

The main purpose of constructing the prototype is to test the fault-tolerant platform architecture chosen for this work. The tests were performed on tiled flors. During the operation tests, three basic operation modes were closely checked and analyzed. These are as listed below.

- Determination of Acceleration and Velocity Parameters
- Steering without Translation
- Translation on a Straight Line Trajectory
- Moves on a Curved Trajectory

5.3.1 Determination of Acceleration and Velocity Parameters

Stepper motors operate one rotation step at a time. The time interval between consecutive steps determines the rotational speed. The parameter that measures the rotational speed is called step rate. The step rate is measured in steps per seconds (pps). However, stepper motors can not change its stepping rate instantaneously. They need to slowly increase or decrease their rotational velocity to the required level. This is called acceleration or slope and is measured in steps per second squared (pps^2).

In order to operate stepper motors under a particular load, the appropriate combination of step rate and slope should be determined. In the particular case of this prototype, the right combination was found through a simple trail and error experiment.

The steering system motors were found to require a combination of 1200 pps^2 (slope) and a step rate of 350 pps. This rotational speed is equivalent to 1.75 revolutions per second. This speed is further reduced 100 times in the gearhead (i.e., gear ratio 100:1). This provides a smooth steering speed of 6.3° per second.

The driving motor was found to have slightly different requirements. After several tests, the correct combination turned out to be 1150 pps^2 and 450 pps for slope and step rate, respectively. This step rate is equivalent to 0.56 rev/s if the reduction of 4:1 at the bevel gear set is taken into account. The platform uses 4"-diameter wheels that translate the rotational speed into a platform's translation speed of 7 in/s. At this speed, the platform moves smoothly and turns without undesirable jerking or jumping.

5.3.2 Steering without Translation

Steering in place without translations is one of the most interesting features of this synchronized drive architecture. This enables the platform to orient itself towards any direction in the horizontal plane without complicated maneuvering.

This test had one objective: evaluate the backlash effect on the steering accuracy and the repeatability in achieving a particular orientation while the platform is at rest. For this test, the imaginary central line of the wheel was aligned with a line drawn on the floor. Then, one of the steering motors was given the required commands to rotate the wheels 90° in both directions. When the motor stopped, the new orientation of the imaginary central line of the wheel was drawn on the floor. This way, it possible to measure the angle of rotation the steering system actually turned.

The test showed that 47 times out of 50 tries (i.e., 94% of the tries) performed on tiled floor, the steering returned to the initial position within 5° or less. This may be considered a poor steering accuracy as it means that the platform drifts 1.05" away from the initial line of the target for every foot traveled.

From this test, it is concluded that the steering system is capable of repeating a predetermined orientation with reasonable frequency. However, the backslash in the gears affects the steering accuracy significantly. Therefore, when constructing a model for real-world operations, the designer should keep close attention to this problem. It should also be noted that problems due to slippage and backslash are common in any platform design and intelligent navigation schemes and close-loop controllers may be used to compensate for position errors.

5.3.3 Translation on a Straight Line Trajectory

After the operation parameters of the driving motor were determined, the most basic type of platform motion was tested. Traveling on a straight-line trajectory was measured by drawing a 5'-line on the floor. The platform was aligned with the line on the floor and commands were given for the motor to travel the distance. One of the steering motors was powered to serve as brake for the steering system.

In the 20 tries performed, the platform always traveled smoothly, and reached the target with an error of ±1 in. or less. Misalignments and component stiffness was the cause for the errors.

5.3.4 Moves on Curved Trajectories

Curved trajectories are achieved by a synchronized actuation of the steering and driving systems. Since the prototype currently operates in open-loop mode only, it is very difficult to make the platform follow a predetermined path with acceptable accuracy. This difficulty increases with the mechanical problems explained before. Instead, some arbitrary curved trajectories were tested for translation smoothness. In this sense, all tests were successful.

One should keep in mind that the focus of this work is to find the appropriate platform architecture to introduce redundant components and fault tolerance capabilities into the platform design. The problems affecting the platform's performance are manufacturing-related and can be easily eliminated if proper tooling and materials are used.

Chapter 6. Conclusions and Recommendations

6.1 Thesis Overview

The work presented in this thesis is a summary of two years of research on fault-tolerant robotic systems. This research has evolved form the evident need for mobile platform designs that take into account reliability and fault tolerance issues in mechanical systems. Particularly, those modular designs that provide hardware redundancies to overcome a failure while safely continue operating with minimal influence on system performance.

The first chapter summarized the knowledge obtained through a vast literature review on previous research work. These include topics such as the effect of kinematic redundancy on the overall system fault tolerance, fault tolerance capacity measure methods, autonomous mobile robots and navigation systems. Furthermore, an explanation of the uniqueness of this work is emphasized in the final section of this chapter.

Chapter 2 is dedicated to introducing important concepts, definitions and methodologies for the design of fault-tolerant systems. In addition, some applications of fault tolerance are detailed to complete this chapter.

The information acquired in preceding chapters was taken in consideration to produce different conceptual alternatives for the structural design of a fault-tolerant mobile platform in the third chapter. This chapter presented procedures and important details on the design and selection of the main components and subsystems of the design alternative that was chosen as more suitable for the task.

The next chapter presented the most commonly used kinematic model of mobile platforms. The advantages of implementing a *synchro drive* model were explained in depth.

Chapter 5 was dedicated to describe the construction a mobile platform prototype as well as the experimental results conducted on the prototype. In addition, a table with useful information on the parts used to build it.

6.2 Conclusions

During the preparation of this thesis work, the knowledge obtained form the initial literature review was used to develop a conceptual design of a fault-tolerant mobile platform. Different hardware redundancy techniques were considered to provide the platform design with a driving and steering system capable of undergoing a failure of one of the actuators and continue to work with no effect in its performance.

In order to practically show the implementation of these techniques, a working prototype was constructed and tested. The test with the prototype showed that the chosen kinematic architecture enables the designer to introduce redundant components and used different fault-tolerant schemes in the design of a mobile platform. If the manufacturing defects are disregarded, it can be safely concluded that the steering and driving systems system worked accurately and smoothly using redundant components.

Working in this prototype has been an experience that helps enhance the mechanical design skills of anyone involved in this type of project. Making decisions on component requirements, finding appropriate sources for most appropriate and economical materials and components as well as dealing with component manufacturers are just a few skills one can develop and exorcise when involve in a project of this type. Time management skills were also put in practice during the completion of this work. In other words, the introduction of hardware redundancy and fault tolerance into the design of a mobile platform has been accomplished successfully not only from the research point of view, but also from a professional formation enhancement perspective.

6.3 Recommendations

This thesis work constitutes the base for further experimentation in different areas in the robotic field. This section gathers the recommendations concerning the mechanical design and the sensor system for future work with this prototype as well as other systems. The recommendations with respect to the mechanical design are listed as follows:

- Design a platform base with a material that allows for a better ball bearing housing.
- Use more precise machining to make the holes that house the ball bearings on the base.
- Redesign the couplings that attach the gear heads and the wheel assembly with its vertical shaft.
- Manufacture a hollow shaft for each wheel assembly to draw motor cables through it up to the controller connector.

- Improve the matching of the steering gears by manufacturing the spacers more precisely.
- Design a fixture to precisely align the wheels.
- Design and place fixtures on top of he base to hold the batteries and electronics in place.
- Experiment the possibility of designing a battery holder attached to the bottom surface of the platform base without interfering with the platform steering.

There are also recommendations to improve the electronic system. These are mentioned below.

- Implement a system of optical encoders to operate the platform in a closed-loop control system.
- Evaluate different techniques for motor failure detection and identification.
- Experiment with different encoder systems using fault-masking schemes to minimize the odometry information errors introduced into the navigation system due to slippage.
- Experiment with a multi sensor-based mapping, path planning and navigation systems.
- Implement, experiment and evaluate performance and reliability issues using teleoperated versus autonomous fault-tolerant navigation systems.
- Construction and test of a prototype using a servo system.
- Compare both prototypes in terms of performance and fault tolerance capabilities.

- Expose the prototypes in conferences and workshops in order to promote Florida International University's advances in robotic research and obtain funding for other application areas.

In order to further improve the platform design and assure the highest level of component reliability, the following items are recommended.

- Evaluate the prototype's performance in real world applications such as nuclear power plants.
- Evaluate component reliability data to find the most appropriate for the application.
- Make improvements in the design according to the results from items above
- Construct a second prototype.

Bibliography

[1] Anderson T., Lee P.A., "Fault Tolerance: Principles and Practice". Computing Laboratory, University of Newcastle Upon Tyne, England. Prentice-Hall International, Inc., 1981.

[2] Borenstein, J., Everett, H.R., Feng, L., 1996, "'Where am I?' Sensors and Methods for Mobile Robot Positioning." Technical Report, The University of Michigan.

[3] Brooks, R.A., "Elephants Don't Play Chess", MIT Artificial Intelligence Laboratory, Cambridge, MA 02139, USA.

[4] Ferrell, C., "Many Sensors, One Robot". Artificial Intelligent Laboratory, Massachusetts Intitute of Technology 545 Technology Square, Room 741, Cambridge, MA 02139. (617)253-7884. ferrell@ai.mit.edu.

[5] Halmintom, D.L., Bennett, J.K., Visinsky, M.L., Member, IEEE, Walker, I.D. Member, IEEE, Cavallaro, J.R., Member, IEEE, "Fault-Tolerant Algorithms and Architectures for Robotics". Department of Electrical and Computer Engineering, Rice University, Houston, TX 77251-1892.

[6] Horak, D.T., "System Failure Isolation in Dynamic Systems". Allied-Signal Aerospace Company, Columbia, MD 21045.

[7] Jamshide, M., Eicker, Patric J., "Robotics and Remote Systems for Hazardous Environments", Prentice-Hall series on environmental and intelligent manufacturing systems; v. 1. ISBN: 0-13-782590-0.

[8] Khatib, O., "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots", 1985 IEEE International Conference on Robotics and Automation, St. Louis, MO, March 25-28, 1990, pp. 500-505.

[9] Koren, Y., Borenstein, J., "Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation", Department of Mechanical Engineering and Applied Mechanics, The University of Michigan, Ann Arbor. Proceedings of the 1991 IEEE International Conference on Robotics and Automation, Sacramento, CA-April 1991.

[10] Koren, Y., Borenstein, J., "Real-Time Obstacle Avoidance for Fast Mobile Robots in Clustered Environments", The 1990 IEEE International Conference on Robotics and Automation, Cincinnati, OH, May 13-18, 1990, pp. 14-23.

[11] Latombe, J-C., "A Fast Path Planner for a Car-Like Indoor Mobile Robot", Robotics Laboratory, Department of Computer Science, Stanford University, Stanford, CA 94305. latombe@cs.stanford.edu.

[12] Maciejewski, A.A., Lewis, C.L., "An Example of Failure Tolerant Operation of a Kinematically Redundant Manipulator", School of Electrical Engineering, Purdue University, West Lafayette, IN 47907-1285.

[13] Maciejewski, A.A., "Fault-Tolerant Properties of a Kinematically Redundant Manipulator" School of Electrical Engineering, Purdue University, West Lafayette, IN 47907-1285.

[14] Maciejewski, A.A., Goel, M. and Balakrishnan, V., "The Design of Control Strategies Tolerant to Undetected Failures in Kinematically Redundant Manipulators". School of Electrical Engineering, Purdue University, West Lafayette, IN 47907-1285.

[15] Meystel, A., "Autonomous Mobile Robots", World Scientific Series in Automation, vol. 1. World Scientific Publishing Co. Pte. Ltd. 1991. ISBN 9971-50-088-4.

[16] Monteverde, V. Tosunoglu, S. "A Measure of Fault Tolerance for Robotic Manipulators", Department of Mechanical Engineering, Florida International University, Miami, FL 33199 USA. E-mail: tosun@fiu.edu.

[17] Parker, L.E., "Alliance: An Architecture for Fault Tolerant, Cooperative Control of Heterogeneous Mobile Robots", Center of Engineering Systems Advanced Research, Oak Ridge National Laboratory, P.O. Box 2008. Oak Ridge, TN, 37831-6364.

[18] Ralph, K.S., Pai, D.K., "Fault Tolerant Locomotion for Walking Robots", Department of Computer Science, University of British Columbia. Vancouver, Canada.

[19] Roumeliotis, S.I., Sukhatme, G.S., and Bekey, G.A., "Fault Detection and Identification in a Mobile Robot Using Multiple-Model Estimation", Department of Computer Science, Institute of Robotics and Intelligent Systems, University of Southern California, Los Angeles, CA 90089-0781.

[20] Sitarama Iyengar, S., Elfes, A., "Autonomous Mobile Robots-Perception Mapping and Navigation", IEEE Computer Society Press, Vol. 1. ISBN 0-8186-2018-8.

[21] Sitarama Iyengar, S., Elfes, A., "Autonomous Mobile Robots-Control, Planning, and Architecture", IEEE Computer Society Press. Vol. 2. ISBN 0-8186-2116-8.

[22] Sousa, B.*, Pereira, F. L.*, da Silva, E. P.**, Martins, A.*, Matos, A.*, Almeida, J.*, Cruz, N.*, Tunes, R.*, Cunha, S.*, "On the Design and Implementation of a Mobile Robotic System", *Instituto de Sistemas e Robotica-Porto e DEEC-Faculdadae de Engenharia da Universidade do Porto, Rua dos Bragas, 4099 Porto Codex, Portugal. **Instituto de sistemeas e Robotica-Porto e Instituto Superior de Engenharia do porto, R. de S. Tome, 4200 Porto, Portugal.

[23] Tosunoglu, S., Monteverde, V., "Development of a Measure for Fault Tolerance Capacity of Robotic Structures", Department of Mechanical Engineering, Florida International University, Miami, FL 33199 USA. E-mail: tosun@fiu.edu.

[24] Tosunoglu, S., Monteverde, V., "Kinematic and Structural Design Assessment of Fault-Tolerant Manipulators", Department of Mechanical Engineering, Florida International University. Miami, FL 33199 USA.

[25] Tosunoglu, S., "Kinematic and Structural Design Issues in the Development of Fault-Tolerant Manipulators". Department of Mechanical Engineering, Florida International University, Miami, FL 33199 USA. E-mail: tosun@fiu.edu.

[26] Tosunoglu, S., Monteverde, V., "Effect of Kinematic and Dual Actuation on Fault Tolerance of Robot Manipulators", Department of Mechanical Engineering, Florida International University, Miami, FL 33199 USA. E-mail: tosun@fiu.edu.

[27] Tosunoglu, S., Tesar, D., Sreevijayan, D., "Architectures for Fault-Tolerant Mechanical Systems", Department of Mechanical Engineering, University of Texas at Austin, BRC/MER, Mail Code 79925, Austin, TX, 78712-1100.

[28] Tosunoglu, S., Ting, Y., Freeman, R., "Torque Redistribution Method for Fault Recovery in Redundant Serial Manipulators", Department of Mechanical Engineering, University of Texas at Austin, BRC/MER, Mail Code 79925. Austin, TX 78712-1100.

[29] Tso, K.S.*, Hecht, M.*, Marzwell, N.I**, "Fault-Tolerant Robotic System for Critical Application", SoHaR Incorporated, Beverly Hills, CA, 90211. **Jet Propulsion Laboratory, Pasadena, CA. 91109.

[30] Vasseura, Hubert A., Pina, Francois G., and Taylor, Jack R.b, "Navigation of Car-Like Mobile Robots in Obstructed Environments Using Convex Polygonal Cells", Oak Ridge National Laboratory, Center for Engineering Systems Advanced Research, P.O Box 2008, Building 6025. Engineering Application Graphics Laboratory, Application Support Dept., Central Engineering, Y-12, Building 9111, MS-8201, Oak Ridge, TN 37831-8201, USA.

[31] Visinsky, M.L., Walker, I.D, Cavallaro, J.R, "Leyered Dynamic Fault Detection and Tolerance for Robots", Department of Electrical and Computer Engineering, Rice University, Houston, TX, 77251-1892

[32] Visinsky, M.L., Walker, I.D, Cavallaro, J.R, "Robotic Fault Detection and Fault Tolerance: a Survey", Department of Electrical and Computer Engineering, Rice University, Houston, TX, 77251-1892

[33] Visinsky, M.L., Walker, I.D, Cavallaro, J.R, "Expert System Framework for Fault Detection and Fault Tolerance in Robotics", Department of Electrical and Computer Engineering, Rice University, Houston, TX, 77251-1892. Computers Elect. Engineering, vol. 20, No. 5, pp. 421-435, 1994.

[34] Visinsky, M.L., Member, IEEE, Walker, I.D, Member, IEEE, Cavallaro, J.R, Member, IEEE, "A Dynamic Fault-Tolerant Framework for Remote Robots". Department of Electrical and Computer Engineering, Rice University, Houston, TX, 77251-1892. IEEE Transactions on Robotics and Automation, vol. 11, No. 4, August 1995.

[35] Willlsky, A. S., "A Survey of Design Methods for Failure Detection in Dynamic Systems", Automatica, vol. 12, pp. 601-611, Pergamon Press, 1976.