PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA

DETECCIÓN Y DIAGNÓSTICO DE FALLAS EN ROBOTS MÓVILES COOPERATIVOS

RODRIGO ARNALDO CARRASCO SCHMIDT

Tesis para optar al grado de
Magíster en Ciencias de la Ingeniería

Profesor Supervisor:
ALDO CIPRIANO Z.

Santiago de Chile, Agosto de 2004

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
Departamento de Ingeniería Eléctrica

DETECCIÓN Y DIAGNÓSTICO DE FALLAS EN ROBOTS MÓVILES COOPERATIVOS

RODRIGO ARNALDO CARRASCO SCHMIDT

Tesis presentada a la Comisión integrada por los profesores:

ALDO CIPRIANO Z.

ÁLVARO SOTO A.

MIGUEL TORRES T.

JUAN DE DIOS RIVERA A.

Para completar las exigencias del grado de
Magíster en Ciencias de la Ingeniería

Santiago de Chile, Agosto de 2004

Pluralitas non est ponenda sine necesítate.

William of Ockham 1285-1349

A mis padres Rodrigo y Loreto y a Daniela, quienes me apoyaron en todo momento.

En recuerdo de Alicia Casanueva de Carrasco.

AGRADECIMIENTOS

Quisiera aprovechar estas líneas para agradecer a todos aquellos que de una u otra forma entregaron su apoyo para la realización de esta tesis.

En primer lugar doy las gracias a mi familia por haber estado a mi lado en todo momento. Todos los logros obtenidos no habrían sido posibles de no ser por su apoyo incondicional y el ánimo que me entregaron. Agradezco a mis padres, Rodrigo y Loreto, por ayudarme en todo lo posible. Su cariño y su confianza han sido factores fundamentales para seguir avanzando. Les agradezco también las revisiones que hicieron de esta tesis, las que permitieron mejorar muchos aspectos y hacerla más entendible para todos.

Agradezco especialmente el apoyo de mi polola Daniela, quién, al igual que mis padres, siempre estuvo al tanto de todos los avances y logros alcanzados. La compañía, el cariño, la paciencia y el amor entregados me dieron la fuerza para seguir trabajando cuando estaba cansado y sus puntos de vista más humanistas me ayudaron a pensar en otras líneas de acción que resultaron mejores. Le agradezco su ayuda en la construcción del equipo de robots y las revisiones que hizo a este documento. También quisiera agradecerle el apoyo a su familia: sus padres Max y Cecilia y a sus hermanos, quienes me entregaron su cariño y me apoyaron muchísmo.

Quisiera agradecer también el apoyo y la entrega de mi profesor supervisor, don Aldo Cipriano, quién siempre tuvo confianza en el tema de investigación y dedicó su tiempo y su paciencia a revisiones y largas conversaciones del tema que desembocaron en estos resultados. A él le debo gran parte de lo logrado en esta tesis.

También debo agradecer a todos mis amigos y compañeros: Rolo, Sergio, Michael y José Luis, quienes estuvieron al tanto de mis avances y me apoyaron

con nuevas ideas y especialmente con ánimo. Agradezco la ayuda entregada por Michael y José Luis en la creación de las placas del robot.

Un agradecimiento especial a todas las personas del Departamento de Ingeniería Eléctrica, quienes siempre mostraron interés en el tema y me apoyaron como fuera posible para hacer más grato el trabajo: a Marcelo Guarini y sus excelentes conversaciones; a Betty y Elena, cuyo apoyo durante toda la carrera fue impagable; y a Errol quién me facilitó materiales y me ayudó en la confección de las placas.

Agradezco también a los miembros de la comisión evaluadora, cuyos comentarios y sugerencias sirvieron para mejorar la calidad de este trabajo: don Álvaro Soto, don Miguel Torres y don Juan de Dios Rivera.

Finalmente, quisiera agradecer el apoyo económico entregado por el proyecto Fondecyt nº 1020741 llamado "Detección y Diagnóstico de Fallas en Sistemas Nolineales y Variantes en el Tiempo", a la empresa Dow Chemical Company por el premio otorgado en 2002 y a Conicyt por la entrega de la Beca de Asistencia a Congresos Internacionales destinada a presentar uno de los trabajos resultantes de esta tesis en el IEEE International Conference on Fuzzy Systems, en julio de 2004 en Budapest, Hungría.

ÍNDICE GENERAL

Pág.

DEDICATORIA ii

AGRADECIMIENTOS iii

ÍNDICE DE TABLAS viii

ÍNDICE DE FIGURAS ix

RESUMEN xii

ABSTRACT xiii

1. Introducción 1
   1.1. Motivación 1
   1.2. Descripción del Problema y Planteamiento de Objetivos 2
   1.3. Descripción de la Tesis 3
   1.4. Contribuciones Originales 4

2. Marco Teórico 5
   2.1. Historia de la Robótica 5
   2.2. Sistemas de Robots Cooperativos 19
   2.3. Detección y Diagnóstico de Fallas en Sistemas Robóticos 29
      2.3.1. Detección, Diagnóstico y Corrección de Fallas 31
      2.3.2. Sistemas de Detección y Diagnóstico Aplicados a Robótica 34
   2.4. Modelo Matemático de un Robot Móvil 39
      2.4.1. Ecuaciones Cinemáticas 41
      2.4.2. Ecuaciones Dinámicas 44

2.4.3. Ecuaciones Electromecánicas 46
2.4.4. Modelo del Robot de Dos Ruedas 48
2.5. Sistema de Control 49

3. Arquitectura Multicapa de Detección y Diagnóstico de Fallas para Robots Cooperativos 53
3.1. Descripción de las Fallas a Estudiar 53
3.2. Descripción de la Arquitectura 59
3.3. Capa 1: Capa de Detección de Fallas Fatales 61
3.3.1. Identificación Bayesiana usando Múltiples Filtros de Kalman 61
3.3.2. Modelos Utilizados y Parámetros de Detección y Diagnóstico 65
3.3.3. Simulación de la Capa 1 72
3.3.4. Análisis de Desempeño de la Capa 1 77
3.3.5. Extensión de la Capa 1: Interacción con el Sistema de Control 81
3.3.6. Adaptación para Diagnosticar Múltiples Fallas 84
3.3.7. Limitaciones del Método Utilizado 86
3.4. Capa 2: Capa de Detección para Robots Cooperativos 88
3.4.1. Detección y Diagnóstico en Sensores Redundantes 89
3.4.2. Criterios de Detección y Diagnóstico 94
3.4.3. Detección y Diagnóstico de Fallas en el Sistema de Comunicación Inalámbrico 97
3.4.4. Análisis de Desempeño de la Capa 2 99
3.5. Interacción entre Capas 100

4. Implementación de la Arquitectura 103
4.1. Descripción del Sistema de Robots Cooperativos 103
4.2. Implementación de Arquitectura y de las Fallas 107
4.2.1. Capa de Detección de Fallas Fatales 107
4.2.2. Capa de Detección para Robots Cooperativos 109
4.3. Resultados y Validación de la Arquitectura 111

4.3.1. Capa de Detección de Fallas Fatales 111
4.3.2. Capa de Detección para Robots Cooperativos 115

5. Conclusiones y Trabajos Futuros 119
5.1. Análisis del Cumplimiento de los Objetivos 119
5.2. Trabajos Futuros 121

BIBLIOGRAFÍA 123

ANEXOS 130

ANEXO A. Filtro de Kalman 131
A.1. Filtro de Kalman para Sistemas Lineales 131
A.2. Filtro de Kalman Extendido 134

ANEXO B. Algoritmos Genéticos 137

ANEXO C. Publicaciones 142

ÍNDICE DE TABLAS

Pág.

Tabla 3.1: Parámetros de la Simulación de Fallas. 72
Tabla 3.2: Matriz de Confusión de la Capa 1. 78
Tabla 3.3: Tiempos de Detección de la Capa 1 80
Tabla 3.4: Tiempos de Diagnóstico de la Capa 1. 81

ÍNDICE DE FIGURAS

Pág.

Figura 2.1: El Clepsydra o Reloj de Agua. 6
Figura 2.2: Robot de la obra R.U.R. de Karel Capek. 7
Figura 2.3: La tortuga Elsie, de Grey Walter. 8
Figura 2.4: Silla de Ruedas Inteligente Wheelesley del MIT. 10
Figura 2.5: El primer brazo biónico, el EMAS. 11
Figura 2.6: Sharkey, el primer robot autónomo. 12
Figura 2.7: El Rover 2 y su predecesor. 13
Figura 2.8: El Beagle 2, enviado por la comunidad europea a Marte. 14
Figura 2.9: Vista de la superficie marciana por el robot Spirit 15
Figura 2.10: La tercera generación de AIBO, el modelo ERS-7. 16
Figura 2.11: Robot Bípedo en el RoboCup 2003. 17
Figura 2.12: Robot Creado con LEGO Mindstorms. 18
Figura 2.13: Afiche del Abierto de RoboCup de Alemania de 2003. 22
Figura 2.14: Partido de la Liga F180 del RoboCup 2003. 23
Figura 2.15: Carnegie Mellon Hammerheads 24
Figura 2.16: Robot Humanoide en el RoboCup 2003. 25
Figura 2.17: Robot ANT del MIT AI Lab. 26
Figura 2.18: Grupo de Robots Heterogéneos. 27
Figura 2.19: CMU Millibots. 28
Figura 2.20: Sistema Multi-Robot Tolerante a Fallas. 31
Figura 2.21: Diagnóstico de Fallas en Base a Modelos. 33
Figura 2.22: Esquema de un Robot Móvil de Dos Ruedas. 40
Figura 2.23: Variables de Posición Absoluta del Robot. 41
Figura 2.24: Relaciones Geométricas para el Ángulo de Giro. 42
Figura 2.25: Esquema del Sistema Electromecánico. 47
Figura 2.26: Estructura de Control en un Sistema Cooperativo de Robots. 50
Figura 2.27: Esquema Habitual de la Capa de Control de Nivel Inferior. 51

Figura 3.1: Comparación de Sistemas Básicos de Localización. 54
Figura 3.2: Navegación con una Falla en la Brújula Digital. 56
Figura 3.3: Estimación de φ con una Falla en la Brújula Digital. 57
Figura 3.4: Estructura del Sistema de DDF. 60
Figura 3.5: Diagrama de Bloques del Método Bayesiano. 62
Figura 3.6: Problemas en el Diagnóstico de algunas Fallas. 71
Figura 3.7: Falla 1: La Rueda Derecha Resbala. 73
Figura 3.8: Falla 2: La Rueda Izquierda Resbala. 73
Figura 3.9: Falla 3: Pérdida de Momento de Torsión en la Rueda Derecha. ... 74
Figura 3.10: Falla 4: Pérdida de Momento de Torsión en la Rueda Izquierda. . 74
Figura 3.11: Falla 5: Falla del Sensor de Velocidad de la Rueda Derecha. 75
Figura 3.12: Falla 6: Falla del Sensor de Velocidad de la Rueda Izquierda. 76
Figura 3.13: Falla 7: Falla en el Giróscopo. 76
Figura 3.14: Falla 8: Pérdida de Momento de Torsión en Ambas Ruedas. 77
Figura 3.15: Porcentaje de Error en el Diagnóstico. 79
Figura 3.16: Interacción entre la Capa 1 y el Control de Navegación. 82
Figura 3.17: Falla 9: La Rueda Derecha Topa con un Obstáculo. 83
Figura 3.18: Falla 11: Ambas Ruedas Topan con un Obstáculo. 84
Figura 3.19: Detección y Diagnóstico de Fallas Consecutivas. 86
Figura 3.20: Disminución del Error por Aumento de Muestras. 92
Figura 3.21: Disminución del Error por Aumento de Muestras y Límites..... 93
Figura 3.22: Detección de Fallas en el Sonar. 95
Figura 3.23: Detección de Fallas en la Brújula. 95
Figura 3.24: Protocolo de Comunicación para Detección de Fallas..... 97
Figura 4.1: Sistema Cooperativo de Robots Móviles. 104
Figura 4.2: Diagnóstico Experimental de la Falla 1. 111
Figura 4.3: Diagnóstico Experimental de la Falla 3. 112
Figura 4.4: Diagnóstico Experimental de la Falla 6. 113
Figura 4.5: Diagnóstico Experimental de la Falla 7. 114
Figura 4.6: Diagnóstico Experimental de la Falla 10. 115

Figura 4.7: Fase uno de la Detección. 116
Figura 4.8: Fase dos de la Detección. 116
Figura 4.9: Fase tres de la Detección. 117
Figura A.1: Estimación de la Posición Utilizando un Filtro de Kalman. 133
Figura A.2: Estimación de la Posición Utilizando un Filtro de Kalman. 134
Figura B.1: Diagrama de Flujo de un Algoritmo Genético. 139

RESUMEN

El uso de robots móviles en diferentes aplicaciones ha aumentado significativamente en el último tiempo, abarcando desde procesos industriales a aplicaciones domésticas, como aspiradoras y cortadoras de pasto. Debido a ello, y a lo dinámico de los ambientes de trabajo, se necesita construir plataformas robustas, que permitan a los robots detectar la presencia de fallas en sus componentes y reaccionar oportunamente a ellas.

La presente Tesis considera el problema de la existencia de fallas en robots móviles que cooperan, y tiene como objetivo el diseño y la posterior implementación de una nueva arquitectura de detección y diagnóstico de fallas de bajo costo. Esta permite entregar más información del estado de los robots tanto a su sistema de control como a los operadores que lo utilizan.

La arquitectura se somete primero a pruebas de evaluación en una plataforma de simulación en MATLAB.

Posteriormente, el sistema de detección y diagnóstico de fallas se implementa en un grupo de robots móviles, mostrando un excelente desempeño.

ABSTRACT

The use of mobile robots has increased over the last years in a wide range of applications, which go from industrial work to more domestic tasks, such as vacuum cleaning or grass cutting. Due to benefits that these systems bring and the complexity of the environments where they are used, it is necessary to improve their reliability, which can be done by adding fault detection systems that are capable of diagnosing the operating state of the robot and react according to it.

This Thesis centres on the problem of failures on cooperative mobile robots, by designing a new low cost architecture of fault detection and diagnosis, which is later implemented. This system allows both, the control structure and the operators of the robot, to take into account the actual operating state of the mobile robot, by giving more information to them.

The architecture is first evaluated using a simulation platform in MATLAB, which also helps to analyze the performance of the system.

Then, the fault detection and diagnosis system is implemented on a cooperative group of small and resource limited mobile robots, showing an excellent performance in the identification of the different faults.

1. Introducción

En el presente capítulo se hace una introducción al tema desarrollado en la Tesis. Primero se plantea la motivación del trabajo, y después se describe el problema a solucionar y los objetivos. Finalmente se resume brevemente el contenido de cada capítulo.

1.1. Motivación

La robótica es un área multidisciplinaría de la ingeniería, hacia la que convergen la ingeniería eléctrica, la ingeniería mecánica y las ciencias de la computación. Dependiendo de la aplicación del robot se suman además muchas otras áreas científicas como la geología, la medicina o la biología marina, convirtiendo a la robótica en una de las áreas más complejas y desafiantes de la ingeniería actual. Junto con esto, la robótica ha adquirido un rol fundamental en la industria y hoy en día su masificación llega incluso a los hogares, con productos como aspiradoras y cortadoras de pasto robotizadas.

Considerando las implicancias futuras que puede alcanzar la robótica, esta Tesis se centra en el área del control automático y el procesamiento de señales aplicadas a robots móviles, teniendo como objetivo primordial, el entregar un aporte a las Ciencias de la Ingeniería a través de un estudio acabado de un problema de esta área. Debido a lo poco explorado del problema, y a los beneficios que puede entregar a la robótica, se ha optado por tratar el tema de detección y diagnóstico de fallas en robots móviles, buscando una solución factible y de bajo costo que apoye el avance en el área.

1.2. Descripción del Problema y Planteamiento de Objetivos

La complejidad y el dinamismo de los ambientes en que circulan los robots móviles son factores fundamentales en la aparición de diferentes fallas en sus componentes, que pueden llegar incluso a inhabilitarlo. Esto genera pérdidas en la eficiencia, tiempo y desempeño pues requieren continuamente de reparaciones.

Por estas razones, la presente Tesis se centra en el estudio acabado de las fallas que aparecen en los robots móviles. El objetivo del trabajo es el diseño de una arquitectura de detección y diagnóstico de las fallas, que aproveche los recursos de cada robot al máximo y lo proteja ante la aparición de diferentes problemas de funcionamiento. Esta arquitectura debe ser capaz de identificar a tiempo la aparición de problemas, evitando que el robot se dañe y entregando información de apoyo tanto a los sistemas de control como a los operadores y reparadores.

Con el fin de lograr estos objetivos, la metodología de trabajo se basa en el desarrollo de un modelo matemático completo de un robot móvil, que permita simular su comportamiento ante la aparición de las diferentes fallas estudiadas. En base a estas simulaciones se diseña una arquitectura de detección y diagnóstico de fallas, la cual, a través del procesamiento de las señales entregadas por el robot, debe lograr diferenciar los comportamientos del robot ante las diferentes fallas.

Junto con el análisis por simulaciones, la validación de los métodos utilizados se basa en la construcción de un pequeño grupo de robots móviles simples, de capacidades computacionales limitadas, en el que posteriormente se implementa la arquitectura, demostrando su correcto funcionamiento.

1.3. Descripción de la Tesis

Con el fin de entregar una visión global de este trabajo, a continuación se presenta un breve resumen del contenido de cada uno de los capítulos.

El segundo capítulo tiene como fin establecer el marco teórico sobre el cual se desarrolla esta Tesis. Se presenta primero una reseña histórica de la robótica, sus aplicaciones y un análisis de los sistemas cooperativos de robots móviles. Posteriormente se desarrolla un estudio detallado de los trabajos preliminares en detección y diagnóstico de fallas, identificando los desarrollos alcanzados y las limitaciones actuales. A continuación se diseña un modelo matemático completo del robot móvil que permite realizar las simulaciones y se describe la estructura de control en la que se basan los sistemas de robots cooperativos.

Teniendo ya la base teórica necesaria, el capítulo tres describe la arquitectura de detección y diagnóstico de fallas en un grupo cooperativo de robots móviles. Se enumeran las fallas que se deben identificar y se presenta el desarrollo teórico de la arquitectura. A continuación se analiza el desempeño del sistema en base a simulaciones.

El cuarto capítulo presenta la implementación de la arquitectura sobre un grupo de pequeños robots móviles. En este capítulo primero se describen los robots a utilizar y se presenta la forma en que se han de generar las diferentes fallas. Posteriormente se resumen los resultados de la implementación y se valida lo obtenido a través de simulaciones.

El quinto y último capítulo de la Tesis abarca todas las conclusiones del trabajo. Se analizan los objetivos alcanzados y los posibles trabajos futuros.

Los anexos incluidos al final de este trabajo entregan un desarrollo más en profundidad de algunos conceptos utilizados en la Tesis. Se presenta primero una

descripción de los filtros de Kalman, seguido de una breve reseña sobre Algoritmos Genéticos. Finalmente, se adjuntan las diferentes publicaciones que han resultado de esta investigación.

1.4. Contribuciones Originales

La presente tesis incluye contribuciones originales en varios ámbitos.

Primero, la arquitectura presentada es un aporte novedoso que permite flexibilidad en el diseño e implementación de sistemas de detección y diagnóstico de fallas, siendo además extendible a otros tipos de procesos.

Junto con ello, esta tesis presenta por primera vez un sistema de detección y diagnóstico que abarca la gran mayoría de las fallas comunes en robots móviles, validando los resultados a través de la implementación.

Otro aporte original de este trabajo es el uso de la interacción entre la arquitectura de detección y la estructura de control. Si bien actualmente los sistemas de detección entregan información a los sistemas de control, en este trabajo la arquitectura de detección solicita acciones a la estructura de control, obteniendo más información y así identificando un mayor número de fallas.

Finalmente, la presente tesis innova al usar la redundancia en sistemas cooperativos de robots, utilizando la información para desarrollar rutinas de detección y diagnóstico de fallas, que no se habían implementado anteriormente.

2. Marco Teórico

Para entender los alcances de la robótica en el mundo actual y la necesidad de contar con sistemas más robustos de monitoreo y control, este capítulo presenta una breve visión general de la robótica, partiendo con su historia, continuando con el estado del arte en robots cooperativos y sistemas de detección y diagnóstico de fallas aplicados en robótica.

Además, en este capítulo se describe el modelo matemático de un robot móvil de dos ruedas independientes, que servirá como base para el diseño posterior de la arquitectura de detección y diagnóstico de fallas, junto con un sistema genérico de control para regular las velocidades angulares de las ruedas.

2.1. Historia de la Robótica

Si bien la robótica se asocia generalmente a tecnología de punta y complejos métodos de control del siglo XX, el origen del concepto data de hace más de dos milenios. El primer ancestro de los actuales robots fue creado en el año 270 AC, por el inventor y físico griego Ctesibus de Alexandria, quién lo diseñó en busca de un método más preciso para la medición del tiempo y reemplazar los relojes de arena usados hasta entonces. Denominado "clepsydra" o reloj de agua, combinaba un ingenioso sistema de sifón para reciclar el agua en su interior y con ello poder medir lapsos de tiempo (Fig. 2.1). Este sistema fue reemplazado recién en la edad media por el reloj de péndulo [*Encyclopædia Britannica, 2004*].

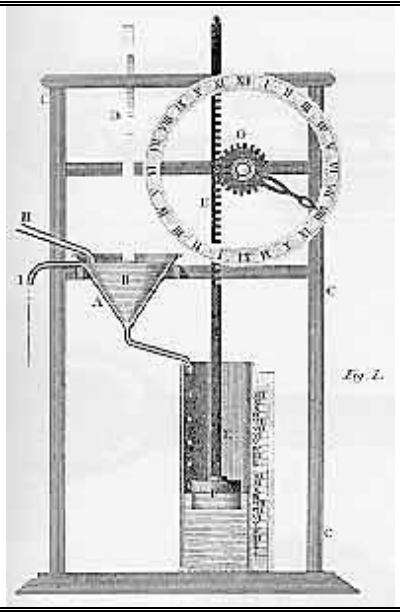

Figura 2.1: El Clepsydra o Reloj de Agua.

No fue sino hasta fines del siglo XVIII que comenzaron a aparecer los primeros sistemas que automatizaban diferentes tareas del quehacer cotidiano de la época. Por ejemplo, en 1774 los inventores suizos Pierre y Henri Jacquet Droz dieron a conocer su escriba automático, una máquina que se asemejaba a un niño y era capaz de dibujar mensajes de hasta 40 caracteres. Otro de sus aportes fue un robot con apariencia de mujer que tocaba el piano. Como se puede apreciar, los aportes de la época no fueron más que máquinas anecdóticas basadas en complejos sistemas mecánicos.

Quizás inspirada en aquellos primeros modelos y lo que podrían significar en el futuro, Mary Selley se adelantó a su tiempo con la obra "Frankenstain", en la cual el famoso Doctor Víctor Frankenstain crea una forma de vida artificial, la que al rebelarse, acaba con su vida.

Durante el siglo XIX los avances fueron lentos. En 1801 Joseph Jaquard inventó una máquina textil, cuyo programa de funcionamiento dependía de tarjetas perforadas, siendo quizás una de las versiones más antiguas de un sistema re-programable. Sólo a fines de siglo apareció la primera grúa

motorizada, creada por Seward Babbitt, al mismo tiempo que aparecía por primera vez la palabra "automatización" en la revista Strand. También realizó sus aportes a fines de 1890 el famoso inventor Nikola Tesla, creando vehículos a control remoto. Los intentos de la época buscaban crear máquinas que no sólo fueran capaces de apoyar al ser humano en tareas agotadoras o peligrosas, sino que además fueran, en algunos casos, máquinas antropomórficas, algo que se ha extendido hasta nuestros días.

Aun cuando hace más de un siglo que estaban en circulación, estas llamadas máquinas maravillosas tomaron el nombre de "robots" recién a inicios del siglo XX. Esta vez fue un dramaturgo checo, Karel Capek, quién nombró "robot" a una maquinaria por primera vez en su obra R.U.R (Rossum's Universal Robots), estrenada en 1921 en Londres (Fig. 2.2). El término robot proviene de la palabra checa *robota* que significa "trabajo forzado" ya que en su obra se mostraba un grupo de trabajadores mecánicos, que acaban con sus amos tras sublevarse [*Currie*].

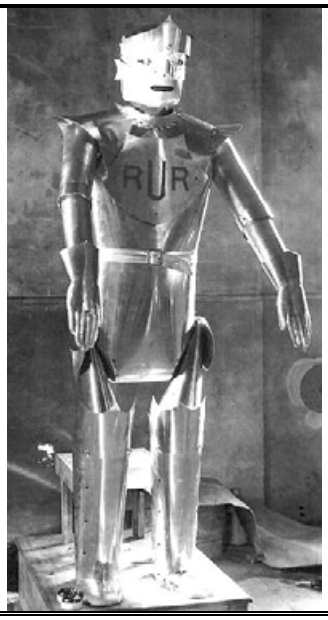

Figura 2.2: Robot de la obra R.U.R. de Karel Capek.

Veinte años más tarde la palabra robótica fue acuñada por el conocido autor de ciencia ficción Isaac Asimov, asociándola a la tecnología necesaria para la creación y desarrollo de los robots. Una vez más es un escritor quién se adelanta a los hechos ya que en su obra "Runaround", escrita en 1942, predice la aparición de robots que apoyarán el desarrollo de la industria. Además, a través de obras como "I Robot", sienta los precedentes y caminos en los que la robótica va a transitar durante los siguientes años.

Durante esa misma época, gracias a los avances en la electrónica, nuevos robots aparecieron en escena, aunque que esta vez ya no con forma humana. Uno de los primeros fueron las tortugas Elsie y Elmer, creadas por el científico inglés Grey Walter en 1950 (Fig. 2.3). Fueron el resultado de un experimento que buscaba analizar los complejos comportamientos que eran capaces de generar simples interconexiones similares a las conexiones nerviosas en seres vivos. Este robot, de pequeñas dimensiones, era capaz de trasladarse por el suelo, cambiando de rumbo según la información que obtenía de su medioambiente, dando un impulso vital a la generación de nuevos sistemas robóticos controlados en base a leyes de comportamiento, [*Sabbatini*].

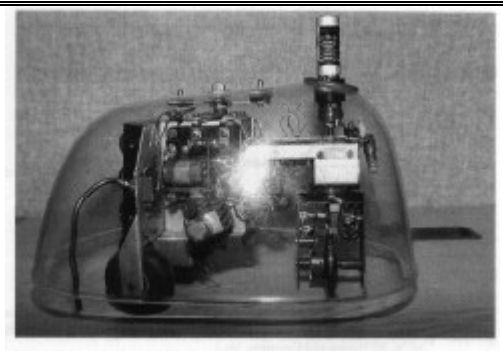

Figura 2.3: La tortuga Elsie, de Grey Walter.

La revolución de la electrónica trajo consigo avances notables en el control y diseño de los nuevos robots. En 1946, al mismo tiempo que J. Presper Eckert y John Mauchly creaban a Eniac, el primer computador electrónico del

mundo, George Devol patentaba un sistema general para el control de máquinas, que permitiría manejar los complejos sistemas mecánicos que se iban desarrollando. No fue sino hasta 1954 en que apareció el primer robot programable. Este fue un nuevo aporte de George Devol, quién junto a Joseph Engelberger, fundan dos años más tarde la primera empresa de robótica del mundo, llamada Unimation, como diminutivo de "Universal Automation". Debido a sus contribuciones, para muchos son considerados como los padres de la robótica, o al menos de la robótica moderna.

En los años posteriores los avances en inteligencia artificial, control automático y electrónica ayudaron significativamente al avance de la robótica. Se fundaron centros de investigación en el Massachusetts Institute of Technology (MIT) y en la Universidad de Stanford especialmente para desarrollar las nuevas áreas emergentes, logrando avances en la automatización de manufactura.

El año 1961 se instaló el primer robot industrial, un Unimate, en las líneas de producción de General Motors (GM). Este se dedicaba a tomar piezas de metal caliente para apilarlas en la zona de enfriado, aumentando el nivel de producción. Será GM, en años posteriores, quién apoyará de manera significativa el desarrollo de nuevos robots industriales para sus procesos a lo largo de Estados Unidos.

Si bien Estados Unidos fue el primer país en adoptar la robótica como medio para automatizar procesos y mejorar eficiencia, su uso no fue popular. En contraste, Japón recién importó su primer robot industrial el año 1967, pero debido al esfuerzo en conjunto de la industria japonesa, un año después, Kawasaki Heavy Industries obtuvo licencias para desarrollar robots industriales en ese país. Este impulso inicial sirvió para que 20 años más tarde, sobre 40 empresas japonesas estuvieran desarrollando nuevos sistemas robóticos industriales, mientras para entonces en Estados Unidos sólo sobrevivía una docena.

Siguiendo otro camino en el desarrollo de la robótica, en 1963 apareció el primer brazo artificial controlado por computadora. Esta vez su función no era la de trabajar en líneas de producción, sino que permitir a personas discapacitadas mejorar su condición de vida. El brazo constaba de seis grados de libertad que le permitían la misma flexibilidad que el brazo humano. El brazo se transformaría en el primer peldaño en el desarrollo de sistemas robóticos de apoyo para las personas, pues posteriormente aparecerían nuevos avances como piernas artificiales o sillas de ruedas inteligentes que ayudaban a incrementar el confort y la movilidad. Ese es el caso del Stanford Cart creado por el Stanford Research Institute (SRI) en 1979, una silla de ruedas con sistemas de video capaces de detectar la distancia a los diferentes objetos para diseñar una ruta de navegación. La figura 2.4 muestra una versión moderna de la silla inteligente creada en el MIT.

Figura 2.4: Silla de Ruedas Inteligente Wheelesley del MIT.

Para el año 1998 el grupo de desarrollo de prótesis del Princess Margaret Rose Orthopedic Hospital en Edimburgo, logró crear el primer brazo biónico, el "Edinburg Modular Arms System" o EMAS (Fig. 2.5). Además de estos equipos prostéticos, robots quirúrgicos, basados en los desarrollos logrados en SRI, MIT e

IBM, han entrado a los pabellones de operación, facilitando el trabajo a los médicos y disminuyendo las complicaciones y tiempos de recuperación en los pacientes.

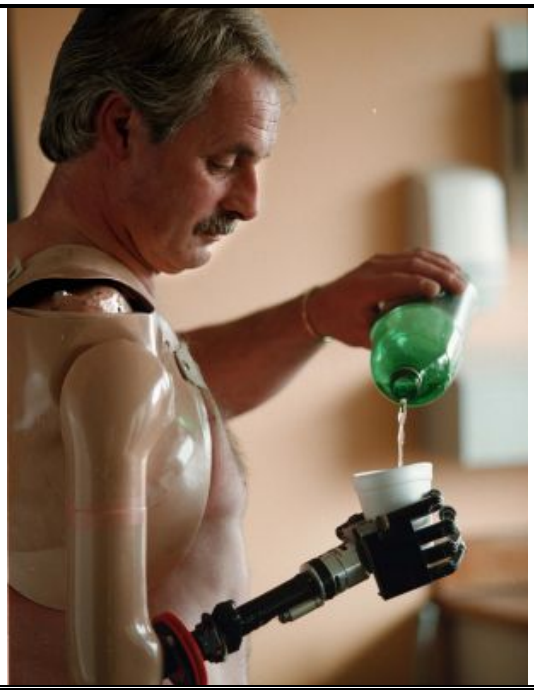

Figura 2.5: El primer brazo biónico, el EMAS.

Continuando con el aumento de complejidad de los sistemas, el SRI creó en 1968 a Sharkey, el primer robot con capacidades autónomas y que podía interactuar con su medioambiente, además de ser uno de los primeros robots móviles en contar con sistemas de video y procesamiento de imágenes (Fig. 2.6). En base a sistemas electrónicos y una limitada memoria, Sharkey era capaz de resolver problemas de navegación con obstáculos. SRI continuó con sus aportes al presentar un año más tarde el Stanford Arm de manos del profesor Scheinman. Era un brazo robótico controlado electrónicamente y diseñado para investigación, lo que lo transformaría posteriormente en un estándar.

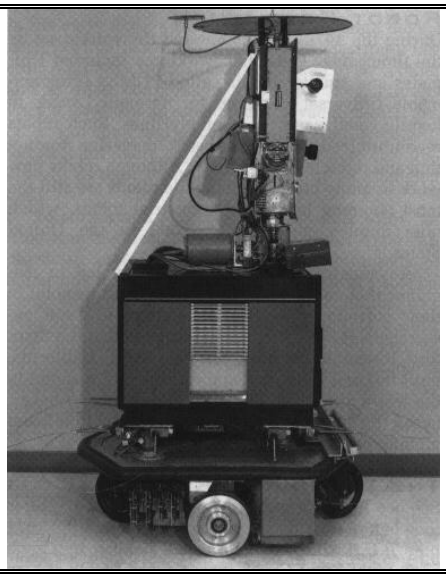

Figura 2.6: Sharkey, el primer robot autónomo.

La década de los 70 trajo nuevos avances y desafíos a la industria de la robótica. En 1973 apareció en el mercado el primer robot industrial con un microcomputador para su control y programación. Fue desarrollado por Richard Horn para la empresa Cincinnati Milacron Corporation, y se llamó T3. Un año más tarde se diseñó el Silver Arm, un brazo para ensamblaje industrial que era capaz de reaccionar en base a la retroalimentación que le entregaban sensores de presión en sus actuadores. Otro avance importante fue Freddy, un robot móvil controlado por inteligencia artificial desarrollado en la Universidad de Edimburgo. Este robot era capaz de juntar piezas separadas puestas sobre una mesa, demostrando lo difícil que era lograr coordinación ojo – mano para el ensamblaje. En 1978 apareció el brazo robótico Puma (Programmable Universal Machine for Assembly), uno de los brazos más utilizados actualmente en el estudio de la cinemática y dinámica en cursos de robótica. Fue desarrollado por Unimate con el apoyo de GM.

El año 1976 se dio el primer paso en la industria de la robótica dirigida a la exploración espacial. Ese año, Estados Unidos envió su primera sonda de investigación a Marte, en busca de datos sobre su composición y el desarrollo de

mapas de su superficie. Las sondas Viking 1 y 2 fueron diseñadas para esta misión, siendo las primeras en contar con brazos robóticos a bordo. Desde entonces se desarrollaron una seguidilla de robots espaciales con el fin de obtener información de los diferentes planetas del sistema solar y en especial de Marte. Así nació la misión del Pathfinder, que sería la primera que contaría con un robot móvil que transitaría por la superficie marciana. El 4 de Julio de 1997 el Pathfiner Lander, llamado ahora Sagan Memorial Station, se posaba en Marte, y minutos más tarde el robot Sojourner circulaba por la superficie obteniendo imágenes y muestras para su posterior análisis. El robot, de menos de 10 kilos y basado en energía solar, estaba equipado con un espectrómetro para analizar la composición química de las rocas y del ambiente, a la vez que una cámara de alta resolución enviaba imágenes a la tierra. El éxito de la misión sentó un precedente para lo que sería la exploración espacial del siguiente siglo [*NASA*]. En la figura 2.7 se observa el Rover 2 y una copia del Sojourner llamado Marie Curie, predecesor del Rover 2.

Figura 2.7: El Rover 2 y su predecesor.

Así, durante los años siguientes se lanzaron diferentes misiones para investigar más sobre Marte, Júpiter y Venus. A fines de 2003 la comunidad europea se sumó a los esfuerzos de investigación logrando hacer llegar a la órbita

de Marte su sonda Mars Express. Al interior de esta se encontraba alojado el Beagle2, diseñado para posarse sobre el terreno de Marte y explorar la composición del mismo (Fig. 2.8). Un brazo le permitía obtener muestras y analizarlas, mientras un pequeño robot móvil llamado Mole ayudaba en la tarea [*Beagle2*].

Figura 2.8: El Beagle 2, enviado por la comunidad europea a Marte.

Para inicios de 2004, otras dos sondas de Estados Unidos se posaron sobre la superficie de Marte, la Opportunity y la Spirit, que buscaban recolectar más información del planeta (Fig. 2.9).

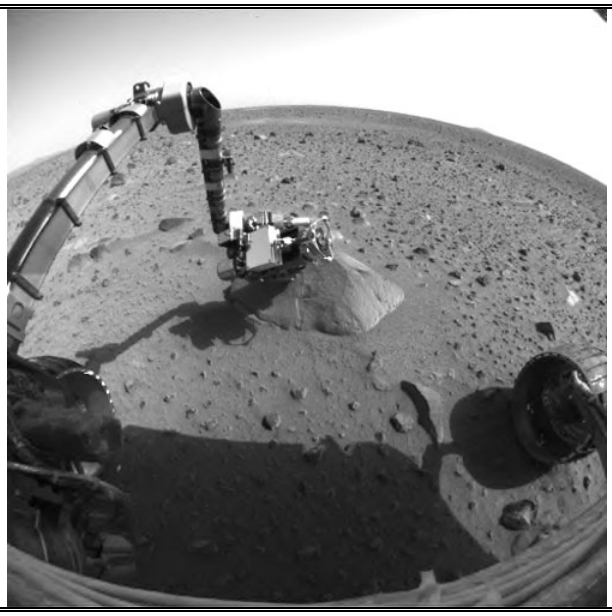

Figura 2.9: Vista de la superficie marciana por el robot Spirit.

Es también la búsqueda de métodos más seguros para la exploración, la que ha contribuido a la creación de otros robots móviles. Tal es el caso de Dante, un robot de seis piernas desarrollado en la Universidad Carnegie Mellon, en Estados Unidos, para la investigación de volcanes. Fue probado exitosamente en 1994 sobre el Monte Spurr en Alaska. Otros robots como el Odyssey IIb, creado en el MIT, buscan explorar el océano a profundidades inalcanzables para el hombre.

Por su parte el siglo XXI trajo consigo una nueva aplicación de la robótica: el entretenimiento. Si bien el uso de títeres electromecánicos era algo común para fines de la década de los 90, la aparición de nuevos sistemas autónomos generó una revolución. Uno de los más impresionantes es el caso de Aibo, un perro robot creado por Sony Corporation en Japón. En el año 2000 se presentó su segunda versión, lanzándose a fines del 2003 la tercera generación, el ERS-7 (Fig. 2.10). Esta cuenta con una red inalámbrica Wi-fi, junto con 20 grados de libertad y sensores que lo convierten en algo único en su tipo. Es capaz de reconocer caras utilizando una cámara y detectar comandos de voz,

además que logra identificar su fuente de alimentación para recargarse en forma autónoma [*Sony*].

Figura 2.10: La tercera generación de AIBO, el modelo ERS-7.

Junto con sus mascotas robóticas, Sony también ha desarrollado su línea Sony Dream Robots o SDR, que mostró en el año 2000 uno de los primeros robots humanoides, siguiendo con lo que ya había logrado Honda y su robot Asimo. Volviendo a la idea de hace ya varios siglos, todos estos son robots bípedos, que han logrado controlar satisfactoriamente los problemas de inestabilidad inherentes de los robots de dos piernas, resolviendo dificultades complejas como patear una pelota o subir una escalera.

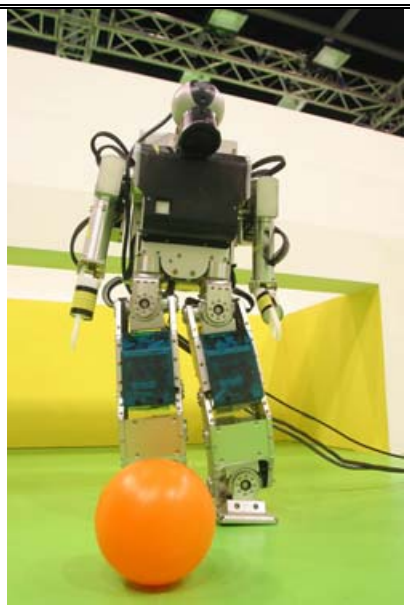

Figura 2.11: Robot Bípedo en el RoboCup 2003.

La atracción que tiene la robótica en la gente ha logrado que nuevos sistemas de menor costo aparezcan para el público en general. Es así como en 1995 LEGO llegó al mercado con su línea Mindstorms. Esta se basa en un cubo llamado RCX Brick, creado en el MIT, que posee un microcontrolador, dándole la posibilidad de controlar sensores y actuadores con mucha simpleza. Si bien fue originalmente diseñado para niños, la línea Mindstorms se usa actualmente en muchas universidades en la enseñanza de robótica e inteligencia artificial, por sus ventajas en costo y flexibilidad [*Wallich, 2001*].

Figura 2.12: Robot Creado con LEGO Mindstorms.

Durante los últimos años han aparecido nuevos robots de uso doméstico. El Roomba, de Robots Inc., es un pionero en este campo al ser el primer robot aspiradora de costo accesible.

De lo expuesto, se puede concluir que, la robótica actual ha seguido por cuatro caminos principales:

La industria, con robots en las líneas de producción que mejoran la eficiencia y disminuyen los costos de producción.

La exploración, donde robots revisan desde el espacio exterior hasta las profundidades del océano, entregando información de manera más segura para los investigadores.

La medicina, con sistemas de apoyo para discapacitados y robots que permiten cirugías más simples, baratas y exactas.

El entretenimiento, destacando aportes como el Aibo o el sistema Mindstorms de LEGO.

Cabe destacar que con tantas aplicaciones y morfologías se hace difícil llegar a alguna definición concreta de que es un robot hoy día. Según el Instituto de Robótica de Estados Unidos, un robot se define como "un manipulador multifuncional y reprogramable, diseñado para mover herramientas, materiales o partes en varios movimientos programados para la realización de diferentes tareas". Una definición más simple aparece en el diccionario Webster, señalándolo como "un aparato automático, que realiza tareas que normalmente son hechas por humanos, o una máquina en la forma de un humano".

2.2. Sistemas de Robots Cooperativos

Esta es una de las áreas más nuevas en la robótica moderna. Sin embargo, existe ya una extensa cantidad de artículos publicados, en que se tratan estos sistemas desde múltiples puntos de vista, e incluso desde varias áreas del conocimiento, que van desde la electrónica hasta la biología y la psicología.

Los sistemas distribuidos de robots presentan un gran número de ventajas en comparación a un sólo robot [*Cao, 1997*]:

- Algunas tareas son imposibles de realizar por un robot único, dada la complejidad o las limitaciones espaciales del problema, haciendo necesario que dos o más de ellos intervengan para efectuarla.

- El utilizar más robots permite un aumento de la eficiencia y efectividad del sistema, lo que claramente puede evidenciarse en sistemas de exploración y recolección de información, donde un sistema de múltiples robots es capaz de abarcar un área mayor.

- La creación de muchos robots simples es más barata y fácil, permitiendo una flexibilidad en cuanto a la atención de tareas que un robot único no es capaz de dar, sin importar que tan equipado esté. Así, actividades como el

movimiento de cargas, que puede realizarse igual de rápido por un robot único, puede efectuarse en forma más flexible por un conjunto de robots que ayudan según la magnitud de la carga.

- El uso de múltiples agentes permite enfrentar fallas de mejor forma, ya que, aunque fallen y salgan del sistema algunos robots, el objetivo final es aún posible de lograr. En el caso de un robot único, su falla significa la imposibilidad de continuar y finalizar la tarea.

Aunque las ventajas son importantes, también vienen asociadas dificultades por su complejidad. Por un lado, presenta las complejidades asociadas a los sistemas distribuidos (como redes o bases de datos), que son las limitaciones de recursos que deben ser manejados para optimizar el resultado final y el diseño de la arquitectura necesaria para resolver problemas, sin perder flexibilidad. A esto se agregan los problemas propios de la robótica móvil, que incluyen todas aquellas dificultades con las que debe enfrentarse un agente autónomo al salir a un ambiente dinámico, donde las condiciones pueden cambiar sin información previa y el robot debe ser capaz de adaptarse a ello. La unión de ambos conlleva a la existencia de otros problemas adicionales como son la generación de trayectorias para evitar colisiones y aquellos referidos a problemas geométricos relacionados.

Los sistemas cooperativos enfocan el comportamiento colectivo del sistema hacia una o varias tareas que tienen un objetivo final, aumentando en general la utilidad del sistema, en comparación al caso en que los agentes no cooperan entre ellos.

Uno de los primeros sistemas de robots múltiples que existieron fueron los creados por el neurofisiólogo Grey Walter en 1950, quien mostró los altos niveles de complejidad que se podían obtener en el comportamiento de sus robots [*Sabbatini*]. Siguiendo esta línea de estudio, en la década de 1970 se crearon los conceptos de inteligencia artificial distribuida y la coordinación de agentes para

la resolución de problemas, aunque todo ello limitado a la teoría y las simulaciones. No fue sino hasta los inicios de la década de 1990 en que comenzaron a aparecer los primeros estudios sobre conjuntos de robots para resolver problemas o simular comportamientos de colonias. Su aparición no fue de exclusiva responsabilidad de quienes trabajaban en robótica en ese entonces, sino también se debió a estudios en otras áreas, como la teoría de juegos en economía o la biología teórica.

En sus inicios los estudios de sistemas cooperativos se limitaron a la teoría y las simulaciones, ya que el costo y la dificultad de construir y mantener un gran número de robots autónomos hacía muy difícil obtener estos sistemas. Existen algunos estudios como los hechos en [*Dodds, 1995*], en los que en vez de trabajar con robots móviles se analizan experimentalmente los aportes que se logran al utilizar más de un brazo robótico para la resolución de una tarea. Estos primeros trabajos aumentaron la eficiencia de los sistemas al contar con un mayor número de mediciones que permiten, al usar fusión sensorial, una mejora en la estimación de la posición de cada elemento del brazo y con ello una mejora en el control. Ya para mediados de la década varios equipos de investigadores contaban con sistemas de 2 o 3 robots que interactuaban, usando principalmente un control basado en simples reglas de comportamiento.

Posiblemente uno de los eventos más apasionantes en la robótica cooperativa son los campeonatos de fútbol robótico o RoboCup (Fig. 2.13) que comenzaron en 1997 en Japón [*RoboCup y Coradeshi, 2002*]. Este concurso actualmente cuenta con difusión mundial y con campeonatos locales en países como Alemania y Japón, a la vez que existen abiertos zonales como el American Open y últimamente un campeonato Latinoamericano que se realizó en el marco de las competencias de robótica de la zona.

Figura 2.13: Afiche del Abierto de RoboCup de Alemania de 2003.

La aparición de estos concursos ha sido una de las piedras angulares en cuanto a la investigación y desarrollo de tecnologías en el área de robots cooperativos. Teniendo diferentes ligas, cada una con sus dificultades propias, han incentivado el estudio en áreas tan variadas como la visión artificial, el control distribuido y el diseño de robots y sus sistemas de comunicación. Por un lado se cuenta con ligas de simulación, donde se hace énfasis en las decisiones que debe tomar cada uno de los jugadores para lograr ganar. Por otro lado, se cuenta con ligas de robots que deben ser capaces de organizarse como equipo para vencer al oponente. Las ligas más importantes en esta competencia son la F180 y la Middle Size Robot League. La F180 (Fig. 2.14) o Small Robot League obtiene su nombre del hecho que cada robot debe caber en un círculo de 180 [mm] de diámetro para poder competir. En general los jugadores cuentan con

una cámara global que le entrega información al sistema de la posición de su equipo, el oponente y la pelota, permitiendo que un agente central tome las decisiones de control.

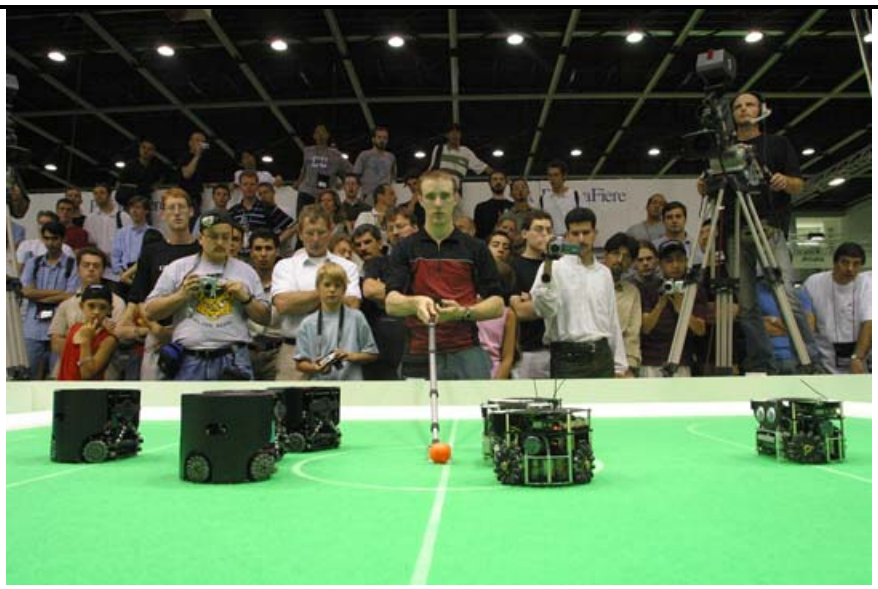

Figura 2.14: Partido de la Liga F180 del RoboCup 2003.

La Middle Size Robot Legue cuenta con robots más grandes, completamente autónomos. Estos poseen sus propios sistemas de visión y deben coordinarse entre ellos sin un agente externo, aumentando el grado de dificultad en cuanto al comportamiento de cada robot. La imagen 2.15 muestra al equipo de la universidad de Carnegie Mellon, los CM Hammerheads.

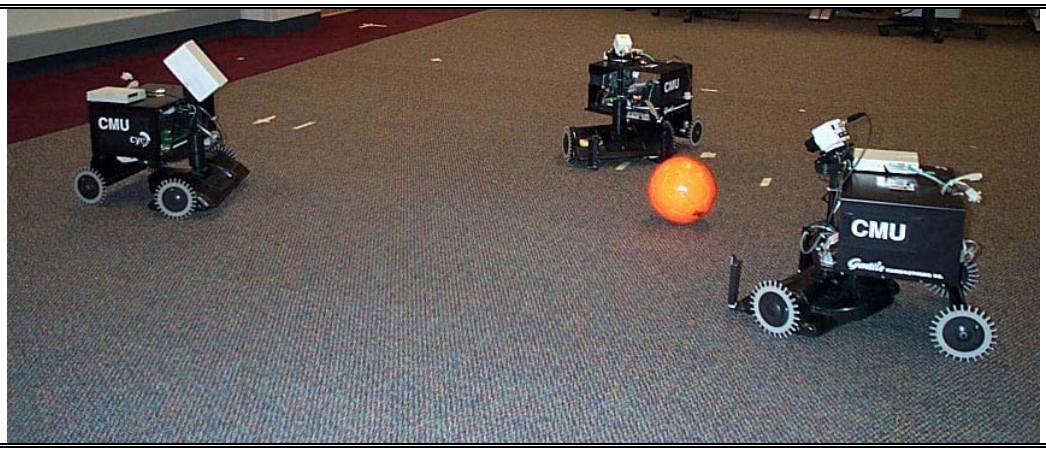

Figura 2.15: Carnegie Mellon Hammerheads.

El campeonato RoboCup también incluye una liga de equipos de perros Aibo que presentan un nivel de dificultad extra al ser robots con piernas en vez de ruedas. Finalmente, desde el campeonato hecho en Japón el 2003 han aparecido varios equipos de robots bípedos, buscando así lograr los objetivos finales del campeonato, que es contar con un equipo de robots humanoides para el año 2050, que sean capaces de competir (y ganar) contra jugadores humanos.

Figura 2.16: Robot Humanoide en el RoboCup 2003.

Las primeras investigaciones en el tema de sistemas de múltiples robots móviles, se centraron en el desarrollo de diferentes arquitecturas que integraran a los robots. Estas arquitecturas pueden ser clasificadas en diferentes aspectos, siendo quizá el más importante de ellos la forma en que se realizará el control. Los sistemas centralizados cuentan con un agente que toma todas las decisiones y se las comunica a los robots para que estos las ejecuten. Por otro lado, los sistemas descentralizados no poseen este agente central y pueden ser jerárquicos (con algunos robots que toman decisiones locales) o distribuidos (donde son todos iguales a nivel de la toma de decisión). La arquitectura descentralizada posee varias ventajas, como ser más tolerante a fallas, permitir mejor distribución de recursos en procesos paralelos, mayor confiabilidad y ser

infinitamente escalable en número, ya que al ser cada agente autónomo e independiente no se requiere de sistemas de control sobre ellos.

Las arquitecturas también pueden diferir en cuanto a los recursos con los que cuenta cada robot. Así, los sistemas homogéneos son aquellos que poseen todos sus robots de iguales características, teniendo un elevado número de elementos redundantes en el sistema. Un ejemplo de esto son los Robot Ants u Hormigas Robots creadas en el Laboratorio de Inteligencia Artificial del MIT (Fig. 2.17), cuyos objetivos son lograr avanzar en la miniaturización de robots, a la vez que se logra crear una comunidad organizada de ellos [*RobotANTS*].

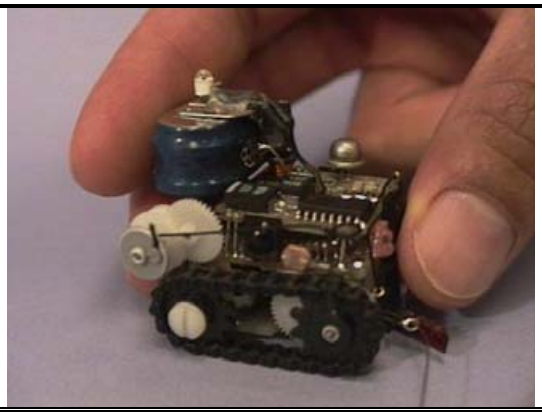

Figura 2.17: Robot ANT del MIT AI Lab.

Por otro lado arquitecturas como la ALLIANCE [*Parker, 1998*] buscan crear un marco de funcionamiento para robots heterogéneos, como el mostrado en la figura 2.18. Diseñado e implementado en el Laboratorio de Inteligencia Distribuida de la Universidad de Tennessee, su objetivo fundamental es realizar búsquedas en forma rápida al interior de edificios y casas [*Alliance*].

Figura 2.18: Grupo de Robots Heterogéneos.

Se puede apreciar en la figura 2.18 que el equipo consta de un grupo grande de robots pequeños, acompañados de dos robots más grandes, los cuales poseen diferentes sensores.

Dando un paso más allá en la heterogeneidad de los equipos de robots, en la Universidad Carnegie Mellon se está desarrollando un proyecto llamado Millibots, basado en un equipo de cinco robots diferentes entre si, dedicados al reconocimiento y vigilancia [*Millibot*]. A diferencia de otros enfoques del tema, este trabajo busca la creación de robots simples y de bajo costo en que la redundancia de sensores sea menor, al acompañar sólo a ciertos agentes del sistema con equipo especializado. Sólo algunos de los robots poseen una cámara, mientras otros poseen sonares de corto y largo alcance. Se hace fundamental entonces el apoyo y cooperación entre robots para poder lograr los objetivos finales, teniendo un equipo flexible y más económico.

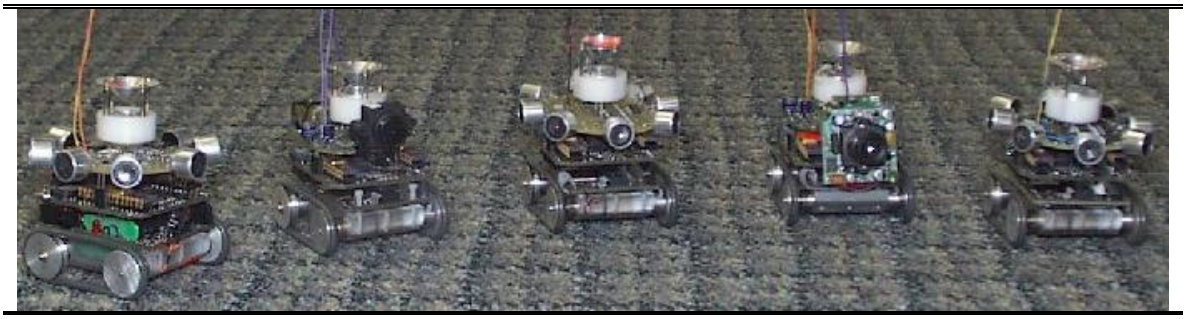

Figura 2.19: CMU Millibots.

Otras clasificaciones de arquitecturas radican en la forma en que se comunican los robots entre si. Puede ser en forma directa (utilizando sistemas de comunicación dedicada); indirectamente, a través del uso de sensores para analizar a sus compañeros de equipo o sensores para analizar el medioambiente y con ello inferir el estado del resto de los miembros, y, finalmente las arquitecturas que carecen de comunicación entre los agentes.

Desde comienzos del siglo XXI, el trabajo en equipos de robots se ha intensificado, gracias a los nuevos desafíos que se plantean y la disminución en los costos de producción de los mismos. Así, la mayoría de los trabajos buscan aprovechar la redundancia de información que existe al interior de los equipos de robots. El trabajo de [*Michaelson, 2000*] analiza la redundancia de actuadores en el sistema con el fin de movilizar una carga. En base a su estudio es posible generar diferentes sistemas de control para el grupo de robots, aprovechando que al tener más de una cierta cantidad de robots alrededor de la carga es posible movilizarla en cualquier dirección, incluso si alguno de los robots falla.

Por otro lado, otros investigadores buscan aprovechar la existencia de redundancia en la información colectiva del grupo para mejorar las estimaciones en posición de cada uno de los agentes. En su trabajo, [*Roumeliotis, 2000*] presenta un filtro de Kalman para la población completa de robots, mejorando las estimaciones de posición y dirección de cada uno en base a la información entregada por otros al momento de juntarse. Otra forma de enfrentar este

problema es la propuesta de [*Stroupe, 2001*], en la cual se utiliza fusión sensorial basada en información estadística de los sensores para mejorar la estimación de la posición de cada robot.

En definitiva, son muchas y muy variadas las áreas involucradas en el desarrollo de sistemas cooperativos de robots, todas ellas aportando desde su punto de vista para ampliar el conocimiento de estos complejos sistemas. Aún así, es un área de la robótica relativamente nueva, que da cabida a una gran cantidad de nuevos caminos y enfoques para resolver los problemas y poder aprovechar los beneficios que significa tener una comunidad que coopere hacia un objetivo común.

2.3. Detección y Diagnóstico de Fallas en Sistemas Robóticos

Desde la aparición de los primeros sistemas de control, los requerimientos de eficiencia y robustez en los diferentes procesos industriales han ido paulatinamente incrementándose. Este ha sido en parte el resultado del aumento en la competencia por el mercado, en que los sistemas más eficientes ahorran tiempo y dinero a sus dueños. Con este objetivo en mente, durante la década de 1970 aparecieron los primeros trabajos de sistemas de detección de fallas, que buscaban indicar cuándo se producía un error en el sistema y así prevenir posibles problemas posteriores. A continuación, se desarrollaron sistemas más complejos de monitoreo y control que no sólo permitían detectar el momento en el que ocurría una falla en el sistema, sino que también tomar las medidas necesarias para corregir el error, como por ejemplo modificar las variables de control para minimizar los efectos de la falla.

La detección y diagnóstico de fallas incluye tres campos de acción. El primero corresponde a las fallas en el proceso mismo. Estas ocurren cuando los parámetros como volúmenes o masas cambian debido a fallas en el proceso,

modificando así su comportamiento frente a las variables de entrada. El segundo campo de acción está referido a las fallas de actuadores. Estas fallas se presentan en elementos como motores y válvulas que pierden sus características o se desacoplan del proceso, haciendo que su acción no afecte el sistema. El tercer campo de acción corresponde a las fallas de sensores, como velocímetros y medidores de nivel, que proporcionan las mediciones del sistema, necesarias para realizar el monitoreo y/o el control.

Por su parte, las fallas pueden ser caracterizadas según su efecto. Las fallas de tipo fuerte son aquellas en las que un parámetro cambia en un instante corto de tiempo, permaneciendo así posteriormente. Tal es el caso cuando los sensores se descomponen y mantienen constante el valor de salida sin importar la variación en la variable medida, o cuando un motor deja de entregar el momento de torsión necesario. Por otro lado, las fallas suaves son aquellas que aparecen lentamente y se van incrementando en el tiempo. Ese es el caso de fallas degenerativas, como el desgaste, que hace cambiar los parámetros en forma paulatina.

Por otra parte, es importante hacer una distinción entre lo que son los sistemas de detección de fallas y lo que son los sistemas tolerantes a fallas. Si bien ambos parecen realizar las mismas acciones, sus objetivos se entremezclan. Los sistemas de detección y diagnóstico buscan identificar lo más eficientemente posible la ocurrencia de una falla y su localización, entregando esta información a otro agente capaz de tomar decisiones en base a ellas. Por su parte, los sistemas tolerantes a fallas no requieren exclusivamente de un sistema de detección de fallas para poder funcionar. Los trabajos de [*Michaelson, 2000*] y [*Parker, 1998*] por ejemplo, presentan estructuras tolerantes a fallas sin utilizar un sistema formal y explícito de detección. Las estructuras que presenta [*Michaelson, 2000*] para el movimiento de objetos utilizando sistemas multi-robot, cuentan con robots redundantes, que al fallar no impiden la realización de la tarea, aunque ninguno de los robots restantes note que hay uno que falta. Un

ejemplo simple de estos sistemas, son dos robots tirando de una misma carga como se observa en la figura 2.20.

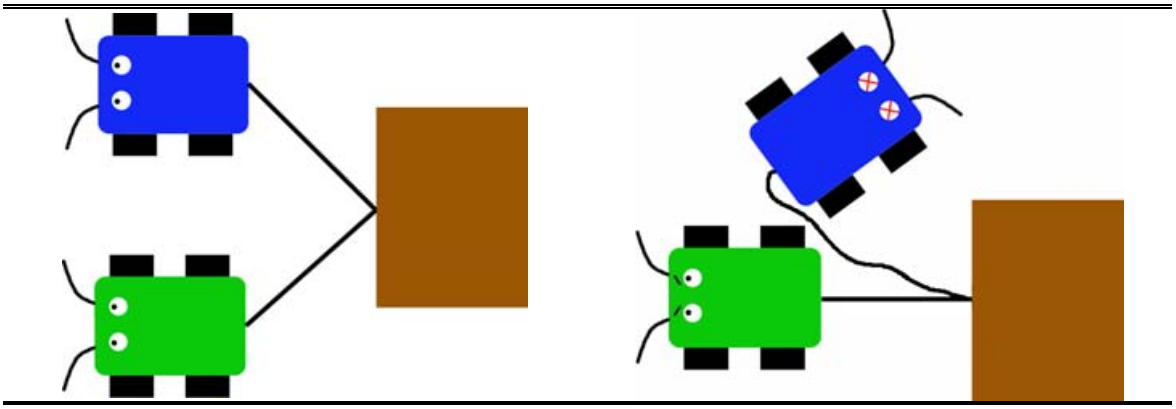

Figura 2.20: Sistema Multi-Robot Tolerante a Fallas.

Asumiendo que la carga puede ser movida por cada uno de los robots en forma individual, la existencia de un robot redundante permite que al fallar uno de ellos, el otro aún pueda finalizar la tarea. Indirectamente, el robot restante es capaz de identificar que existe una falla en el sistema, pues debe aumentar el torque de sus ruedas y su consumo de energía para mover la carga en comparación a cuando tenía a su compañero funcional, aunque no es capaz de determinar que es lo que falló en su compañero.

2.3.1. Detección, Diagnóstico y Corrección de Fallas

El proceso completo que permite detectar una falla y las acciones posteriores realizadas en base a la información obtenida, puede dividirse en tres etapas principales: detección, diagnóstico y corrección o acomodación de la falla [*Basseville, 1998*].

La detección de fallas es el paso inicial para todo método de trabajo con fallas, que permite al sistema identificar cuándo ésta ha ocurrido y que hace que el sistema no funcione según parámetros establecidos. En general, este proceso

puede implementarse en formas simples y permiten dar indicaciones o alarmas a los operadores indicando que ha ocurrido una falla.

En su trabajo de análisis de métodos de detección y diagnóstico, [*Isermann, 1995*] clasifica los procesos de detección y diagnóstico de fallas en tres categorías según su complejidad.

En los niveles más bajos se encuentran los procesos de monitoreo y de protección automática, que son sistemas que sólo realizan detección de fallas sobre el proceso en observación. El más básico de ambos, el monitoreo, sólo entrega la observación de un conjunto de variables a los operadores, quienes deben compararlos con niveles críticos de funcionamiento para tomar decisiones en cuanto a reparar o detener el proceso para evitar problemas mayores.

En el caso de los sistemas de protección automática, éstos, además de contar con un monitoreo de variables, en caso de detectar un falla, automáticamente realizan acciones para evitar mayores daños, sin necesidad de encontrar en forma precisa la causa que motivó la falla.

En la clasificación que propone Isermann, el tercer y más complejo grupo son los sistemas de supervisión con diagnóstico de fallas. Estos, además de detectar, incorporan el diagnóstico como una herramienta para mejorar la utilidad del sistema. El diagnóstico de fallas responde a la pregunta "¿Qué es lo que falló?". Su funcionamiento se basa en un análisis de las mediciones realizadas para poder generar un grupo de síntomas, los cuales sirven como base para determinar qué elemento al interior del sistema completo fue el que falló, siendo a veces factible incluso determinar por qué sucedió. Así, el sistema puede entregar al operador o a un sistema posterior de toma de decisiones más información relevante a la falla, la que permite tomar mejores decisiones o incluso logran que el sistema continúe en funcionamiento aún cuando la falla esté presente.

Uno de los métodos más utilizados en el diagnóstico de fallas es el uso de modelos. Este método se basa en el empleo de modelos del proceso para el estado normal y/o para cada una de las diferentes fallas que se desean observar, y posteriormente analizar qué tan parecidas son las estimaciones de cada modelo con las observaciones realizadas del proceso real, tal como lo muestra la figura 2.21. La forma de analizar las diferencias entre los modelos y las observaciones, llamados residuos, es muy variada, pasando desde los sistemas expertos, hasta complejas redes neuronales.

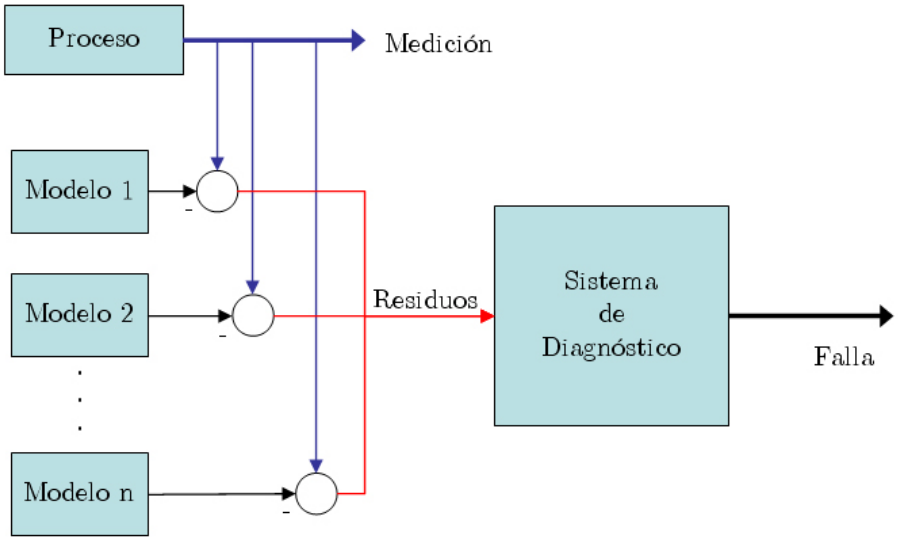

Figura 2.21: Diagnóstico de Fallas en Base a Modelos.

El último paso es la corrección de la falla, lo cual puede realizarse de múltiples formas. Tal como se indicaba anteriormente, los sistemas tolerantes a falla son capaces de tomar decisiones para corregir la falla ocurrida sin necesidad de existir un sistema formal de detección y diagnóstico. Pero también existen procesos que cuentan con sistemas de detección y diagnóstico para identificar el efecto de la falla y contrarrestarlo. Ejemplos de esto se presentan en los trabajos de [*Shin, 1999*] y [*Visinsky, 1993*]. Ambos proponen sistemas de control para brazos robóticos que cuentan con procesos simples de detección y diagnóstico

que permiten al control adaptarse a una falla y poder continuar con la tarea programada.

2.3.2. Sistemas de Detección y Diagnóstico Aplicados a Robótica

El hecho que los sistemas robóticos actuales, tanto brazos mecánicos como robots móviles estén diseñados principalmente para el trabajo en ambientes extremos para el hombre, hace trascendental que se cuente con sistemas paralelos de apoyo que permitan identificar fallas y posibles riesgos durante el funcionamiento de los mismos. Esto permite evitar la necesidad de poner en peligro la vida de personas que realicen el diagnóstico. También permiten incrementar la utilidad de los sistemas robóticos, ya que reduce la necesidad de realizar detenciones rutinarias de revisión, pues son los propios sistemas los que se encargan de realizar un diagnóstico de sus estado y de las posibles complicaciones que se presenten.

Existe un gran número de trabajos, tanto teóricos como prácticos, de sistemas de detección y diagnóstico de fallas aplicadas a sistemas robóticos. La mayoría de estos trabajos se encuentran enfocados a la detección y diagnóstico de un sinnúmero de diferentes fallas en brazos mecánicos, debido principalmente a que en la industria actual éstos tienen mayor aplicación. La gama de formas en las que se realiza la detección y el diagnóstico es casi tan amplia como el número de trabajos existentes, ya que cada autor busca una forma diferente de resolver el problema en estos sistemas, que poseen un complejo grupo de ecuaciones cinemáticas y dinámicas.

Una línea ha sido la de un estudio analítico de la redundancia presente en las mediciones obtenidas en un brazo robótico, para poder identificar diferentes fallas. El trabajo de [*Visinski, 1993*] presenta uno de los primeros sistemas de detección en brazos robóticos usando esta técnica, en la que se busca generar un

control adaptivo para fallas en las diferentes articulaciones del brazo. Este trabajo se ha desarrollado en una línea muy similar en [*Dixon, 2000*], filtrando las mediciones de torque aplicadas a cada articulación para determinar la existencia de fallas, incluso bajo condiciones de incertidumbre en algunos parámetros del brazo. La teoría del uso de redundancia analítica no-lineal para detectar y diagnosticar las fallas se encuentra posteriormente desarrollada en [*Leuschen, 2002*] y se demuestran sus aplicaciones en brazos mecánicos, detectando fallas tanto en los sensores montados como en los actuadores. También, centrado en el aprovechamiento de la redundancia analítica, el trabajo de [*Kmelnitsky, 2002*] busca utilizar métodos similares en conjunto con la creación y comparación de modelos del brazo para determinar cuando ocurre una falla.

Usando un sistema diferente, [*Schneider, 1996*] aprovecha las ventajas de la lógica difusa para trabajar con las no-linealidades inherentes de los sistemas robóticos. En su método, el uso de modelos permite detectar la aparición de una falla, mientras que un análisis de los residuos basado en lógica difusa permite diagnosticar la falla.

El área de robótica móvil no se encuentra exenta de trabajos destinados al estudio de fallas. El objetivo del sistema de DDF no sólo es entregar información de apoyo al sistema de control de los robots y así permitir que el sistema reaccione ante las fallas que se presenten, sino también entregar información del estado del robot a sus operadores. El fin de utilizar robots móviles, en la mayoría de sus aplicaciones, es proteger a los operadores de ambientes nocivos y peligrosos, por lo que la reparación en el campo de trabajo se ve muchas veces imposibilitada. El análisis estadístico de fallas ocurridas en robots móviles realizado en [*Carlson, 2004*], indica que uno de los problemas más graves que generan las fallas es el alto costo en horas – hombre necesarias para repararlos, debido a que primero se necesita detectar la falla en el sistema, y muchas veces no se tiene conocimiento de qué fue lo que ocurrió mal. El contar

con un sistema de DDF que entregue al operador información del estado del robot es una herramienta indispensable al momento de corregir la falla, ya que los registros del robot indican qué es lo que se encuentra funcionando en forma errónea, agilizando así el proceso de reparación. Además, un sistema de DDF que entregue una correcta identificación de la fallas es una gran ventaja, ya que muchas de las que ocurren, especialmente aquellas en sensores, pueden ser acomodadas permitiendo que el robot continúe con su trabajo. Si bien puede que el robot quede trabajando en forma limitada, no requerirá de detenciones para ser reparado.

Dado el ambiente dinámico en el que se debe considerar este tipo de robots, la medición de variables se ve afectada por el ruido, las inexactitudes y el efecto de las no-linealidades del sistema, haciendo que no sean aplicables los sistemas más simples de detección de fallas. Por ello los métodos de detección y diagnóstico para robots móviles se basan en sistemas más complejos que utilizan redundancia analítica, específicamente filtros de Kalman y métodos estadísticos, para identificar las fallas que se presenten en el sistema.

Para poder tener éxito en tan difíciles condiciones, el trabajo de [*Kawabata, 2002*] presenta un sistema sensorial expandido, que agrupa los componentes de los robots móviles en módulos, cada uno de los cuales cuenta con sensores que monitorean su funcionamiento. Si bien el trabajo sólo presenta un sistema de monitoreo, demuestra que su aplicación permite identificar una amplia gama de fallas en el robot.

Otro de los métodos utilizados se basa en un banco de filtros de Kalman, que permiten realizar una estimación más precisa de las variables de estado del robot y compararlas con las fallas. El trabajo de [*Roumeliotis, 1998a*] presenta uno de los primeros métodos de detección aplicados a robots móviles. Este trabajo utiliza un conjunto de filtros de Kalman, cada uno de los cuales considera un modelo del robot bajo una de las siguientes fallas: la existencia de

una rueda pinchada (decrecimiento en el radio de una de las ruedas) y la existencia de una perturbación periódica en una rueda. En forma paralela, en [*Roumeliotis, 1998b*] se presenta un método estadístico para detectar fallas en los sensores del robot, incluyendo fallas en el giróscopo (que permite identificar la velocidad rotacional del robot) y en el encoder de una rueda (que permite determinar el giro de la rueda). Si bien ambos sistemas presentan una excelente respuesta en cuanto a detectar las fallas, requieren de un gran número de observaciones para poder determinarlas, haciendo retardar su respuesta en el tiempo.

Continuando con el uso de bancos de filtros de Kalman, [*Goel, 2000*] presenta un esquema de detección de fallas en un robot de cuatro ruedas; cada filtro se encarga de una de las fallas a detectar: giróscopo, encoders y un pinchazo de 1 o 2 ruedas en cada lado, y posteriormente una red neuronal para analizar los residuos y así determinar qué falla es la que ocurrió. Si bien este sistema presenta una muy buena respuesta en tiempo, las fallas de proceso no son diferenciables entre si, es decir, el sistema no es capaz de determinar si fueron 1 o 2 las ruedas pinchadas. Por su parte, [*Washington, 2000*] crea un sistema donde mezcla el uso de filtros de Kalman y modelos de Markov para mejorar la identificación de fallas aprovechando las ventajas de ambos métodos.

El trabajo de [*Hashimoto, 2001*] presenta otra versión del uso de sistemas estadísticos para el diagnóstico de fallas en sensores. Mediante su sistema de detección y diagnóstico el robot es capaz de identificar fallas en sus encoders, el potenciómetro que indica la dirección de la rueda libre y el giróscopo. Su método estadístico demuestra ser muy eficiente en la detección, teniendo un tiempo de respuesta aceptable.

Siguiendo con su trabajo en redundancia analítica en brazos, [*Dixon, 2001*] extiende su aplicación a robots móviles. Su sistema de detección basado en la medición del torque aplicado permite identificar fallas de cambio de radio en

las ruedas y resbalamiento en una rueda, sin necesidad de contar con valores exactos para los parámetros del robot. Si bien el sistema entrega un excelente resultado, se encuentra limitado a la detección de ese tipo de fallas, sin poder incluir fallas de actuadores o sensores.

Finalmente, tomando una dirección diferente, el trabajo de [*Soika, 1997*] presenta un sistema de detección de fallas en los sensores del robot, basado en la creación de mapas de entorno. El proceso se inicia con la creación de un mapa a medida que el robot avanza, el cual se compara con las mediciones proporcionadas por los sensores para identificar fallas en ellos y realizar una calibración de los mismos. El sistema demuestra ser muy bueno y estable, pero se ve limitado a lugares donde existan referencias en las cuales pueda realizar la comparación, al mismo tiempo que las fallas sólo se identifican en sensores de entorno (como sonares) y no de navegación.

En el caso de sistemas distribuidos, los principales aportes se enfocan a robustecer sistemas para evitar la existencia de fallas. En su trabajo, [*Koushanfar, 2002*] muestra como la redundancia existente en los sistemas distribuidos permite obtener mejores mediciones, a la vez que entrega la posibilidad de detectar los elementos que se encuentran con fallas sin necesidad de complejos sistemas de diagnóstico en cada agente. Esto permite pensar que el utilizar sistemas de múltiples robots se simplifica la tarea de detección y diagnóstico, al dividirla entre varios robots.

Los estudios de fallas en sistemas de múltiples robots son escasos. Tal como se ha indicado anteriormente, los trabajos se enfocan en analizar los efectos de la redundancia de agentes, sin especificar sistemas para diagnosticar las fallas que pudiesen ocurrir. Una excepción es el artículo de [*Tinós, 2002*], en el que se analiza un sistema de dos manipuladores robóticos, que al actuar en paralelo permite identificar fallas en base a un sistema simple. El sistema aprovecha el hecho que existen relaciones geométricas cuando ambos brazos se utilizan para

mover un objeto rígido, empleando las ecuaciones para detectar cuándo una articulación se ha quedado pegada o cuándo el motor asociado no entrega el torque requerido, dejando la articulación libre. Este trabajo también confirma que el uso de más agentes en un sistema robótico puede facilitar el trabajo de detección y diagnóstico.

Como se puede observar de los estudios realizados, existe muy poca investigación dirigida a la detección y diagnóstico de fallas en el campo de la robótica móvil, área en la cual es indispensable tener sistemas de monitoreo que permitan lograr una mayor autonomía de los robots. Menor aún es la investigación en los sistemas de múltiples robots, donde queda claro que la existencia de agentes redundantes permitiría una flexibilidad importante, a la vez que entregaría la posibilidad de simplificar los sistemas de detección y diagnóstico que se apliquen.

2.4. Modelo Matemático de un Robot Móvil

Con el fin de poder realizar un estudio teórico de la detección y diagnóstico de fallas en robots móviles, se requiere disponer de un modelo lo más completo posible de un robot real. Esto permite observar, a través de simulaciones, los efectos de las diferentes fallas sin necesidad de contar con elementos defectuosos, y así diseñar estrategias para poder enfrentarlos.

La figura 2.22 presenta un esquema simplificado de un robot móvil de dos ruedas independientes o de tracción diferencial.

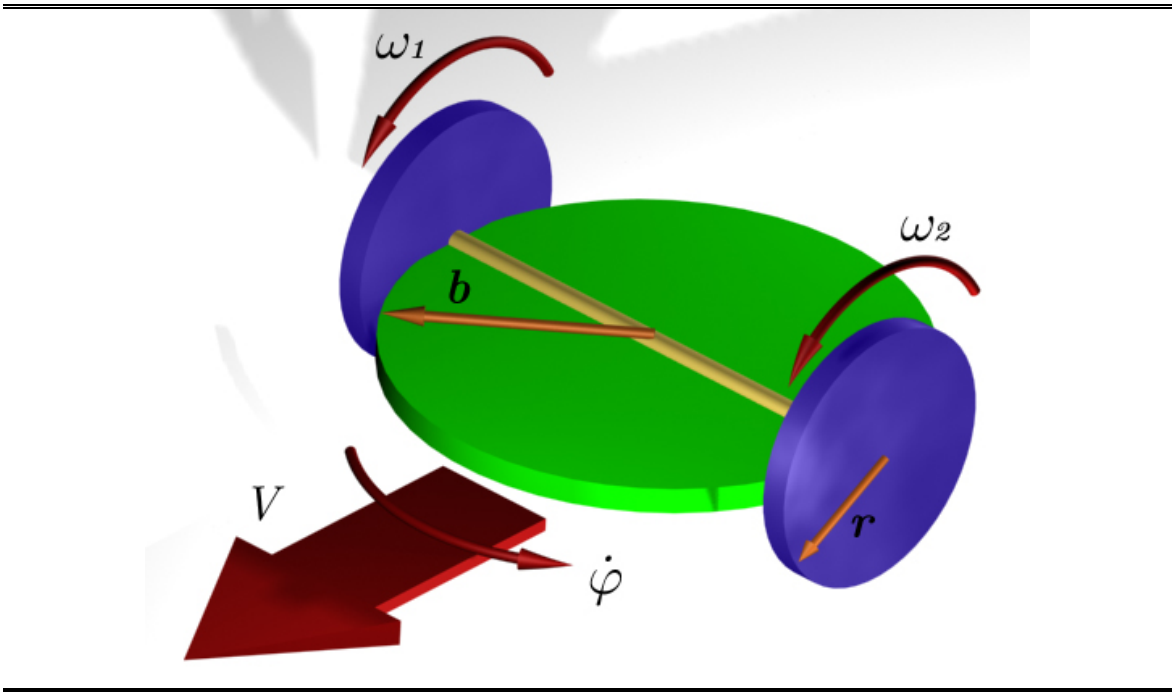

Figura 2.22: Esquema de un Robot Móvil de Dos Ruedas.

Este esquema muestra tanto los parámetros físicos del robot, como las variables dinámicas. Los parámetros físicos del robot son: la longitud del eje (o distancia entre las ruedas) denominada ***2b***; el radio de cada rueda, r_1 y r_2; la masa del cuerpo del robot, ***M***; y la masa de cada una de las ruedas, ***m***.

Las variables relacionadas con el movimiento del robot son: la velocidad lineal que posee el cuerpo del robot, V; la velocidad angular que posee el cuerpo, $\dot{\varphi}$; y las velocidades angulares de las ruedas, $\omega_1 = \dot{\theta}_1$ para la rueda derecha y $\omega_2 = \dot{\theta}_2$ para la rueda izquierda.

Además, entre las variables dinámicas se encuentra también la posición absoluta del robot en el espacio. Esta posición queda definida por las coordenadas bi-dimensionales del centro de masa (x,y) y el ángulo entre la dirección de movimiento del robot y el eje X, denominado φ, tal como se observa en la figura 2.23.

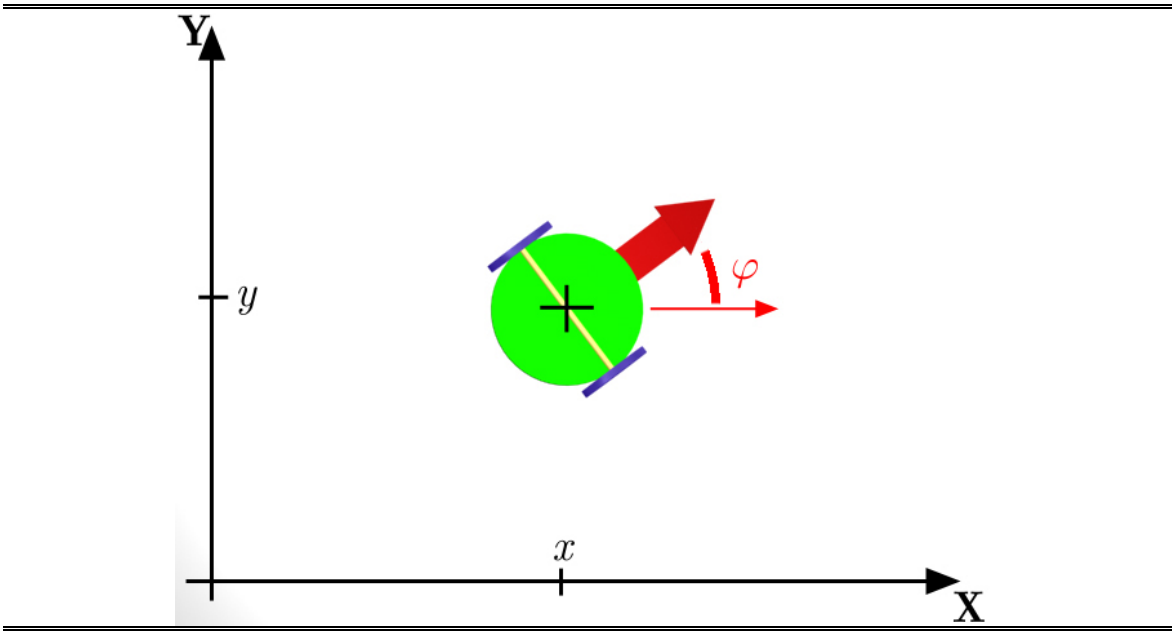

Figura 2.23: Variables de Posición Absoluta del Robot.

El modelo también consta de parámetros eléctricos, asociados a los motores DC que se utilizan para entregar la tracción a las ruedas del robot.

Para generar el modelo completo, el problema se divide en tres partes: las ecuaciones cinemáticas, las ecuaciones dinámicas y finalmente las ecuaciones eléctricas del sistema [*Wise, 1999*].

2.4.1. Ecuaciones Cinemáticas

Las ecuaciones cinemáticas son aquellas que relacionan la velocidad de giro de cada una de las ruedas con las variables de la posición del robot: (x, y, φ) [*Angeles, 1997*].

Considerando al robot como un cuerpo rígido, la velocidad lineal del centro de masa se obtiene en base al promedio de las velocidades lineales de sus extremos, que es donde se encuentran las ruedas. A su vez, la velocidad lineal de cada una de sus ruedas se obtiene como el producto de la velocidad angular

(velocidad de giro) y el radio de ellas. Así, la velocidad del centro de masa queda definida por:

$$V = \frac{r\left(\dot{\theta}_1 + \dot{\theta}_2\right)}{2} \qquad (2.1)$$

El ángulo de giro del robot se determina en base a las relaciones geométricas entre el movimiento de cada lado del robot, tal como se muestra en la figura 2.24.

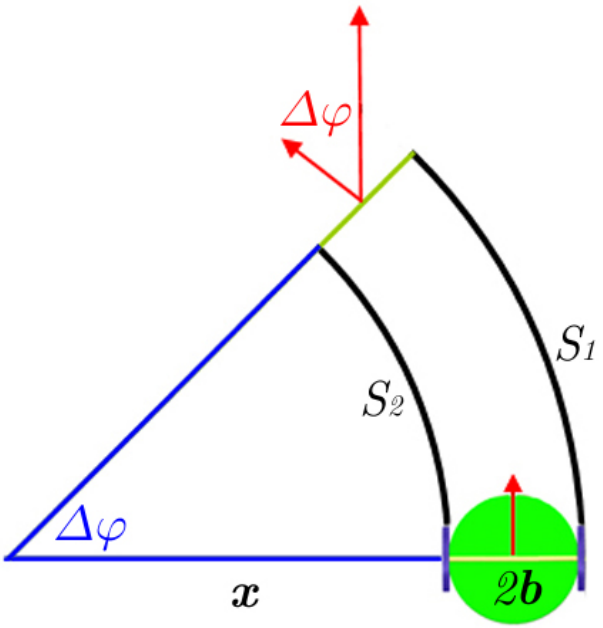

Figura 2.24: Relaciones Geométricas para el Ángulo de Giro.

Según se observa en la figura, el ángulo de giro del robot es igual al ángulo del arco sostenido por la trayectoria. Dado que por definición el ángulo de dirección del robot, φ, aumenta en contra de los punteros del reloj, y considerando que la rueda derecha gira a una mayor velocidad que la izquierda, el ángulo de dirección debe aumentar en $\Delta\varphi$.

Según se aprecia en la figura, la rueda izquierda sostiene un arco de radio x, por lo que la distancia recorrida por esa rueda esta dada por:

$$S_2 = r\Delta\theta_2 = x\Delta\varphi \qquad (2.2)$$

La rueda derecha, que se encuentra más lejos del centro de la circunferencia que determina la trayectoria, recorre una distancia mayor en el mismo tiempo, dada por:

$$S_1 = r\Delta\theta_1 = (x + 2b)\Delta\varphi \qquad (2.3)$$

Calculando el valor de la diferencia S_1-S_2, y dividiendo por el tiempo transcurrido Δt, se obtiene la relación entre la velocidad de giro del robot, y la velocidad de cada una de sus ruedas, como se indica en la ecuación 2.4:

$$\dot{\varphi} = \lim_{\Delta t \to 0} \frac{\Delta\varphi}{\Delta t} = \frac{r\left(\dot{\theta}_1 - \dot{\theta}_2\right)}{2b} \qquad (2.4)$$

Para obtener las coordenadas de la posición del centro de masa del robot, se requiere primero descomponer la velocidad lineal del robot en las velocidades asociadas a cada eje del plano **XY**. Estas velocidades son la proyección de la velocidad expresada en la ecuación 2.1 en cada eje, con lo que se obtiene que:

$$\begin{aligned} \dot{x} &= V\cos(\varphi) \\ \dot{y} &= V\sin(\varphi) \end{aligned} \qquad (2.5)$$

Integrando las expresiones obtenidas en 2.4 y 2.5, se obtienen las coordenadas absolutas del centro de masa del robot en el plano **XY**, también denominada la postura del robot:

$$\begin{aligned} x(t) &= x(0) + \int_0^t \frac{r\left(\dot{\theta}_1 + \dot{\theta}_2\right)}{2}\cos\left(\varphi(t)\right)dt \\ y(t) &= y(0) + \int_0^t \frac{r\left(\dot{\theta}_1 + \dot{\theta}_2\right)}{2}\sin\left(\varphi(t)\right)dt \\ \varphi(t) &= \varphi(0) + \int_0^t \frac{r\left(\dot{\theta}_1 - \dot{\theta}_2\right)}{2b}dt \end{aligned} \qquad (2.6)$$

Así, en base a la postura inicial y a las velocidades angulares de las ruedas, es posible determinar la posición relativa a esa postura inicial en todo momento, que es el sistema básico de posicionamiento usado en robots móviles, llamado posicionamiento relativo o basado en odometría.

2.4.2. Ecuaciones Dinámicas

Las ecuaciones dinámicas, tal como lo indica su nombre, relacionan las variables dinámicas del robot para determinar la aceleración de cada rueda en base al momento de torción aplicado por cada motor [*Craig, 1989*].

Considerando el cuerpo del robot y sus ruedas como discos rígidos de masas M y m respectivamente, las ecuaciones dinámicas se determinan en base al cálculo del Lagrangeano del sistema. Dado que el robot se mantiene en un nivel de altura fijo, la energía potencial se mantiene constante, reduciendo el problema del Lagrangeano al cálculo de la energía cinética del cuerpo (K_C) y de cada rueda (K_{r1} y K_{r2}). Entonces, el Lagrangeano está dado por:

$$\mathfrak{L} = K_C + K_{r_1} + K_{r_2} \qquad (2.7)$$

La energía cinética del cuerpo del robot está dada por la suma de la energía cinética debido a la translación del cuerpo (asociado a la velocidad lineal) y la energía cinética de la rotación (asociada a la velocidad angular), como se indica en la siguiente expresión:

$$K_c = \tfrac{1}{2} M V^2 + \tfrac{1}{2} I_c \dot{\varphi}^2 \qquad (2.8)$$

En la ecuación 2.8, el parámetro $\boldsymbol{I_C}$ representa el momento de inercia del cuerpo del robot con respecto a su centro de giro, que está dado por:

$$I_c = \tfrac{1}{2} M b^2 \qquad (2.9)$$

En forma similar, la energía cinética de cada rueda queda determinada por la siguiente expresión:

$$K_{r_i} = \tfrac{1}{2} m v_i^{\,2} + \tfrac{1}{2} I_r \dot{\theta}_i^{\,2}, i = 1,2 \qquad (2.10)$$

En este caso v_i representa la velocidad lineal de cada rueda, donde $v_i = r\dot{\theta}_i$, e $\boldsymbol{I_r}$ simboliza el momento de inercia de la rueda, dado por:

$$I_r = \tfrac{1}{2} m r^2 \qquad (2.11)$$

Utilizando las ecuaciones 2.7 a 2.11 en conjunto con las ecuaciones cinemáticas derivadas en el punto anterior, la ecuación del Lagrangeano queda definida como:

$$\mathfrak{L} = \left[\frac{3r^2}{16}\left(M + 4m\right)\right]\left(\dot{\theta}_1^2 + \dot{\theta}_2^2\right) + \left[\frac{Mr^2}{8}\right]\dot{\theta}_1\dot{\theta}_2 \qquad (2.12)$$

Tal como se indica en [*Craig, 1989*], las ecuaciones dinámicas pueden obtenerse a partir del Lagrangeano en base a la siguiente ecuación:

$$\tau_i = \frac{d}{dt}\left(\frac{\partial}{\partial \dot{\theta}_i}\mathfrak{L}\right) - \frac{\partial}{\partial \theta_i}\mathfrak{L} \qquad (2.13)$$

En esta expresión τ es un vector columna que contiene el valor del torque aplicado a cada una de las ruedas. Calculando las derivadas necesarias en base al Lagrangeano obtenido en 2.21, las ecuaciones dinámicas que se obtienen son las siguientes:

$$\begin{bmatrix}\tau_1 \\ \tau_2\end{bmatrix} = \begin{bmatrix}\frac{3r^2}{8}\left(M + 4m\right) & \frac{Mr^2}{8} \\ \frac{Mr^2}{8} & \frac{3r^2}{8}\left(M + 4m\right)\end{bmatrix}\begin{bmatrix}\ddot{\theta}_1 \\ \ddot{\theta}_2\end{bmatrix} \qquad (2.14)$$

Con el fin de dar más exactitud al modelo, se debe agregar también el efecto del roce dinámico sobre cada rueda. El roce dinámico puede ser modelado

en forma simple, como un par de torción que ejerce una fuerza contraria y proporcional, con constante de proporcionalidad $\boldsymbol{\rho}$, a la velocidad de giro de cada una de las ruedas.

$$\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} = \begin{bmatrix} \frac{3r^2}{8}(M+4m) & \frac{Mr^2}{8} \\ \frac{Mr^2}{8} & \frac{3r^2}{8}(M+4m) \end{bmatrix}^{-1} \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} - \begin{bmatrix} \rho\dot{\theta}_1 \\ \rho\dot{\theta}_2 \end{bmatrix} \qquad (2.15)$$

Integrando en el tiempo esta ecuación se pueden obtener las velocidades de cada rueda, necesarias para resolver las ecuaciones cinemáticas.

2.4.3. Ecuaciones Electromecánicas

El último paso para generar un modelo completo es la incorporación de los motores utilizados, los cuales proporcionan el torque necesario a las ruedas según el voltaje aplicado a cada uno.

Tal como se indica en la figura 2.25, los motores se encuentran acoplados a las ruedas a través de un sistema de reducción, los cuales disminuyen la velocidad de giro de las ruedas, a cambio de un aumento en el torque entregado.

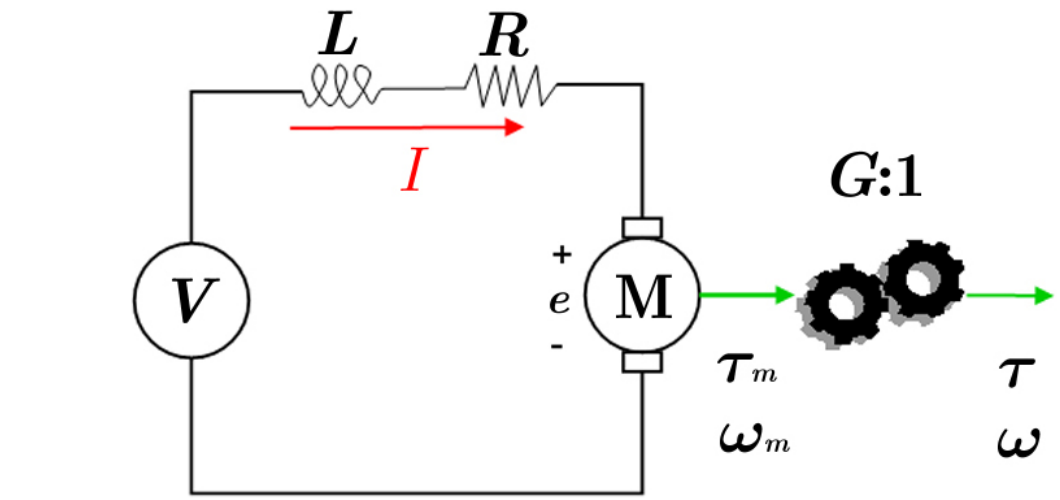

Figura 2.25: Esquema del Sistema Electromecánico.

En el circuito equivalente de la figura 2.25, $\boldsymbol{R}$ representa la resistencia interna y $\boldsymbol{L}$ la inductancia del motor. Recorriendo la malla completa del circuito equivalente se obtiene la siguiente ecuación:

$$V = L\frac{di}{dt} + Ri + e \qquad (2.16)$$

Las ecuaciones del motor DC establecen que el torque aplicado es proporcional a la corriente que circula por el circuito y que el voltaje inducido, e, es proporcional a la velocidad angular del eje. Por lo tanto:

$$\begin{aligned} e &= K_e\omega_m \\ \tau_m &= K_t i \end{aligned} \qquad (2.17)$$

El parámetro $\boldsymbol{K_e}$ se denomina constante del motor, mientras $\boldsymbol{K_t}$ es la constante de armadura.

Al acoplar los motores a las ruedas a través de una reducción, la velocidad angular de las ruedas se ve reducida $\boldsymbol{G}$ veces con respecto a la velocidad del eje del motor, mientras que el torque aplicado a las ruedas se incrementa $\boldsymbol{G}$ veces:

$$\dot{\theta} = \frac{\omega_m}{G}$$
$$\tau = G\tau_m \qquad (2.18)$$

Uniendo las ecuaciones 2.16, 2.17 y 2.18 se obtienen las ecuaciones electromecánicas del robot, que relacionan el voltaje aplicado a cada motor con el torque que recibe la rueda respectiva, como se indica a continuación:

$$V = L\frac{di}{dt} + Ri + K_e G\dot{\theta}$$
$$\tau = K_t G i \qquad (2.19)$$

2.4.4. Modelo del Robot de Dos Ruedas

En base a las ecuaciones cinemáticas, dinámicas y electromecánicas derivadas anteriormente, es posible componer un modelo matemático que describe el comportamiento del robot en base al voltaje aplicado a cada motor, y así tener una herramienta que permite analizar con mayor facilidad los efectos de cambios en los parámetros y fallas en el sistema.

Las ecuaciones que describen al robot son las siguientes:

$$L\frac{di_1}{dt} = V_1(t) - Ri_1 - K_e G\dot{\theta}_1$$
$$L\frac{di_2}{dt} = V_2(t) - Ri_2 - K_e G\dot{\theta}_2 \qquad (2.20)$$

$$\begin{bmatrix}\ddot{\theta}_1\\ \ddot{\theta}_2\end{bmatrix} = \begin{bmatrix}\frac{3r^2}{8}(M+4m) & \frac{Mr^2}{8}\\ \frac{Mr^2}{8} & \frac{3r^2}{8}(M+4m)\end{bmatrix}^{-1}\begin{bmatrix}K_t G i_1\\ K_t G i_1\end{bmatrix} - \begin{bmatrix}\rho\dot{\theta}_1\\ \rho\dot{\theta}_2\end{bmatrix} \qquad (2.21)$$

$$\dot{\theta}_1(t) = \dot{\theta}_1(0) + \int_0^t \ddot{\theta}_1 dt$$
$$\dot{\theta}_2(t) = \dot{\theta}_2(0) + \int_0^t \ddot{\theta}_2 dt \qquad (2.22)$$

$$x(t) = x(0) + \int_0^t \frac{r\left(\dot{\theta}_1 + \dot{\theta}_2\right)}{2} \cos\left(\varphi(t)\right) dt$$
$$y(t) = y(0) + \int_0^t \frac{r\left(\dot{\theta}_1 + \dot{\theta}_2\right)}{2} \sin\left(\varphi(t)\right) dt \qquad (2.23)$$
$$\varphi(t) = \varphi(0) + \int_0^t \frac{r\left(\dot{\theta}_1 - \dot{\theta}_2\right)}{2b} dt$$

Según los sensores utilizados, varias de estas variables pueden ser medidas durante el funcionamiento del robot. Las velocidades angulares de cada rueda se miden empleando encoders, mientras que el ángulo de dirección puede obtenerse utilizando, por ejemplo, una brújula digital. Métodos de navegación más sofisticados pueden incluir un Sistema de Posicionamiento Global (o GPS), obteniéndose así la posición (x, y) del robot en forma absoluta y sin necesidad de realizar localización relativa. También se pueden utilizar acelerómetros y giróscopos para determinar la velocidad lineal y la de giro respectivamente.

2.5. Sistema de Control

Una parte fundamental de un robot móvil es su sistema de control, por lo que no puede ser dejado de lado.

Un sistema de robots cooperativos se basa generalmente en una estructura jerarquizada de sistemas físicos y de control que permiten operar cada robot y coordinar los esfuerzos independientes en un sistema conjunto. Debido a lo complejo de las ecuaciones cinemáticas y dinámicas que describen el comportamiento de un robot móvil, la estrategia de control es siempre

subdividida en tareas más simples, evitando así la implementación de sistemas de control muy complejos y que requieran mucha potencia computacional. La figura 2.26 ilustra la división realizada de la estructura de control, según el nivel de control que posee cada nivel.

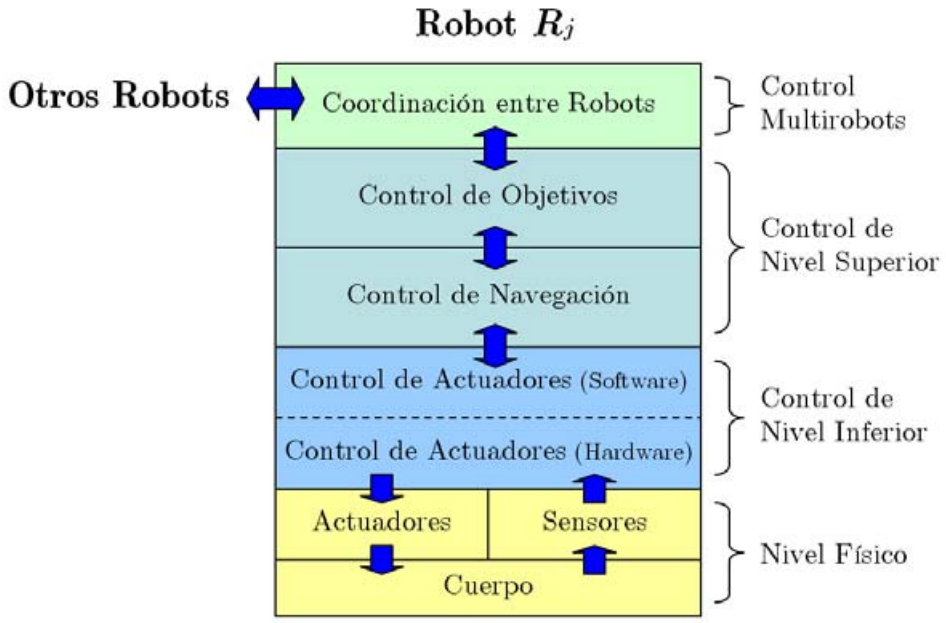

Figura 2.26: Estructura de Control en un Sistema Cooperativo de Robots.

En el nivel físico de la estructura se encuentran el cuerpo del robot y sobre este los sensores, que entregan la información del estado del robot, y los actuadores, que generan el movimiento en el cuerpo.

La capa de control de nivel inferior se encarga de regular las velocidades o momentos de torsión aplicados por los actuadores, utilizando como retroalimentación la información de los sensores, y recibiendo su referencia de control de parte de la capa de control superior. Esta capa puede subdividirse en dos. Primero, una capa de control por hardware, que genera el bucle de control básico sobre las velocidades de los actuadores, lidiando con la dinámica del robot. Gracias a que las ecuaciones dinámicas son lineales, generalmente a este nivel se utilizan sistemas de control convencionales como lo es un controlador

PID. Debido a que las ecuaciones cinemáticas presentan nolinealidades, el control de nivel superior es un elemento más complejo y requiere del uso de herramientas más apropiadas como la lógica difusa y los sistemas de control heurísticos. Estos se encargan de determinar las velocidades angulares de las ruedas, teniendo como objetivo que el robot siga una cierta trayectoria. En [*Carrasco, 2003*] se explica el funcionamiento de un control para robots móviles basado en lógica difusa, que es el que se utiliza en las simulaciones para probar el sistema de detección de fallas. La figura 2.27 muestra el funcionamiento de estos dos sub-niveles en conjunto con el cuerpo del robot.

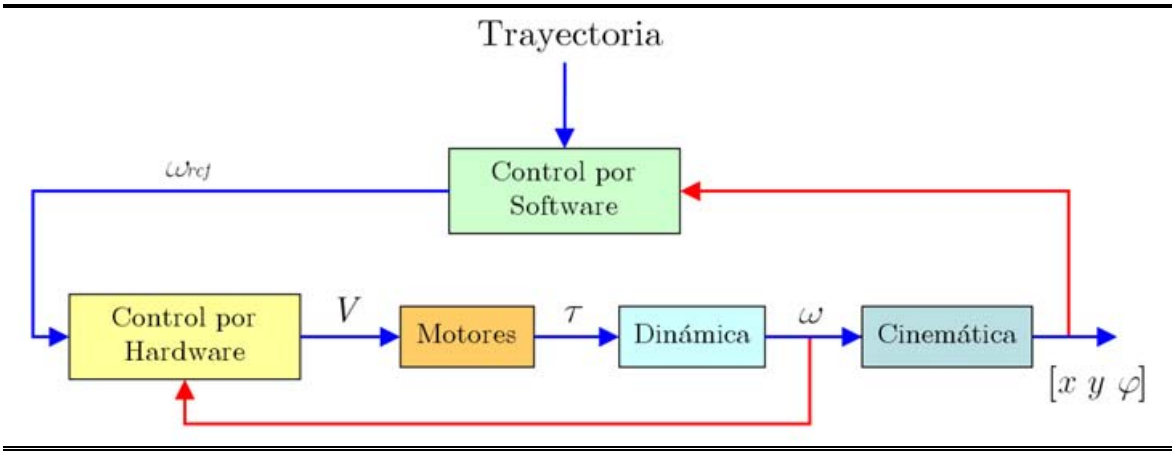

Figura 2.27: Esquema Habitual de la Capa de Control de Nivel Inferior.

La capa de control de nivel superior se encarga de controlar los movimientos del robot según los objetivos determinados para esa unidad. El control de navegación permite trazar las trayectorias, esquivar los obstáculos, y movilizar el robot de un punto a otro. Sobre este, un control de objetivos se encarga de determinar los puntos a los que debe navegar el robot y qué actividades debe realizar en cada una de sus paradas. A nivel de un único robot, este es el control superior que determina todas las actividades que debe realizar.

En el caso de sistemas cooperativos de robots, existe una capa superior de control que se encarga de administrar los recursos de cada robot y permite coordinar los esfuerzos del sistema global para completar la o las tareas

designadas. Este control es especialmente importante en sistemas heterogéneos de robots, ya que no todos los robots cuentan con los mismos equipos, haciendo fundamental el coordinar los esfuerzos correctamente para no desperdiciar recursos. Es importante destacar que esta capa de control no necesariamente se implementa a base de un comando central, sino que también puede ser un sistema de control distribuido, donde a nivel local de cada robot se toman decisiones según el estado del resto de los robots buscando cumplir el objetivo común.

Esta estrategia permite además una flexibilidad al aplicar los sistemas de control sobre diferentes modelos de robots móviles, ya que sólo se requiere adaptar las capas inferiores según el robot utilizado, dejando las capas superiores intactas.

3. Arquitectura Multicapa de Detección y Diagnóstico de Fallas para Robots Cooperativos

El presente capítulo describe en forma detallada la arquitectura de detección y diagnóstico de fallas desarrollada. Primero, se detallan las fallas que debe detectar y diagnosticar el sistema, seguido de una descripción general de la arquitectura utilizada. Posteriormente, se describen en profundidad las dos capas que componen la arquitectura de detección y diagnóstico, analizando su robustez y eficiencia a través de simulaciones. Finalmente se explica la interacción que existe entre ambas capas y como esta mejora el funcionamiento del sistema completo.

3.1. Descripción de las Fallas a Estudiar

El estudio realizado en [*Carlson, 2004*] analiza una gran gama de fallas frecuentes en robots móviles durante un período de 3 años, tanto para aquellos robots de exploración en ambientes dinámicos como aquellos en ambientes estructurados. En base a este estudio se han determinado las fallas más frecuentes y/o más importantes, para incluirlas en el grupo de fallas diagnosticables por la arquitectura, dejando de lado fallas muy poco frecuentes como lo son las fallas en los sistemas de potencia.

Las fallas a analizar pueden ser agrupadas en dos conjuntos dependiendo del grado en que afectan en correcto funcionamiento del robot: las fallas fatales y las fallas secundarias.

El grupo de fallas fatales está compuesto por aquellas fallas en actuadores y sensores que impiden que el robot continúe en funcionamiento o que disminuyen significativamente la entrega de ayuda al resto del grupo. En el caso de las fallas en actuadores, la principal falla que inhabilita un robot es la pérdida

del momento de torsión en las ruedas. Esto se debe a que se ha desacoplado la rueda al motor o a que la rueda se ha atascado y el motor no es capaz de continuar girando. Esta falla puede ocurrir en cada una de las ruedas por separado o en ambas al mismo tiempo, por lo que se consideran tres casos diferentes.

La otra falla común asociada a los actuadores y afecta en forma importante el movimiento del robot es que alguna de las ruedas resbale. Actualmente todos los robots móviles sufren en cierto grado de este tipo de fallas, la cual puede ser disminuida significativamente utilizando técnicas de odometría calibrada [*Borestein, 1996a*] y el uso del filtro de Kalman. La figura 3.1 muestra la diferencia entre la localización basada en odometría y la basada en odometría calibrada utilizando un filtro de Kalman. En el Anexo A se presenta una descripción más detallada de la utilización y los fundamentos de un filtro de Kalman.

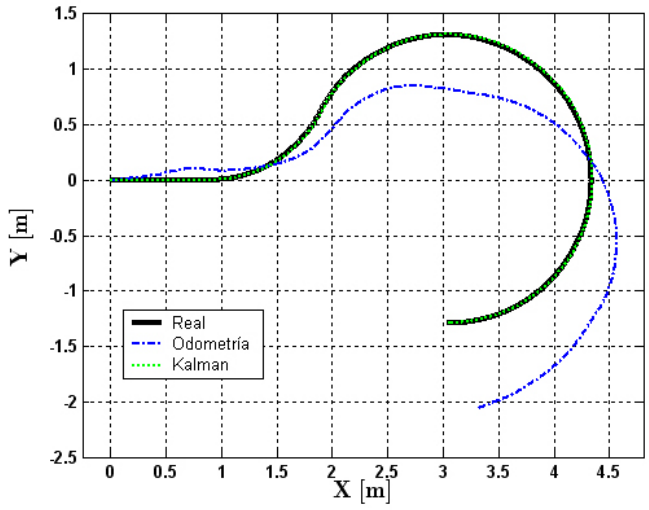

Figura 3.1: Comparación de Sistemas Básicos de Localización.

El problema se agrava cuando el resbalamiento ocurre en forma importante y afecta el movimiento continuo del robot, por lo que debe incluirse entre las fallas fatales. A nivel de simulaciones, el efecto causado por el

resbalamiento es equivalente a la reducción del radio de la rueda correspondiente, siendo el extremo (cuando el robot solo resbala) equivalente a un radio de rueda igual a cero. Esto hace necesario modificar levemente el modelo matemático presentado en las ecuaciones 2.20 a 2.23 extendiéndolo para incorporar este elemento. Las ecuaciones 3.1 a 3.4 presentan el modelo extendido:

$$\begin{aligned} L\frac{di_1}{dt} &= V_1(t) - Ri_1 - K_e G\dot{\theta}_1 \\ L\frac{di_2}{dt} &= V_2(t) - Ri_2 - K_e G\dot{\theta}_2 \end{aligned} \qquad (3.1)$$

$$\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} = \begin{bmatrix} \frac{3r_1^2}{8}(M+4m) & \frac{Mr_1r_2}{8} \\ \frac{Mr_1r_2}{8} & \frac{3r_2^2}{8}(M+4m) \end{bmatrix}^{-1} \begin{bmatrix} K_t G i_1 \\ K_t G i_1 \end{bmatrix} - \begin{bmatrix} \rho\dot{\theta}_1 \\ \rho\dot{\theta}_2 \end{bmatrix} \qquad (3.2)$$

$$\begin{aligned} \dot{\theta}_1(t) &= \dot{\theta}_1(0) + \int_0^t \ddot{\theta}_1 dt \\ \dot{\theta}_2(t) &= \dot{\theta}_2(0) + \int_0^t \ddot{\theta}_2 dt \end{aligned} \qquad (3.3)$$

$$\begin{aligned} x(t) &= x(0) + \int_0^t \frac{r_1\dot{\theta}_1 + r_2\dot{\theta}_2}{2}\cos(\varphi(t))dt \\ y(t) &= y(0) + \int_0^t \frac{r_1\dot{\theta}_1 + r_2\dot{\theta}_2}{2}\sin(\varphi(t))dt \\ \varphi(t) &= \varphi(0) + \int_0^t \frac{r_1\dot{\theta}_1 - r_2\dot{\theta}_2}{2b}dt \end{aligned} \qquad (3.4)$$

A nivel de los sensores, se consideran como fallas fatales aquellas que afectan a los sensores de navegación del robot. Los sensores de velocidad en cada una de las ruedas (o encoders) y el giróscopo para determinar la velocidad de rotación del cuerpo del robot son los elementos básicos con los cuales se realiza navegación por odometría [*Borestein, 1996b*]. Es por ello que las fallas sobre estos sensores se consideran a nivel de fallas fatales. Como ambos sensores

requieren estar mecánicamente acoplados a los elementos que giran para obtener la medición, la falla que más ocurre es la que aparece cuando el sensor se desacopla y entrega una medida de velocidad cero.

La clasificación de fallas secundarias puede a su vez subdividirse en fallas de sensores y en fallas del sistema de comunicación.

La gama de sensores que pueden adicionarse a un robot para realizar operaciones de navegación, mejorar la localización y para cumplir con otros objetivos requeridos es muy amplia, y cada uno de ellos puede presentar fallas. Es por esto que en el caso de fallas secundarias de sensor se analizan los sensores más utilizados para el funcionamiento normal de un robot.

Dado que la localización por odometría diverge en el largo plazo, es muy común el uso de sensores absolutos como brújulas magnéticas, el llamado "Sun Sensor" [*Roumeliotis, 1997*], GPS o radiofaros para triangulación, que mejoran la localización del robot. Si bien este tipo de sensores mejoran la localización del robot, las fallas que se presentan en ellos causan un grave problema para la estimación de la posición, como se ilustra en las siguientes figuras.

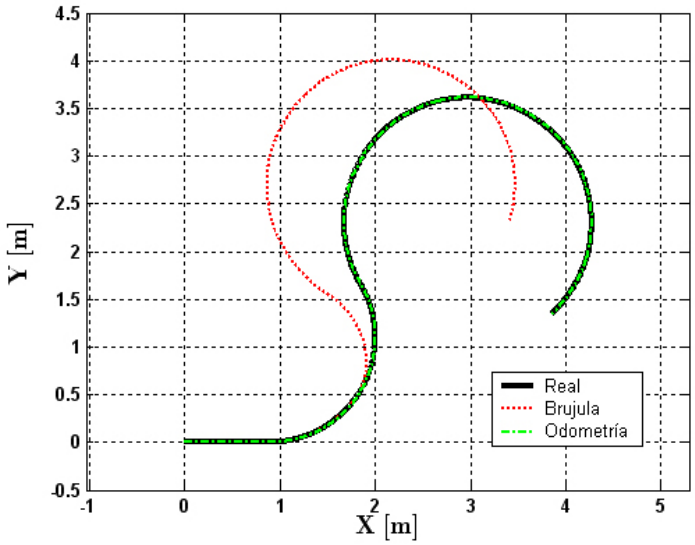

Figura 3.2: Navegación con una Falla en la Brújula Digital.

En la figura 3.2 se observa que al ocurrir una falla aditiva sobre la brújula, la estimación de la posición se ve afectada en el monto de la falla, perdiendo las ventajas de contar con un sensor absoluto de posición, mientras que la odometría calibrada sin el uso de la brújula mantiene una estimación mejor.

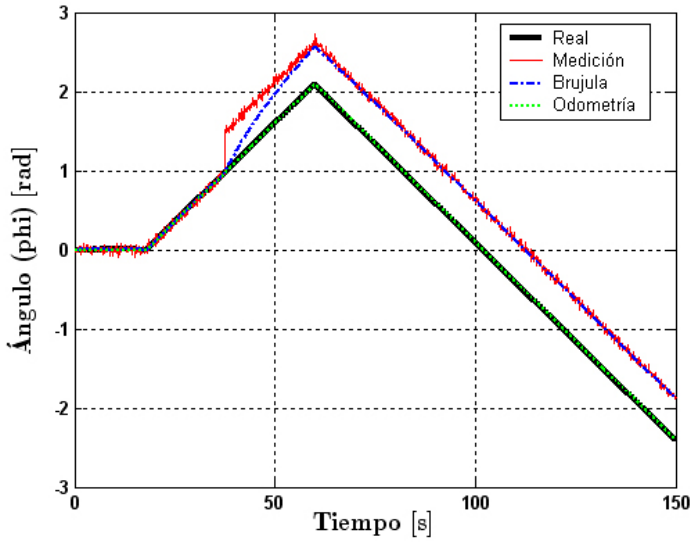

Figura 3.3: Estimación de φ con una Falla en la Brújula Digital.

La figura 3.3 muestra como el uso de la brújula mejora la estimación del ángulo de dirección del robot al usar un filtro de Kalman. Si bien el valor medido posee ruido, el uso del filtro de Kalman entrega una estimación excelente del ángulo real. Al ocurrir la falla aditiva, la estimación obtenida a través del filtro se ve afectada por la medición errónea, mientras que la estimación sólo por odometría calibrada se mantiene más cerca del valor real.

Debido a que este tipo de sensores se utilizan comúnmente en robot móviles, se considera que el robot cuenta con una brújula magnética que entrega al robot el ángulo de dirección en la cual se está moviendo.

A nivel de navegación, es muy frecuente el uso de sensores de distancia para esquivar obstáculos, como sonares, sistemas infrarrojos y scanner láser.

Para este trabajo se considera que el robot cuenta con un sonar frontal, que le permite identificar obstáculos y la distancia a la que se encuentran.

Para ambos sensores, las fallas más comunes y que son las incorporadas en el diseño de la arquitectura de DDF, son las de tipo aditivo y las llamadas fallas de "valor permanente de salida" o también llamadas de tipo "stuck". Las primeras desfasan la medición en un valor determinado y generalmente son de tipo degenerativo. Esto significa que va aumentando el valor del desfase paulatinamente. Las fallas de "valor permanente a la salida" ocurren cuando el sensor mantiene el valor de salida, ya sea por un mal funcionamiento interno o por factores externos que desfasan la medición. Por ejemplo, en el caso de una brújula magnética, si algún elemento que contiene hierro queda sobre el robot, dependiendo de la cercanía con la brújula puede generar tanto fallas aditivas como de valor permanente de salida. Si está lejos, sólo desfasa la medición de la brújula, mientras que si está muy cerca dejará un valor constante a la salida.

En cuanto a las fallas del sistema de comunicación, estas no son por ahora tan comunes como las anteriores, debido a que actualmente el uso de sistemas inalámbricos de comunicación es muy limitado. A medida que aumente su uso mayor será la posibilidad de que estos ocurran, haciendo entonces necesario el contar con una arquitectura de DDF que permita incluir algo tan vital en la coordinación de grupos de robots. Son tres las fallas principales en el sistema de comunicación: falla del transmisor, falla del receptor y que el robot se encuentre fuera de rango, que es la falla más común de los sistemas de transmisión inalámbricos para robots móviles.

Finalmente se asume que los robots cuentan con sensores que les permiten detectar a otros robots, como por ejemplo una cámara. Esto les permite identificar a otros miembros del grupo para realizar tareas cooperativas.

3.2. Descripción de la Arquitectura

Como se indicó en el capítulo 2.3, existe una gran gama de trabajos dedicados a generar sistemas tolerantes a falla o de acomodación de fallas, pero la mayoría de ellos asumen la existencia de un sistema de DDF capaz de diagnosticar el estado del sistema. Además, los trabajos que si presentan un sistema de DDF sólo poseen un número muy limitado de fallas para identificar, en especial considerando las fallas de tipo fatal. Todo esto hace necesario el desarrollo de una arquitectura de DDF complementaria a la estructura de control del robot, que sea aplicable a diferentes tipos de robots sin necesidad de realizar cambios importantes sobre el robot mismo, ni su sistema de control, comunicándose con los diferentes niveles para entregar la información necesaria.

Dados los requerimientos necesarios, el diseño del sistema de DDF ha sido desarrollado en capas, cada una de las cuales es responsable de diagnosticar un grupo de fallas. Esto se realiza según los requerimientos de cada tarea y la importancia de las diferentes fallas, minimizando las necesidades computacionales para la implementación y aprovechando la existencia de elementos redundantes al interior del grupo de robots. Además, esto permite aprovechar la ventaja de trabajar en conjunto con las capas del sistema de control, ya que cada capa al interior de la arquitectura de DDF puede aprovechar al máximo su interacción con las capas del sistema de control.

Otro beneficio de un sistema de DDF diseñado en capas, es que cada capa se activa sólo cuando ocurre una falla que debe ser detectada por esa capa, sin gastar los recursos del sistema cuando no le corresponde o cuando no es factible la detección. Además, esto permite adaptar el sistema de DDF a la estructura existente en el sistema cooperativo, según sean los sensores y actuadores con que cuente cada robot y la arquitectura del conjunto.

La figura 3.4 muestra la estructura del sistema de DDF diseñado, indicando la interacción existente con los diferentes niveles del robot.

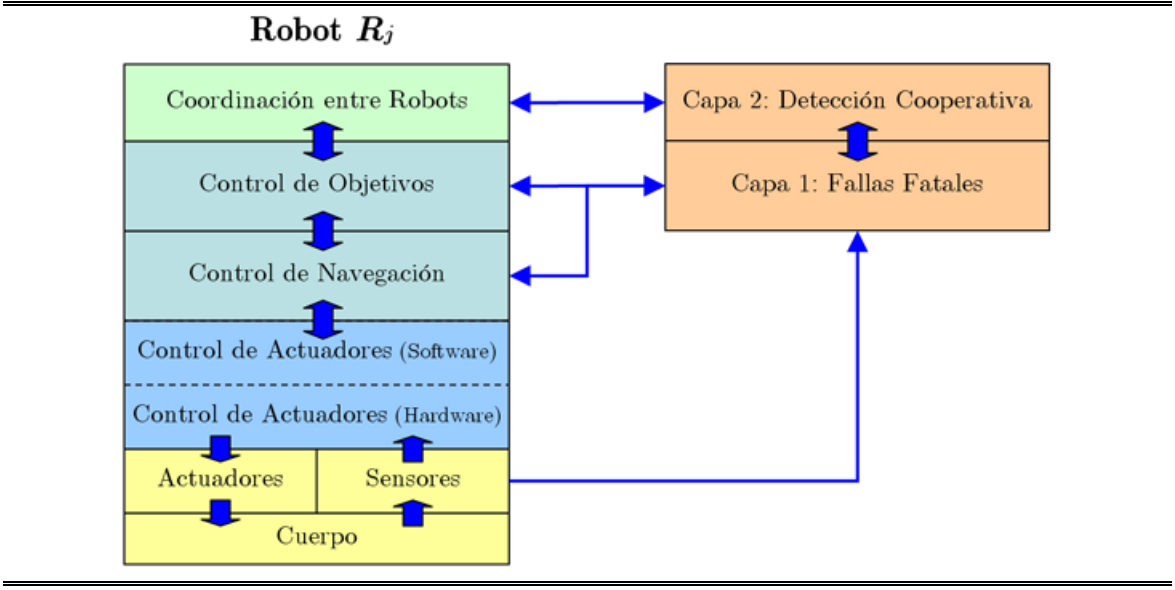

Figura 3.4: Estructura del Sistema de DDF.

La capa 1, llamada "Capa de Fallas Fatales" se encarga de revisar los sistemas que son indispensables para el funcionamiento del robot y cuya falla significaría la eliminación del robot del sistema. Esto requiere de un monitoreo permanente y de una respuesta rápida, que informe al sistema de control si se puede contar con dicho robot para alguna operación. Debido a los requerimientos necesarios, esta capa se implementa sobre cada robot sin necesitar de otros para activarse, lo cual permite un tiempo de respuesta corto y un monitoreo constante de las fallas más importantes.

Las fallas consideradas como secundarias pueden ser monitoreadas a intervalos más largos, por lo que no se requiere de un sistema de revisión continuo ni que presente una respuesta rápida. La capa 2, llamada "Capa de Detección Cooperativa", aprovecha la existencia de redundancia en un sistema de múltiples robots para detectar e identificar las fallas consideradas secundarias que ocurran en un robot.

3.3. Capa 1: Capa de Detección de Fallas Fatales

El uso de múltiples modelos para detección y diagnóstico permite un monitoreo continuo de las fallas que se modelen, haciendo de este método el ideal para implementar la capa 1 de la arquitectura. El problema es que los modelos nunca son exactos, y el ruido presente en las mediciones hace más difícil lograr una correcta determinación del modelo adecuado.

Las redes neuronales y la lógica difusa se han utilizado constantemente para generar sistemas de análisis de los residuos de cada modelo, resultando en sistemas de alto costo computacional y que no entregan un gran nivel de desempeño. Por otro lado, el uso de métodos probabilísticos ha demostrado ser de gran eficiencia, presentando un tiempo de respuesta rápido y un costo computacional más reducido [*Maybeck, 1979*].

La identificación Bayesiana propuesta por [*Maybeck, 2000*] aprovecha las ventajas del método probabilístico de múltiples hipótesis, en conjunto con la información de correlación existente en un banco de filtros de Kalman para determinar qué modelo es más probable que represente correctamente el estado actual del sistema. Dado que todas las fallas a ser consideradas por esta capa son modelables al interior de un filtro de Kalman y que este método presenta los tiempos de respuesta necesarios, la capa 1 se basa en la utilización de la identificación Bayesiana para la detección y el diagnóstico de cada falla.

3.3.1. Identificación Bayesiana usando Múltiples Filtros de Kalman

El método de identificación Bayesiana se basa en el diseño de un banco de $n+1$ filtros de Kalman, para la identificación de n diferentes fallas. Cada uno de los filtros tiene incorporada en su estructura un estado diferente de operación en el sistema, n estados de falla y uno (n=0) de operación normal. Este último

generalmente ya se encuentra implementado en los sistemas de localización de robots móviles para mejorar la estimación de la posición.

La estructura básica de cada uno de los modelos es la siguiente:

$$\mathbb{M}_i : \begin{matrix} X_{k+1,i} = A_i X_{k,i} + B_i U_k + \omega_k \\ Z_{k+1,i} = C_i X_{k+1,i} + \mu_k \end{matrix} ; i = 0..n \qquad (3.5)$$

En la ecuación 3.5 $X_{k,i}$ es el vector de estado del sistema en el tiempo k según el modelo i, $Z_{k,i}$ es el vector de medición según el modelo i para el tiempo k, U_k es la entrada en el tiempo k (que es igual para todos los modelos) y las matrices A_i, B_i y C_i son las que describen el sistema para la falla i, donde i=0 es el estado normal. Las variables ω_k y μ_k representan el ruido Gausseano aditivo presente en el proceso y en la medición respectivamente.

Cada filtro de Kalman del banco, asociado a uno de los *n+1* modelos existentes, entrega una estimación óptima del vector de estado, $X_{k,i}$, y del vector de medición, $Z_{k,i}$, para el sistema según el modelo $\mathbb{M}_i$ y las mediciones del proceso, Y_k, obtenidas.

El diagrama de bloques para este método se ilustra en la figura 3.5.

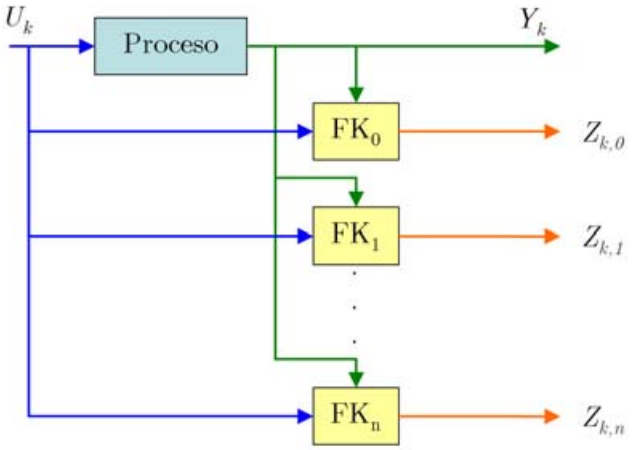

Figura 3.5: Diagrama de Bloques del Método Bayesiano.

El objetivo es determinar la probabilidad de cada hipótesis $\mathbb{H}_i$, que establece que el sistema se encuentra funcionando de acuerdo a las ecuaciones del modelo $\mathbb{M}_i$.

El cálculo de la probabilidad de cada hipótesis $\mathbb{H}_i$ en el tiempo *k+1* se hace utilizando la siguiente ecuación, similar al teorema de Bayes:

$$\mathbb{P}_{k+1}\left(\mathbb{H}_i\right) = \frac{f\left(Y_{k+1} / \mathbb{M}_i, \left[Y_0^T .. Y_k^T\right]\right)\mathbb{P}_k\left(\mathbb{H}_i\right)}{\sum_{j=0}^{n} f\left(Y_{k+1} / \mathbb{M}_j, \left[Y_0^T .. Y_k^T\right]\right)\mathbb{P}_k\left(\mathbb{H}_j\right)} \qquad (3.6)$$

Para determinar la probabilidad de la ecuación 3.6, se requiere calcular la función de densidad condicional *f(.)* con que la medición en el tiempo *k+1* representa el estado del sistema, asumiendo que ha ocurrido la falla i y que se conocen las mediciones anteriores:

$$f\left(Y_{k+1} / \mathbb{M}_i, \left[Y_0^T .. Y_k^T\right]\right) = \beta_{k+1,i} e^{-\frac{1}{2} D_{k+1,i}} \qquad (3.7)$$

Denotando la matriz de covarianza del residuo del modelo i en el tiempo *k+1* como $S_{k+1,i}$, el término $D_{k+1,i}$ en la ecuación 3.7 se define como:

$$D_{k+1,i} = \left[Y_{k+1} - Z_{k+1,i}\right]^T S_{k+1,i}^{-1} \underbrace{\left[Y_{k+1} - Z_{k+1,i}\right]}_{r_{k+1,i}} \qquad (3.8)$$

Mientras que $\beta_{k+1,i}$ es un parámetro determinado por:

$$\beta_{k+1,i} = \frac{1}{\left(2\pi\right)^{\frac{m}{2}} \left|S_{k+1,i}\right|^{\frac{1}{2}}} \qquad (3.9)$$

En la ecuación 3.9, m es la dimensión del vector de. La matriz de covarianza $S_{k+1,i}$ se obtiene directamente del filtro de Kalman asociado, como se describe en el Anexo A.

El término $D_{k+1,i}$ definido en la ecuación 3.8 es la distancia de Mahalanobis del residuo para el modelo *i*, en el tiempo *k+1*. Debido a que los diferentes elementos del vector de medición Y_{k+1} poseen varianzas diferentes, es necesario que la medición de distancia utilizada para el cálculo de la función de densidad condicional en base al residuo considere la matriz de covarianzas para ese residuo. La distancia de Mahalanobis utiliza esta información, a diferencia de la distancia Euclideana.

Es importante destacar que si en la ecuación 3.6 la probabilidad de alguna hipótesis es 0, esta continuará siéndolo de ese instante en adelante. Por esta razón, artificialmente se coloca el valor 0.0001 como límite inferior de la probabilidad de cada hipótesis, eliminando la posibilidad que una de las hipótesis quede descartada.

Este proceso se realiza en forma iterativa y en conjunto con la estimación realizada por cada filtro de Kalman, determinando la probabilidad con la que cada modelo representa el estado actual del sistema dadas las mediciones realizadas. De este modo se puede inferir si el robot dejó el estado normal y cual es el estado de falla más probable en el cual se encuentra, permitiendo el diagnóstico de la falla.

El sistema de detección de fallas se activa si la probabilidad de la hipótesis $\mathbb{H}_0$ es inferior al límite $\mathbb{P}_{\mathrm{DT}}$. Para reducir el número de diagnóstico erróneos, se establece otro límite para activar el diagnóstico tras detectada la ocurrencia de una falla. Si la probabilidad de alguna de las hipótesis de falla supera la probabilidad $\mathbb{P}_{\mathrm{DG}}$ se asume que el sistema está operando con dicha falla activa. $\mathbb{P}_{\mathrm{DT}}$ y $\mathbb{P}_{\mathrm{DG}}$ son los dos parámetros libres que deben ser sintonizados en base a las observaciones para optimizar la eficiencia de la capa. Un nivel muy alto para $\mathbb{P}_{\mathrm{DT}}$ puede generar un gran número de falsas alarmas, mientras uno muy bajo implica un mayor tiempo de detección desde el momento que ocurre la

falla. De igual modo el valor de $\mathbb{P}_{DG}$ determina qué tan rápido o tan confiable se puede realizar el diagnóstico.

3.3.2. Modelos Utilizados y Parámetros de Detección y Diagnóstico

El objetivo de esta capa de DDF es monitorear en forma continua el estado del robot, sin ocupar muchos recursos computacionales. Generar modelos lineales que contengan en su interior la estructura completa descrita en el capítulo 2.4 aumenta mucho el costo de procesamiento. Es por ello que el diseño de los diferentes modelos se basa en una forma simplificada que utiliza sólo las ecuaciones cinemáticas. Gracias a que los sistemas de control para robots móviles, como se describió anteriormente, se basan siempre en un doble lazo de control, donde el lazo superior trabaja con la cinemática del robot, mientras uno inferior trabaja con la dinámica, el sistema de DDF puede emplear directamente las ecuaciones cinemáticas, obviando los elementos dinámicos. Estos quedan entonces "transparentes"para el sistema de DDF, teniendo un sistema de control propio que se encarga de generar los voltajes necesarios en los motores para obtener las velocidades angulares requeridas.

El vector de estado del sistema que se utiliza en esta capa considera sólo las variables afectadas por las fallas fatales. Estas son: la velocidad angular del robot y la velocidad angular de cada una de las ruedas. Así, el vector de estado simplificado queda determinado por:

$$X_k = \begin{bmatrix} \dot{\varphi}_k & \omega_{1k} & \omega_{2k} \end{bmatrix}^T \qquad (3.10)$$

En base a este vector de estado y considerando que las entradas son las velocidades angulares determinadas por el sistema de control, el modelo del sistema sin fallas es el siguiente:

$$\mathbb{M}_0: \quad \begin{aligned} X_{k+1,0} &= \underbrace{\begin{bmatrix} 0 & \frac{r_1}{2b} & \frac{-r_2}{2b} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{A_0} X_{k,0} + \underbrace{\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{B_0} U_k \\ Z_{k+1,0} &= \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{C_0} X_{k+1,0} \end{aligned} \qquad (3.11)$$

A continuación se muestran las variaciones del modelo de estado normal que incorporan cada una de las fallas que debe diagnosticar la capa 1.

Falla 1: La Rueda Derecha Resbala

El efecto más notorio de este tipo de fallas se ve a nivel de las ecuaciones cinemáticas, pues que la rueda resbale es equivalente a una reducción del radio y afecta directamente en la velocidad lineal de dicha rueda. Por ende, esta falla causa una variación en la velocidad angular de giro del cuerpo del robot. Debido a que se requiere una holgura en la medición del radio de la rueda, para evitar que errores de medición de los parámetros del robot generen fallas inexistentes, se debe escoger un valor para el radio equivalente que permita dicha holgura. Una reducción de un 50% en el radio de la rueda asegura tener al menos un 10% de error en la medición de los parámetros del robot sin activar la alarma de detección. Reducciones equivalentes a más de un 90% son identificables por este único modelo. El resultando es el siguiente modelo:

$$\mathbb{M}_1: \begin{cases} X_{k+1,1} = \underbrace{\begin{bmatrix} 0 & \mathbf{0.5}\frac{r_1}{2b} & \frac{-r_2}{2b} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{A_1} X_{k,1} + \underbrace{\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{B_1} U_k \\ Z_{k+1,1} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{C_1} X_{k+1,1} \end{cases} \qquad (3.12)$$

Falla 2: La Rueda Izquierda Resbala

En forma equivalente al modelo de la falla 1, el modelo para la falla 2 agrega la reducción equivalente en la otra rueda:

$$\mathbb{M}_2: \begin{cases} X_{k+1,2} = \underbrace{\begin{bmatrix} 0 & \frac{r_1}{2b} & \mathbf{0.5}\frac{-r_2}{2b} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{A_2} X_{k,2} + \underbrace{\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{B_2} U_k \\ Z_{k+1,2} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{C_2} X_{k+1,2} \end{cases} \qquad (3.13)$$

Falla 3: Pérdida del Momento de Torsión en la Rueda Derecha

Como se ha explicado anteriormente, la pérdida de torque en una rueda puede deberse a varios motivos. Puede haber una ruptura en la conexión mecánica entre la rueda y el motor, el motor puede estar descompuesto, o la rueda se ha atascado. En los tres casos el resultado es que la velocidad angular de la rueda afectada sea cero, independiente del valor requerido por el sistema de control. El modelo entonces queda de la siguiente forma:

$$\mathbb{M}_3: \begin{cases} X_{k+1,3} = \underbrace{\begin{bmatrix} 0 & \frac{r_1}{2b} & \frac{-r_2}{2b} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{A_3} X_{k,3} + \underbrace{\begin{bmatrix} 0 & 0 \\ \mathbf{0} & 0 \\ 0 & 1 \end{bmatrix}}_{B_3} U_k \\ Z_{k+1,3} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{C_3} X_{k+1,3} \end{cases} \qquad (3.14)$$

Falla 4: Pérdida del Momento de Torsión en la Rueda Izquierda

El modelo para la falla 4 es equivalente al anterior, modificando la rueda afectada.

$$\mathbb{M}_4: \begin{cases} X_{k+1,4} = \underbrace{\begin{bmatrix} 0 & \frac{r_1}{2b} & \frac{-r_2}{2b} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{A_4} X_{k,4} + \underbrace{\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & \mathbf{0} \end{bmatrix}}_{B_4} U_k \\ Z_{k+1,4} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{C_4} X_{k+1,4} \end{cases} \qquad (3.15)$$

Falla 5: Falla del Sensor de Velocidad de la Rueda Derecha

Los sensores de velocidad se basan en los llamados “encoders”, que son sensores que entregan un tren de pulsos a una frecuencia que es proporcional a la velocidad con la que gira la rueda. Es por ello, que parte del sensor debe estar acoplado mecánicamente a la rueda o al motor para identificar la velocidad de giro. Una falla habitual es que el sensor se desacople o que el sistema óptico utilizado quede mal alineado y no pueda detectar si la rueda gira. En ambos

casos el resultado es que el sensor no entrega pulsos y por ende se asume que el motor se encuentra detenido. Esta falla sólo afecta a la matriz asociada a la medición.

$$\mathbb{M}_5: \quad \begin{aligned} X_{k+1,5} &= \underbrace{\begin{bmatrix} 0 & \frac{r_1}{2b} & \frac{-r_2}{2b} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{A_5} X_{k,5} + \underbrace{\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{B_5} U_k \\ Z_{k+1,5} &= \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & \mathbf{0} & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{C_5} X_{k+1,5} \end{aligned} \qquad (3.16)$$

Falla 6: Falla del Sensor de Velocidad de la Rueda Izquierda

Esta falla posee un modelo equivalente al de la falla 5.

$$\mathbb{M}_6: \quad \begin{aligned} X_{k+1,6} &= \underbrace{\begin{bmatrix} 0 & \frac{r_1}{2b} & \frac{-r_2}{2b} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{A_6} X_{k,6} + \underbrace{\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{B_6} U_k \\ Z_{k+1,6} &= \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \mathbf{0} \end{bmatrix}}_{C_6} X_{k+1,6} \end{aligned} \qquad (3.17)$$

Falla 7: Falla en el Giróscopo

Una falla en el giróscopo implica que el robot no es capaz de detectar la velocidad de giro de su cuerpo. Al igual que con los sensores de velocidad para las ruedas, se asume que la falla produce una salida nula en el giróscopo.

$$\mathbb{M}_7: \begin{cases} X_{k+1,7} = \underbrace{\begin{bmatrix} 0 & \frac{r_1}{2b} & \frac{-r_2}{2b} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{A_7} X_{k,7} + \underbrace{\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{B_7} U_k \\ Z_{k+1,7} = \underbrace{\begin{bmatrix} \mathbf{0} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{C_7} X_{k+1,7} \end{cases} \qquad (3.18)$$

Falla 8: Pérdida del Momento de Torsión en Ambas Ruedas

La última falla agregada al sistema es la de pérdida de torque en ambas ruedas. Este es equivalente a unir los modelos de las fallas 3 y 4:

$$\mathbb{M}_8: \begin{cases} X_{k+1,8} = \underbrace{\begin{bmatrix} 0 & \frac{r_1}{2b} & \frac{-r_2}{2b} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{A_8} X_{k,8} + \underbrace{\begin{bmatrix} 0 & 0 \\ \mathbf{0} & 0 \\ 0 & \mathbf{0} \end{bmatrix}}_{B_8} U_k \\ Z_{k+1,8} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{C_8} X_{k+1,8} \end{cases} \qquad (3.19)$$

Es interesante observar que cada falla afecta en forma diferente las ecuaciones del modelo normal, lo cual permite determinar qué falla es la que está ocurriendo según el comportamiento observado.

La capa 1 de la arquitectura de DDF se diseña en base a los 9 modelos presentados anteriormente. Así, la capa se preocupa de actualizar cada uno de los filtros implementados para cada instante de muestreo, y junto con ello calcula las probabilidades para cada hipótesis usando la ecuación 3.6.

El proceso de detección se realiza en base a la observación de la probabilidad de la hipótesis de operación normal $\mathbb{P}_k(\mathbb{H}_0)$. Si esta decrece a un valor menor que $\mathbb{P}_{DT}$=0.01, se considera que el sistema ha salido de operación normal. Para mejorar la robustez del sistema de detección y evitar las falsas alarmas producidas por el ruido en la medición, la alarma de detección de falla se activa cuando la probabilidad de la hipótesis $\mathbb{H}_0$ es menor al límite establecido durante 3 intervalos consecutivos de muestreo.

Tras detectar una falla, el diagnóstico se hace mediante la observación de las otras probabilidades, escogiendo la primera hipótesis que tiene una probabilidad superior a $\mathbb{P}_{DG}$=0.99. Si bien se podría escoger la hipótesis de falla con mayor probabilidad al momento de detección de la operación en estado de falla, esto demostró generar muchas equivocaciones en el diagnóstico.

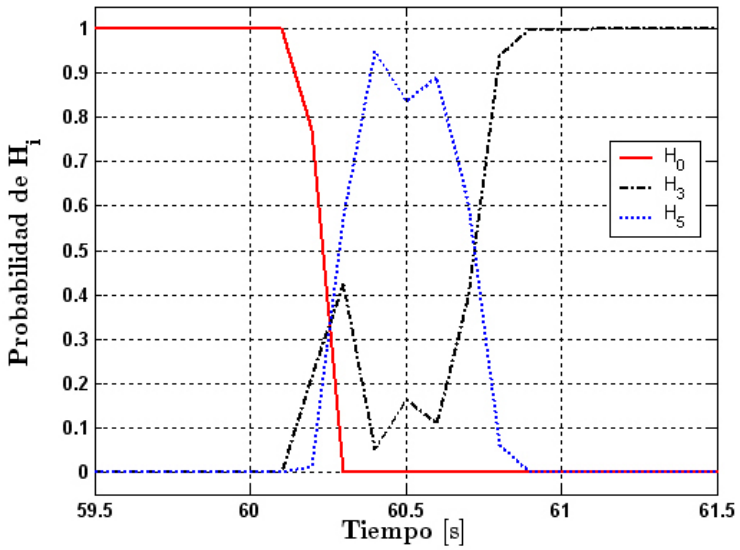

Figura 3.6: Problemas en el Diagnóstico de algunas Fallas.

Como los efectos de la pérdida del momento de torsión y de la falla en el sensor de velocidad son inicialmente muy similares, ambas hipótesis presentan una probabilidad alta al momento de detectarse la falla, tal como se observa en

la figura 3.6. Dependiendo del efecto del ruido en el instante de detección, las simulaciones muestran que es posible que la probabilidad de uno de estos dos tipos de fallas sea mayor, sin ser necesariamente la falla real que está ocurriendo. Como se ilustra en la figura 3.6, al momento de detección de la falla, en t=60.3 [s], la probabilidad más alta la presenta la falla 5, cuando en realidad es la falla 3 la que ha ocurrido. Si se espera, en cambio, a que la probabilidad de una hipótesis sea superior a 0.99, se incrementa el tiempo de diagnóstico pero se eliminan casi por completo los errores en el diagnóstico.

3.3.3. Simulación de la Capa 1

Las figuras siguientes ilustran los resultados de algunas simulaciones obtenidas para cada una de las distintas fallas diagnosticables, utilizando los parámetros definidos en la tabla 3.1 para cada simulación.

La desviación estándar del ruido utilizado en el proceso y para la medición es del 10% del máximo valor que puede alcanzar el actuador o que es capaz de medir el sensor respectivamente.

Frecuencia de Muestreo (f_M)	10 [Hz]
Tiempo en que Ocurre la Falla (t_F)	30 [s]

Tabla 3.1: ***Parámetros de la Simulación de Fallas.***

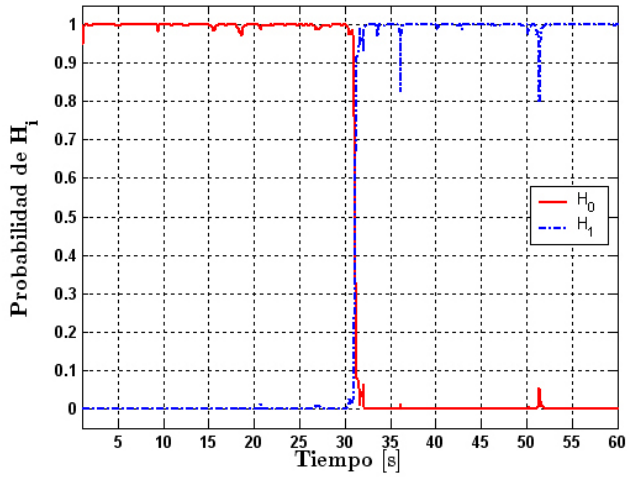

Figura 3.7: Falla 1: La Rueda Derecha Resbala.

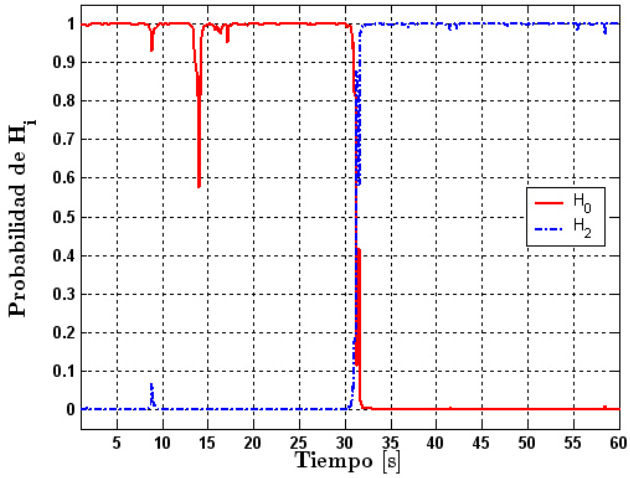

Figura 3.8: Falla 2: La Rueda Izquierda Resbala.

Como se observa en las figuras 3.7 y 3.8, existe una disminución rápida en la probabilidad del estado normal al momento de ocurrida la falla.

Es importante notar que el sistema se comporta en forma robusta ante variaciones en los parámetros del robot. Una variación del 10% en el radio equivalente utilizado para los modelos en comparación al radio real de la rueda,

que se puede atribuir a la incertidumbre en la medición de los parámetros del robot, no gatilla una falla. Por otro lado, variaciones mayores activan de inmediato la falla asociada a que una rueda resbale. De igual forma, variaciones en la masa de las ruedas o del cuerpo del robot tampoco origina una detección de fallas inexistentes.

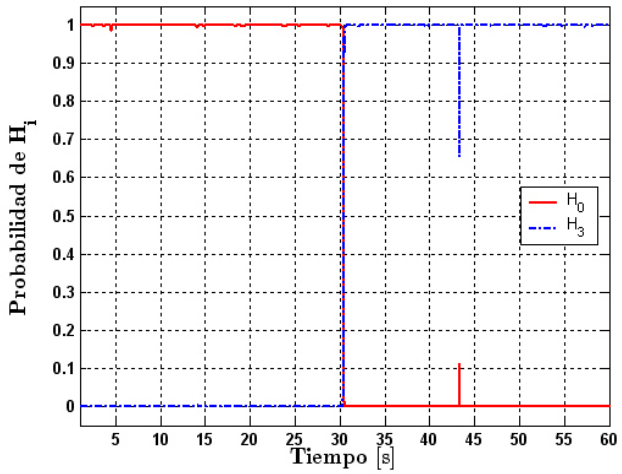

Figura 3.9: Falla 3: Pérdida de Momento de Torsión en la Rueda Derecha.

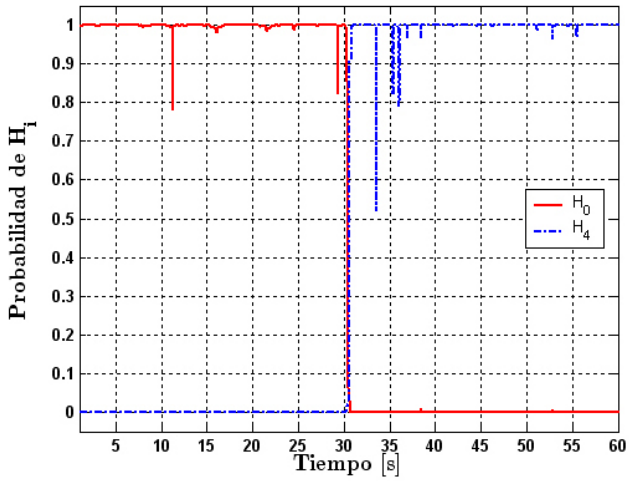

Figura 3.10: Falla 4: Pérdida de Momento de Torsión en la Rueda Izquierda.

En las figuras 3.9 y 3.10 se aprecia que el tiempo de respuesta ante la pérdida de momento de torsión en una de las ruedas es más corto que la detección de si una rueda resbala.

Si bien no se alcanza a identificar en las figuras, existe un aumento momentáneo de la probabilidad de las fallas relacionadas al sensor de velocidad de la rueda correspondiente, pero que disminuye rápidamente.

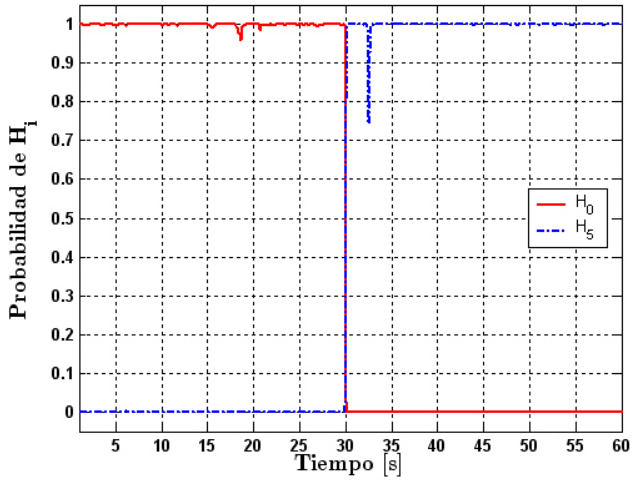

Figura 3.11: Falla 5: Falla del Sensor de Velocidad de la Rueda Derecha.

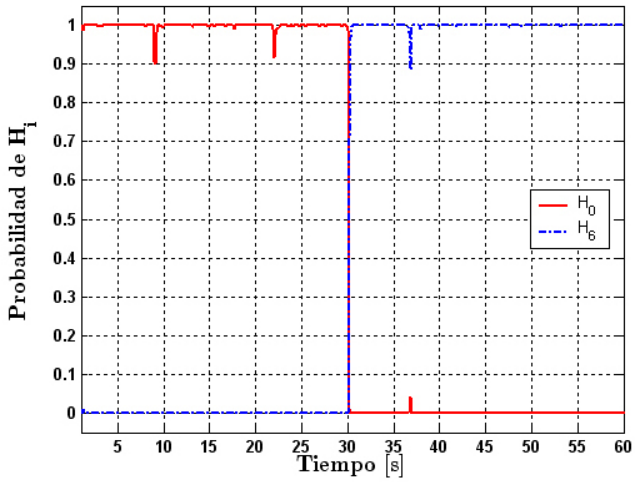

Figura 3.12: Falla 6: Falla del Sensor de Velocidad de la Rueda Izquierda.

Las figuras 3.11 y 3.12 muestran lo rápido que identifica el sistema la existencia de una falla en el sensor de velocidad de la rueda. Esto se debe a que las ruedas se encuentran girando en todo momento y por ende se hace fácilmente identificable este tipo de fallas.

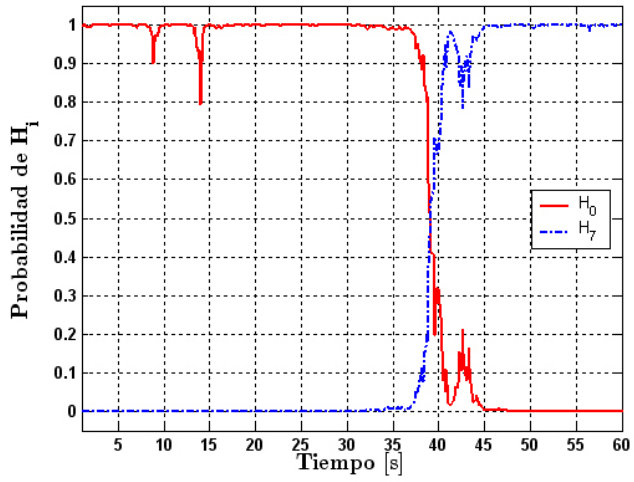

Figura 3.13: Falla 7: Falla en el Giróscopo.

Como se observa en la figura 3.13, las fallas asociadas al giróscopo presentan un tiempo de retardo mucho mayor. Cabe mencionar que el mayor

tiempo de retardo se debe también a que el giróscopo sólo entra en funcionamiento cuando el robot rota. Esto implica que, mientras el robot se mueva en una línea recta, no se detectará falla alguna en el giróscopo pues este entrega un valor nulo en ese momento, con o sin falla presente, a diferencia de los sensores de velocidad de las ruedas, que están activos durante todo el tiempo de funcionamiento del robot.

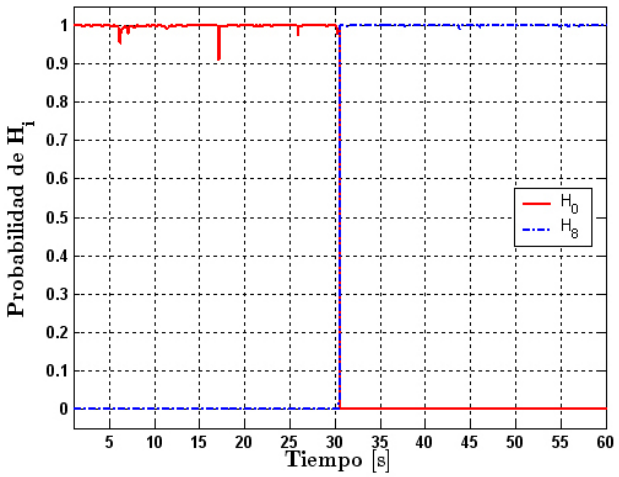

Figura 3.14: Falla 8: Pérdida de Momento de Torsión en Ambas Ruedas.

Finalmente, como se ilustra en la figura 3.14, al igual que con la detección de pérdida del momento de torsión en una rueda, la identificación de la pérdida en ambas ruedas se realiza en forma rápida.

3.3.4. Análisis de Desempeño de la Capa 1

Con el fin de analizar el desempeño de esta capa, se simuló el robot 1000 veces, eligiendo al azar en cada oportunidad el modo de operación del robot y en el caso de los modos de operación con falla, el tiempo en que ocurre la falla. Esto permite analizar estadísticamente el comportamiento de la capa de DDF, observando los resultados del diagnóstico para cada estado simulado. Para

analizar el desempeño de la capa 1 se utilizan cuatro criterios: número de falsas alarmas, la matriz de confusión, el tiempo de detección y el tiempo de diagnóstico.

Tras las 1000 simulaciones se observó que, gracias al criterio de detección utilizado, no se producen falsas alarmas aún cuando se modifique el nivel de ruido en el sistema hasta en un 10%. Si el número de muestras utilizadas en el criterio de detección se reduce a una, se observa en las simulaciones que el número de falsas alarmas aumenta a 7%.

En la tabla 3.2 se presenta la matriz de confusión arrojada por las simulaciones.

$$Conf_{capa1} = \begin{bmatrix} 100 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 98 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 98.3 & 0 & 1.7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 100 \end{bmatrix}$$

Tabla 3.2: Matriz de Confusión de la Capa 1.

La matriz de confusión es una forma simple de analizar el desempeño de un sistema de diagnóstico. La matriz está compuesta por elementos c_{ij}, que indican el porcentaje de casos en lo que un modo de operación $\mathbb{M}_i$, se diagnostica como modo de operación $\mathbb{M}_j$. Un sistema de diagnóstico perfecto presenta una matriz diagonal, ya que cada falla se diagnostica correctamente. Como se puede calcular a partir de la matriz de confusión, el sistema de DDF diagnostica exitosamente en un 99.6% de los casos simulados.

En las 1000 simulaciones sólo hay 4 errores de diagnóstico, los que se deben a la confusión que se produce inicialmente entre los modos de operación de falla en el encoder y de pérdida del momento de torsión en una rueda, tal como se indicó anteriormente. Debido a esto, las fallas de pérdida de torque en una rueda son identificadas como fallas en el sensor de velocidad de la misma. Es posible eliminar estos errores de diagnóstico por medio de un incremento en el número de muestras necesarias para activar la detección de fallas. Un incremento de 3 a 8 muestras elimina por completo estos errores de diagnóstico, pero aumenta el tiempo de detección en 0.5 [s] para todos los tipos de fallas.

El porcentaje de errores en el diagnóstico depende de la cantidad de muestras utilizadas en el criterio de detección. Al incrementar el número de intervalos de muestreo, aumenta el tiempo de espera antes del diagnóstico, permitiendo que se estabilice la probabilidad de la hipótesis de falla correspondiente. La figura 3.15 ilustra la relación entre el porcentaje de error en el diagnóstico y la cantidad de muestras usadas en la simulación, para $\mathbb{P}_{DT}$=0.05 y $\mathbb{P}_{DG}$=0.9.

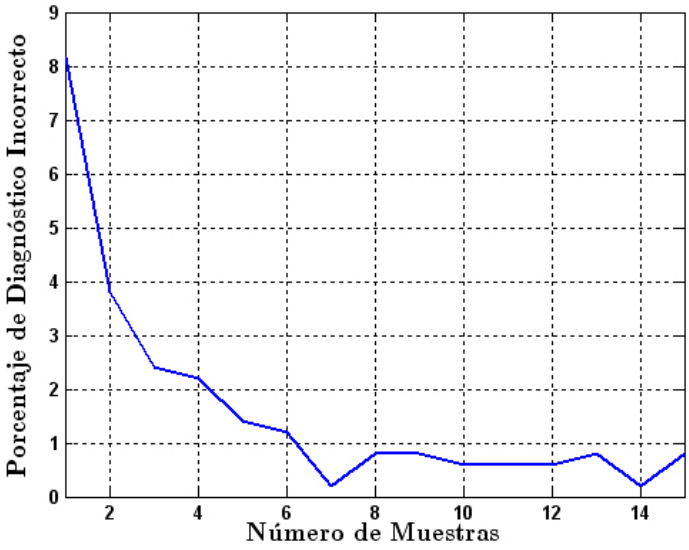

Figura 3.15: Porcentaje de Error en el Diagnóstico.

Con respecto a los tiempos de detección, la tabla 3.3 presenta los resultados para cada una de las fallas. Estos indican el retraso que existe entre el momento de ocurrida la falla, y el momento en que el sistema detecta que ya no se encuentra en estado normal, que es cuando $\mathbb{P}_{k}(\mathbb{H}_{0}) < \mathbb{P}_{DT}$.

Falla	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Tiempo [s]	2.7	2.6	0.7	0.6	0.2	0.2	20.5	0.6

Tabla 3.3: Tiempos de Detección de la Capa 1

Considerando que la constante de tiempo dinámica del robot es del orden de los 0.15 [s], se puede apreciar que los tiempos de detección son pequeños para las fallas de pérdida de torque y las fallas en los sensores de velocidad. Un retardo mayor se presenta en las fallas donde una rueda resbala y uno mucho mayor en las fallas en el giróscopo. Esto se debe a que ambas dependen principalmente de las mediciones del giróscopo para ser identificadas. Dado que el robot presenta una velocidad máxima de 1.5 [rad/s] para cada rueda, las velocidades medidas por el giróscopo son muy pequeñas en comparación a las de entregadas por los encoders, haciendo que discrepancias entre el valor estimado de cada modelo y la medición entregada por este sensor deban acumularse para poder afectar la probabilidad de la hipótesis correspondiente. Además, dado que en alguna de las simulaciones las fallas ocurren mientras el robot se encuentra moviéndose en línea recta, las fallas en el giróscopo presentan un retardo mucho mayor, ya que se requiere que el robot rote para poder identificarlas.

Debido a la forma en que se ha diseñado el sistema de diagnóstico, existe un retraso en la identificación de la falla con respecto al momento de la detección. La tabla 3.3 presenta los tiempos de diagnóstico, que ocurren tras la detección, cuando $\mathbb{P}_{k}(\mathbb{H}_{i}) > \mathbb{P}_{DG}$ para i=1..8.

Falla	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Tiempo [s]	2.7	2.6	0.8	0.8	0.2	0.2	20.8	0.6

Tabla 3.4: Tiempos de Diagnóstico de la Capa 1.

Se puede apreciar que el retardo no es significativo, ya que para la mayoría de las fallas el diagnóstico ocurre en el mismo momento en que ocurre la detección.

3.3.5. Extensión de la Capa 1: Interacción con el Sistema de Control

La interacción con la estructura de control del robot permite obtener una mayor información del estado del robot y las condiciones actuales de operación. Esta es una ventaja que no se ha explotado en ninguno de los trabajos previos, que permite aumentar el número de fallas diagnosticables.

Si bien la Capa 1 muestra muy buenos resultados en la detección y diagnóstico de las fallas, no es posible diferenciar entre las fallas de pérdida del momento de torsión causadas por un desacople mecánico entre el motor y la rueda, y aquellas fallas debido a que la rueda ha topado con algo. Si el coeficiente de roce entre la rueda y el suelo no es lo suficientemente grande, el que la rueda tope con un obstáculo puede activar las fallas relacionadas con que una rueda resbale en vez de las fallas de pérdida del momento de torsión.

Para distinguir esta nueva falla, se debe recurrir a una interacción con el sistema de control, aprovechando que esta capa del sistema de DDF puede identificar también el regreso al estado normal si es que la falla desaparece. Esto permite detectar si la rueda derecha topa con un obstáculo (falla 9), si lo hace la rueda izquierda (falla 10) o si ocurre en ambas ruedas en forma simultánea (falla 11).

La figura 3.16 muestra el diagrama de flujos que describe la forma de interacción entre la Capa 1 y el Control de Navegación.

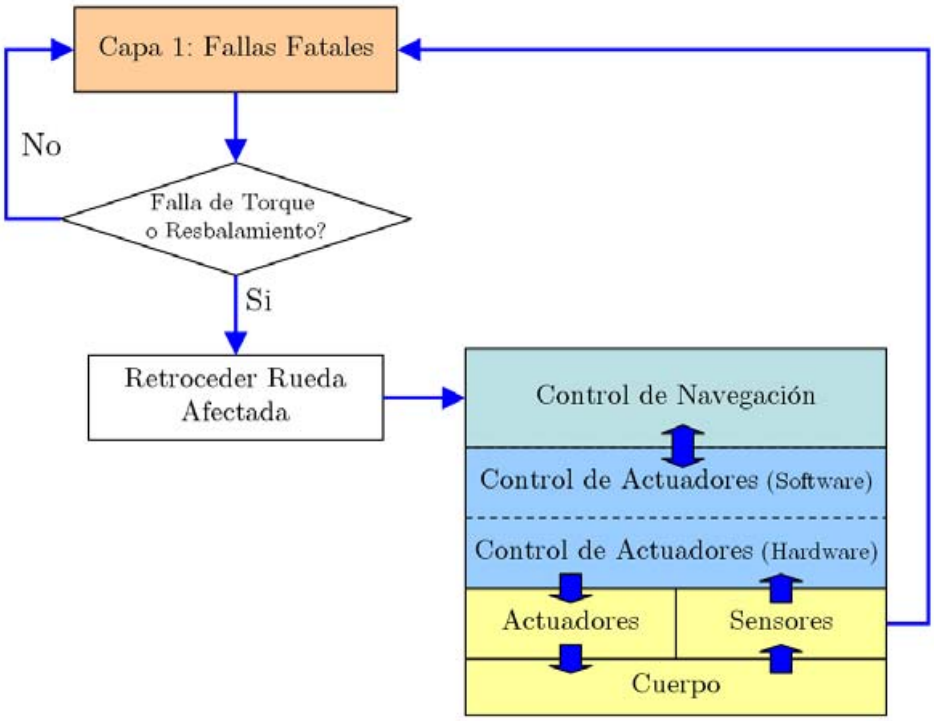

Figura 3.16: Interacción entre la Capa 1 y el Control de Navegación.

El objetivo del algoritmo es, que si se ha detectado una falla relacionada con la pérdida del momento de torsión de una o de las dos ruedas (fallas 3, 4 o 8) o una falla de resbale de una rueda (fallas 1 o 2), la Capa 1 de diagnóstico solicita al Control de Navegación un cambio en la dirección de movimiento de la o las ruedas afectadas. Si la falla se debe a que la rueda topa con algún obstáculo, el cambio de dirección implicaría un regreso al estado normal, indicando entonces que la falla ha desaparecido. En cambio, si la falla se debe a un desacoplamiento entre la rueda y el motor, el cambio de dirección no hace que el sistema vuelva al estado normal y la falla permanece. Esto puede utilizarse a su vez como un "sensor virtual de colisiones", que identifica cuando el robot topa con algún obstáculo sin necesidad de agregar sensores alrededor del cuerpo del robot.

La figura 3.17 muestra como el sistema identifica que el robot vuelve a trabajar en su estado normal al cambiar de dirección cuando la falla simulada corresponde a la colisión de la rueda derecha con un obstáculo.

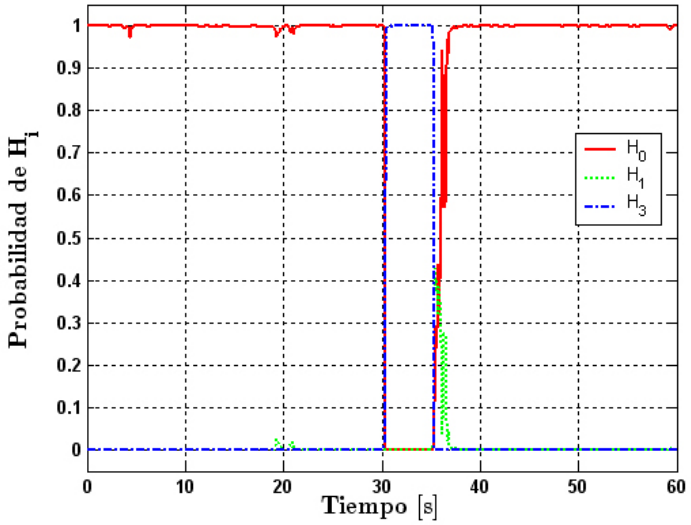

Figura 3.17: Falla 9: La Rueda Derecha Topa con un Obstáculo.

En este caso, asumiendo que el coeficiente de roce entre la rueda y el suelo es muy alto, la Capa 1 identifica el evento como una falla de pérdida del momento de torsión en la rueda derecha (falla 3) y procede a cambiar la dirección de rotación de esa rueda. Se ha generado un retardo de 5 [s] en la reacción del robot para hacer observable el cambio de un estado a otro, aunque el sistema es capaz de generar la orden al Control de Navegación apenas identifica la falla, que ocurre tras 0.8 [s] en promedio. Una vez revertida la dirección de movimiento de la rueda, la probabilidad de $\mathbb{H}_0$ vuelve a aumentar, indicando que el robot regresa a su estado de operación normal.

Como se ilustra en la figura 3.18, el sistema funciona también si ambas ruedas se han atascado. Esto puede ocurrir por ejemplo cuando el cuerpo del robot colisiona con un obstáculo.

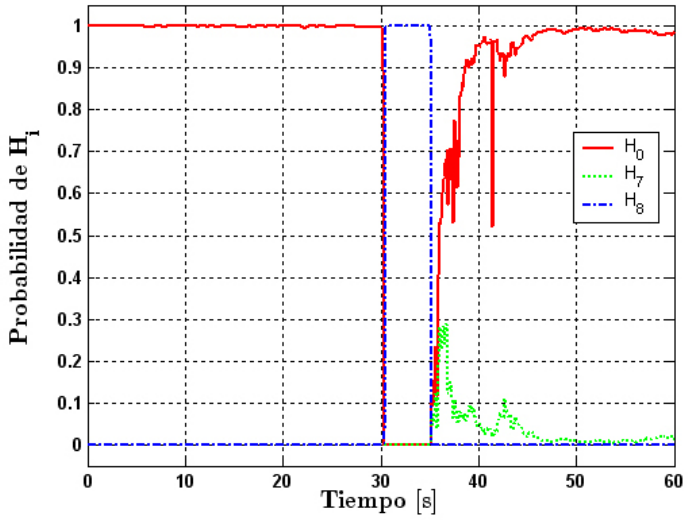

Figura 3.18: Falla 11: Ambas Ruedas Topan con un Obstáculo.

3.3.6. Adaptación para Diagnosticar Múltiples Fallas

Cuando el Control de Navegación y/o el Control de Objetivos están diseñados para acomodar las fallas menos graves, el sistema debe adaptarse para continuar con su monitoreo del sistema, sin ser interferido por la falla actual. Así por ejemplo, si se detecta una falla en el giróscopo, pero el robot cuenta con otros sistemas de localización, éste puede continuar con su funcionamiento correcto, pero debe ser capaz de detectar alguna de las otras fallas que si pueden resultar invalidantes.

En las ecuaciones 3.12 a 3.19, que corresponden a cada uno de los estados de falla, es fácilmente identificable el elemento que cambia con respecto al modelo normal (ecuación 3.11). Como en cada modelo de falla, excepto en la pérdida del momento de torsión en ambas ruedas, la modificación es única, se pueden utilizar estos elementos como factores para introducir la falla asociada a cualquiera de los otros modelos. Así, una vez que se identifica una falla, la modificación que esta incorpora sobre el modelo de estado normal se incluye en

todos los modelos de falla permitiendo, que si otra falla ocurre, el sistema sea capaz de detectarla. Es importante aclarar que el modelo de estado normal no se modifica, ya que este permite identificar si se han corregido las fallas.

Por ejemplo, si el robot sufre una falla en el sensor de velocidad de la rueda izquierda, el modelo modificado para la falla de pérdida del momento de torsión en la rueda derecha es:

$$\mathbb{M}_{3,\mathbb{M}_6}: \quad \begin{aligned} X_{k+1,3} &= \underbrace{\begin{bmatrix} 0 & \frac{r_1}{2b} & \frac{-r_2}{2b} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{A_3} X_{k,3} + \underbrace{\begin{bmatrix} 0 & 0 \\ \mathbf{0} & 0 \\ 0 & 1 \end{bmatrix}}_{B_3} U_k \\ Z_{k+1,3} &= \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \underline{\mathbf{0}} \end{bmatrix}}_{C_3} X_{k+1,3} \end{aligned} \qquad (3.20)$$

En la ecuación 3.20 se ha subrayado el elemento adicional con respecto al modelo $\mathbb{M}_3$ normal, que permite identificar una falla de pérdida del momento de torsión en la rueda derecha, dado que ya ha ocurrido una falla en el sensor de velocidad.

Es interesante notar que el sistema es capaz de adaptarse a las fallas existentes con el fin de detectar nuevas fallas, pero sin necesidad de mantener todos los otros modelos que contengan ambas fallas. Sólo se debe modificar el elemento asociado a la falla que ha ocurrido, lo cual no requiere de un aumento en los requerimientos computacionales del sistema.

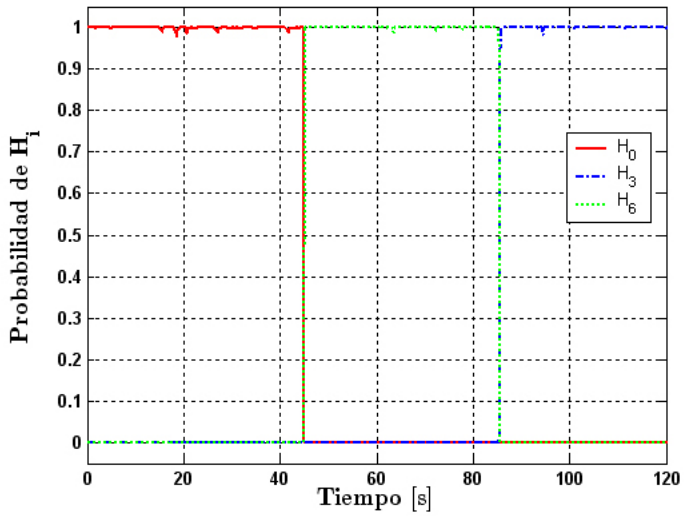

Figura 3.19: Detección y Diagnóstico de Fallas Consecutivas.

La figura 3.19 muestra los resultados de la simulación utilizando la Capa 1 adaptiva. En t=45 [s] se simula la aparición de una falla en el sensor de velocidad izquierdo (falla 6), tras la cual el sistema cambia las matrices de cada modelo de falla permitiendo detectar la falla de pérdida del momento de torsión en la rueda derecha (falla 3), simulada en t=85 [s]. Si bien tras ocurrida la segunda falla se observa que sólo se diagnostica la falla 3, dado que el sistema no volvió a su estado normal se deduce que la falla 6 no ha sido corregida, permitiendo al sistema identificar fallas consecutivas.

3.3.7. Limitaciones del Método Utilizado

Aunque la Capa 1 Extendida con comportamiento adaptivo permite correctamente diagnosticar múltiples fallas fatales para un robot móvil, el método utilizado posee limitaciones importantes que no permiten utilizar su estructura para agregar cualquier tipo de falla.

La primera gran limitación que presenta este método es que requiere que la falla se pueda incorporar en la estructura de un filtro de Kalman. Esto deja de lado todas aquellas fallas que no poseen modelos, como ocurre con la mayoría de las fallas de los sensores de percepción del entorno, como cámaras y sonares. Por igual motivo, fallas aditivas y fallas tipo valor permanente de salida tampoco son identificables utilizando este método. Si bien se pueden crear modelos nolineales que contengan estos tipos de fallas, su implementación en la estructura de un filtro de Kalman es muy limitada y no permite obtener la información necesaria de la matriz de covarianzas del residuo, haciendo inútil el uso del método. En general, el análisis de estos tipos de fallas requiere de sistemas más complejos de inferencia, que utilizan métodos heurísticos o análisis en base a redes neuronales y lógica difusa para poder identificar el patrón de falla en los residuos obtenidos. Esto requiere de un poder computacional mayor y por ende limita su aplicabilidad a sistemas simples.

La segunda limitación importante que presenta el método se refiere a la detección e identificación cuando existen variables absolutas en el vector de estado que describe el sistema. Las variables de la postura en un robot móvil $[x\ y\ \varphi]$ son un ejemplo de ellas. Debido a que los diferentes modelos que utiliza el método de diagnóstico se actualizan desde mucho antes que ocurra la falla, se produce una divergencia entre los valores estimados por cada modelo y los valores reales del vector de mediciones. Esto implica que, tras ocurrir la falla, el residuo asociado al modelo que contiene la falla tendrá un valor suficientemente grande como para disminuir la probabilidad de la hipótesis asociada a cero. En el caso de las variables absolutas, es el tiempo en el que ocurre la falla el elemento fundamental para determinar el efecto que ésta tendrá sobre el robot. Por ende se requiere que los modelos utilizados para identificar comiencen a ser actualizados en el momento mismo de ocurrencia de la falla, siendo inicializados con los valores que tiene el vector de estado en el momento justo antes de la falla. Como siempre existe un tiempo de retardo entre la ocurrencia de la falla y

su detección, es imposible activar los diferentes modelos sin que exista un desfase de tiempo y en consecuencia, un residuo significativo para cada uno de los modelos utilizados.

Por ejemplo, en el caso que se utilice la posición del robot en el vector de estado, el modelo que contiene la falla de pérdida del momento de torsión en ambas ruedas, estimará que la posición del robot se mantiene constante, aunque mientras no se produzca realmente la falla, el robot se moverá. Al ocurrir la falla el robot quedará detenido, pero como el lugar donde ocurrió la falla es diferente del estimado por el modelo con falla, el residuo resultante será grande, indicando erróneamente que ese modelo no es el que identifica mejor el estado de operación actual.

Debido a estas falencias, el método no permite detectar fallas que se produzcan en una gran cantidad de sensores que pueden ser adicionados a un robot móvil, y que en definitiva se requieren para que los objetivos del mismo sean cumplidos. Por ello se debe recurrir a otras herramientas que permitan utilizar la redundancia existente para detectar y diagnosticar estas fallas, sin necesariamente ocupar una gran cantidad de recursos computacionales para lograrlo.

3.4. Capa 2: Capa de Detección para Robots Cooperativos

La principal dificultad en detectar las fallas secundarias utilizando modelos, es que los modelos requeridos deben ser mucho más complejos, utilizando más recursos computacionales. Métodos como el filtro de Kalman extendido permiten estimar el vector de medición del robot, pero el error en la estimación no permite una buena identificación. Otros métodos como filtro de Kalman difuso propuesto en [*Carrasco, 2004*], entregan una mejor estimación,

pero implican un alto costo computacional, que limita su implementación en robots simples, además que no se aprovecha que exista un grupo de robots.

Todo grupo de robots móviles cuenta con redundancia en sus sensores y actuadores, la cual puede aprovecharse para mejorar las mediciones o para detectar fallas al interior del grupo [*Tinós, 2001*]. La ventaja principal de contar con un sistema distribuido, es que la tarea de detección y diagnóstico puede dividirse entre todos los robots del grupo. Esto permite reducir los requerimientos computacionales del sistema de DDF, logrando una detección y diagnóstico de la falla sin necesidad de complejos sistemas de identificación. La principal desventaja del método es el aumento en el tiempo de respuesta. Estas características permiten que el método sea utilizado para el monitoreo de las fallas secundarias, siendo la base del diseño de la capa 2.

Es importante destacar que esta capa sólo puede detectar y diagnosticar fallas sobre elementos que posean redundancia al interior del grupo cooperativo de robots. Si sólo un robot posee un cierto sensor o actuador, no será posible aplicar este método para detectar una falla en el sensor o actuador.

La detección y el diagnóstico de fallas secundarias en sensores, se realiza a través de múltiples mediciones entre los robots, mientras que las fallas en el sistema de comunicación se detectan utilizando un algoritmo diferente.

3.4.1. Detección y Diagnóstico en Sensores Redundantes

Para este análisis se considera que todos los robots están equipados con los mismos sensores, permitiendo que se realicen mediciones independientes de una misma variable con cada uno de los sensores a bordo. Como los sensores poseen un error en la medición, no basta con comparar dos mediciones realizadas por sensores diferentes para determinar si existe una discrepancia. Se requiere

generar un método adaptivo que permita determinar las diferencias máximas que pueden existir entre las mediciones de dos sensores, antes de determinar que una falla ha ocurrido.

El método diseñado se basa en el uso de múltiples mediciones por parte de cada sensor, para reducir el efecto del ruido en la medición, y con ello determinar en forma óptima los límites que definen cuando una falla ha ocurrido.

La medición realizada por un sensor cualquiera, S_i, se considera como una medición ideal de la variable, M, con un ruido Gausseano aditivo, μ, de media cero y varianza conocida:

$$S_i = M + \mu \qquad (3.21)$$

$$\mu \sim N\left(0, \sigma_i^2\right) \qquad (3.22)$$

La diferencia entre las mediciones realizadas por dos sensores independientes, debido al ruido presente, tiene también una distribución normal dada por:

$$D_{ij} = S_i - S_j \sim N\left(0, \sigma_d^2\right) \qquad (3.23)$$

$$\sigma_d^2 = \sigma_i^2 + \sigma_j^2 \qquad (3.24)$$

Considerando la ecuación 3.23 para la diferencia entre las mediciones de dos sensores, es posible establecer un límite óptimo que determine que una falla se ha presentado en uno de los sensores, dado que se desea realizar una detección correcta con probabilidad $\mathbb{P}_n$, o equivalentemente, que las falsas alarmas ocurran con probabilidad 1-$\mathbb{P}_n$.

Muchas veces los límites deseados para la detección son más pequeños que lo que permite el nivel de ruido en la medición. Por esta razón se debe recurrir al uso de múltiples mediciones que reduzcan el efecto del ruido y así permitir la utilización de límites más pequeños.

Si cada robot toma n mediciones de la misma variable, la varianza de la distribución del promedio de las diferencias distribuye de la siguiente forma:

$$Dn_{ij} = \frac{1}{n}\sum_{k=1}^{n}\left(D_{ij}\right)_k = \frac{1}{n}\sum_{k=1}^{n}\left(S_i\right)_k - \frac{1}{n}\sum_{k=1}^{n}\left(S_j\right)_k \sim N\left(0,\sigma_n^2\right) \qquad (3.25)$$

$$\sigma_n^2 = \frac{1}{n}\sigma_d^2 \qquad (3.26)$$

Esto permite reducir la varianza del ruido a medida que se consideran más mediciones. Al aumentar el número de mediciones, se puede asegurar con una mayor probabilidad, que si la diferencia promedio Dn_{ij} supera un límite D_{TH}, entonces ha ocurrido una falla en uno de los sensores, en comparación al caso en que se utiliza una única medición D_{ij}.

Utilizando la ecuación 3.26, es posible determinar el número de mediciones que se deben realizar para que, dado el límite de detección D_{TH}, se determine correctamente que ha ocurrido una falla con una probabilidad $\mathbb{P}_n$:

$$n = \frac{\sigma_d^2}{\sigma_n^2} \qquad (3.27)$$

$$\sigma_n^2 = \frac{D_{TH}}{D_P} \qquad (3.28)$$

En la ecuación 3.28, el valor del límite D_P se calcula en base a la distribución normal estándar y a la probabilidad buscada $\mathbb{P}_n$ tal que:

$$erf\left(\frac{D_P}{\sqrt{2}}\right) = \sqrt{\frac{2}{\pi}}\int_{x=0}^{D_P} e^{-\frac{x^2}{2}}dx = \mathbb{P}_n \qquad (3.29)$$

Como se ilustra en el ejemplo de la figura 3.20, al aumentar el número de mediciones a n=20, dado que σ_d=2, se puede asegurar con una probabilidad de un 99.9% que si el promedio de la diferencia supera el límite D_{TH}=1.5, ha ocurrido una falla. En cambio, si se utiliza una única medición, sólo se puede asegurar con una probabilidad de un 54.7%.

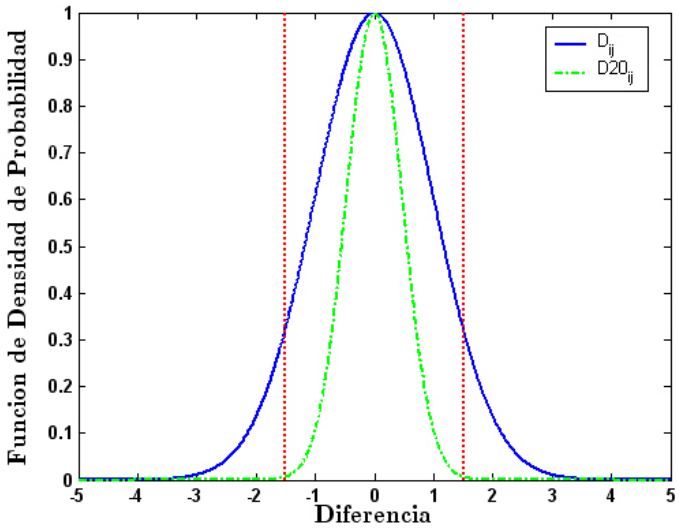

Figura 3.20: Disminución del Error por Aumento de Muestras.

Si bien con este método se puede obtener el número óptimo de muestras para asegurar con cierta probabilidad que ha ocurrido una falla, dependiendo del nivel de ruido en los sensores, es posible que el número de mediciones necesarias sea muy elevado para el tiempo o memoria disponibles. Utilizando las mismas ecuaciones antes descritas, es posible determinar el valor óptimo del límite D_{TH}, que permita detectar la falla dada una probabilidad $\mathbb{P}_n$ y restringiendo el número máximo de mediciones posibles a n_{max}.

Cuando el número de mediciones necesarias supera el límite n_{max}, el número de mediciones a efectuar se acota en este valor, y se procede a calcular el valor óptimo del límite D_{TH} utilizando la siguiente ecuación:

$$D_{TH} = \sigma_d \frac{D_P}{\sqrt{n_{max}}} \qquad (3.30)$$

Al igual que en el caso anterior, el valor de D_P se calcula utilizando la ecuación 3.29.

En el ejemplo ilustrado en la figura 3.21, utilizando las mismas condiciones del ejemplo anterior, se coloca un límite máximo de mediciones de n_{max}=5. Esto reduce la incertidumbre en la medición, pero el límite D_{TH} debe aumentarse a D_{TH}=2.94 para asegurar con una probabilidad del 99.9% que es correcta la detección de la falla.

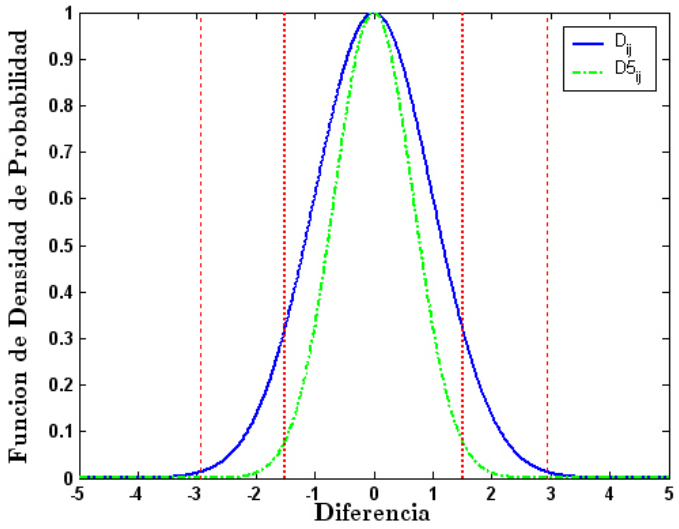

Figura 3.21: Disminución del Error por Aumento de Muestras y Límites.

Una ventaja adicional de este método es que permite detectar fallas en sensores sin el uso de modelos, los cuales requieren un alto costo computacional. Además, siendo conocidas las varianzas de cada sensor utilizado, el número de mediciones necesarias se debe calcular una sola vez. Esto también reduce en forma importante el costo computacional de la implementación.

3.4.2. Criterios de Detección y Diagnóstico

El uso de esta capa de la arquitectura requiere de la presencia de al menos dos robots que cuenten con los mismos sensores. Por ello, cada vez que dos robots se encuentren cerca, se debe aprovechar el momento para que ambos hagan una revisión de sus sensores y determinar si existen fallas presentes. La capa puede ser también activada cada cierto tiempo, tras el cual el robot que realiza la revisión de sus sistemas sale en busca de alguno de los otros robots del grupo para hacerla. La búsqueda de los otros miembros se realiza utilizando la cámara con que cuenta cada robot.

Una vez que se encuentran ambos robots, se realiza una revisión de los sensores redundantes, utilizando los parámetros calculados, detectando si existe alguna falla.

Aunque el método permite una determinación óptima de los límites y número de mediciones necesarias para detectar una falla, contar con sólo dos mediciones independientes no permite realizar un diagnóstico correcto que determine cual de los dos robots posee el sensor que ha fallado [*Tinós, 2002*]. Es por ello que cuando dos robots detectan una falla en sus sensores, al no contar con una tercera medición, activan una señal de alerta en cada uno de sus sistemas, indicando que el sensor puede presentar una falla. Tras esto, ambos robots buscan otro miembro del grupo para realizar la revisión y así diagnosticar de cuál robot es el sensor que ha fallado.

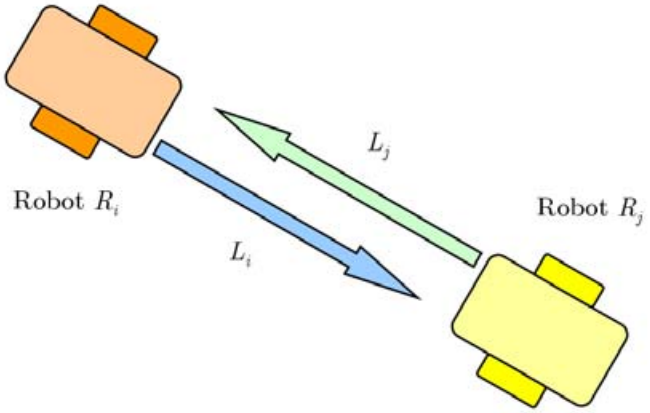

Figura 3.22: Detección de Fallas en el Sonar.

En el caso de los sonares, la revisión se realiza en base a la medición de distancia de un robot a otro, como se ilustra en la figura 3.22. Ambas mediciones de distancia, L_i y L_j, deben ser iguales, pero debido al ruido en la medición, la diferencia entre ambas tiene distribución de probabilidad normal, con media cero y varianza igual al doble de la varianza de uno de los sensores. Considerando que los sonares miden la distancia con un error de desviación estándar σ_s=2 [cm], y que la detección se realiza con un límite D_{THs}=±5 [cm], se requieren 5 mediciones para que la probabilidad de detección correcta sea de $\mathbb{P}_n$=99.99%.

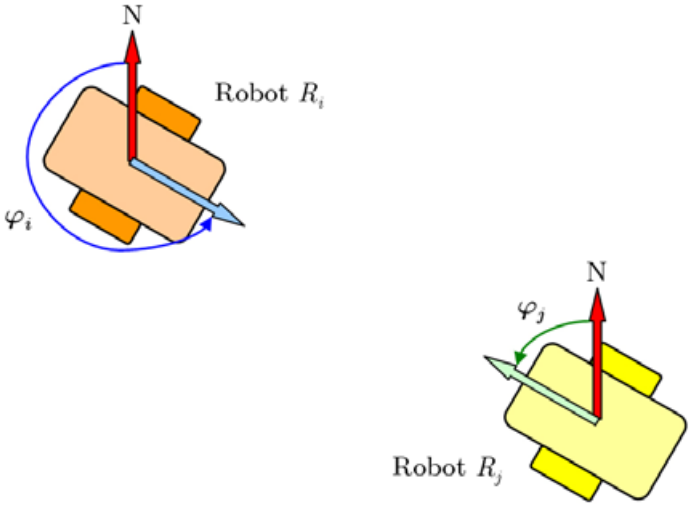

Figura 3.23: Detección de Fallas en la Brújula.

La forma más simple de comparar las mediciones de cada brújula, φ_i y φ_j, es colocando a ambos robots R_i y R_j, uno al frente del otro como se muestra en la figura 3.23. Esto se hace utilizando la cámara, ya que si ambos centran al otro robot en su imagen, quedan alineados. En este caso, la diferencia entre ambas mediciones no debería ser cero. Geométricamente, si ambos robots están apuntando uno al otro, la diferencia entre el mayor y el menor de los ángulos medidos debe ser 180 [°]. Considerando el ruido en la medición, esta diferencia presenta la siguiente distribución, donde σ_c es la desviación estándar del error de medición de la brújula:

$$\max\{\varphi_i, \varphi_j\} - \min\{\varphi_i, \varphi_j\} \sim N\left(180, 2\sigma_c^2\right) \qquad (3.31)$$

En este caso, la detección se realiza con un límite de D_{THc}=±0.5 [°] sobre los 180 [°]. Dado que σ_c=0.15 [°], se requieren tres mediciones para estimar correctamente la falla con una probabilidad de $\mathbb{P}_n$=99.99%.

El diagnóstico del sensor fallido y del tipo de falla sobre el sensor se realiza tras detectar que ha ocurrido una falla, mediante la revisión del sensor junto con otro miembro del grupo de robots. Una vez detectada la falla, se almacena en la memoria del robot el valor promedio de la diferencia entre las mediciones de ambos robots, y se procede a ubicar a otro robot para realizar nuevamente la comparación. Si el robot R_i detecta nuevamente una falla, es su sensor el que esta fallando, mientras que si no detecta falla alguna, es el sensor del robot R_j el que falla. El diagnóstico del tipo de falla se hace comparando la diferencia almacenada D_1, con la nueva, D_2:

$$|D_1 - D_2| < 0.1 \max\{D_1, D_2\} \qquad (3.32)$$

Si se cumple la ecuación 3.32, la falla se considera aditiva. En caso contrario, la falla se identifica como de tipo valor permanente de salida.

3.4.3. Detección y Diagnóstico de Fallas en el Sistema de Comunicación Inalámbrico

A diferencia de las fallas de sensor, las fallas del sistema de comunicación deben detectarse en base a pruebas de comunicación entre diferentes robots. El algoritmo presentado a continuación permite detectar si ha ocurrido una falla, y tras ello, diagnosticar qué parte del sistema de comunicación es la que esta fallando: el transmisor, el receptor o si el robot está fuera de alcance de cualquier otro.

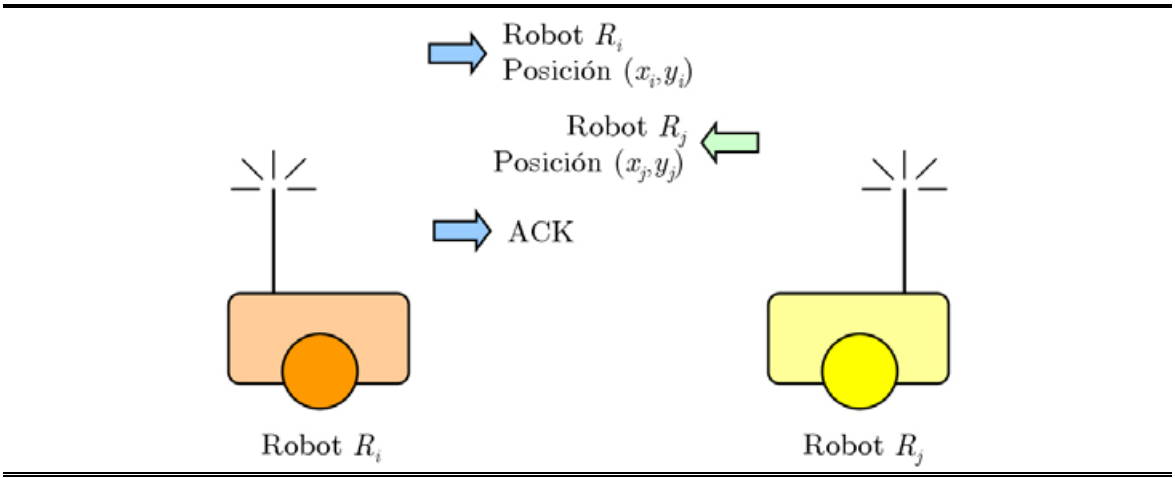

Figura 3.24: Protocolo de Comunicación para Detección de Fallas.

La detección de fallas se realiza mediante el uso de señales de "acknowledge" (ACK) entre los diferentes robots, como se muestra en la figura 3.24. Cada vez que se activa la capa 2 en el robot R_i, se envía una señal que indica la posición y el número del robot. Esto se hace con un cierto orden, para evitar que todos los robots revisen su sistema de comunicación al mismo tiempo. Como cada robot en el grupo cuenta con un identificador numérico, se utiliza ese número para determinar el tiempo en que se debe hacer la revisión.

Una vez enviado el mensaje, los robots que lo han recibido contestan con una señal de ACK, entregando su posición y número. Esta respuesta debe realizarse en base al orden numérico de cada robot, para evitar que el robot R_i

reciba múltiples respuestas al mismo tiempo. El problema se resuelve mediante el uso de ventanas de tiempo, asignadas a cada robot. Así, el robot R_j debe esperar *j-1* ventanas de tiempo antes de responder el mensaje, y el robot que envía el mensaje original debe esperar n ventanas antes de tomar alguna acción, donde n es el número de robots en el grupo.

Finalmente, al interior de cada ventana de tiempo el robot R_i avisa que recibió el mensaje con un ACK al robot correspondiente.

El funcionamiento de este algoritmo requiere que los sistemas de transmisión y recepción de los diferentes robots estén en correcto funcionamiento, por lo que si cualquier etapa en al cadena presenta un error, se detecta que ha ocurrido una falla.

El diagnóstico de la falla correspondiente se realiza en base al siguiente algoritmo para el robot R_i que hace la revisión:

1. Si el robot no recibe ningún ACK tras esperar las n ventanas de tiempo, se acerca a la última posición conocida de algún robot del grupo (robot R_j) y lo busca mediante la cámara. Una vez encontrado, el robot envía nuevamente su mensaje.

2. Si ahora el robot recibe el ACK correspondiente, quiere decir que la falla detectada anteriormente se debe a que estaba fuera del alcance del sistema de comunicación. Por el contrario, si no recibe ningún ACK, la falla se debe a un problema en el transmisor o en el receptor de alguno de ambos robots.

3. Para diagnosticar la falla, el robot se acerca a otro miembro del grupo, R_k, y realiza la rutina nuevamente. Si se recibe una señal de ACK, la falla era del robot Rj, mientras que si no detecta respuesta, la falla está en el robot R_i.

Al no existir otro método de compartir información entre los robots, el diagnóstico inmediato queda limitado a diferenciar entre fallas de "fuera de rango" y fallas en alguno de los elementos de comunicación.

Para determinar si es el transmisor o el receptor el que ha fallado, el robot que ha detectado la falla debe mantenerse cerca de otro robot del grupo hasta que a este le corresponda hacer una revisión de su sistema de comunicación. Si en ese momento no recibe ningún mensaje, es el receptor el que está fallando, mientras que si lo recibe es su transmisor.

La segunda señal de ACK del protocolo de revisión diseñado, permite que el robot que recibe un mensaje de revisión pueda determinar si su sistema de comunicación está funcionando correctamente. Si el robot R_j recibe el mensaje original, pero no recibe un ACK a su respuesta, detecta que ha ocurrido una falla, y puede diagnosticar de inmediato que se debe a que su sistema de transmisión ha fallado.

3.4.4. Análisis de Desempeño de la Capa 2

Debido a que el uso de la capa 2 requiere la presencia de al menos dos robots en un mismo lugar, no es útil la aplicación de los tiempos de detección y diagnóstico como medidas de desempeño. Para esta capa, las medidas de desempeño utilizadas son el número de falsas alarmas y la matriz de confusión.

El análisis de desempeño del sistema de detección de sensores redundantes se realiza en base a 1000 simulaciones, en las cuales se prueba el método de detección de fallas asumiendo que dos robots se han encontrado. En cada simulación se determina en forma aleatoria cual es el modo de operación de ambos robots (normal o con falla), asumiendo que sólo uno de los sensores redundantes puede fallar a la vez. Además, en el caso de fallas aditivas se asume

que el valor de la falla es de al menos un 10% del máximo valor medible por el sensor.

Las simulaciones muestran que, dado que la probabilidad de detección correcta de fallas es de un 99.99%, la cantidad de falsas alarmas es nula, ya que estadísticamente se requieren sobre las 10000 pruebas para que ocurra una falsa alarma. Es interesante notar que el número de falsas alarmas se mantiene nulo incluso cuando se varía hasta en un 10% el nivel de ruido presente en las mediciones.

Dado que la forma de detectar una falla del sensor S_i es independiente del método para detectar una en el sensor S_j, con $i \neq j$, la matriz de confusión para la detección de fallas es perfectamente diagonal, al no existir posibilidad que el sistema confunda fallas de un sensor con otras. Las simulaciones muestran además que la matriz de confusión para el diagnóstico de fallas es también diagonal, lo cual implica que dado que se ha detectado una falla en el sensor S_i, siempre se identifica correctamente cuál es el robot que posee el sensor con la falla y que tipo de falla es la que ha ocurrido sobre el sensor.

3.5. Interacción entre Capas

Así como la interacción de las capas de la arquitectura de DDF con las capas de la estructura de Control permite mejorar la información del estado de operación de cada robot, la interacción entre las capas de la arquitectura permite obtener nuevos datos para inferir el estado actual del robot.

Los sensores de navegación del robot, específicamente el sonar en el caso de los robots estudiados, deben ser capaces de evitar que el robot colisione con los obstáculos que se le presenten. Si se presenta alguna falla en el sonar, la navegación se dificulta y el robot tiende a atascarse con los obstáculos en su

camino. La capa 1 de DDF permite la implementación de un sensor virtual de colisiones, el cual se activa cuando el robot ha topado con algún obstáculo. Si se diagnostica esta falla sin que el sonar haya detectado el obstáculo, es factible que el sonar se encuentre en estado de falla. Por ello, cada vez que se identifique una falla 9, 10 u 11 en la capa 1, se activa la capa 2 para probar el estado del sonar y con ello determinar si el obstáculo no se detectó por una falla en el sonar.

En este caso, la interacción entre ambas capas genera la señal de activación de la capa superior, independientemente del tiempo transcurrido desde la última revisión, y disminuyendo el tiempo de detección para las fallas de sonar.

Otra forma de interacción entre ambas capas permite activar la capa superior ante fallas de la brújula. Dado que la capa 1 mantiene un filtro de Kalman que estima óptimamente la velocidad de giro del cuerpo del robot, mediante la integración en el tiempo de dicha variable, se puede estimar el valor del ángulo de dirección del robot como:

$$\varphi_k = \varphi_{k-1} + \tfrac{1}{2}\left(\dot{\varphi}_{k-1} + \dot{\varphi}_k\right)\Delta T \qquad (3.33)$$

En la ecuación 3.33, ΔT es el tiempo de muestreo utilizado. Para estimar mejor el nuevo ángulo de dirección se emplea el promedio de la velocidad en el último intervalo de muestreo.

Utilizando las mismas técnicas de selección adaptiva de los límites usada en la capa 2, es posible determinar la máxima diferencia aceptable entre las mediciones de la brújula y la estimación por odometría, considerando que es posible sólo tomar una muestra en cada intervalo de tiempo. Dado que la varianza del error de la medición en la brújula es conocida, y se puede calcular la varianza del error de estimación por odometría, es posible determinar el límite óptimo tras el cual hay una probabilidad $\mathbb{P}_n$ de que la brújula presente una falla. Se requiere de una mayor potencia computacional para aplicar este criterio, ya

que la varianza del error de la estimación por medio de la odometría varía en el tiempo. Si la desviación estándar del error en la medición del giróscopo es σ_{gyro}, la varianza del error de estimación por odometría en el tiempo k es:

$$\sigma_{odo,k}^2 = \sigma_{odo,k-1}^2 + \frac{\Delta T^2}{2}\sigma_{gyro}^2 \qquad (3.34)$$

Una vez calculada la varianza para la estimación por odometría, se puede calcular σ_d como:

$$\sigma_{d,k}^2 = \sigma_{odo,k}^2 + \sigma_c^2 \qquad (3.35)$$

En la ecuación 3.35, σ_c es la desviación estándar del error en la medición de la brújula. Con este valor calculado, se determina el valor del límite D_{TH} en base a la ecuación 3.30, con n_{max}=1. Si la diferencia entre ambas estimación supera el límite, hay una falla en la brújula y se activa la capa 2 para revisar el sensor.

4. Implementación de la Arquitectura

Este capítulo presenta la implementación de la arquitectura de DDF en un grupo de tres robots móviles simples. Se describe primero el sistema de robots cooperativos, detallando el equipamiento y las limitaciones que poseen. A continuación, se presenta la implementación de las fallas en el sistema de robots, y se indica que elementos de la arquitectura son factibles de probar en base a los recursos disponibles. Posteriormente se presentan detalles de la implementación, para concluir con la presentación de los resultados obtenidos, que validan el correcto funcionamiento de la arquitectura.

4.1. Descripción del Sistema de Robots Cooperativos

Con el fin de evaluar los resultados obtenidos a través de las simulaciones de la arquitectura, se ha construido un pequeño grupo de robots móviles sobre los cuales se implementa cada una de las capas del sistema de DDF.

El sistema cooperativo está compuesto por un grupo homogéneo de tres robots móviles no-holonómicos independientes, los que se observan en la figura 4.1.

Figura 4.1: Sistema Cooperativo de Robots Móviles.

Cada robot se encuentra equipado con dos motores servo independientes que permiten un control diferencial de movimiento. Los motores han sido modificados, permitiendo una rotación continua y un control de dirección y velocidad en base a la señal PWM de entrada. Como se aprecia en la figura 4.1, los motores están acoplados a ruedas de espuma, las que tienen un radio de r=2 [cm] y están separadas a $2b$=13 [cm].

El control de cada robot se realiza con un microcontrolador PIC 16F877 que genera las señales PWM para regular la velocidad y dirección de rotación de los motores, y adquiere las muestras de cada uno de los sensores. Posteriormente, las mediciones se almacenan en una memoria externa de 64 [Kbytes], a la cual se comunica el microcontrolador utilizando el protocolo I2C.

A nivel de sensores, los robots están equipados con: un sonar, una brújula magnética, un giróscopo, encoders en ambas ruedas y una cámara de video, utilizada para reconocer a los diferentes robots utilizando el marcador rojo que poseen sobre su cuerpo. Los sensores se proban extensivamente para determinar sus características estadísticas y rango de operación, las que son necesarias para la implementación de la arquitectura de DDF.

El sonar, modelo SRF-04, se encuentra ubicado en la parte frontal del robot, permitiendo identificar la distancia a obstáculos en un rango entre 3 [cm] y 3 [m]. La distancia al obstáculo se obtiene midiendo el ancho del pulso entregado por el sensor, el cual es proporcional al tiempo que tarda el eco de la señal ultrasónica en regresar tras el rebote en el obstáculo. El máximo valor del ancho del pulso es 36 [ms], lo cual implica que la tasa de muestreo con este sensor no puede superar los 25 [Hz]. En base a la medición de diferentes distancias, se observa que la desviación estándar del error en la medición es de σ_s=1.31 [cm].

A diferencia del sonar, la brújula magnética modelo CMPS-03, posee un microcontrolador en su interior que entrega los datos en forma digital, usando el protocolo I2C con una frecuencia de reloj máxima de 1 [MHz]. La dirección medida por la brújula se obtiene con una resolución de 0.1 [°], y las observaciones arrojan que la desviación estándar del error en la medición es de σ_c=0.157 [°]. Es importante mencionar que las brújulas deben ser calibradas inicialmente, lo cual genera una leve diferencia entre las mediciones de cada brújula. La mayor diferencia que se observa entre dos brújulas es de 3.5 [°], lo cual debe ser considerado al momento de implementar la capa 2 de la arquitectura de DDF.

El giróscopo permite medir la velocidad de giro que tiene el cuerpo del robot, por lo que requiere estar firmemente acoplado al robot, lo más cercano posible a su centro de masa. Se utiliza un sensor modelo PG-02, modelo comercial que se utiliza para helicópteros y aviones a control remoto. Debido al alto costo que tienen, sólo un robot se encuentra equipado con este sensor. La obtención de la velocidad en este tipo de sensores se obtiene mediante la comparación de dos señales de PWM. El giróscopo, en los modelos radio controlados, se utiliza para modificar la señal de PWM enviada a un motor servo, en base a la velocidad de rotación del avión o helicóptero, por lo que se debe ingresar una señal de PWM y determinar la variación que sufre a la salida

del sensor. Este sensor presenta un error en la medición de desviación estándar σ_{gyro}=2.22 [°/s].

La medición de la velocidad de cada rueda se hace en base a encoders colocados en cada una. Como se alcanza a apreciar en la figura 4.1, en la parte interior de cada rueda existe un disco dividido en 20 zonas, las cuales son de color blanco y negro en forma alternada. Se han colocados frente al disco un LED infrarrojo y un fototransistor, permitiendo que la luz del LED se refleje. El voltaje en el fototransistor depende de la cantidad de luz reflejada por el disco, permitiendo distinguir las zonas blancas de las negras. Este efecto genera un tren de pulsos a la salida del fototransistor cuando la rueda se encuentra en movimiento. La frecuencia de los pulsos es proporcional a la velocidad de giro de la rueda con la siguiente relación:

$$\omega = \frac{2}{20} 2\pi f = 0.628 f \left[rad/s \right] \qquad (4.1)$$

Este sensor muestra un error en la medición de varianza igual a σ_{enc}=0.0418 [rad/s].

Cada robot además dispone de una cámara de video de baja resolución modelo CMU-CAM01, desarrollada en el Robotics Institute de la Universidad de Carnegie Mellon. Esta cámara cuenta además con un procesador que permite realizar varias operaciones sobre la imagen. El uso de la cámara se remite a detectar la posición de otros robots en el ambiente, mediante la detección del marcador rojo que cada robot tiene. Esta operación no requiere usar el microcontrolador del robot, ya que el procesador de la cámara se encarga de determinar la posición del marcador en la imagen, entregándosela al sistema de control del robot. Este mueve las ruedas para dejar el marcador centrado en la imagen, y de esa forma se alinean ambos robots.

Debido al aumento significativo en el costo y la complejidad, no se implementó una red inalámbrica de comunicación, por lo cual no fue posible probar experimentalmente el algoritmo de DDF para el sistema de comunicación.

4.2. Implementación de Arquitectura y de las Fallas

Las capas de la arquitectura de DDF se implementaron en forma separada. A continuación se detalla la forma en que se hace la implementación de cada una de las capas.

4.2.1. Capa de Detección de Fallas Fatales

La primera capa en ser probada corresponde a la capa de detección de fallas fatales. La implementación de esta capa se realiza sobre un robot de prueba, que cuenta, junto con los encoders, del giróscopo para medir su velocidad de giro. En cada prueba se utilizó la misma trayectoria, la cual es almacenada en el programa de control del robot.

Debido a las limitaciones computacionales del microcontrolador a bordo del robot de prueba, no es posible implementar el método de diagnóstico en tiempo real, por lo que se utiliza la memoria externa del robot para almacenar los datos de los sensores y realizar el diagnóstico posteriormente en un computador. Considerando que la frecuencia de muestreo es de 10 [Hz] y que la medición de cada sensor se almacena en 2 [bytes], la memoria externa permite acumular hasta 18 [min] de datos.

Esta capa contempla el diagnóstico de 11 fallas diferentes, cuya implementación se describe a continuación.

Las fallas 1 y 2 corresponden a que la rueda derecha e izquierda, respectivamente, resbalen. Ambas fallas se implementan colocando un obstáculo pesado frente a la rueda correspondiente, mientras el robot se traslada sobre una superficie de cerámica. Dado que el roce no permite que la rueda se atasque, ésta resbala al topar con el obstáculo.

Las fallas 3 y 4, se implementaron en forma similar a las anteriores. En este caso la falla es la pérdida del momento de torsión en una de las ruedas, lo cual es equivalente a que la rueda se atasque. Para ello se colocó un obstáculo frente a la rueda correspondiente, pero en esta ocasión el movimiento del robot es sobre alfombra. Al ser mayor el coeficiente de roce la rueda se atasca en vez de resbalar y se emula una falla de pérdida de torque.

Las fallas 5 y 6 ocurren cuando uno de los encoders deja de entregar la medición de velocidad de la rueda a la que está adherido. En este caso la simulación de la falla se hace colocando un trozo de cartulina negra frente a la pareja LED – Fototransistor, eliminando el tren de pulsos que este último entrega al microcontrolador. De esta forma la velocidad medida es cero, aun cuando la rueda se encuentre girando.

La falla en el giróscopo (falla 7) se simula manteniendo desacoplado el sensor al cuerpo del robot. En el momento que se desea simular la falla, el sensor se levanta y se mantiene sin girar aunque el robot lo haga. Esto indica al microcontrolador que la velocidad de giro del robot es cero.

Al igual que las fallas 3 o 4, la falla 8 se implementa colocando un obstáculo frente al cuerpo del robot, haciendo que ambas ruedas se atasquen.

Dado que la capa 1 no se encuentra funcionando en tiempo real, es necesario simular el comportamiento que tendría el robot para detectar las fallas 9, 10 y 11. Estas son las fallas de activación del sensor virtual de colisiones, por lo que se programa el robot para que avance durante 30 [s], tras lo cual

retrocede la rueda afectada por la falla, según corresponda. Se coloca el obstáculo frente a la rueda correspondiente (para las fallas 9 y 10) o frente al cuerpo del robot (para la falla 11) de tal manera que alcance a topar el robot con el obstáculo durante unos pocos segundos antes de retroceder. Los datos se transfieren posteriormente al computador para determinar que hubiera ocurrido si se hubiera activado el sensor virtual de colisiones.

4.2.2. Capa de Detección para Robots Cooperativos

La segunda capa de la arquitectura de DDF se implementó en tiempo real en el grupo de robots, gracias a que la detección y el diagnóstico se realizan en forma distribuida entre los robots. Por ello, el trabajo computacional es dividido y se puede implementar en los microcontroladores utilizados.

Al no estar los robots conectados con una red inalámbrica de comunicación, el pedido de ayuda de un robot a otro se hace en forma manual mediante un botón. Este se presiona cuando un robot solicita la ayuda a otro, tras lo cual ambos robots se alinean utilizando la cámara. Posteriormente se toman las medidas con el sonar y la cámara, se promedian y se almacenan en la memoria externa. A continuación cada robot busca al tercero del grupo, con el cual realiza las mediciones una vez más, y se almacenan en la memoria. Todas las mediciones se ingresan al computador y se determina si ha ocurrido una falla, que sensor es el que ha fallado y finalmente se diagnostica si la falla es aditiva o de tipo valor permanente de salida.

La implementación de las fallas se hace por software, al agregar a la rutina de medición del microcontrolador que activa la capa 2, el efecto de la falla. En el caso de las fallas aditivas, se agrega un valor al azar, mayor al 1% del máximo valor del sensor. Esto implica que en el sonar se suma al menos 3 [cm], mientras en la brújula se suma al menos 3.7 [°]. En el caso de las fallas tipo

"stuck", se elige un valor al azar, el cual se asume que lee el microcontrolador del sensor correspondiente, cada vez que realiza la medición.

Dado que se conocen las desviaciones estándar del error en la medición de ambos sensores, se calcula de antemano el número de mediciones necesarias para el sonar y la brújula. Se considera un aumento del 10% en la desviación estándar del error en cada sensor, para evitar falsas alarmas. Con el fin de no requerir de muchos cálculos, se ha establecido que el máximo número de mediciones que se puede solicitar a un sensor es n_{max}=5.

En el caso del sonar, se deben diagnosticar fallas aditivas sobre los 3 [cm], por lo que se requiere de al menos siete mediciones para que con D_{THs}=±3 [cm], la probabilidad de detección correcta sea de $\mathbb{P}_{\mathrm{n}}$=99.99%. Debido a que el número de mediciones es superior al máximo establecido, se modifica el límite a D_{THs}=±3.55 [cm], que permite mantener la probabilidad, pero con solo cinco mediciones.

Al existir un desfase entre las mediciones de las distintas brújulas se debe seleccionar un umbral de detección mayor que la máxima diferencia entre ellas, evitando así falsas alarmas. Considerando entonces que D_{THc}=±3.5 [°], se necesita sólo una medición para que $\mathbb{P}_{\mathrm{n}}$=99.99%.

Como se indicó anteriormente, las fallas del sistema de comunicación no se han implementado. Al no existir este medio de comunicación y dado que la capa 1 no se prueba en línea, no es posible probar el funcionamiento de la interacción entre las capas de la arquitectura de DDF.

4.3. Resultados y Validación de la Arquitectura

A continuación se presentan los resultados de la implementación de cada capa de la arquitectura de DDF.

4.3.1. Capa de Detección de Fallas Fatales

Cada una de las fallas diagnosticables por la capa 1 se han implementado en el robot de prueba y los datos se han almacenado en la memoria para su procesamiento posterior utilizando MATLAB. Las siguientes figuras ilustran el resultado del cálculo de la probabilidad de cada hipótesis para algunas de las fallas, utilizando los datos obtenidos del robot.

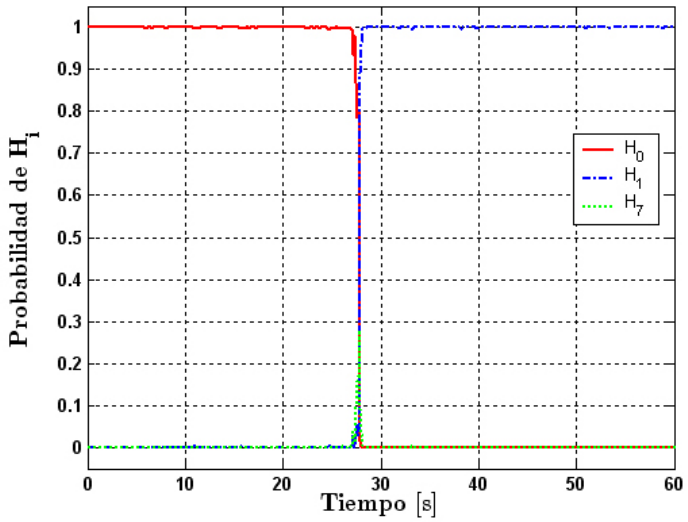

Figura 4.2: Diagnóstico Experimental de la Falla 1.

Como se observa en la figura 4.2, si una rueda resbala el diagnóstico de la falla se logra exitosamente. No es posible realizar una comparación de los tiempos de respuesta con respecto a los de las simulaciones, ya que la medición de los tiempos de falla en la implementación es muy inexacta. Sólo a partir de

una comparación gráfica se puede determinar que el tiempo de detección es similar. Una diferencia con respecto a las simulaciones es que tras ocurrida la falla existe una leve alza de la probabilidad de la hipótesis $\mathbb{H}_7$, lo cual puede ser explicable por las diferencias existentes entre los parámetros del robot (radio de las ruedas, largo del eje y los valores de la desviación estándar del ruido en cada sensor) y los utilizados en el banco de filtros de Kalman.

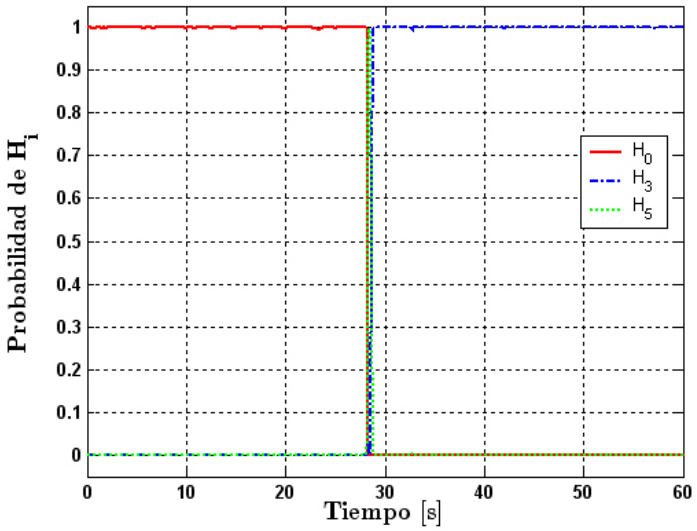

Figura 4.3: Diagnóstico Experimental de la Falla 3.

En el caso de las fallas de pérdida del momento de torsión, la respuesta es similar a la de las simulaciones, identificando correctamente la falla en todos los casos probados. Como se ilustra en la figura 4.3, existe un aumento más marcado en este caso de la probabilidad de la hipótesis $\mathbb{H}_5$, que al igual que en el caso anterior se puede deber a las insertazas en los parámetros de los modelos.

A continuación la figura 4.4 muestra el diagnóstico de una falla en el sensor de velocidad izquierdo.

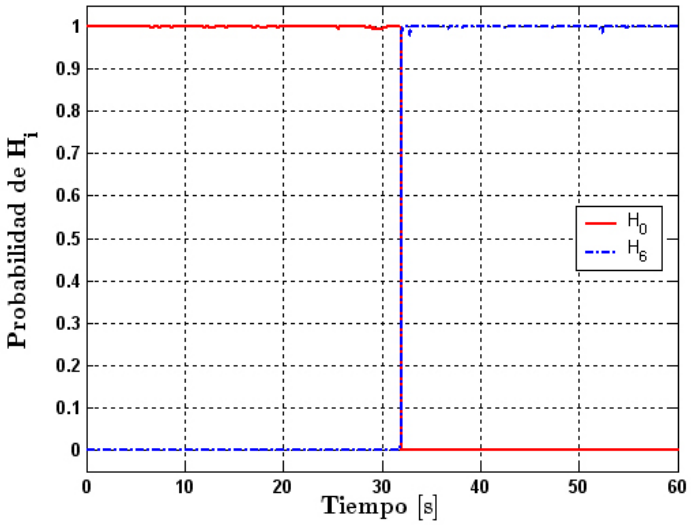

Figura 4.4: Diagnóstico Experimental de la Falla 6.

Se puede apreciar en la figura que el diagnóstico de la falla se realiza correctamente. La diferencia en los niveles de ruido observados en estas figuras, en comparación a las obtenidas en las simulaciones se debe a que las desviaciones estándar del ruido de las simulaciones son mucho mayores que la presente en los sensores utilizados. Esto permite obtener estimaciones más certeras en la implementación, y con ello error menor en el cálculo de las probabilidades de cada hipótesis.

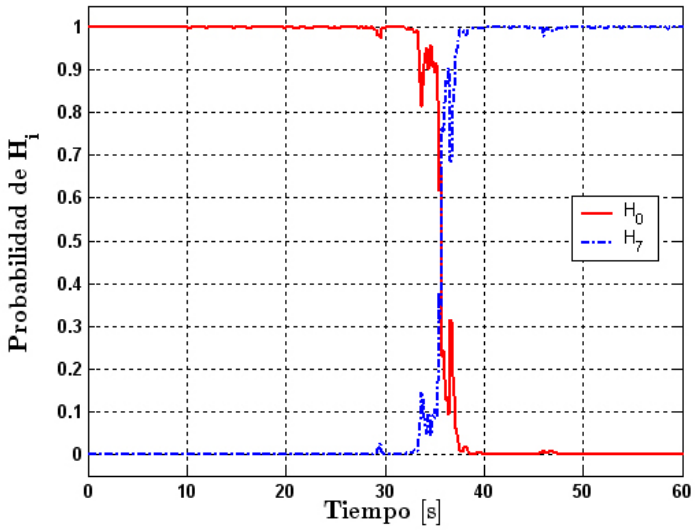

Figura 4.5: Diagnóstico Experimental de la Falla 7.

Los datos obtenidos para una falla en el giróscopo (falla 7) entregan resultados similares a los de las simulaciones. Como se observa en la figura 4.5, la diferencia fundamental radica en que el tiempo de respuesta es mucho menor en la implementación. Esto se debe a que el nivel de ruido en la medición del giróscopo es menor que la utilizada en los sensores. Otro factor para esta diferencia es la velocidad de rotación de la ruedas en los robots implementados. A diferencia de las simulaciones, donde se estimaba una velocidad de rotación cercana a 1 [rad/s], las ruedas de los robots implementados rotan a velocidades cercanas a los 3 [rad/s], aumentando también la velocidad de rotación del cuerpo del robot. Esto hace que el efecto del ruido sea menor aún.

Tal como se explicó anteriormente, la implementación del sensor virtual de colisiones no puede realizarse en tiempo real, por lo que se simula el comportamiento requerido en la programación del robot.

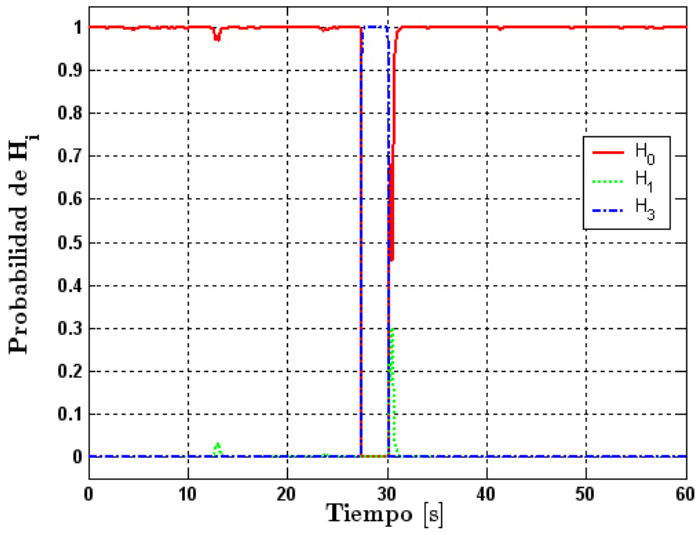

Figura 4.6: Diagnóstico Experimental de la Falla 10.

La figura 4.6 ilustra el cálculo de las probabilidades para cada hipótesis en base a los datos experimentales. Se observa que aproximadamente en t=27 [s] el robot colisiona con el obstáculo, ya que la probabilidad de $\mathbb{H}_3$ aumenta. Como la probabilidad del estado de operación normal vuelve a aumentar tras el cambio de velocidad en t=30 [s], se puede diagnosticar que la falla se debe a un obstáculo, y no a una pérdida del momento de torsión del motor derecho. Esto activa el sensor virtual de colisiones.

4.3.2. Capa de Detección para Robots Cooperativos

Gracias a los bajos requerimientos de esta capa de DDF, es posible implementar en tiempo real su funcionamiento en los robots. Cada prueba se realizó comenzando con los robots en posiciones aleatorias y tras unos segundos de funcionamiento, el robot R_1 activa su capa de detección y comienza a buscar otro robot del grupo para revisar sus sistemas. La figura 4.7 ilustra la fase uno

de la prueba experimental, en que los robots se encuentran realizando sus labores en forma independiente.

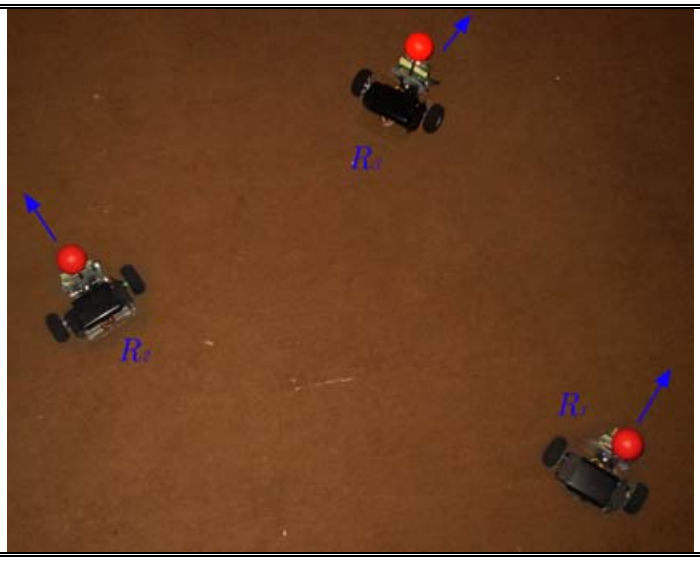

Figura 4.7: Fase uno de la Detección.

Al activarse el sistema de revisión de R_1, este busca a otro robot (R_2) que le ayude y comunica su intención pidiéndole ayuda. Dada la ausencia del sistema de comunicación, esto se hace manualmente como se explicó con anterioridad.

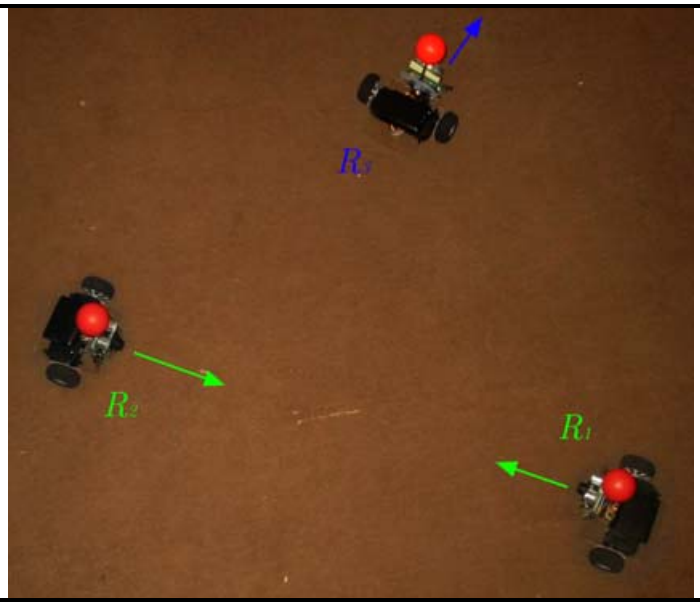

Figura 4.8: Fase dos de la Detección.

La figura 4.8 ilustra la siguiente fase en la detección de las fallas. Una vez que ambos robots están alineados gracias a sus cámaras, se toma la cantidad de mediciones necesarias y se almacenan en la memoria.

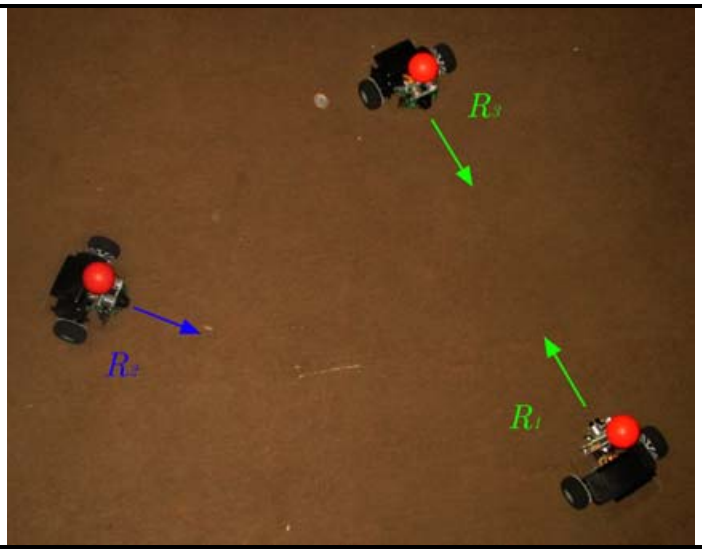

Figura 4.9: Fase tres de la Detección.

Posteriormente, el robot R_1 busca a un tercer robot, R_3, con el cual revisa una vez más sus sensores, almacenando las nuevas mediciones.

Tras esta operación se obtienen los datos almacenados en las memorias de los tres robots y se comparan las mediciones realizadas con los límites previamente establecidos.

Las 10 pruebas experimentales realizadas muestran que el sistema es capaz de detectar correctamente las fallas, diagnosticando cuál es el sensor con problemas e incluso determinar que tipo de falla es la que tiene el sensor. Además se observa que no existen falsas alarmas, al igual que en las simulaciones realizadas anteriormente.

Debido a que la segunda capa de la arquitectura de DDF no utiliza modelos para detectar las fallas, no se aprecian mayores diferencias entre los

datos obtenidos en las simulaciones y aquellos resultantes de las pruebas experimentales.

5. Conclusiones y Trabajos Futuros

El presente capítulo resume las reflexiones finales sobre esta Tesis. Para ello se hace un breve análisis de los objetivos alcanzados y se dejan propuestas futuras líneas de investigación referentes al mismo tema.

5.1. Análisis del Cumplimiento de los Objetivos

De acuerdo a lo indicado inicialmente, esta Tesis se centra en esencia en dos objetivos principales:

- Desarrollar una arquitectura de detección y diagnóstico de fallas aplicada a robots móviles, que aproveche la redundancia existente al interior de grupos cooperativos, y
- Validar los datos obtenidos a través de simulaciones, mediante la construcción de un grupo de robots en los cuales implementar la arquitectura.

Observando los resultados obtenidos tanto a través de las simulaciones como en los robots construidos, se puede decir que los objetivos de esta Tesis se han cumplido plenamente.

En base al modelo matemático del robot móvil, desarrollado al comienzo de este trabajo, se logró diseñar una arquitectura de detección y diagnóstico de fallas. Esta aprovecha la existencia de información redundante, tanto en los robots en forma individual como en el conjunto de ellos, generando una arquitectura nueva y de bajo costo, la cual es capaz de detectar una gran cantidad de fallas y diagnosticar exitosamente el modo de operación de cada robot. Esto permite que el operador del robot obtenga nueva información, agilizando el proceso de reparación y con ello aumentando la eficiencia de los

robots utilizados. Esta arquitectura proporciona además una base formal sobre la cual desarrollar nuevas extensiones en la detección de fallas sobre grupos cooperativos de robots, tema que no se encuentra actualmente desarrollado.

Junto con las simulaciones, la construcción de los robots permitió reafirmar que la arquitectura diseñada entrega buenos resultados, incluso en grupos cooperativos de limitados recursos computacionales. Si bien la capa 1 de la arquitectura no opera en tiempo real, se puede observar que el método de detección y diagnóstico funciona correctamente, aunque exista incertidumbre en la medición de los parámetros del robot. Los resultados obtenidos muestran que, aunque un robot no tenga la capacidad de realizar los cálculos a bordo, basta que cuente con un sistema de almacenamiento de las variables medidas, para que un operador pueda diagnosticar correctamente la falla que ha ocurrido, sin requerir de nuevas pruebas de funcionamiento. Además, considerando la rapidez con que aumenta la capacidad computacional de los microcontroladores, es posible afirmar que la implementación en tiempo real, en robots tan pequeños como los diseñados, es posible en un corto plazo.

Lamentablemente, debido al mayor costo relativo que implica la implementación de una red inalámbrica para los robots, no es posible probar por completo la amplia gama de posibilidades que permite esta arquitectura.

En conclusión, se ha logrado abarcar satisfactoriamente todos los objetivos propuestos en la presente Tesis, diseñando e implementando la arquitectura deseada, limitada eso si por el costo y las tecnologías disponibles.

5.2. Trabajos Futuros

Debido a que el área de la robótica móvil, y más específicamente los sistemas cooperativos de robots, son de exploración reciente, existe una gran cantidad de posibilidades de extensión del trabajo aquí presentado.

Con respecto a la arquitectura diseñada, se requiere primero de una validación completa y en tiempo real del funcionamiento del sistema. Para ello se necesita la incorporación de un sistema de comunicación entre los robots, y el diseño de una plataforma dedicada para detección y diagnóstico de las fallas. Esta debe contar con un procesador de mayores capacidades computacionales que los utilizados, que permita identificar las fallas sin necesidad de un computador externo, agilizando aún más el proceso de reparación ante la presencia de problemas.

Dada la flexibilidad que permite esta arquitectura, se puede ampliar también el número de sensores utilizados, usando sistemas de posicionamiento absoluto como GPS, u otros sensores de las variables ambientales, que interesen para alguna aplicación.

Además, considerando que ya se encuentra implementada una plataforma de pruebas, es posible probar nuevos sistemas de detección de fallas, que agilicen el proceso o simplemente permitan a estudiantes trabajar con sistemas tan complejos como lo son los grupos de múltiples robots.

Dando un paso más allá de los objetivos de esta Tesis, un trabajo muy interesante es la construcción de sistemas de control tolerante a fallas. Estos requieren de la información entregada por las arquitecturas de detección y diagnóstico de fallas para determinar otros cursos de acción en el control del movimiento del robot, adaptando el sistema al modo de operación existente.

La cantidad de estudios que se encuentran en curso actualmente, sobre sistemas de múltiples robots son innumerables, por lo que el campo de estudio parece ilimitado. El trabajo presentado en esta Tesis es sólo un pequeño aporte a una de las tantas aristas que posee la robótica, y el número de trabajos futuros sobre este tipo de sistemas sólo esta limitado por la imaginación con que se enfrentan los desafíos que se presenten.

BIBLIOGRAFÍA

ALLIANCE, University of Tennessee, *Distributed Intelligence Laboratory,* [Online]. Disponible en: http://www.cs.utk.edu/~parker/Distributed-Intelligence-Lab/index.html

ANGELES, J. (1997) *Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms*. Springer.

BASSEVILLE, M. y NIKIFOROV, I.V. (1998, Noviembre). *Detection of Abrupt Changes – Theory and Application* [Online]. Disponible en: http://www.irisa.fr/sigma2/kniga/

BEAGLE2, *Beagle2 Website* [Online]. Disponible en: http://www.beagle2.com

BORESTEIN, J. y FENG, L. (1996) Measurement and correction of systematic odometry errors in mobile robots. *IEEE Trans. on Robotics and Automation*, vol. 12**,** nº 5, 869-879.

BORESTEIN, J., EVERETT, H.R. y FENG, L. (1996) *Where am I? Sensors and Methods for Mobile Robot Positioning*. University of Michigan.

CAO, Y., FUKUNAGA, A. y KAHNG, A. (1997) Cooperative Mobile Robotics: Antecedents and Directions. *Autonomous Robots*, vol. 4, 1-23.

CARLSON, J., MURPHY, R.R. y NELSON, A. (2004) Follow-up Analysis of Mobile Robot Failures. *Proc. of the IEEE International Conference on Robotics and Automation*, Abril 2004, 4987-4994, New Orleans, EE.UU.

CARRASCO, R. y CIPRIANO, A. (2003) Queen-bee Genetic Optimization of an Heuristic Based Fuzzy Control Scheme for a Mobile Robot. *Proc. of the First IEEE Latin-American Conference on Robotics and Automation*, Noviembre 2003, 61-66, Santiago, Chile.

CARRASCO, R. y CIPRIANO, A. (2004) Fuzzy Logic Based Nonlinear Kalman Filter Applied to Mobile Robots Modelling. *Proc. of the IEEE International Conference on Fuzzy Systems*, Julio 2004, 1485-1490, Budapest, Hungría.

CORADESCHI, S., TADOKORO, S. y BIRK, A. (2002) *RoboCup 2001: Robot Soccer World Cup V*. Springer Verlag.

CRAIG, J. (1989) *Introduction to Robotics: Mechanics and Control*. 2nd ed., Addison-Wesley Pub. Co.

CURRIE, A. *The History of Robotics by Adam Currie* [Online]. Disponible en: http://cache.ucr.edu/~currie/roboadam.htm

DIXON, W.E., WALKER, I.D., DAWSON, D.M. y HARTRANFT, J.P. (2000) Fault Detection for Robot Manipulators with Parametric Uncertainty: A Prediction-Error-Based Approach. *Proc. of the IEEE International Conference on Robotics and Automation*, Abril 2000, vol. 4, 3628-3634, San Francisco, EE.UU.

DIXON, W.E., WALKER, I.D. y DAWSON, D.M. (2001) Fault Detection for Wheeled Mobile Robots with Parametric Uncertainty. *Proc. of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Julio 2001, vol. 2, 1245-1250, Como, Italia.

DODDS, G., WILSON, G. y ZATARI, A. (1995) Cooperative Arm Tasks and Sensor Fusion. *IEE Colloquium on Intelligent Measuring Systems for Control Applications*, Abril 1995, 6/1-6/5, Londres, Inglaterra.

BRITANNICA (2004) *Robot*. Encyclopædia Britannica Premium Service.

GOEL, P., DEDEOGLU, G., ROUMELIOTIS, S.I. y SUKHATME, G.S. (2000) Fault Detection and Identification in a Mobile Robot Using Multiple Model

Estimation and Neural Network. *Proc. of the IEEE International Conference on Robotics and Automation*, Abril 2000, 2302-2309, San Francisco, EE.UU.

GOLDBERG, D.E. (1989) *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Pub. Co.

HASHIMOTO, M., KAWASHIMA, H., NAKAGAMI, T. y OBA, F. (2001) Sensor Fault Detection and Identification in Dead-Reckoning System of Mobile Robot: Interacting Multiple Model Approach. *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Octubre 2001, vol. 3, 1321-1326, Maui, EE.UU.

HOLLAND, J.H. (1992) *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. MIT Press.

ISERMANN, R. (1995) Model Based Fault Detection and Diagnosis Methods. *Proc. of the American Control Conference*, Junio 1995, vol. 3, 1605-1609, Seattle, EE.UU.

JUNG, S.H. (2003) Queen-Bee Evolution for Genetic Algorithms. *IEE Electronic Letters*, 20 Marzo 2003, 575-76.

KALMAN, R.E. (1960) A New Approach to Linear Filtering and Prediction Problems. *ASME Journal of Basic Engineering*, vol. 86, 35-45.

KAWABATA, K., AKAMATSU, T. y ASAMA, H. (2002) A Study of Self – Diagnosis of an Autonomous Mobile Robot: Expansion of State Sensory System. *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Octubre 2002, vol. 2, 1802-1807, Lausanne, Suiza.

KMELNITSKY, V.M. (2002) Automated On-Line Diagnosis and Control Configuration in Robotic Systems Using Model Based Analytical Redundancy,

M. Sc. thesis, Department of Mechanical Engineering, Worcester Polytechnic Institute, Worcester, EE.UU.

KOUSHANFAR, F., SLIJEPCEVIC, S., POTKONJAK, M. y SANGIOVANNI, A. (2002) Error Tolerant Multimodal Sensor Fusion. *IEEE CAS Workshop on Wireless Communication and Networking*, Septiembre 2002, Pasadena, EE.UU.

LEUSCHEN, M.L., CAVALLARO, J.R. y WALKER, I.D. (2002) Robotic Fault Detection Using Nonlinear Analytical Redundancy. *Proc. of the IEEE International Conference on Robotics and Automation*, Mayo 2002, vol. 1, 456-463, Washington DC, EE.UU.

LIN, C.T. y LEE, C.S.G. (1996) *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*. Prentice Hall.

MAYBECK, P.S. (1979) *Stochastic Models, Estimation, and Control*. Academic Press.

MAYBECK, P.S. y HANLON, P.D. (2000) Multiple-Model Adaptive Estimation Using a Residual Correlation Kalman Filter Bank. *IEEE Trans. on Aerospace and Electric Systems*, vol. 36, nº 2, 393-406.

MICHAELSON, D.G. y JIANG, J. (2000) Modeling of Redundancy in Multiple Mobile Robots. *Proc. of the American Control Conference*, Junio 2000, vol. 2, 1083-1087, Chicago, EE.UU.

MILLIBOT, Universidad Carnegie Mellon, *The CMU Millibots* [Online]. Disponible en: http://www.contrib.andrew.cmu.edu/~rjg/millibots/millibot_project.html.

NASA, *Jet Propulsion Laboratory Website* [Online]. Disponible en: http://www.jpl.nasa.gov

PARKER, L. (1998) ALLIANCE: An Architecture for Fault Tolerant Multirobot Cooperation. *IEEE Trans. on Robotics and Automation*, vol. 14, nº 2, 220-240.

ROBOCUP, *RoboCup Website* [Online]. Disponible en: http://www.robocup.org

ROBOTANTS, Artificial Intelligence Laboratory, MIT, *The Ants: A Community of Microrobots* [Online]. Disponible en: http://www.ai.mit.edu/ projects/ants/

ROUMELIOTIS, S.I. y BEKEY, G.A. (1997) An Extended Kalman Filter for Frequent Local and Infrequent Global Sensor Data Fusion. *Proc. of the Sensor Fusion and Decentralized Control in Autonomous Robotic Systems*, Octubre 1997, 11-22, Pittsburgh, EE.UU.

ROUMELIOTIS, S.I., SUKHATME, G.S. y BEKEY, G.A. (1998) Fault Detection and Identification in a Mobile Robot using Multiple-Model Estimation. *Proc. of the IEEE International Conference on Robotics and Automation*, Mayo 1998, 2223-2228, Leuven, Bélgica.

ROUMELIOTIS, S.I., SUKHATME, G.S. y BEKEY, G.A. (1998) Sensor Fault Detection and Identification in a Mobile Robot. *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Octubre 1998, 1383-1388, Victoria, Canadá.

ROUMELIOTIS, S.I. y BEKEY, G.A. (2000) Collective Localization: A Distributed Kalman Filter Approach to Localization of Groups of Mobile Robots. *Proc. of the IEEE International Conference on Robotics and Automation*, Abril 2000, 2958-2965, San Francisco, EE.UU.

SABBATINI, R., *W. Grey Walter: The Machina Speculatrix* [Online]. Disponible en: http://www.epub.org.br/cm/n09/historia/documentos_i.htm

SCHNEIDER, H. y FRANK, P.M. (1996) Observer-Based Supervision and Fault Detection in Robots Using Nonlinear and Fuzzy Logic Residual Observation. *IEEE Trans. on Control System Technology*, vol. 4, nº 3, 274-282.

SHIN, J.H., LEE, C.Y. y LEE, J.J. (1999) Robust Fault-Tolerant Control Framework for Robot Manipulators with Free-Swinging Joint Failures: Fault Detection, Identification, and Accommodation. *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Octubre 1999, vol. 1, 185-190, Kyongju, Corea.

SOIKA, M. (1997) Grid Based Fault Detection and Calibration of Sensors on Mobile Robots. *Proc. of the IEEE International Conference on Robotics and Automation*, Abril 1997, vol. 3, 2589-2594, Albuquerque, EE.UU.

SONY, *Aibo Website* [Online]. Disponible en: http://www.aibo.com

STROUPE, A., MARTIN, M. y BALCH, T. (2001) Distributed sensor fusion for object position estimation by multi-robot systems. *Proc. of the IEEE International Conference on Robotics and Automation*, Mayo 2001, vol. 2, 1092-1098.

TINÓS, R., NAVARRO-SERMENT, L.E. y PAREDIS, C.J.J (2001) Fault Tolerant Localization for Teams of Distributed Robots" *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Maui, USA, October 2001, vol. 2, 1061-1066.

TINÓS, R., TERRA, M.H. y BERGERMAN, M. (2002) Fault Tolerance in Cooperative Manipulators. *Proc. of the IEEE International Conference on Robotics and Automation*, Mayo 2002, vol. 1, 470-475, Washington DC., EE.UU.

VISINSKY, M., CAVALLARO, J.R. y WALKER, I.D. (1993) Layered Dynamic Fault Detection and Tolerance for Robots. *Proc. of the IEEE International*

Conference on Robotics and Automation, Mayo 1993, vol. 2, 180-187, Atlanta, EE.UU.

WALLICH, P. (2001) Mindstorms: not just a kid's toy. *IEEE Spectrum*, vol. 38, nº 9, 52-57.

WASHINGTON, R. (2000) On-board Real-Time State and Fault Identification for Rovers. *Proc. of the IEEE International Conference on Robotics and Automation*, Abril 2000, vol. 2, 1175-1181, San Francisco, EE.UU.

WELCH, G. y BISHOP, G. (2003). *An Introduction to the Kalman Filter* [Online]. Disponible en: http://www.cs.unc.edu/~welch/kalman/kalmanIntro.html

WISE, E. (1999) *Applied Robotics*. Delmar Learning.

YAO, X. (1999) Evolving Artificial Neural Networks. *Proc. of the IEEE*, vol. 87, nº 9, 1423-1447.

ANEXOS

ANEXO A. FILTRO DE KALMAN

La presencia de ruido en señales de datos, hace siempre necesaria la utilización de herramientas de filtrado, que permitan obtener información útil o más certera, de las señales que se analizan. Esto no sólo se aplica a sistemas de comunicación sino también a sistemas de control, en que las mediciones necesarias para lograr el control se ven afectadas por el ruido, disminuyendo la precisión de las medidas y con ello, del sistema de control.

En el año 1960, R.E. Kalman entrega una nueva herramienta que permite minimizar los efectos del ruido sobre un proceso lineal, obteniendo una estimación en base a las mediciones realizadas, pero cuyo error es mucho menor que si sólo se utiliza en forma ciega la información de los sensores [*Kalman, 1960*]. Si bien originalmente su trabajo estuvo centrado en la navegación de sistemas aeroespaciales, sus aplicaciones actuales cubren todo rango de procesos, incluso aquellos no-lineales, gracias a las extensiones que se han realizado posteriormente. Descripciones más detalladas de la formulación del filtro pueden encontrarse en [*Welch, 2003*] y [*Maybeck, 1979*].

A.1. FILTRO DE KALMAN PARA SISTEMAS LINEALES

El filtro de Kalman es una herramienta iterativa, que permite estimar el estado de un sistema lineal en forma óptima, al minimizar el efecto del ruido en él. Un sistema lineal es aquel que puede ser expresado de la siguiente forma:

$$\begin{aligned} X_{k+1} &= AX_k + BU_k + \omega_{k+1} \\ Y_{k+1} &= CX_{k+1} + \mu_{k+1} \end{aligned} \qquad (A.1)$$

Donde $X_k \in \mathbb{R}^n$ es el vector de estado del sistema en el tiempo k, $U_k \in \mathbb{R}^m$ es la entrada aplicada al sistema en k, $Y_k \in \mathbb{R}^p$ es el vector de medición en el tiempo

k y, ω_k y μ_k son variables aleatorias que representan el ruido del proceso y de la medición respectivamente. Las variables aleatorias poseen una distribución de probabilidades normal, independientes entre si y con matrices de covarianza Q y R respectivamente:

$$\begin{aligned} \omega &\sim N(0,Q) \\ \mu &\sim N(0,R) \end{aligned} \qquad (A.2)$$

El proceso por el cual se aplica el filtro de Kalman puede subdividirse en dos etapas: una etapa de estimación y otra de actualización.

La etapa de estimación del valor actual del vector de estado, $\tilde{X}$, y del vector de medición, $\tilde{Z}$, se hace a través de la utilización de la ecuación A.1, pero considerando que las variables aleatorias poseen un valor nulo:

$$\begin{aligned} \tilde{X}_{k+1} &= AX_k + BU_k \\ \tilde{Z}_{k+1} &= C\tilde{X}_{k+1} \end{aligned} \qquad (A.3)$$

Además, esta etapa también consta de la estimación de la matriz de covarianza para el error, $\tilde{P}$:

$$\tilde{P}_{k+1} = AP_kA^T + Q \qquad (A.4)$$

Una vez realizada la estimación para el tiempo *k+1*, la etapa de actualización se utiliza para mejorar el valor estimado del vector de estado X_{k+1}, del vector de medición Z_{k+1} y de la matriz de covarianza P_{k+1}, en base a la medición realizada, Y_{k+1}.

$$\begin{aligned} X_{k+1} &= \tilde{X}_{k+1} + K_{k+1}\left(Y_{k+1} - \tilde{Z}_{k+1}\right) \\ Z_{k+1} &= CX_{k+1} \\ P_{k+1} &= \left(I - K_{k+1}C\right)\tilde{P}_{k+1} \end{aligned} \qquad (A.5)$$

El error entre la medición realizada y la medición estimada se pondera por la denominada ganancia de Kalman. Esta ganancia da un peso relativo a la

medición en comparación con la estimación realizada. Así, si la ganancia es muy pequeña, significa que la estimación a priori, tiene más peso en la estimación final del estado del sistema, mientras una ganancia grande refleja que la medición es más certera que lo estimado anteriormente. La ganancia del filtro para el tiempo *k+1* se calcula de la siguiente forma, donde S_{k+1} representa la matriz de covarianza del residuo:

$$K_{k+1} = \tilde{P}_{k+1} C^T S_{k+1}^{-1}$$
$$S_{k+1} = C\tilde{P}_{k+1} C^T + R \qquad (A.6)$$

El algoritmo se aplica en forma iterativa, obteniéndose la estimación del vector de estado para el sistema en cada tiempo de muestreo.

La siguiente figura muestra los resultados de la estimación de la posición de un auto en base a la aceleración entregada y a la medición de la posición, la cual se logra con un error de desviación estándar de 40 [m].

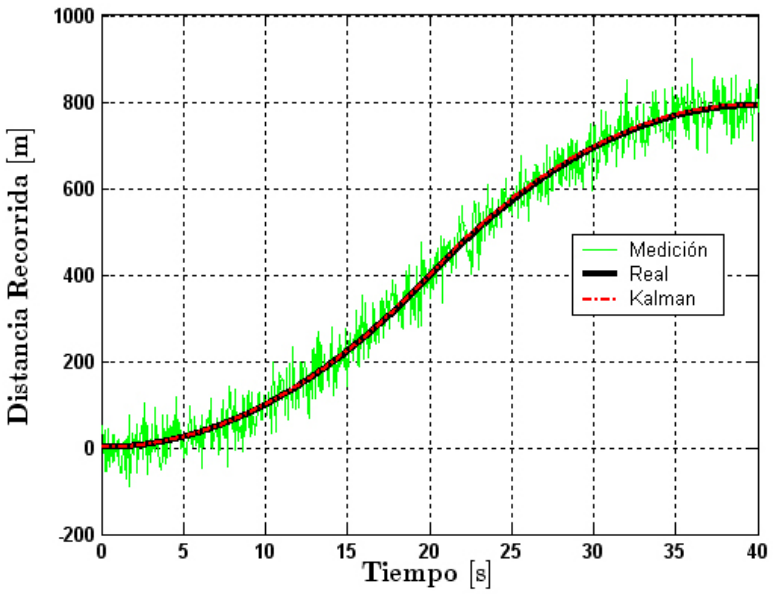

Figura A.1: Estimación de la Posición Utilizando un Filtro de Kalman.

Tal como se observa en la figura A.1, si bien existe un error considerable en las mediciones, la estimación entregada por el filtro se ajusta muy bien a la posición real que tiene el auto en cada momento.

La figura A.2 muestra la diferencia que existe entre el error de la medición y el error de la estimación arrojada por el filtro de Kalman.

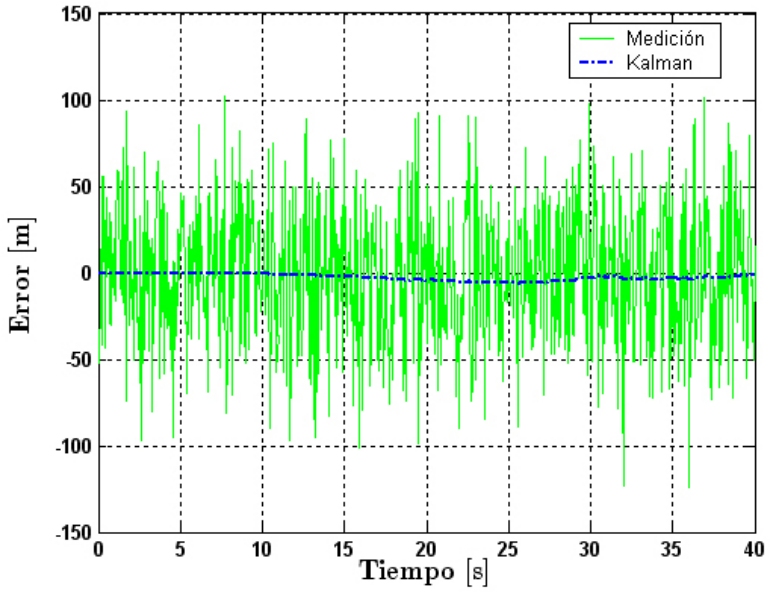

Figura A.2: Estimación de la Posición Utilizando un Filtro de Kalman.

A.2. Filtro de Kalman Extendido

La principal desventaja que posee el filtro de Kalman, es que su aplicación se ve limitada a sistemas lineales, dejando de lado la amplia gama de procesos no-lineales que requieren de sistemas de filtrado similares.

Una forma de abordar este problema es extender las ecuaciones del filtro de Kalman lineal para aplicarlo a problemas no-lineales, entregando una solución

que si bien es sub-óptima, permite estimar de mejor manera el estado de un sistema ante la presencia de ruido blanco.

Un sistema no-lineal se representa de la siguiente forma:

$$\begin{aligned} X_{k+1} &= f\left(X_k, U_k, \omega_{k+1}\right) \\ Z_{k+1} &= g\left(X_{k+1}, \mu_{k+1}\right) \end{aligned} \qquad (A.7)$$

El algoritmo para aplicar el filtro de Kalman extendido es similar al del filtro lineal, por lo que primero se requiere estimar el estado del sistema considerando que no existe ruido:

$$\begin{aligned} \tilde{X}_{k+1} &= f\left(X_k, U_k, 0\right) \\ \tilde{Z}_{k+1} &= g\left(X_{k+1}, 0\right) \end{aligned} \qquad (A.8)$$

Para estimar la matriz de covarianza del error (ecuación A.4) se requiere determinar las matrices que representan el sistema, las que se obtienen linealizando el sistema en torno al punto de operación actual:

$$\begin{aligned} A_{k+1[i,j]} &= \frac{\partial f_{[i]}}{\partial X_{[j]}}\left(X_k, U_k, 0\right) \\ C_{k+1[i,j]} &= \frac{\partial g_{[i]}}{\partial X_{[j]}}\left(X_{k+1}, 0\right) \end{aligned} \qquad (A.9)$$

Además, se requiere calcular la linealización del sistema con respecto al ruido blanco, para determinar el efecto que tienen las nolinealidades sobre el ruido y corregir la estimación:

$$\begin{aligned} \Omega_{k+1[i,j]} &= \frac{\partial f_{[i]}}{\partial \omega_{[j]}}\left(X_k, U_k, 0\right) \\ \mathrm{M}_{k+1[i,j]} &= \frac{\partial g_{[i]}}{\partial \mu_{[j]}}\left(X_{k+1}, 0\right) \end{aligned} \qquad (A.10)$$

Contando con estas matrices, la estimación de la matriz de covarianza para el error de estimación queda determinada por:

$$\tilde{P}_{k+1} = A_{k+1} P_k A_{k+1}^T + \Omega_{k+1} Q \Omega_{k+1}^T \qquad (A.11)$$

La etapa de actualización es idéntica a la del filtro de Kalman lineal:

$$\begin{aligned} X_{k+1} &= \tilde{X}_{k+1} + K_{k+1}\left(Z_{k+1} - \tilde{Z}_{k+1}\right) \\ P_{k+1} &= \left(I - K_{k+1} C_{k+1}\right)\tilde{P}_{k+1} \end{aligned} \qquad (A.12)$$

A diferencia del cálculo realizado en el filtro de Kalman lineal, el cálculo de la ganancia para el filtro de Kalman extendido requiere utilizar la linealización con respecto al ruido en la medición, para obtener correctamente la matriz de covarianza del residuo:

$$\begin{aligned} K_{k+1} &= \tilde{P}_{k+1} C_{k+1}^T S_{k+1}^{-1} \\ S_{k+1} &= C_{k+1} \tilde{P}_{k+1} C_{k+1}^T + \mathrm{M}_{k+1} R \mathrm{M}_{k+1}^T \end{aligned} \qquad (A.13)$$

Al igual que en el filtro de Kalman lineal, la aplicación se realiza en forma iterativa, estimando para cada instante de tiempo el valor del vector de estado y reduciendo así el efecto del ruido tanto en el proceso como en la medición.

ANEXO B. ALGORITMOS GENÉTICOS

Los procesos evolutivos presentes en la naturaleza inspiraron la creación de algoritmos capaces de adaptarse a su ambiente y que pudieran "aprender", de pruebas anteriores, a resolver los problemas de manera más eficiente. Uno de estos son los algoritmos genéticos, cuyas bases están contenidas en la publicación de John H. Holland en 1962. Estos algoritmos han sido capaces de optimizar sistemas altamente complejos y mejorar procesos a través de técnicas de aprendizaje.

Los algoritmos genéticos son procesos de búsqueda dirigida basados en los procesos evolutivos de las especies. Si bien existen muchas variantes de como implementar un algoritmo genético, todos ellos poseen una estructura similar, comenzando con una población inicial de posibles soluciones o cromosomas que se procesan en forma iterativa en cuatro pasos principales llamados evaluación, selección, combinación y reemplazo, los cuales serán detallados posteriormente. La iteración se realiza hasta que se cumple alguna condición de corte impuesta previamente.

El primer paso para implementar un algoritmo genético, consiste en expresar o codificar los elementos del conjunto de búsqueda en cromosomas. Dado que estos algoritmos pueden aplicarse a sistemas que abarcan desde aquellos que trabajan con elementos numéricos (como los valores de entrada en una función) hasta aquellos que lo hacen con características simbólicas (como los patrones en una imagen), la forma en que los datos son codificados varía significativamente. Así, si el problema a resolver es del tipo de trayectoria óptima, donde se busca determinar por que nodos debe pasar un robot para llegar de un punto inicial a otro final, la codificación puede ser "binaria" donde cada cromosoma será un conjunto de 1's y 0's que indiquen cuales nodos están incluidos en la trayectoria. Este tipo de codificación se utiliza principalmente para problemas del tipo "On – Off" en que el objetivo es determinar qué

elementos pertenecen a la solución. Otra forma muy utilizada es la de codificar en base a un vector de números reales; esto se hace en problemas de optimización en que las funciones a evaluar poseen valores numéricos de entrada. También se utiliza en el entrenamiento de redes neuronales [*Yao, 1999*], en que los pesos asociados a cada neurona son valores reales y el sistema completo puede ser codificado en un cromosoma en el cual cada gen es el valor del peso de una neurona. En [*Carrasco, 2003*] se puede observar una forma de codificar las funciones de pertenencia en base a vectores de números reales que permiten modificar su forma para optimizar el control difuso asociado.

Una vez que se conoce la forma de codificar el espacio de búsqueda, se genera una población inicial de cromosomas, los cuales en general se producen en forma aleatoria o en base a algún conocimiento previo del sistema.

Del número de cromosomas que hay en la población inicial depende el requerimiento computacional del algoritmo y la cobertura que este logrará en cada iteración. Cuando el número de cromosomas es elevado es posible evaluar más soluciones en forma simultánea, pero se requiere de una mayor potencia computacional para manejar apropiadamente el problema.

En la figura B.1 se observa el esquema general de un algoritmo genético, que indica los pasos fundamentales del algoritmo, sin ser está la única forma para secuenciar los pasos. A continuación se describen las tareas que se realiza en cada paso:

1. Evaluación o Prueba: cada cromosoma se evalúa mediante una función objetivo llamado “fitness” o “aptitud” que describe la fortaleza de dicho cromosoma como solución del problema estudiado. En general este es un valor numérico que permite comparar todos los cromosomas para determinar cuales son más aptos para resolver el problema.

Figura B.1: Diagrama de Flujo de un Algoritmo Genético.

2. Selección: la selección de los padres de la futura generación de cromosomas puede realizarse de múltiples formas. Si bien intuitivamente se deberían escoger sólo los cromosomas más aptos, en la práctica se utilizan muchos métodos de selección que combinan tanto los más aptos como aquellos que no lo son, esperando que los genes de estos últimos puedan aportar a cromosomas futuros. Entre las formas más comunes de selección se encuentra la "rueda de ruleta", en que a cada cromosoma se le asigna un área en una ruleta, proporcional a la aptitud del cromosoma, y luego se van escogiendo al azar números en la ruleta y usando los cromosomas asociados para ser los padres de la nueva generación. Esto aumenta la probabilidad de reproducirse de aquellos cromosomas que son más aptos, pero no descarta al resto. En [*Jung, 2003*] y [*Carrasco, 2003*] se utiliza un nuevo método de selección

basado en las colonias de abejas, donde sólo la abeja reina (la solución más apta) es aquella seleccionada para producir la próxima generación de soluciones. También en esta etapa se aplica el concepto de "elitismo", que implica copiar en forma idéntica el cromosoma (o cromosomas) más apto en forma directa a la próxima generación, conservando así la mejor solución hasta el momento y permitiendo que el algoritmo pueda ser detenido en cualquier momento asegurando que la mejor solución es mejor o igual a la mejor solución de la población inicial.

3. Combinación o Mezcla: una vez seleccionados los padres, estos deben combinarse de alguna forma para, con alguna probabilidad $\mathbb{P}_{\mathrm{C}}$, crear el nuevo cromosoma de la siguiente generación. Dependiendo de la forma de codificación existen diferentes opciones para la combinación. Por ejemplo, si la codificación es binaria, se puede utilizar el método de "punto de cruce", en que se eligen los primeros genes (o bits) de un padre y se utilizan los últimos del otro para generar el nuevo cromosoma, existiendo así un único punto de cruce en el cromosoma hijo que divide lo que era de un padre de lo que provenía del otro. De igual forma se puede hacer una mezcla con más puntos de cruce, o escogiendo aleatoriamente de cada padre un gen y entregándoselo al hijo. En el caso de los cromosomas codificados con vectores numéricos, el cruce se puede hacer también utilizando puntos de cruce en el vector, o mezclando algebraicamente ambos vectores padre, donde por ejemplo el cromosoma hijo será el promedio de los cromosomas de ambos padres. Durante la combinación también se aplica el concepto de mutación, por el cual, con alguna probabilidad $\mathbb{P}_{\mathrm{M}}$, algunos de los cromosomas sufren variaciones aleatorias en sus genes. Esto permite que el algoritmo no converja a mínimos locales, obteniéndose una mejor exploración del espacio de soluciones. En general los valores utilizados para $\mathbb{P}_{\mathrm{C}}$ van entre el 80% y 95%, mientras que $\mathbb{P}_{\mathrm{M}}$ varía entre un 0.5% y un 1%.

4. Reemplazo: el paso final de la iteración es el reemplazo, mediante el cual se eliminan los cromosomas de la generación anterior y se dejan sólo los hijos. En el caso de utilizar elitismo, en este paso se copian los cromosomas más aptos a la nueva generación. Una vez realizado el reemplazo, la nueva población se somete a prueba para determinar la aptitud de cada cromosoma, y el ciclo vuelve a repetirse.

Uno de los principales problemas de los algoritmos genéticos es que no existe una condición general de término. Dado que no se puede asumir convergencia, es difícil determinar un criterio de finalización para las iteraciones, por lo que generalmente se utiliza el número de generaciones como condición de término. En algunos casos, como la evolución de redes neuronales por ejemplo, se puede utilizar como condición de término el error con el que la red entrega la respuesta esperada, pero esto no asegura que se logre dicha condición de término y por ende el algoritmo puede iterar infinitamente.

Un desarrollo más detallado con todos los alcances y variaciones que presentan los algoritmos genéticos puede encontrarse en [*Goldberg, 1989*], [*Holland, 1992*] y [*Lin, 1996*].

ANEXO C. PUBLICACIONES

CARRASCO, R. y CIPRIANO, A. (2003) Queen-bee Genetic Optimization of an Heuristic Based Fuzzy Control Scheme for a Mobile Robot. *Proc. of the First IEEE Latin-American Conference on Robotics and Automation*, Noviembre 2003, 61-66, Santiago, Chile.

CARRASCO, R. y CIPRIANO, A. (2004) Fuzzy Logic Based Nonlinear Kalman Filter Applied to Mobile Robots Modelling. *Proc. of the IEEE International Conference on Fuzzy Systems*, Julio 2004, 1485-1490, Budapest, Hungría.

CARRASCO, R. y CIPRIANO, A., Layered Low Cost Architecture for Fault Detection and Diagnosis in Cooperative Robots. Presentado para evaluación en la revista *IEEE Transactions on Robotics*.