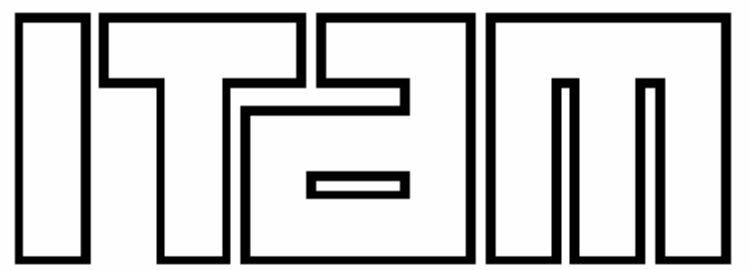
INSTITUTO TECNOLÓGICO AUTÓNOMO DE MÉXICO



SISTEMA DE VISIÓN PARA EL EQUIPO DE ROBOTS AUTÓNOMOS DEL ITAM

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

P R E S E N T A:

LUIS ALFREDO MARTÍNEZ GÓMEZ

MÉXICO, D.F. **2004**

Con fundamento en los artículos 21 y 27 de la Ley Federal del Derecho de Autor y como titular de los derechos moral y patrimonial de la obra titulada “SISTEMA DE VISIÓN PARA EL EQUIPO DE ROBOTS AUTÓNOMOS DEL ITAM” , otorgo de manera gratuita y permanente al Instituto Tecnológico Autónomo de México y a la Biblioteca Raúl Baillères Jr., autorización para que fijen la obra en cualquier medio, incluido el electrónico, y la divulguen entre sus usuarios, profesores, estudiantes o terceras personas, sin que pueda percibir por tal divulgación una contraprestación.

LUIS ALFREDO MARTÍNEZ GÓMEZ

_____

FECHA

_____

FIRMA

Pensar es más interesante que saber, pero menos interesante que mirar.

Goethe

AGRADECIMIENTOS

Quiero agradecer en primer lugar a mis padres que han sido el mejor ejemplo de vida que un hijo puede esperar. Gracias a su amor he alcanzado mis metas. Todos mis logros son suyos.

A mis hermanos. Lili, porque por tu talento, cariño y alegría eres la prueba irrefutable de que algo pequeño puede ser mas grande que el universo mismo. Ceci, porque por tu admirable pasión en todo lo que realizas eres el faro que iluminas a los que venimos detrás de ti. Adrián, porque por tu decisión, sinceridad y amistad te has convertido en cómplice irreemplazable de mis ocurrencias.

A mi abuelita que se nos adelantó hace poco, sé que aunque no estas presente físicamente estas siempre con nosotros.

A mi asesor Dr. Alfredo Weitzenfeld por su invaluable orientación y confianza en esta tesis y en todos los proyectos que hemos realizado en el Laboratorio de Robótica. También quiero agradecer a mis sinodales Dr. Ma. Elena Algorri y Dr. Jesús Savage por su tiempo y comentarios.

A mi familia, mis abuelos (Carlos, Clotilde y Luis), mis tíos, tías, primos, primas, sobrinos y sobrinas.

A todos mis maestros de quienes aprendí mucho mas de lo que enseñaban en sus cursos, gracias por transmitir su conocimiento y exigir siempre lo mejor de sus alumnos. A todas las personas que de una u otra forma hicieron de mi estancia en el ITAM una grata experiencia y en especial a Rosa Urtilla y Rafael Martínez (de los Laboratorios de Ingeniería), Edith, Luzma y Eva.

Quiero agradecer a Fernando Flores que se convirtió en un asesor extraoficial y el mejor crítico de esta tesis. Gracias por los consejos, sugerencias e ideas que moldearon en gran medida el contenido de este trabajo.

A los amigos y amigas que encontré en el ITAM. En especial a los robóticos y ex-robóticos con los que pase momentos memorables: Luis A. Chávez (Ouicho), Nicolás Hakim, Enrique Coronado (Kike), Alex Valadez, Horacio Flores, David Farías (Coco), Francisco Moneo, Víctor Soto, Juan Pablo François (Oso), David Sotelo, Sahima Murguía, Antonio Medrano y Alejandro Chávez (Chemo).
A todos mis amigos: Ana, Andrés, Héctor, Nadia Lazcano, Edgar, Liliana, Jorge, Juan Juan, Juvie, Gaby Sosa, Nadia del Villar, a las tocayas (Karla, Karla Karina y Ana Karina), a los de CANNES (Rodolfo y Pablo), Fernando Nava, Liz, Carlos Aquiles (Cobi), Rogelio, Quauhtli, Pop, Maribel, Gilberto, Maria, Toni, Adrián Puente, Paola, Arlette y Liliana (Pili).

ÍNDICE

1. INTRODUCCIÓN 1

1.1 ROBOCUP..... 1
1.1.1 Ligas de RoboCup..... 1
1.1.2 Arquitectura de un equipo de la Liga de Robots Pequeños..... 4
1.2 PLANTEAMIENTO..... 6
1.3 OBJETIVOS..... 6
1. 4 JUSTIFICACIÓN Y TRABAJOS ANTERIORES..... 6
1. 5 ALCANCE..... 7
1. 6 ORGANIZACIÓN DEL DOCUMENTO..... 7

2. MARCO TEÓRICO 9

2.1 VISIÓN POR COMPUTADORA..... 9
2.2 PROCESAMIENTO DIGITAL DE IMÁGENES..... 10
2.2.1 Adquisición de imagen..... 11
2.2.1.1 Espacios de color..... 13
2.2.1.1.1 RGB..... 13
2.2.1.1.2 YUV..... 14
2.2.1.1.3 HSV..... 15
2.2.2 Pre-procesamiento..... 16
2.2.3 Segmentación..... 18
2.2.4 Representación y descripción..... 19
2.2.5 Reconocimiento e interpretación..... 21
2.3 CALIBRACIÓN DE CÁMARA..... 22
2.3.1 Modelo de la cámara de punta de alfiler..... 22
2.3.2 Distorsión de lente..... 23
2.3.3 Métodos de calibración..... 24
2.3.3.1 Método de Tsai..... 24

3. ANÁLISIS DEL SISTEMA 26

3.1 REQUERIMIENTOS 26
3.1.1 Reglas de la liga de robots pequeños..... 26
3.1.1.1 El campo de juego..... 26

3.1.1.2 La pelota..... 28
3.1.1.3 El número de robots..... 29
3.1.1.4 El equipamiento robótico..... 29
3.1.2 Requerimientos dados por las reglas de la liga de robots pequeños..... 30
3.1.3 Requerimientos de desempeño..... 30
3.2 ACTORES..... 32
3.3 DESCRIPCIÓN FUNCIONAL DEL SISTEMA..... 34
3.3.1 Inicialización del Sistema..... 35
3.3.2 Calibración de Cámara..... 38
3.3.3 Calibración de Objetos..... 39
3.3.4 Procesamiento de Video..... 41
3.3.5 Mejoramiento de Imagen..... 44

4. DISEÑO DEL SISTEMA 45

4.1 ARQUITECTURA..... 45
4.2 MÓDULOS..... 46
4.3 DIAGRAMA DE CLASES..... 50
4.4 DIAGRAMAS DE SECUENCIAS..... 54
4.4.1 Inicialización del Sistema..... 55
4.4.2 Calibración de Cámara..... 56
4.4.3 Calibración de Objetos..... 57
4.4.4 Mejoramiento de Imagen..... 58
4.4.5 Procesamiento de Video..... 58
4.5 ALGORITMOS..... 60
4.5.1 Algoritmo de Segmentación..... 60
4.5.2 Algoritmo de formación de regiones..... 62
4.5.3 Algoritmo de Identificación..... 64
4.5.4 Algoritmo de Localización..... 65

5. IMPLEMENTACIÓN DEL SISTEMA 67

5.1 HERRAMIENTAS..... 67
5.1.1 Hardware..... 67
5.1.2 Software..... 68
5.1.2.1 DirectShow..... 68
5.1.2.2 COM..... 70
5.1.2.3 OpenCV..... 72

5.2 IMPLEMENTACIÓN DEL PROTOTIPO FUNCIONAL..... 73
5.2.1 Filtro de Transformación..... 73
5.2.2 Aplicación..... 76

6. PRUEBAS Y RESULTADOS 84

6.1 VELOCIDAD DE PROCESAMIENTO..... 84
6.2 PORCENTAJE DE PÉRDIDA O LOCALIZACIÓN ERRÓNEA DE LA PELOTA..... 85
6.3 PORCENTAJE DE PÉRDIDA O LOCALIZACIÓN ERRÓNEA DE ROBOTS CONTRARIOS..... 86
6.4 PORCENTAJE DE PÉRDIDA O LOCALIZACIÓN ERRÓNEA DE NUESTROS ROBOTS..... 87
6.5 PORCENTAJE DE IDENTIFICACIÓN ERRÓNEA..... 88
6.6 VARIACIÓN DE LA POSICIÓN..... 89
6.7 VARIACIÓN DE LA ORIENTACIÓN..... 89

7. CONCLUSIONES 91

7.1 CONCLUSIONES PARTICULARES DEL SISTEMA..... 91
7.2 VERSIONES DEL SISTEMA..... 92
7.3 LÍNEAS FUTURAS..... 92
7.2.1 Sistemas de Visión Global..... 93
7.2.1 Sistemas de Visión Local..... 93
7.4 LÍMITACIONES..... 94
7.3 CONCLUSIONES GENERALES..... 95

BIBLIOGRAFÍA 95

APÉNDICE A 100

ÍNDICE DE FIGURAS

Figura 1.1 Liga de Simulación…….. 2
Figura 1.2 Liga de robots medianos…… 2
Figura 1.3 Liga de robots cuadrúpedos……... 3
Figura 1.4 Liga Humanoide……... 3
Figura 1.5 Liga de robots pequeños……..... 4
Figura 1.6 Arquitectura de un equipo de la liga de robots pequeños…… 4
Figura 1.7 Diagrama Lógico 5

Figura 2.1 Etapas del procesamiento digital de imágenes…….. 10
Figura 2.2 Vídeo entrelazado…… 11
Figura 2.3 Cubo de color RGB…….. 13
Figura 2.4 Cono hexagonal de color HSV…….. 15
Figura 2.5 Transformación de valor…… 17
Figura 2.6 Transformación geométrica en un plano…….. 17
Figura 2.7 Segmentación……... 19
Figura 2.8 Código de cadena…… 19
Figura 2.9 Aproximación poligonal……... 20
Figura 2.10 Esqueleto de una región……... 20

Figura 3.1 Campo de juego..... 27
Figura 3.2 Vista superior de un robot..... 30
Figura 3.3 Saltos en el movimiento con diferentes tasas de procesamiento..... 31
Figura 3.4 Actor: Usuario..... 33
Figura 3.5 Actor: Sistema de Inteligencia Artificial..... 33
Figura 3.6 Actor: Archivo de parámetros de cámara..... 33
Figura 3.7 Actor: Archivo de segmentación..... 33
Figura 3.8 Actor: Dispositivo de Captura..... 34
Figura 3.9 Diagrama de Casos de Uso en su vista principal..... 34

Figura 4.1 Arquitectura del Sistema..... 45
Figura 4.2 Diagrama de clases..... 50
Figura 4.3 Inicialización del Sistema: Seleccionar Cámara..... 55
Figura 4.4 Inicialización del Sistema: Carga Archivo de Segmentación..... 55
Figura 4.5 Inicialización del Sistema: Abre Archivo Parámetros de Cámara..... 56
Figura 4.6 Calibración de Cámara..... 56
Figura 4.7 Calibración Objetos..... 57
Figura 4.8 Mejoramiento de Imagen..... 58
Figura 4.9 Procesamiento de Vídeo: Activación/Desactivación..... 58
Figura 4.10 Procesamiento Vídeo: Procesa Cuadro..... 59
Figura 4.11 Método de fusión de corridas..... 63
Figura 4.12 Trayectoria cerrada simple..... 65
Figura 4.13 Ordenamiento por ángulos..... 65

Figura 5.1 Pasos para escribir una aplicación de DirectShow..... 70
Figura 5.2 Filtro de Transformación..... 73
Figura 5.3 Pantalla de Selección de Cámara..... 78
Figura 5.4 Gráfica de procesamiento..... 80
Figura 5.5 Procedimiento de calibración de la cámara..... 82
Figura 5.6 Pantalla Principal del Sistema de Visión..... 82
Figura 5.7 Interface del Sistema IA..... 83

Figura 6.1 Iluminación del campo de juego..... 86
Figura 6.1 Máximos y mínimos de la orientación..... 90

ÍNDICE DE TABLAS

Tabla 2.1 Valores de colores en RGB……. 13
Tabla 2.2 Valores de colores en YCrCb…….. 15
Tabla 2.3 Valores de colores en HSV…….. 16

Tabla 6.1 Mediciones de velocidad de procesamiento..... 85
Tabla 6.2 Resultados prueba localización pelota..... 85
Tabla 6.3 Resultados de prueba localización robots contrarios..... 87
Tabla 6.4 Resultados de prueba localización robots de nuestro equipo..... 88
Tabla 6.5 Resultados de prueba de identificación errónea..... 89

CAPÍTULO 1: INTRODUCCIÓN

1. INTRODUCCIÓN

Este primer capítulo es una introducción al trabajo desarrollado en la tesis, comienza presentando los objetivos y metas de RoboCup, continúa con la descripción de las diferentes ligas que lo integran, profundizando en las características de la liga en donde el Sistema de Visión diseñado tiene aplicación, después se expone el planteamiento, los objetivos que se fijaron, la justificación y trabajos en los que se sustenta, el alcance que delimitó el proyecto, y finalmente la organización del documento.

1.1 ROBOCUP

Robot World Cup (RoboCup) es un proyecto conjunto internacional para promover la investigación en IA, robótica y campos relacionados. Su objetivo es que las innovaciones en tecnología puedan ser aplicadas en la industria y en problemas de importancia social. Su meta final es [ROB 98]: "Para el año 2050, desarrollar un equipo de robots humanoides completamente autónomo que gane un partido de fútbol contra la selección campeona del mundo".

RoboCup escogió al fútbol soccer como problema estándar debido a su atractivo como ambiente multi-agente en tiempo real desde el punto de vista de investigación en IA distribuida. En un partido se tiene a dos equipos competidores con una meta común: ganar el juego. La meta de cada uno de los equipos es incompatible entre si y por tanto el equipo oponente puede ser visto como una fuerza dinámica obstructora. Para ganar, es necesario que los robots sean cooperativos y para lograrlo es indispensable incorporar principios de diseño de agentes autónomos, ejecución y adquisición de estrategias, razonamiento en tiempo real y percepción del ambiente dinámico en el que se desenvuelven [KIT 95].

1.1.1 Ligas de Robocup

RoboCup esta organizado en cinco ligas para poder abarcar áreas de investigación interdisciplinarias que incluyen electrónica, mecatrónica, robótica, ciencias de la computación e IA. Las ligas son: Liga de simulación, Liga de robots medianos, Liga de robots cuadrúpedos, Liga de humanoides y Liga de robots pequeños.

- La liga de simulación consiste en dos equipos oponentes de once agentes virtuales basados en un sistema de computadora que provee una simulación realista de los robots con sensores y acciones.

Cada agente es un proceso separado que envía al servidor de comunicación del simulador instrucciones de movimiento al jugador que representa y recibe información de su estado, la cual incluye observaciones ruidosas y parciales del ambiente que lo rodea.
Una imagen del simulador es mostrada en la Figura 1.1.

Figura 1.1 Liga de Simulación

- La liga de robots medianos tiene dos equipos de cuatro robots que cuentan con todos los sensores y cámara abordo. Los objetos relevantes son identificados por sus colores. La comunicación entre los robots es mediante una red inalámbrica.
  No se permite intervención humana excepto para colocar o remover los robots dentro del campo de juego.
  La Figura 1.2 muestra una escena de un partido de esta liga en RoboCup 2003.

Figura 1.2 Liga de robots medianos

- En la Liga de robots cuadrúpedos los equipos utilizan cuatro robots AIBO de Sony que tienen integrados los sensores y las partes mecánicas. Los robots utilizan colores para localizarse dentro de la cancha e identificar objetos de interés como la pelota y otros robots. La comunicación entre los robots es por medio de una red inalámbrica y no se permite la intervención humana.

A diferencia del ambiente simulado, en esta liga se puede evaluar y comparar la eficiencia de los algoritmos en robots reales ya que se cuenta con una plataforma mecánica y de procesamiento común.

Dos robots AIBO disputando una pelota son mostrados en la Figura 1.3.

Figura 1.3 Liga de robots cuadrúpedos

- La Liga de humanoides presenta robots que muestran las habilidades básicas de un jugador de fútbol como caminar, pegarle a la pelota y defender una portería. Se utilizan colores para distinguir objetos y la intervención humana esta permitida ya que algunos robots son tele-operados.

  La Figura 1.4 muestra un robot humanoide pegándole a una pelota.

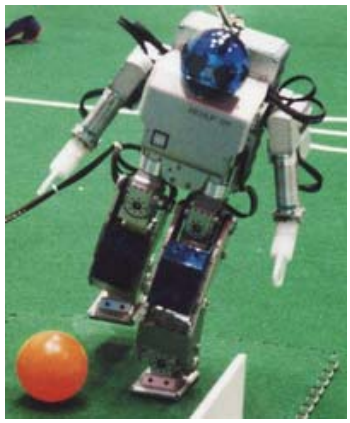


Figura 1.4 Liga Humanoide

- En la Liga de robots pequeños juegan equipos de 5 robots. Los robots deben tener un diámetro máximo de 18 cm y una altura máxima de 15 cm. Pueden utilizar visión global o visión local.

  La visión global utiliza una o varias cámaras colocadas por encima del campo de juego para proveer retroalimentación de las posiciones de la pelota y jugadores a una computadora externa. Los robots son distinguidos por patrones codificados de colores colocados en su parte superior.

En los sistemas de visión local los robots cuentan con una cámara integrada y el procesamiento de video es hecho a bordo del robot. Cada robot debe ser capaz de ubicarse a si mismo, a los robots contrarios, a sus compañeros y a la pelota.
La Figura 1.5 muestra una imagen de la liga de robots pequeños.

Figura 1.5 Liga de robots pequeños

El Sistema de Visión desarrollado en esta tesis es un sistema de visión global diseñado para la Liga de robots pequeños y en el siguiente apartado se describe las características y arquitectura de un equipo de esta liga.

1.1.2 Arquitectura de un equipo de la Liga de Robots Pequeños

En general, la arquitectura de un equipo de esta liga esta integrada por tres componentes principales: el ***Sistema de Visión*** **(SV)**, el Sistema de Inteligencia Artificial y los robots. En la Figura 1.6 se muestra un esquema de la arquitectura con todos sus componentes.

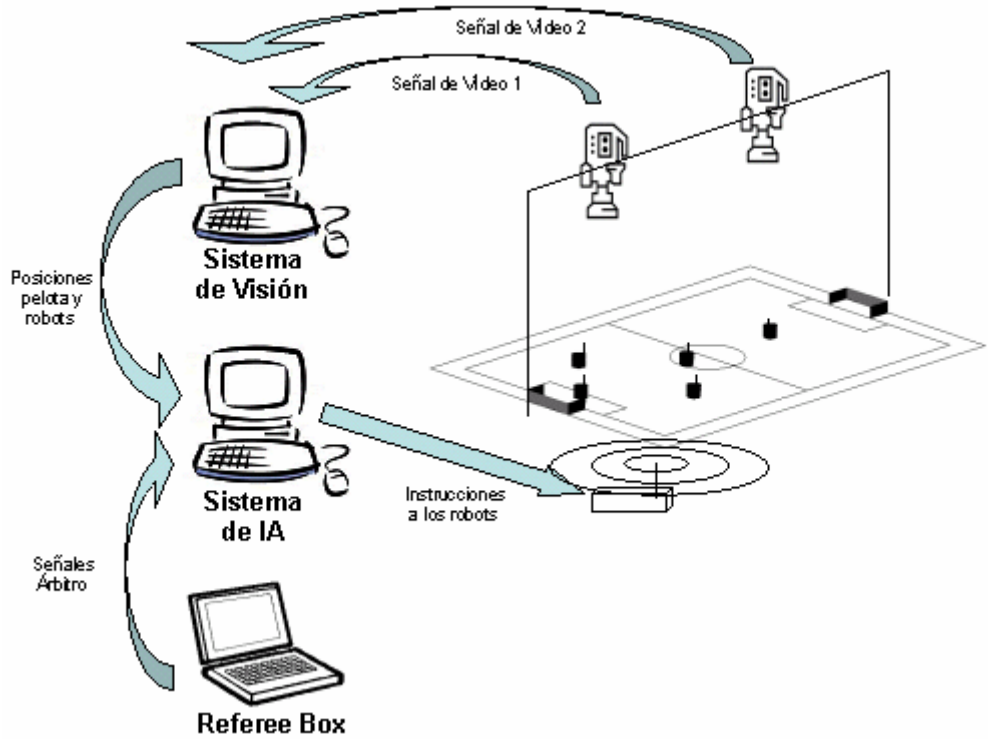


Figura 1.6 Arquitectura de un equipo de la liga de robots pequeños

La principal característica de la arquitectura es que el procesamiento de IA y visión no se hace en los robots sino en una o varias computadoras externas, esto permite eliminar las restricciones de procesamiento inherentes a robots pequeños con capacidades limitadas.

Las señales de video de las cámaras colocadas sobre el campo de juego son capturadas y procesadas por el SV, éste se encarga de calcular la posición de la pelota y de todos los robots dentro del campo de juego; también calcula la orientación de los robots de su equipo y transmite toda la información al Sistema de IA.
El Sistema de IA utiliza esa información para tomar decisiones estratégicas. El accionar del equipo está basado en un conjunto de roles (portero, defensa, delanteros, etc.) que exhiben un comportamiento de acuerdo al estado del juego. Para evitar las colisiones con los robots contrarios se utiliza un módulo de evasión de obstáculos Las decisiones son transformadas a instrucciones de movimientos o acciones mecánicas que son enviadas a los robots por medio de un enlace inalámbrico. Los robots reciben estos comandos y los ejecutan. Cada robot cuenta con motores para poder desplazarse por el campo de juego, un mecanismo para patear la pelota y otro para controlarla. El árbitro del partido utiliza una computadora (Referee Box) para informar sus decisiones al Sistema de IA, de esta manera se puede indicar el inicio o fin de los tiempos de un partido, marcar la anotación de un gol y las infracciones cometidas. La Figura 1.7 muestra el diagrama lógico de la arquitectura.

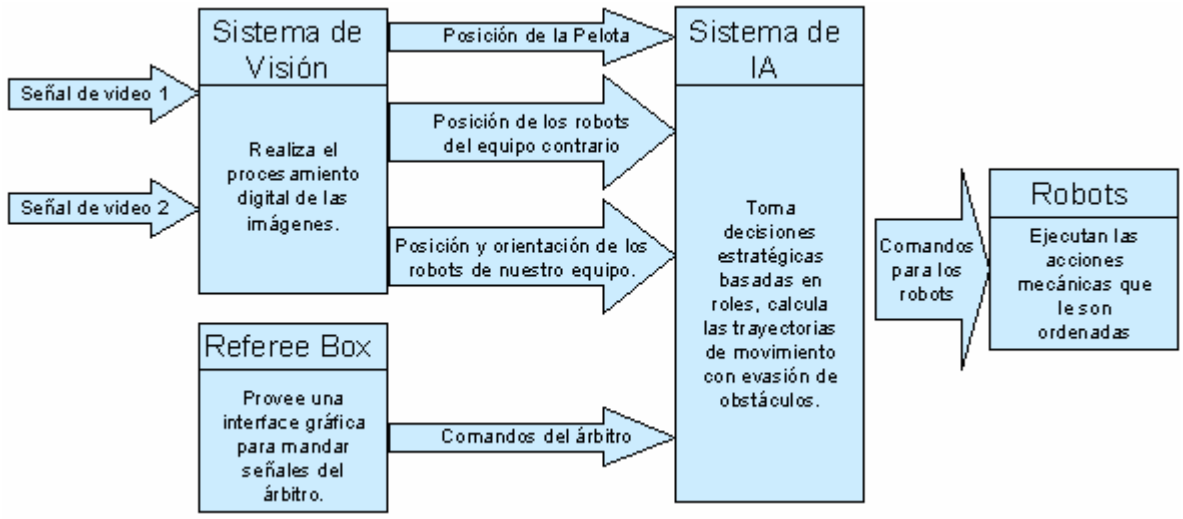


Figura 1.7 Diagrama Lógico

Todo el sistema es completamente autónomo, no existe intervención humana salvo para colocar y remover los robots del campo de juego.

1.2 PLANTEAMIENTO

El SV es el responsable de proveer retroalimentación al Sistema de IA de la posición, orientación y acciones de todos los robots dentro del campo de juego. También debe localizar a la pelota y transmitir la información.
Debido a que en un partido de la liga de robots pequeños los objetos alcanzan fácilmente velocidades superiores a 2m/s, el SV debe ser un sistema en tiempo real para poder seguir con una precisión aceptable los objetos dentro del campo de juego.

Para facilitar esta tarea, las reglas de la liga especifican un conjunto de colores que distinguen a los objetos dentro del campo de juego, de esta manera, la pelota es el único objeto de color naranja, los robots de un equipo tienen una serie de parches de colores pegados en su parte superior que ayudan a distinguir de qué equipo es el robot e identificar de qué jugador se trata.
Por lo tanto, el SV debe estar basado en la detección de colores para encontrar e identificar a los robots y a la pelota.

1.3 OBJETIVOS

Los objetivos que se fijaron para el trabajo realizado en esta tesis son:

- Desarrollar un Sistema de Visión para el Laboratorio de Robótica para su uso en competencias nacionales e internacionales del equipo de robots pequeños del ITAM.

- Proveer una arquitectura tecnológica sobre la cual se basen futuros desarrollos en visión por computadora, identificando y señalando líneas de trabajo futuras.

1.4 JUSTIFICACIÓN Y TRABAJOS ANTERIORES

Ésta tesis forma parte del proyecto que el ITAM, por medio del Laboratorio de Robótica, inició con el apoyo de la División Académica de Ingeniería (DAI) para sumarse a la iniciativa de RoboCup. Nuestro equipo, Eagle Knights, ganó el tercer lugar en el RoboCup American Open 2003 y el segundo lugar en el RoboCup USOpen 2004, convirtiendo al ITAM en la primera universidad de México en desarrollar e implementar su propio equipo de robots para participar en las competencias de la liga de robots pequeños.

Por otra parte, otros sistemas de visión han sido propuestos e implementados para RoboCup por algunas universidades de distintos países del mundo, sin embargo, estos sistemas se basan en dispositivos de hardware especializados [KAR 99] que son difíciles de conseguir y

tienen alto costo. Además, no son software libre ni están disponibles al público, o bien si lo están, son librerías de bajo nivel para procesamiento digital de imágenes en tiempo real [BRU 00] que requieren de soluciones adicionales para su utilización dentro del proyecto RoboCup del ITAM. Por lo tanto se decidió realizar un sistema propio, que se integrara de manera sencilla al sistema de IA desarrollado en el Laboratorio de Robótica y que permitiera desarrollos futuros en visión por computadora.

En el Laboratorio de CANNES del ITAM, dentro del proyecto MIRO en Internet, se han desarrollado prototipos de visión por computadora para el control de robots móviles en sistemas distribuidos neurobiológicos [WEI 03], se utilizan colores para diferenciar los depredadores de las presas, cada objeto azul detectado (presas) genera un campo de atracción representado como un estímulo positivo, mientras que los objetos rojos (depredadores) generan un campo repulsivo como estímulo negativo logrando así que el robot persiga a sus presas y huya de sus depredadores.

1.5 ALCANCE

Se debe analizar, diseñar, desarrollar, implementar y probar el SV; el cual debe:

- Capturar el video proveniente de una o varias cámaras en tiempo real.
- Reconocer el conjunto de colores que distinguen a los objetos de interés (pelota y robots).
- Ubicar cada objeto dentro de un plano de referencia.
- Identificar a los robots de nuestro equipo.
- Transmitir la información al sistema de IA.
- Adaptarse a diferentes campos de juegos con distintos materiales y condiciones de luz.

1.6 ORGANIZACIÓN DEL DOCUMENTO

En el primer capítulo se explica el contexto y planteamiento del problema sobre el cual se desarrolla el trabajo de tesis, los objetivos planteados para este proyecto, los trabajos previos que lo soportan y el alcance del mismo.

En el capítulo dos se presenta el marco teórico necesario para estructurar los conocimientos básicos que se requieren para entender los conceptos de visión por computadora, el procesamiento digital de imágenes y calibración de cámara.

En el capítulo tres se describen los requerimientos funcionales del sistema, basándose en las reglas de la liga de robots pequeños y en las especificaciones técnicas de la arquitectura de nuestro equipo.

En el capítulo cuatro se presenta el diseño del sistema con los módulos que lo componen, la estructura del procesamiento del video y la arquitectura correspondiente.

En el capítulo cinco se muestra la implementación del prototipo funcional y las herramientas de hardware y software utilizadas.

En el capítulo seis se presentan las pruebas y los resultados obtenidos del SV desarrollado en esta tesis.

En el capítulo siete se detallan las conclusiones obtenidas a lo largo del proyecto y se incluyen reflexiones y lineamientos para trabajos futuros.

2. MARCO TEÓRICO

En este capítulo se presenta el marco teórico necesario para estructurar los conocimientos básicos que se requieren para entender los conceptos de visión por computadora, el procesamiento digital de imágenes y calibración de cámara.

2.1 VISIÓN POR COMPUTADORA

Los humanos obtenemos la mayor parte de nuestra entrada sensorial a través de nuestro sistema visual, y se ha hecho un enorme esfuerzo para mejorar de forma artificial este sentido. Lentes, binoculares, telescopios, radares, sensores infrarrojos, todos funcionan para mejorar nuestra vista del mundo y del universo. Por lo tanto, era un paso lógico utilizar computadoras para incrementar la capacidad de este sentido [PAR 97].

La visión por computadora tiene como objetivo duplicar el efecto de la visión humana percibiendo y entendiendo una imagen por medios electrónicos. Utiliza los resultados y métodos de las matemáticas, reconocimiento de patrones, inteligencia artificial, ciencias de la computación, electrónica y otras disciplinas científicas para conseguir su objetivo.
En visión por computadora se distinguen dos niveles: procesamiento de imágenes de bajo nivel y entendimiento de imágenes de alto nivel [SON 99].

Los métodos de bajo nivel utilizan muy poco conocimiento del contenido de la imagen. Algunos ejemplos son los métodos de compresión de imágenes, filtros para eliminar ruido y extracción de bordes. El procesamiento de bajo nivel recibe y genera como salida datos en matrices que representan el brillo o color en una posición de la imagen.

El procesamiento de alto nivel está basado en el conocimiento y es común utilizar métodos de IA. La visión por computadora de alto nivel trata de imitar la cognición humana y la toma de decisiones basada en información contenida en la imagen.
La visión de alto nivel comienza con un modelo formal del mundo, después, la realidad percibida en forma de imágenes digitalizadas es comparada con ese modelo. Al hallar diferencias se hacen comparaciones parciales y se utilizan los métodos de bajo nivel para extraer información. Este proceso es repetido de forma iterativa y el entendimiento de la imagen es el resultado de la interacción entre los procesos de alto nivel y de bajo nivel. El ciclo de retroalimentación toma los resultados parciales del procesamiento de alto nivel y crea tareas para los métodos de bajo nivel y así, eventualmente, se converge a la meta global.

Las técnicas de visión por computadora de bajo y alto nivel se traslapan en gran medida con las del procesamiento digital de imágenes. La siguiente sección profundiza en el tema.

2.2 PROCESAMIENTO DÍGITAL DE IMÁGENES

En esta sección se describen los pasos fundamentales para realizar el procesamiento digital de imágenes. La Figura 2.1 muestra un esquema de la secuencia de estos pasos.

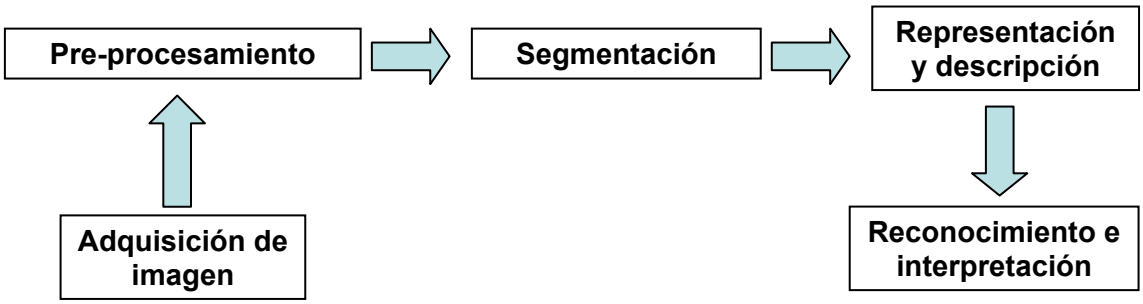


Figura 2.1 Etapas del procesamiento digital de imágenes

- El primer paso es la adquisición de la imagen; para lograrlo, se requiere de una cámara y la capacidad de digitalizar la señal producida por ella.
- Después de que una imagen digital ha sido obtenida, el siguiente paso es el pre-procesamiento de la imagen. El objetivo de este paso es mejorar la imagen para aumentar la probabilidad de éxito en los siguientes pasos. Algunos ejemplos de métodos utilizados en esta etapa son aumentar el contraste, remover el ruido, etc.
- La siguiente etapa es la segmentación en donde se realiza una partición de la imagen en sus partes constituyentes u objetos.
- Los datos obtenidos en la etapa de segmentación deben ser convertidos a una estructura apropiada para su posterior procesamiento. La descripción, también llamada selección de características, extrae información cuantitativa o características relevantes para diferenciar un objeto de los demás.
- La última etapa es el reconocimiento e interpretación. El reconocimiento es el proceso que asigna un identificador a un objeto basado en la información provista por sus descriptores. La interpretación es asignar un significado al objeto reconocido. En general, los sistemas de procesamiento que incluyen reconocimiento e interpretación están asociados a aplicaciones de análisis de imágenes cuyo objetivo es la extracción automática de información [GON 93].

En las siguientes secciones se describe con más detalle cada una de las etapas del procesamiento digital de imágenes.

2.2.1 Adquisición de imagen

Tres elementos son necesarios para adquirir imágenes digitales de video. El primero es un dispositivo físico que sea sensible a la banda visible del espectro electromagnético y que produzca una señal eléctrica proporcional al nivel de energía percibido. El segundo elemento es un cable por donde la señal eléctrica sea transmitida, y el tercero es un digitalizador, encargado de convertir la señal eléctrica del dispositivo físico a una forma digital [GON 93] aunque en la actualidad existen cámaras que producen una señal digital (IEEE 1394) que puede ser transmitida a la computadora sin necesidad de un digitalizador.

Si el dispositivo de captura es una cámara analógica, entonces la señal eléctrica que produce puede ser uno de los tres estándares de video mas comunes: NTSC, PAL o SECAM. Dado que NTSC es el estándar en México es el que se utilizará en este trabajo.
PAL (Phase Alternation Line) se usa en Europa. SECAM (Système Electronique Couleur avec Mémoire) es usado exclusivamente en Francia y en la ex-Unión Soviética. PAL y SECAM fueron diseñados para evitar problemas de distorsión de color que pueden ocurrir en NTSC, operan en sistemas de 625 líneas por cuadro y 25 cuadros por segundo.
NTSC (National Television System Committee) es un estándar de video entrelazado, esto quiere decir que un cuadro de video es generado por dos campos. El campo puede ser par o non y al ser desplegados secuencialmente forman la imagen. La Figura 2.2 muestra este proceso.

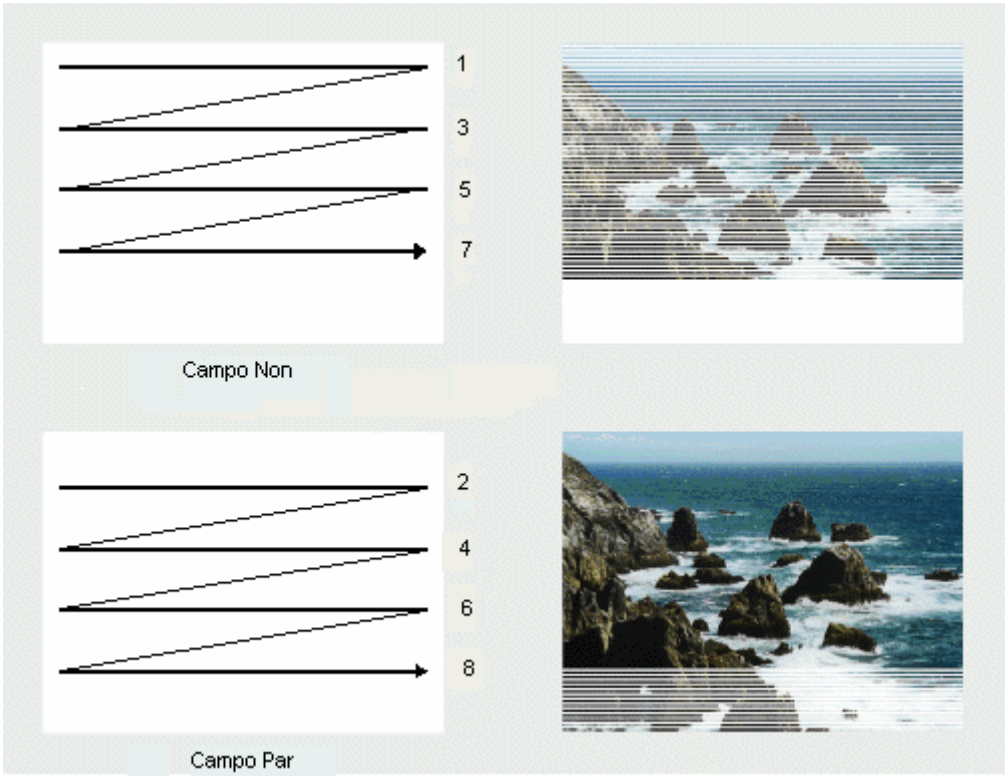


Figura 2.2 Vídeo entrelazado

NTSC consta de 525 líneas horizontales de información de video actualizadas 30 veces por segundo. Entonces, a cada campo le toma la mitad de ese tiempo para desplegarse, es decir 1/60 de segundo.

Para la transmisión de la señal de video es necesario un cable. Existen muchos tipos, pero los mas populares son el de S-Video y el video compuesto.

El video compuesto combina ambos elementos del video (color y luminosidad) en una sola señal que es transmitida en un solo alambre. La calidad de la imagen es generalmente buena pero depende en gran medida del alambre usado. Sin embargo, se presentan fenómenos como el “arrastre de puntos” que ocurre cuando un punto o píxel se mueve de una línea a otra y el “desvanecimiento de color” donde el color se va convirtiendo en otro de manera gradual [LAW 01].
En S-Video los datos de color y luminosidad son separados y enviados en alambres diferentes lo que asegura que las señales no interfieran una con otra. Cada alambre tiene su propio blindaje para prevenir interferencia externa. La imagen producida es muy limpia y notablemente mejor que la imagen generada con el video compuesto. El “arrastre de puntos” es virtualmente eliminado y el “desvanecimiento de color” es reducido [LAW 01].

El último elemento para la adquisición de la imagen son los digitalizadores, también conocidos como tarjetas de captura de video, éstos permiten convertir la amplitud de la señal analógica de video a su representación digital.

Una imagen capturada es un conjunto de valores numéricos. La resolución es la medida básica de cuanta información es visible en una imagen. Generalmente es descrita en términos de “h”x“v”. Donde “h” es la resolución horizontal y “v” es la resolución vertical. Entre mas grandes sean éstos números es mejor debido a que la imagen tendrá un mayor detalle. Existe una relación que guarda el ancho de la imagen con su altura conocida como cociente de aspecto, este puede ser 4:3 o 16:9. Es decir, cuando se tiene un cociente de aspecto de 4:3 significa que la resolución vertical es ¾ la resolución horizontal, por ejemplo, con una resolución horizontal de 640 la resolución vertical es ¾ de 640 = 640/4 * 3 = 480.
De esta forma algunas resoluciones posibles son: 1280x960, 640x480, 320*240, 160x120,etc.
Cada imagen esta compuesta de elementos individuales conocidos como píxeles. Si se tiene una resolución de 640 x 480 en realidad se dice que se tienen 480 líneas horizontales y 640 píxeles por cada línea, entonces, la imagen tendrá un total de 307,200 píxeles.

Existen distintos formatos para la representación de los valores del color de forma digital y estos son conocidos de manera general como espacios de color.

2.2.1.1 Espacios de color

Un espacio de color es un modelo para la representación matemática de un conjunto de colores [KEI 01]. Se les llama espacios de color porque son representaciones en 2,3 o 4 dimensiones. Cada punto dentro del espacio corresponde a un color. Los tres espacios de color mas populares son el RGB, YUV y HSI.

2.2.1.1.1 RGB

Este espacio de color tiene tres componentes: rojo, verde y azul que corresponden a los tres colores primarios aditivos (los componentes individuales son mezclados para formar el color deseado) y son representados en un sistema cartesiano tridimensional como se muestra en la Figura 2.3

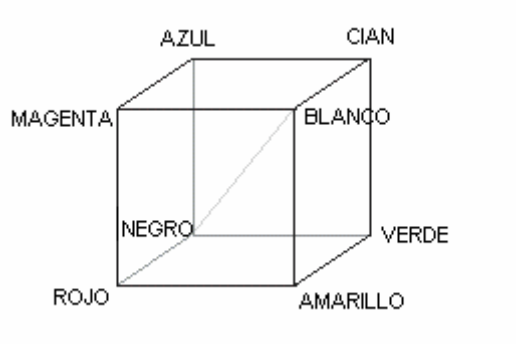


Figura 2.3 Cubo de color RGB

La Tabla 2.1 muestra los valores en RGB para cada uno de los vértices del cubo de color RGB.

	Rango	Blanco	Amarillo	Cian	Verde	Magenta	Rojo	Azul	Negro
R	0 a 255	255	255	0	0	255	255	0	0
G	0 a 255	255	255	255	255	0	0	0	0
B	0 a 255	255	0	255	0	255	0	255	0

Tabla 2.1 Valores de colores en RGB

Este espacio de color es generalmente utilizado en procesamiento de imágenes, pero presenta una desventaja para aplicaciones de visión por computadora debido a que no se puede separar la luminosidad del color, es decir, se necesita de los tres componentes para poder obtener la luminosidad, haciendo difícil la adaptación de la aplicación a condiciones de luz variable.

La principal ventaja de este espacio de color es que es muy utilizado en gráficas por computadora. La mayoría de los monitores y tarjetas de video trabajan con este modelo por lo que resulta sumamente práctico su empleo para aprovechar rutinas de software existentes y optimización por hardware.

2.2.1.1.2 YUV

Para hacer compatible el cambio de televisión en blanco y negro a televisión a color se creo el espacio de color YUV. Éste espacio de color es utilizado por los estándares de video PAL, NTSC y SECAM [KEI 01].

El sistema de televisión en blanco y negro utiliza solamente la información de luminosidad (Y). La información de color (U y V) fue agregada de manera tal que los televisores en blanco y negro siguieran siendo capaces de desplegar su imagen sin interferencia. Los televisores a color codifican la información adicional de los componentes U y V para desplegar una imagen a colores.

La luminosidad (Y) es la parte de la señal de video relacionada a la cantidad de brillo en cualquier punto de la imagen. Si la luminosidad es alta, la imagen es brillante, y si es baja, la imagen es obscura, es decir, el nivel de gris en la imagen.
La cromaticidad (U y V) se define como la diferencia entre un color y una referencia blanca con la misma cantidad de luminosidad [LI 00]. U se obtiene al restar la luminosidad al componente azul (U = B-Y) y V al componente rojo (V = R-Y) donde B es el componente azul y R el componente rojo.
Los términos “YUV” y “YCrCb” se refieren al mismo espacio de color. Y aunque no es parte de ninguna especificación, la literatura utiliza el término “YUV” para indicar la representación analógica de la señal de video, mientras que “YCrCb” es utilizado para la representación digital.

La conversión entre RGB y YCrCb esta dada por la siguiente matriz [KEI 01]:

$$\begin{bmatrix} Y \\ Cr \\ Cb \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ \left(\frac{0.701}{2\times 0.701}\times\frac{224}{219}\right) & \left(-\frac{0.587}{2\times 0.701}\times\frac{224}{219}\right) & \left(\frac{0.114}{2\times 0.701}\times\frac{224}{219}\right) \\ \left(-\frac{0.299}{2\times 0.886}\times\frac{224}{219}\right) & \left(-\frac{0.587}{2\times 0.886}\times\frac{224}{219}\right) & \left(\frac{0.886}{2\times 0.886}\times\frac{224}{219}\right) \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

La Tabla 2.2 muestra los valores en YCbCr para los colores mas representativos y su rango de valores.

	Rango	Blanco	Amarillo	Cian	Verde	Magenta	Rojo	Azul	Negro
Y	16 a 235	180	162	131	112	84	65	35	16
Cb	16 a 240	128	44	156	72	184	100	212	128
Cr	16 a 240	128	142	44	58	198	212	114	128

Tabla 2.2 Valores de colores en YCbCr

El espacio de color YUV al tener codificada la cromaticidad en dos componentes (U y V) y luminosidad en un tercero (Y) puede ser útil para aplicaciones de visión porque pueden adaptarse a condiciones de luz variable al ignorar el componente de luminosidad sin tener impacto en la detección de colores.

2.2.1.1.3 HSV

El espacio de color HSV fue creado para poder manejar el color de manera intuitiva y fue diseñado en base a la manera en que los humanos percibimos e interpretamos el color [KEI 01].

El matiz (H) es lo que normalmente pensamos como color. La saturación (S) es la cantidad de blanco que es mezclado con el color. Una saturación de cero indica que no hay matiz y el color esta dentro de la escala de grises. La componente V indica la luminosidad.

La Figura 2.4 muestra el cono hexagonal del modelo de color HSV.

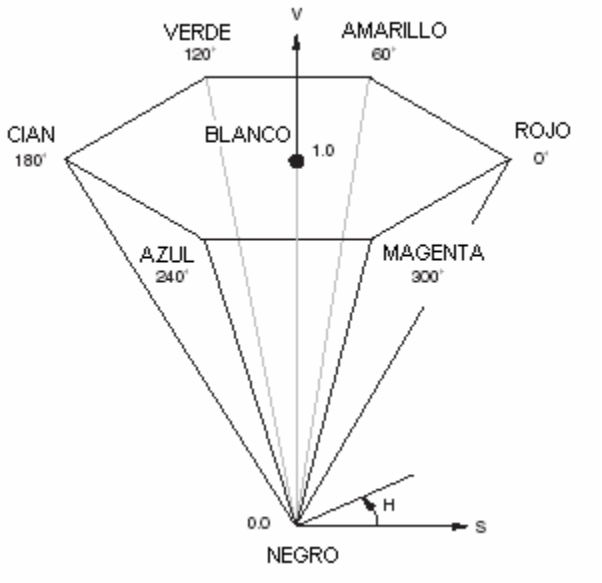


Figura 2.4 Cono hexagonal de color HSV

La tapa del cono corresponde al valor de luminosidad mas intenso (V =1). La punta del cono es el color negro (sin luminosidad). El matiz (H) es el ángulo alrededor del eje de luminosidad (V), por ejemplo un matiz con valor de 120º corresponde al verde. La saturación (S) tiene un rango

de 0 a 1. El punto S =0, V =1 es el color blanco. Cualquier color con V =1 y S =1 es un pigmento puro. Añadir blanco equivale a disminuir S sin cambiar V y añadir negro es disminuir V sin cambiar S.

La Tabla 2.3 muestra los valores en HSV para algunos colores.

	Rango	**Blanco**	**Amarillo**	**Cian**	**Verde**	**Magenta**	**Rojo**	**Azul**	**Negro**
H	0º a 360º	-	60º	180º	120º	300º	0º	240º	-
S	0 a 1	0	1	1	1	1	1	1	0
V	0 a 1	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0

Tabla 2.3 Valores de colores en HSV

El espacio de color HSV presenta las mismas ventajas que el espacio de color YUV para aplicaciones de visión debido a la fácil separación entre luminosidad y cromaticidad, sin embargo, al no estar soportado directamente por hardware el manejo de este espacio de color puede resultar en un procesamiento muy costoso en tiempo.

2.2.2 Pre-procesamiento

El pre-procesamiento es el segundo paso del procesamiento digital de imágenes. Se le da este nombre a las operaciones en las imágenes al más bajo nivel de abstracción, es decir, la entrada y salida de este paso son imágenes con valores de intensidad similares a los datos originales capturados por el digitalizador.

El pre-procesamiento de imágenes no incrementa la cantidad de información de una imagen, sin embargo, es muy útil porque ayuda a suprimir información que no es relevante para los objetivos particulares de análisis en un caso dado. Por lo tanto, el objetivo del pre-procesamiento es una mejora de los datos de la imagen que suprima las distorsiones indeseadas e incremente las características relevantes para su posterior procesamiento [SON 99]. Algunos métodos de pre-procesamiento son las transformaciones de intensidad y las transformaciones geométricas.

- En una transformación de intensidad se modifica el valor del píxel sin importar su posición dentro de la imagen, algunas transformaciones de intensidad incluyen modificar el brillo, contraste, matiz, saturación, gamma, etc.
  El brillo es la cantidad de luz que emite la imagen [LAU 04].
  El contraste se refiere a que tan lejos están los valores mas blancos de los valores mas negros en una imagen. Si el valor mas blanco esta muy lejos del valor mas negro se

dice que la imagen tiene un alto contraste. Con un alto contraste la imagen es muy nítida. Si los dos valores están muy cerca uno de otro se dice que la imagen tiene un pobre o bajo contraste y no se puede distinguir la diferencia entre blanco y negro provocando que la imagen aparezca grisácea [KEI 01].

La saturación es la cantidad de color presente. Es decir, cuánto pigmento es usado para hacer el color. Entre menos pigmento, el color es menos saturado, como si se añadiera blanco al color puro [KEI 01].

El matiz es la longitud de onda del color. Esto significa que el matiz es un término empleado para el color base (rojo, amarillo, etc.). El matiz es independiente de la intensidad y la saturación del color. Por ejemplo, un matiz rojo puede aparecer café cuando hay baja saturación, rojo claro cuando hay mucha saturación y rosa con un alto nivel de brillo y sin embargo los tres tienen el mismo valor de matiz [KEI 01].

El resultado de una transformación de valor en una imagen es mostrado en la Figura 2.5.

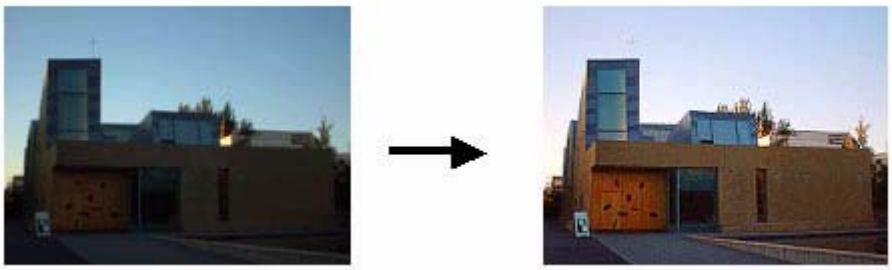


Figura 2.5 Transformación de valor

- Las transformaciones geométricas son comunes en gráficas por computadora y también son utilizadas en análisis de imágenes. Permiten la eliminación de distorsiones geométricas que ocurren cuando una imagen es capturada, como por ejemplo las provocadas por los lentes de la cámara.

  Una transformación geométrica es una función **T** que coloca al píxel *(x , y)* en una nueva posición *(x ', y')*. La figura 2.6 ilustra el concepto.

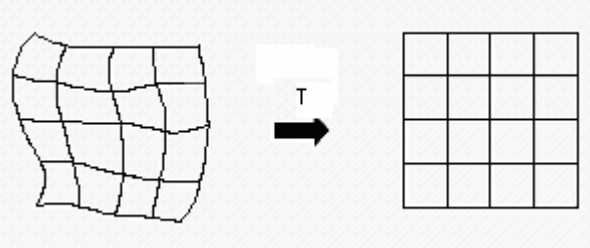


Figura 2.6 Transformación geométrica en un plano

2.2.3 Segmentación

El tercer paso del procesamiento digital de imágenes es la segmentación que subdivide a una imagen en sus partes u objetos constituyentes. El nivel al que se lleva esta subdivisión depende del problema que se este resolviendo, esto es, la segmentación debe detenerse cuando los objetos de interés para una aplicación han sido aislados [GON 93].

En general, la segmentación automática es una de las tareas más difíciles en el procesamiento de imágenes. Este paso del proceso determina el eventual éxito o fracaso del análisis de la imagen.

Los algoritmos de segmentación de imágenes tienen tres formas comunes: métodos basados en bordes, técnicas basadas en regiones y técnicas de umbral [CHA 96].

- Los métodos basados en bordes se centran en la detección de contornos. Delimitan el borde de un objeto y segmentan los píxeles dentro del contorno como pertenecientes a ese objeto. Su debilidad consiste en conectar contornos separados o incompletos, lo que los hace susceptibles a fallas.

- Las técnicas basadas en regiones, usualmente operan de la siguiente forma: la imagen es dividida en regiones agrupando píxeles vecinos con niveles de intensidad similares. Las regiones adyacentes son unidas bajo cierto criterio que involucra la homogeneidad y agudeza de las fronteras de la región. Un criterio muy estricto provoca fragmentación, un criterio poco estricto ocasiona uniones indeseadas.

- Las técnicas de umbral segmentan la imagen píxel por píxel, es decir, no toman en consideración el valor de los píxeles vecinos para el proceso. Si el valor de un píxel esta dentro del rango de valores especificado para un objeto el píxel es segmentado. Son efectivas cuando los objetos y el fondo de la imagen tienen rangos de valores diferentes y existe un contraste marcado entre ellos.
  Como la información de los píxeles vecinos es ignorada, las fronteras de regiones borrosas pueden ocasionar problemas.

La elección de una técnica de segmentación está determinada por las características particulares del problema a resolver. La salida de esta etapa son los valores de los píxeles que forman la frontera de una región o bien la región misma.

La Figura 2.7 muestra la imagen original y segmentada de la misma escena.

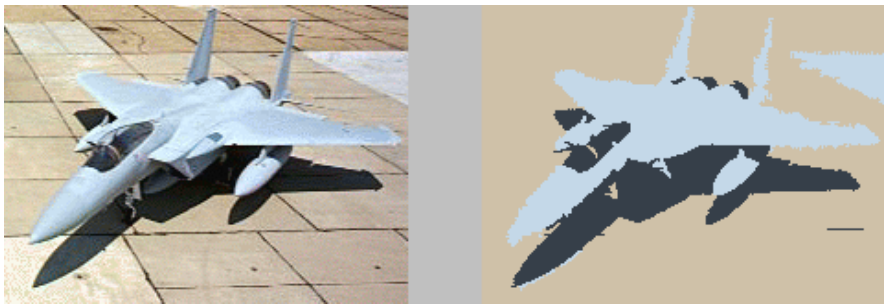


Figura 2.7 Segmentación

2.2.4 Representación y descripción

Después que una imagen ha sido segmentada en regiones, el conjunto de píxeles segmentados son usualmente representados y descritos en una estructura adecuada para su posterior procesamiento. La representación es conveniente porque en lugar de procesar una gran cantidad de datos se procesa una estructura sencilla que contiene la misma información.
Se puede representar la región en términos de sus características externas (frontera) o bien, representarla en términos de sus características internas (los píxeles que componen a la región). Una vez que se ha escogido se debe describir la región de acuerdo a la decisión tomada. Por ejemplo, una región puede ser representada por su frontera y describirla por características como su longitud, la orientación de la línea recta entre sus puntos extremos, etc. Generalmente, una representación externa es escogida cuando el enfoque principal esta en las características de la figura. Una representación interna es seleccionada cuando interesan las propiedades de los píxeles.

Algunos esquemas de representación son: códigos de cadena, aproximaciones poligonales y esqueleto de una región:

- Los códigos de cadena son usados para representar una frontera mediante una secuencia de segmentos conectados de línea recta. La dirección de cada segmento es codificada utilizando un esquema de numeración. La Figura 2.8 ilustra el esquema.

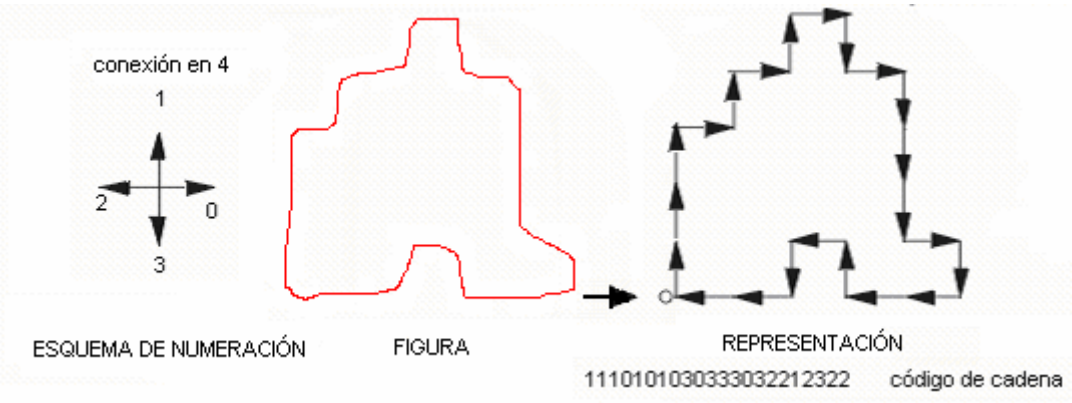


Figura 2.8 Código de cadena

- La meta de una aproximación poligonal es capturar la esencia de la figura con el mínimo posible de segmentos poligonales. Se basa en el principio de que un borde de píxeles consecutivos puede ser aproximado con cierto grado de precisión por un polígono. El problema consiste en encontrar los puntos de las esquinas o puntos de corte que lleven a una buena aproximación. La Figura 2.9 muestra una aproximación poligonal.

Figura 2.9 Aproximación poligonal

- Un esqueleto de región es la representación de la forma estructural de la figura reducida a su gráfica. Una figura en dos dimensiones puede ser representada por sus ejes medios (esqueleto) y las distancias de los ejes a su borde. La Figura 2.10 muestra una región con su esqueleto.

Figura 2.10 Esqueleto de una región

Para describir a los esquemas de representación se pueden utilizar algunos descriptores sencillos como son:

- Perímetro. Es el número de píxeles alrededor del contorno.
- Momentos. Son propiedades numéricas que se pueden obtener de una determinada imagen. Para una función discreta el momento de orden (p+q) se define como:

$$M(p,q) = \sum_x \sum_y x^p y^q f(x,y)$$

- Área. El momento simple de orden 0 representa el área de la figura en imágenes binarias. Es la suma de los valores de todos los píxeles.

$$M(0,0) = \sum_x \sum_y f(x,y)$$

- El centroide (X,Y) es el centro de masa de la figura y se obtiene por medio de los momentos de orden 1 ($M(1,0), M(0,1)$) y de orden 0 (área):

$$M(1,0) = \sum_x \sum_y xf(x,y)$$
$$M(0,1) = \sum_x \sum_y yf(x,y)$$
$$X = M(1,0) \Big/ M(0,0)$$
$$Y = M(0,1) \Big/ M(0,0)$$

2.2.5 Reconocimiento e interpretación

El último paso del procesamiento digital de imágenes es el reconocimiento e interpretación que consiste en descubrir, identificar y entender patrones que son relevantes en el desempeño de una tarea dada [GON 93].
Un patrón es una descripción cuantitativa o estructural de un objeto o entidad en una imagen. En general, un patrón está formado por uno o más descriptores (también conocidos como características) y son agrupados en clases. Una clase es una familia de patrones que comparten propiedades comunes. El reconocimiento de patrones comprende técnicas para asignar cada patrón a su respectiva clase de manera automática y sin intervención humana [RIP 96].

Existen tres principales tipos de patrones, los vectores (para descripciones cuantitativas), las secuencias y los árboles (para descripciones estructurales):

- Los patrones de vector son representados por letras minúsculas en negrillas y toman la forma:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

donde cada componente x_i representa al i-ésimo descriptor y *n* es el número de descriptores. La naturaleza de los componentes en el patrón de vector depende de la técnica de medición utilizada para obtenerlos.

- Para problemas de reconocimiento en donde se necesita tanto las mediciones cuantitativas como las relaciones espaciales entre los descriptores para determinar la pertenencia a una clase se utilizan secuencias de descripción. Son útiles en objetos o entidades cuya estructura está basada en una conexión de primitivas simples generalmente asociadas a su forma. La secuencia se forma ordenando las primitivas de manera tal que describan al patrón.
- Los árboles, se utilizan para representaciones con una estructura mas jerárquica. La raíz del árbol representa toda la escena, los niveles inferiores van clasificando la imagen en sus partes constituyentes hasta que se llega a un nivel en donde no se puede reconocer mas.

Los métodos de reconocimiento suponen que los descriptores de los patrones de vector fueron escogidos apropiadamente, y por lo tanto, la semejanza entre los objetos de cada clase resulta de su proximidad en el espacio de características. Cada clase entonces podría ser separada por una curva de discriminación.

2.3 CALIBRACIÓN DE CÁMARA

La calibración de la cámara es un proceso que relaciona un modelo ideal con el dispositivo físico y determina la posición y orientación de ésta con respecto al sistema de referencia del mundo [IOC 98].
Dependiendo del modelo utilizado existen diferentes parámetros que tienen que ser determinados.

2.3.1 Modelo de la cámara de punta de alfiler

El modelo de cámara de punta de alfiler esta basado en el principio de colinealidad, donde cada punto en el espacio del objeto es proyectado en una línea recta a través del centro de proyección del plano de imagen [HEI 97].
La relación entre un punto M en $3\mathrm{D}$ y su proyección en la imagen m esta dado por la fórmula:

$$m = A[Rt]M$$

donde A es la matriz intrínseca de la cámara:

$$A = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

y (c_x, c_y) son las coordenadas del punto principal;

(f_x, f_y) son las longitudes focales sobre los ejes *x* y *y*,

Los parámetros intrínsecos describen la geometría interna y características ópticas de los lentes y del dispositivo de proyección de la imagen.
La longitud focal es la distancia entre el lente y el plano de imagen.
El punto principal es el centro de la imagen en coordenadas de píxeles.

(R,t) son los parámetros extrínsecos que describen la posición y orientación de la cámara en el sistema de referencia del mundo. Relacionan el sistema de coordenadas del mundo con el sistema de coordenadas de la cámara.
R es la matriz de rotación y t el vector de translación.

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, t = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

Un punto en el marco de referencia de la cámara $\left[X^c, Y^c, Z^z\right]^T$ esta relacionado con su correspondiente punto en el marco de referencia del mundo $\left[X^W, Y^W, Z^W\right]^T$ por:

$$P^c = RP^W + t$$

2.3.2 Distorsión del lente

Una cámara generalmente exhibe distorsión producida por el lente. La distorsión es descrita por cuatro coeficientes: dos coeficientes de distorsión radial k_1, k_2, y dos tangenciales p_1, p_2.
La distorsión radial provoca que un punto en la imagen sea desplazado radialmente del plano de imagen. La distorsión tangencial es provocada porque no siempre los centros de la curvatura de la superficie del lente son estrictamente colineales [SLA 80] .

Si (u,v) es la coordenada de un píxel de la imagen, es decir, la coordenada de la proyección ideal, y $(\breve{u},\breve{v})$ es la correspondiente coordenada de la imagen con distorsión y similarmente, (x,y) es la coordenada física ideal (sin distorsión) y $(\breve{x},\breve{y})$ es la coordenada física real (con distorsión).
Las coordenadas físicas con distorsión y sin distorsión se relacionan por [HEI 97]:

$$\begin{cases} \breve{x} = x + x\left[k_1 r^2 + k_2 r^4\right] + \left[2p_1 xy + p_2\left(r^2 + 2x^2\right)\right] \\ \breve{y} = y + y\left[k_1 r^2 + k_2 r^4\right] + \left[2p_2 xy + p_2\left(r^2 + 2y^2\right)\right] \end{cases}$$

donde $r^2 = x^2 + y^2$.

El segundo término de las ecuaciones anteriores describe la distorsión radial y el tercer término la distorsión tangencial.

Si el centro de la distorsión radial es igual al punto principal (c_x, c_y) entonces:

$$\begin{cases} \breve{u} = u + (u - c_x)\left[k_1 r^2 + k_2 r^4 + 2p_1 y + p_2\left(\frac{r^2}{x} + 2x\right)\right] \\ \breve{v} = v + (v - c_y)\left[k_1 r^2 + k_2 r^4 + 2p_2 x + p_1\left(\frac{r^2}{y} + 2y\right)\right] \end{cases}$$

2.3.3 Métodos de calibración

Existen en la literatura varios métodos para la calibración geométrica [HEI 97]. El enfoque clásico [SLA 80] que se origina del campo de la fotogrametría resuelve el problema minimizando una función no lineal de error. Debido a la lentitud y carga computacional de esta técnica fueron sugeridas soluciones de forma cerrada: [TSA 87], [MEL 94]. Sin embargo, éstos métodos están basados en simplificaciones del modelo de la cámara, y por lo tanto, no proveen resultados tan buenos como la minimización no lineal.

También existen procedimientos de calibración en donde se usan tanto minimización no lineal como soluciones de forma cerrada: [WEN 92]. En éstos métodos de dos pasos, los valores de los parámetros iniciales son calculados linealmente y los valores finales son obtenidos con una minimización no lineal.

Los métodos donde el modelo de cámara esta basado en parámetros físicos, como la longitud focal y el punto principal son llamados métodos explícitos. En la mayoría de los casos, los valores de esos parámetros en sí mismos no tienen utilidad, porque solo se requiere las relaciones entre coordenadas de referencia en 3D y coordenadas de la imagen en 2D. En la calibración implícita de la cámara, los parámetros físicos son remplazados por un conjunto de parámetros implícitos no físicos que son usados para interpolar entre puntos de enlace conocidos [WEI 93]. El método de Tsai es un método explicito.

2.3.3.1 Método de Tsai

Un conocido método para calibrar la cámara es el propuesto por Tsai. El método se basa en el conocimiento de la posición de algunos puntos en el mundo y sus correspondientes proyecciones en la imagen [IOC 98]. Es necesario que la cámara sea dirigida a un patrón de

calibración (generalmente un patrón similar a un tablero de ajedrez) de donde se obtiene las posiciones de los puntos en el mundo.

El método consiste de dos pasos. El primer paso es resolver una ecuación lineal para encontrar los parámetros externos (R,t) (exceptuando t_3) seguido de una optimización no lineal sobre $t_3, f, k_1, k_2, p_1, p_2$. En el segundo paso c_x, c_y son determinados mediante una optimización no lineal.
El procedimiento puede ser iterado para mejorar la exactitud.

3. ANÁLISIS DEL SISTEMA

En este capítulo se describen los requerimientos funcionales del sistema, basándose en las reglas de la liga de robots pequeños y en las especificaciones técnicas de la arquitectura de nuestro equipo. Además se presentan los actores y casos de uso del sistema utilizando notación UML (Unified Modeling Language).

3.1 REQUERIMIENTOS

Los requerimientos para el ***Sistema de Visión*** (**SV**) pueden ser divididos en dos categorías:

- La primera categoría consiste en los requerimientos que imponen las reglas de la liga de robots pequeños (también conocida como Liga F180) ya que definen el ambiente sobre el cual el sistema se desenvuelve.
- La segunda categoría son los requerimientos de desempeño necesarios para lograr que el sistema sea preciso, robusto y en tiempo real.

3.1.1 Reglas de la liga de robots pequeños

Definir el ambiente de un sistema de visión es recomendable para encontrar las características que permitan la extracción de información visual. Sin el previo conocimiento de éstas características es muy poco probable que sea eficiente.
Para el SV desarrollado en esta tesis, el ambiente está definido en las reglas de la liga de robots pequeños de RoboCup.
Las reglas completas se pueden consultar en el Apéndice A y en el resto de éste apartado se describen solo los aspectos relevantes de las reglas para el SV.

3.1.1.1 El campo de juego

El campo de juego representa el “mundo” del SV. Éste y los objetos que se encuentran dentro de él son los elementos de interés; cualquier objeto que se encuentre fuera del campo de juego es ignorado.
Uno de los principales objetivos es el de localizar a los robots, ésta localización se tiene que hacer dentro de un espacio delimitado, conocido y bien definido; de esta forma, el campo de juego puede ser visto como este espacio que sirve de referencia para la localización, asumiendo un papel de entorno estático en donde los elementos dinámicos (robots y pelota)

cambian continuamente pero siempre en referencia al entorno en donde actúan, haciendo posible su localización.

Las características relevantes son:

- Color: verde y blanco.
- Tamaño total horizontal : 5.40 m
- Tamaño total vertical: 4.00 m
- Barra de montaje de equipo a 4 m de altura sobre el campo de juego.

En la Figura 3.1 se muestra un esquema del campo de juego.

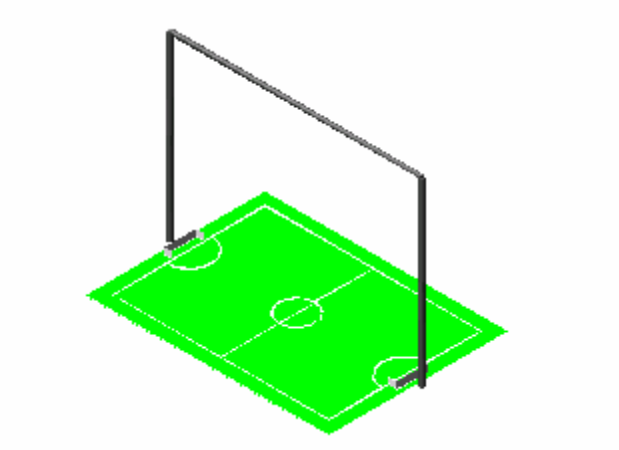


Figura 3.1 Campo de juego

Al tener solo dos colores (verde y blanco) se facilita el encontrar objetos que no pertenecen a él, pues se puede hacer una discriminación basada en colores: cualquier objeto que no sea verde o blanco es un elemento dinámico (robots o pelota).

Conociendo la altura a la que se coloca la cámara y el tamaño del campo de juego se puede determinar el número de cámaras necesarias para obtener una imagen completa del campo.

Si se utiliza una sola cámara, los ángulos de apertura del lente (horizontal y vertical) se calculan de la siguiente forma:
El ángulo de apertura horizontal es:

$$2\tan^{-1}\left(\frac{longitud\_horizontal}{2}\Big/ altura\right) =$$

$$2\tan^{-1}\left(\frac{540}{2}\Big/ 400\right) = 1.1874 radianes = 68.03º$$

El ángulo de apertura vertical es:

$$2\tan^{-1}\left(\frac{longitud\_vertical}{2}\Big/altura\right)=$$

$$2\tan^{-1}\left(\frac{400}{2}\Big/400\right)=0.9272radianes=53.13º$$

Encontrar una cámara con un lente gran angular de esas especificaciones es complicado y costoso, por lo que la solución obvia es utilizar mas de una cámara.

Utilizando dos cámaras, la longitud horizontal se divide entre dos, lo que nos da 2.70 m, de forma idéntica la longitud vertical se divide entre dos y nos da: 2.00 m.
Calculando los ángulos de apertura para cada cámara:

El ángulo de apertura horizontal es:

$$2\tan^{-1}\left(\frac{270}{2}\Big/400\right)=0.6509radianes=37.29º$$

El ángulo de apertura vertical es:

$$2\tan^{-1}\left(\frac{200}{2}\Big/400\right)=0.4899radianes=28.07º$$

Éstos ángulos de apertura se encuentran en cualquier cámara comercial con un lente gran angular. Un lente de este tipo introduce distorsión geométrica en la imagen que debe ser corregida para calcular adecuadamente la posición de los objetos.

El tamaño del campo de juego es importante porque determina directamente el área que representa un píxel de la imagen a una resolución dada.
Con una resolución de 320 x 240 píxeles, el área que representa un píxel esta dada por:

$$\left(Longitud\_total\Big/\mathrm{Re}solución\_horizontal\right)*\left(Ancho\_total\Big/\mathrm{Re}solución\_vertical\right)=$$

$$\left(270\Big/320\right)*\left(200\Big/240\right)=0.70cm^2$$

Para tener una precisión mayor al localizar los objetos es necesaria una resolución mayor.
Con una resolución de 640 x 480 píxeles, el área que representa un píxel esta dada por:

$$\left(270\Big/640\right)*\left(200\Big/480\right)=0.17cm^2$$

3.1.1.2 La pelota

La pelota es uno de los objetos de mayor interés para el SV, su localización precisa es fundamental debido a su importancia dentro de un partido; todas las acciones estratégicas del equipo están influenciadas en gran medida por la posición de la pelota.

Las características relevantes de la pelota para el SV son:

- Esférica.
- Color naranja.
- 43 mm de diámetro aproximadamente.

Debido a su forma esférica, la pelota aparece siempre en la imagen como un círculo y al ser el único objeto color naranja dentro del campo de juego su localización es mas sencilla.

Conociendo el diámetro de la pelota su área es: $2\pi\left(4.3/2\right) = 14.52cm^2$. En una resolución de 640 x 480 píxeles la pelota aparece en la imagen como un círculo naranja de aproximadamente 85 píxeles.

3.1.1.3 El número de robots

Los objetos que el SV debe localizar son los robots y la pelota. Al conocer cuántos objetos se están buscando (no mas de 10 robots y una pelota) se simplifica el procesamiento, ya que se pueden eliminar falsos positivos, es decir, no es posible tener mas de 10 robots y una pelota, así se eliminan aquellos objetos que posean las características más débiles que el SV utiliza para identificarlos, de esta manera se conservan solo los objetos que tienen las mejores características aumentando la probabilidad de haberlos identificado exitosamente.

3.1.1.4 El equipamiento robótico

Los robots poseen características que facilitan su identificación y su localización, estas son:

- Un robot debe caber dentro de un cilindro de 18 cm de diámetro.
- Altura máxima de 15 cm.
- Colores y parches.

Al tener los robots unas dimensiones máximas (cilindro de 18 cm de diámetro y altura máxima de 15 cm) el SV podría reconocer a un objeto con esas dimensiones como un robot, sin embargo, la tarea se facilita con el uso de colores y parches. Los colores permitidos para la construcción del robot son negro y blanco, que sirven de fondo visual para los parches colocados en la parte superior del robot. En cada robot se coloca un parche central de color amarillo para un equipo y azul para el otro, estos parches deben estar localizados en el centro visual del robot observado desde arriba. De esta manera, al localizar un parche amarillo o azul se determina la posición de un robot.

Adicionalmente, el uso de parches extras está permitido, éstos se utilizan para identificar y determinar la orientación del robot respecto a un plano, los colores permitidos para estos parches extras son: cian, verde y rosa claro.

La Figura 3.2 muestra como se observaría un robot desde arriba.

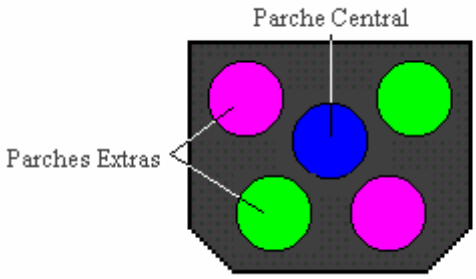


Figura 3.2 Vista superior de un robot

3.1.2 Requerimientos dados por las reglas de la liga de robots pequeños

Los requerimientos impuestos por las reglas de la liga de robots pequeños son los siguientes:

- El mínimo número de cámaras necesarias para que el Sistema de Visión pueda observar todo el campo de juego es de dos. Esto implica tener dos tarjetas de captura en la misma computadora funcionando en paralelo, o bien, una tarjeta de captura con la capacidad de digitalizar dos señales de video al mismo tiempo.
- La distorsión geométrica introducida por el lente gran angular debe ser corregida.
- La resolución mínima necesaria para la digitalización de la imagen es de 640 x 480 píxeles.
- El SV debe poder segmentar los diferentes objetos dentro del campo de juego mediante colores. Las reglas señalan que el campo de juego es verde con líneas blancas. La pelota es naranja. Los robots tienen parches centrales amarillos para un equipo y azul para el otro. Los colores de los parches adicionales son cian, verde y rosa claro.

3.1.3 Requerimientos de desempeño

El SV es la única fuente de información sobre el estado del juego. Esto significa que la precisión de la información es de extrema importancia y debido a que la introducción de ruido en las imágenes es inevitable, es importante tener un sistema robusto y tolerante a fallas.

La información obtenida por el SV debe ser enviada a uno o varios Sistemas de IA para que sea utilizada en otros procesos. Por tanto, uno de los requerimientos es poder transmitir la información a través de un socket de red ya que es el medio de recepción de datos del Sistema de IA.

El SV debe ser capaz de adaptarse a diferentes campos de juegos con condiciones de luz variable. Entonces, es indispensable tener un mecanismo de calibración para poder indicar los rangos de valores que definen a cada color de interés. Se utiliza el espacio de color YUV para poder tener separados en diferentes canales los componentes de intensidad y de color.

La velocidad de procesamiento de las imágenes es también un aspecto a considerar, ya que es necesario mantener una tasa de procesamiento suficientemente alta para captar los movimientos de los objetos sin saltos considerables, es decir, cuando un objeto se mueve, lo deseado es obtener la mayor cantidad de posiciones que ha recorrido en su trayectoria, por ejemplo, si se procesa a una tasa de 15 fps y si el objeto se mueve a una velocidad de 2 m/s (lo cual es bastante común en un partido) tendremos saltos en las posiciones capturadas de 2/15 m = 13.3 cm, lo cual no es muy bueno considerando que el diámetro del robot es de 18 cm.
Por tanto, tener una tasa lo mas cercana a la limitante introducida por las señales de video provenientes de las cámaras es lo mas adecuado. Como se explicó en el Capítulo 2, el estándar de video entrelazado NTSC funciona a 30 cuadros por segundo, cada cuadro contiene dos campos (par o non) y cada campo es capturado en la mitad de ese tiempo: 1/60 s. Entonces la máxima tasa de procesamiento es 60 fps. A esa tasa podremos tener posiciones de un objeto cada (2/ 60) = 1 /30 m = 3.3 cm si su velocidad es de 2m/s.
El SV debe tener una tasa de procesamiento entre 57 y 60 fps para tener saltos entre 3.5 cm y 3.3 cm.

La Figura 3.3 ilustra los saltos en el movimiento de los robots a diferentes tasas de procesamiento.

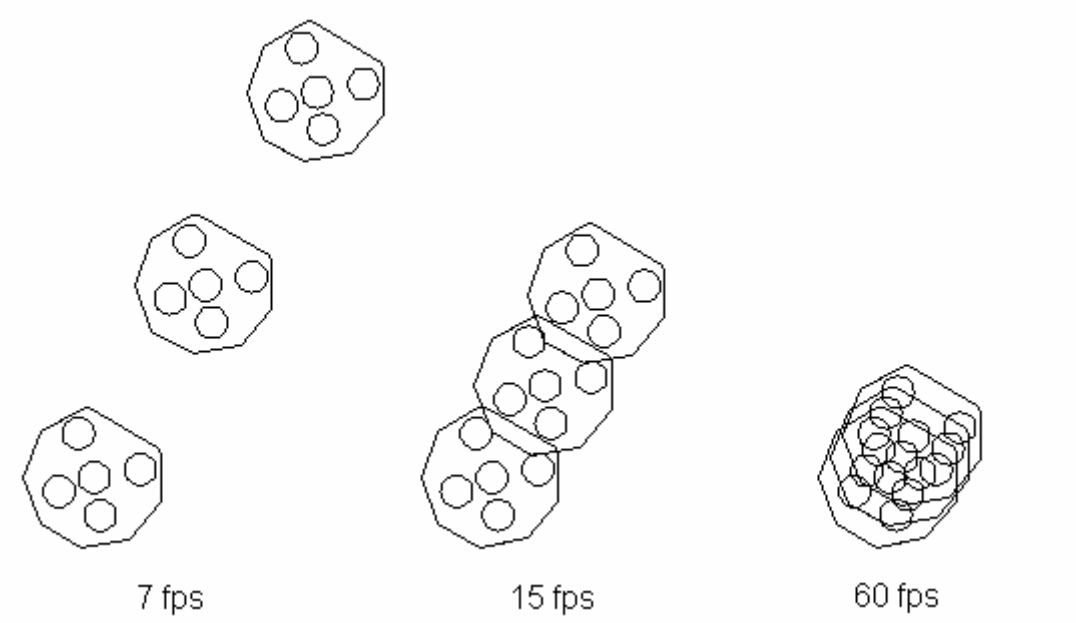


Figura 3.3 Saltos en el movimiento con diferentes tasas de procesamiento

Finalmente, el SV debe poder identificar a cada uno de los robots de nuestro equipo y calcular su orientación porque el Sistema de IA necesita conocer la posición y orientación de cada robot para poder computar un vector de movimiento y transmitir los comandos adecuados al robot correcto. Si la identificación es incorrecta, los comandos serán recibidos por un robot distinto al que queríamos controlar provocando movimientos erráticos.

La identificación de los robots es posible utilizando un código binario de colores en los parches extras. El verde representa al cero y el rosa al 1. Para obtener el código binario de los parches, el SV debe ser capaz de ordenarlos de manera correcta (en sentido contrario a las manecillas del reloj alrededor del parche central) y así identificar al robot. Por ejemplo:

- El robot uno podría ser 0001 (Verde, Verde, Verde, Rosa).
- El dos 0010 (Verde, Verde, Rosa, Verde).
- El tres 0011 (Verde, Verde, Rosa, Rosa).
- El cuatro 0100 (Verde, Rosa, Verde, Verde).
- El cinco 0101(Verde, Rosa, Verde, Rosa).

Sin embargo, cualquier otra combinación puede ser utilizada para identificar a los robots, siempre y cuando está se encuentre entre 0 (0000) y 16 (1111).

3.2 ACTORES

Los actores del sistema sirven para identificar las interacciones que tiene el sistema con el exterior, en concreto, un actor representa [SCO 01]:

- Un rol que un usuario puede jugar con respecto al sistema.
- Una entidad, como otro sistema o base de datos que reside fuera del sistema.

Los actores que intervienen en el sistema son:

- **Usuario.** Es la persona encargada de la administración y configuración del sistema. Sus responsabilidades incluyen:
  - Establecer los rangos de valor que definen cada uno de los colores de interés.
  - Salvar y cargar archivos con los rangos de valor para los colores.
  - Asistir en el proceso de calibración geométrica presentando frente a la cámara un patrón de ajedrez (patrón de calibración) y configurando parámetros de iniciación del proceso.
  - Guardar y cargar archivos con los parámetros obtenidos del proceso de calibración geométrica.
  - Ajustar parámetros para mejorar la calidad del video capturado.

- Seleccionar los objetos (robots contrarios y nuestros) que el sistema va a identificar y localizar.

Figura 3.4 Actor: Usuario

- **Sistema de Inteligencia Artificial.** Es un programa que recibe la información generada por el SV por medio de un socket de red y la utiliza para tomar decisiones estratégicas y enviar comandos de movimientos y acciones a los robots.

Figura 3.5 Actor: Sistema de Inteligencia Artificial

- **Archivo de parámetros de cámara.** En este archivo se guardan parámetros intrínsecos y extrínsecos de la cámara utilizados para eliminar la distorsión introducida por el lente gran angular, como son el punto principal, longitud focal, matriz de rotación y vector de translación, coeficientes de distorsión radial y tangencial. Son utilizados para poder recuperar parámetros de calibraciones anteriores sin necesidad de volver a realizar todo el proceso.

Figura 3.6 Actor: Archivo de parámetros de cámara

- **Archivo de segmentación.** En este archivo se guardan los rangos de valor que definen a cada color. Todos los colores están definidos por un valor mínimo y máximo para cada componente del espacio de color YUV.

Figura 3.7 Actor: Archivo de segmentación

- **Dispositivo de Captura.** Envía mensajes al SV cuando un cuadro de vídeo ha sido capturado en alguna de las dos cámaras.

Figura 3.8 Actor: Dispositivo de Captura

3.3 DESCRIPCIÓN FUNCIONAL DEL SISTEMA

Para la descripción funcional del sistema se emplean casos de uso con su correspondiente flujo de eventos. Un Caso de Uso es una secuencia de acciones que un actor realiza dentro del sistema para alcanzar una meta particular [BRO 02]. Un flujo de eventos es la descripción de los sucesos y eventos que son necesarios para obtener el comportamiento deseado del sistema.

El Sistema de Visión consta de los cinco actores mencionados. La manera general en que interactúan con el sistema es mostrada en la Figura 3.8. En ésta se puede observar los cinco casos de uso y las interacciones existentes entre ellos y los actores del sistema.

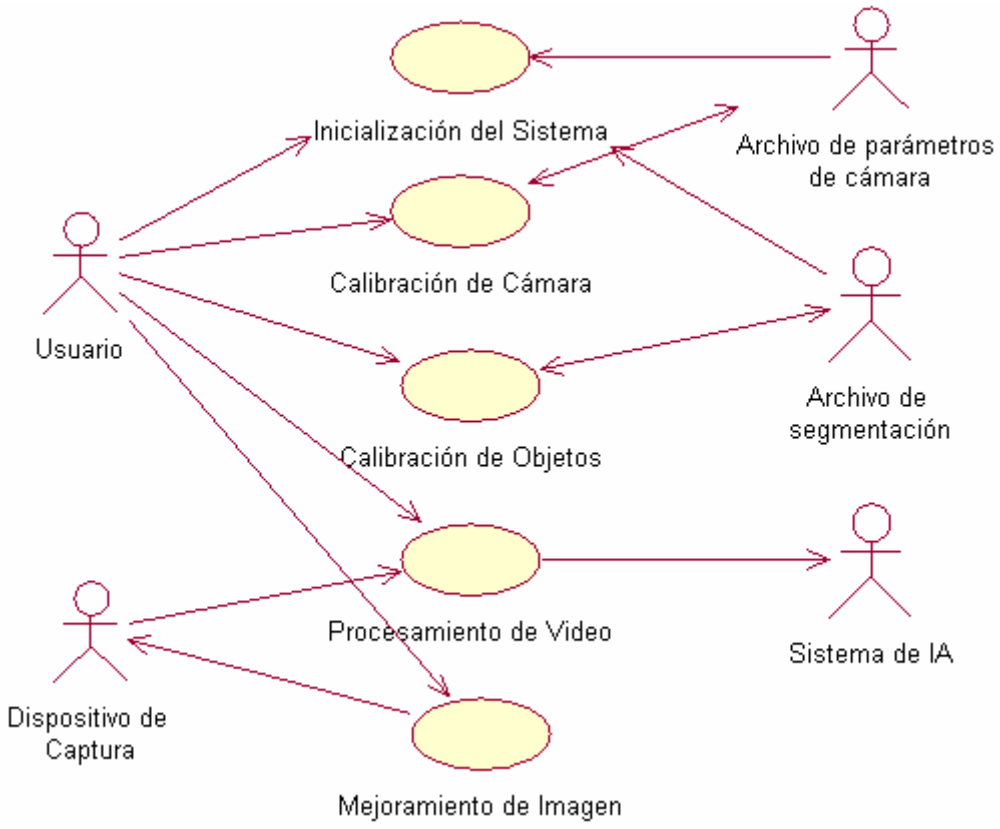


Figura 3.9 Diagrama de Casos de Uso en su vista principal

Los casos de uso son: Inicialización del Sistema, Calibración de Cámara, Calibración de Objetos, Procesamiento de Vídeo y Mejoramiento de Imagen. A continuación se presenta una descripción de los casos de uso.

3.3.1 Inicialización del Sistema

Este caso de uso es iniciado por el Usuario. Otorga la capacidad de seleccionar la cámara para cada mitad de la cancha, cargar archivos de segmentación y de parámetros de cámara. El flujo de eventos de este caso de uso es el siguiente:

CASO DE USO	Inicialización del Sistema
Precondiciones	Antes de que este caso de uso comience su ejecución es necesario que el Usuario haya colocado, encendido y conectado cada una de las dos cámaras que observan el campo de juego a una de las tarjetas de captura de video de la computadora por medio de un cable.
Flujo Principal	Este caso de uso comienza inmediatamente después de la ejecución del sistema, el Usuario será capaz de efectuar diferentes acciones referentes a la inicialización del sistema como son: SELECCIÓN DE LA CÁMARA 1, SELECCIÓN DE LA CÁMARA 2. Una vez que se han seleccionado las dos cámaras el Usuario puede efectuar acciones de configuración las cuales incluyen: CARGA DE ARCHIVO DE SEGMENTACIÓN, CARGA DE ARCHIVO DE PARÁMETROS DE CÁMARA 1 y CARGA DE ARCHIVO DE PARÁMETROS DE CÁMARA 2. Si la actividad seleccionada es SELECCIÓN DE LA CÁMARA 1 es llevado a cabo el subflujo S-1. Si la actividad seleccionada es SELECCIÓN DE LA CÁMARA 2 es llevado a cabo el subflujo S-2. Si la actividad seleccionada es CARGA DE ARCHIVO DE SEGMENTACIÓN es llevado a cabo el subflujo S-3. Si la actividad seleccionada es CARGA DE ARCHIVO DE PARÁMETROS DE CÁMARA 1 es llevado a cabo el subflujo S-4. Si la actividad seleccionada es CARGA DE ARCHIVO DE PARÁMETROS DE CÁMARA 2 es llevado a cabo el subflujo S-5. Si se han realizado todas las inicializaciones necesarias el caso de uso termina.

Subflujos	S-1: Selección de cámara 1. Este flujo solo puede presentarse al iniciar el sistema o después de haber seleccionado el dispositivo correspondiente a la cámara para la segunda mitad del campo (S-2). El Usuario debe seleccionar de una lista de dispositivos de captura registrados en la computadora aquel que este conectado a la cámara colocada sobre la primera mitad del campo. Si el Usuario selecciona ACEPTAR su elección del dispositivo es comparada con el registro obtenido de S-2, si es diferente o no existe registro, su elección es registrada y el caso de uso comienza nuevamente (E-1); si es igual, se procede a reiniciar el subflujo (E-2), si selecciona CANCELAR su elección no es registrada y el caso de uso comienza nuevamente (E-1). S-2: Selección de cámara 2. Este flujo solo puede presentarse al iniciar el sistema o después de haber seleccionado el dispositivo conectado a la cámara para la primera mitad del campo (S-1). El Usuario debe seleccionar de una lista de dispositivos de captura registrados en la computadora aquel que este conectado a la cámara colocada sobre la segunda mitad del campo. Si el Usuario selecciona ACEPTAR su elección del dispositivo es comparada con el registro obtenido de S-1, si es diferente o no existe registro, su elección es registrada y el caso de uso comienza nuevamente (E-1); si es igual, se procede a reiniciar el subflujo (E-2), si selecciona CANCELAR no se hace un registro nuevo de su elección y el caso de uso comienza nuevamente (E-1). S-3: Carga de Archivo de segmentación. Este subflujo se utiliza para recuperar parámetros guardados con anterioridad que se desean utilizar nuevamente. Se presenta una forma donde el Usuario debe escoger un Archivo de Segmentación con el cual se obtienen los rangos de valor de segmentación de cada objeto de interés (Pelota, Parche Central Azul, Parche Central Amarillo, Parche Extra No. 1, Parche Extra No. 2) para cada cámara.: - Valor mínimo de la componente Y. - Valor máximo de la componente Y. - Valor mínimo de la componente U.

	- Valor máximo de la componente U - Valor mínimo de la componente V. - Valor máximo de la componente V. Si el Usuario selecciona ACEPTAR un total de 60 parámetros son registrados en el sistema (30 para cada cámara) y el caso de uso comienza nuevamente (E-1), si selecciona CANCELAR, ningún parámetro es registrado y el caso de uso comienza nuevamente (E-1). S-4: Carga de Archivo de parámetros de cámara 1. El sistema despliega una forma en la cual el Usuario debe escoger un Archivo de parámetros de donde se obtiene los valores intrínsecos y extrínsecos de la cámara colocada sobre la primera mitad del campo y que son utilizados para eliminar la distorsión introducida por el lente gran angular, los cuales son: el punto principal (cx,cy), longitud focal (fx,fy), matriz de rotación (r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, r_{31}, r_{32} y r_{33}), vector de translación (t_1,t_2,y t_3), coeficientes de distorsión radial (k_1 y k_2) y tangencial (p_1 y p_2). Si el Usuario selecciona ACEPTAR un total de 20 parámetros son registrados en el sistema y el caso de uso comienza nuevamente (E-1), si selecciona CANCELAR, ningún parámetro es registrado y el caso de uso reinicia (E-1). S-5: Carga de Archivo de parámetros de cámara 2. El sistema muestra una forma en la cual el Usuario debe escoger un Archivo de parámetros de donde se obtiene los valores intrínsecos y extrínsecos de la cámara colocada sobre la segunda mitad del campo y que son utilizados para eliminar la distorsión introducida por el lente gran angular. Si el Usuario selecciona ACEPTAR un total de 20 parámetros son registrados en el sistema y el caso de uso comienza nuevamente (E-1), si selecciona CANCELAR, ningún parámetro es registrado y el caso de uso reinicia (E-1).
Flujos Alternativos	E-1 El Sistema de Visión ha registrado dos dispositivos diferentes para cada mitad del campo y ahora se pueden efectuar las acciones de configuración. E-2 Se despliega un mensaje de error y se reinicia el subflujo.

3.3.2 Calibración de Cámara

Este caso de uso es iniciado por el Usuario. Otorga la capacidad de calibrar cada una de las dos cámaras y obtener sus parámetros intrínsecos y extrínsecos de una manera semiautomática ya que solo requiere que el Usuario introduzca algunos valores iniciales y presente frente a la cámara un patrón de calibración que el proceso utiliza para hacer los cálculos. El flujo de eventos de este caso de uso es el siguiente:

CASO DE USO	Calibración de Cámara
Precondiciones	Antes de que este caso de uso comience es necesario haber seleccionado los dispositivos de captura para ambas cámaras.
Flujo Principal	Este caso de uso comienza cuando el Usuario selecciona la opción de CALIBRACIÓN. El Usuario puede calibrar cada una de las cámaras mediante las opciones: CALIBRACIÓN GEOMÉTRICA CÁMARA 1 y CALIBRACIÓN GEOMÉTRICA CÁMARA 2. Si la actividad seleccionada es CALIBRACIÓN GEOMÉTRICA CÁMARA 1 es llevado a cabo el subflujo S-1 en el video de la cámara colocada sobre la primera mitad del campo. Si la actividad seleccionada es CALIBRACIÓN GEOMÉTRICA CÁMARA 2 es llevado a cabo el subflujo S-1 en el video de la cámara que observa la segunda mitad del campo. Si se han realizado todas las operaciones deseadas el caso de uso termina.
Subflujos	S-1: Calibración Geométrica de la Cámara. El sistema despliega una forma y el Usuario debe introducir la siguiente información: - Cuadrados por renglón.- El número de cuadrados que contiene el patrón de ajedrez horizontalmente. - Cuadrados por columna.- El número de cuadrados que contiene el patrón de ajedrez verticalmente. - Longitud lineal del cuadrado. Distancia (en cualquier unidad) entre esquinas adyacentes del cuadrado. - Número de imágenes a procesar. Cantidad de imágenes que se utilizarán en el proceso de calibración geométrica.. - Tiempo de espera.- Tiempo en ms (milisegundos) que el sistema esperará para capturar una nueva imagen y

	procesarla. El Usuario debe colocar frente a la cámara un patrón de ajedrez con el número de cuadrados por renglón y columna especificados en los datos introducidos en la forma. El Usuario puede escoger entre las opciones EMPEZAR, CARGAR, ACEPTAR y CANCELAR. Si la opción seleccionada es EMPEZAR el proceso de calibración geométrica comienza capturando y procesando el número de imágenes introducidas en la forma en el tiempo especificado. El Usuario debe mover el patrón de manera aleatoria durante el proceso. Al concluir el proceso se despliegan los parámetros intrínsecos y extrínsecos obtenidos de la cámara (el punto principal (cx,cy), longitud focal (fx,fy), matriz de rotación (r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, r_{31}, r_{32} y r_{33}), vector de translación (t_1,t_2,y t_3), coeficientes de distorsión radial (k_1 y k_2) y tangencial (p_1 y p_2)) . Si la opción seleccionada es CARGAR, el Usuario debe escoger un Archivo de parámetros en el cual se encuentran los valores intrínsecos y extrínsecos de la cámara utilizados para eliminar la distorsión introducida por el lente gran angular. Si el archivo no existe se procede con E-1. Una vez que el sistema ha obtenido los parámetros de la cámara (mediante las opciones EMPEZAR o CARGAR) el Usuario puede guardarlos en un Archivo de parámetros de cámara seleccionando la opción SALVAR, o bien, visualizar el video eliminando la distorsión geométrica activando la opción QUITAR DISTORSIÓN. Si la opción seleccionada es ACEPTAR los parámetros de la cámara son registrados en el sistema y el subflujo termina. Si la opción seleccionada es CANCELAR el subflujo termina.
Flujos Alternativos	E-1: Si el archivo no existe se despliega un mensaje de error y se reinicia el caso de uso.

3.3.3 Calibración de Objetos

Este caso de uso es iniciado por el Usuario. Otorga la capacidad de calibrar los rangos de valor para todos los objetos de interés (Pelota, Parche Central Azul, Parche Central Amarillo, Parche

Extra No. 1, Parche Extra No. 2) para las dos cámaras. El flujo de eventos de este caso de uso es el siguiente:

CASO DE USO	Calibración de Objetos
Precondiciones	Antes de que este caso de uso comience es necesario haber seleccionado los dispositivos de captura para las dos cámaras.
Flujo Principal	Este caso de uso comienza cuando el Usuario selecciona la opción CALIBRACIÓN DE OBJETOS. El Usuario puede seleccionar que objeto calibrar mediante las opciones: PELOTA, EQUIPO AZUL, EQUIPO AMARILLO, PARCHE EXTRA 1, PARCHE EXTRA 2. Adicionalmente, puede escoger para cual cámara se registran los rangos de valores obtenidos activando la opción adecuada: CÁMARA 1 o CÁMARA 2. Si la opción seleccionada es PELOTA es llevado a cabo el subflujo S-1. Si la opción seleccionada es EQUIPO AZUL es llevado a cabo el subflujo S-2. Si la opción seleccionada es EQUIPO AMARILLO es llevado a cabo el subflujo S-3. Si la opción seleccionada es PARCHE EXTRA 1 es llevado a cabo el subflujo S-4. Si la opción seleccionada es PARCHE EXTRA 2 es llevado a cabo el subflujo S-5. Si se han realizado todas las operaciones deseadas el caso de uso termina.
Subflujos	S-1: Pelota. El sistema busca los rangos de valores registrados en el sistema para la pelota. Si no se encuentran se ejecuta E-1. Se procede con el subflujo S-6. S-2: Equipo Azul. El sistema busca los rangos de valores registrados en el sistema para el equipo azul. Si no se encuentran se ejecuta E-1. Se procede con el subflujo S-6. S-3: Equipo Amarillo. El sistema busca los rangos de valores registrados en el sistema

	para el equipo amarillo. Si no se encuentran se ejecuta E-1. Se procede con el subflujo S-6. S-4: Parche Extra 1. El sistema busca los rangos de valores registrados en el sistema para el parche extra 1. Si no se encuentran se ejecuta E-1 Se procede con el subflujo S-6. S-5: Parche Extra 2. El sistema busca los rangos de valores registrados en el sistema para el parche extra 2. Si no se encuentran se ejecuta E-1. Se procede con el subflujo S-6. S-6: Cambio gráfico de rangos de valores. Se despliega una forma con controles gráficos para poder modificar cada uno de los parámetros utilizados para asignar los tres rangos de valores en el espacio YUV a los objetos de interés: valor mínimo de la componente Y, valor máximo de la componente Y, valor mínimo de la componente U, valor máximo de la componente U, valor mínimo de la componente V y valor máximo de la componente V. Si el Usuario cambia cualquier valor de los parámetros, el sistema despliega en tiempo real el efecto del cambio modificando el color (de acuerdo al objeto del que se trate) de los píxeles de la imagen que entran dentro de los tres rangos señalados por el Usuario. Si el Usuario selecciona la opción ACEPTAR, los rangos de valores son registrados para el objeto correspondiente en la cámara seleccionada. El subflujo termina. Si el Usuario selecciona la opción CANCELAR, se descartan los nuevos valores y se restablecen los valores anteriores. El subflujo termina.
Flujos Alternativos	E-1: Los valores de los parámetros son inicializados en cero.

3.3.4 Procesamiento de video

Este caso de uso es inicializado por el Usuario cuando desea empezar a procesar el flujo de video. Otorga la capacidad de iniciar el proceso por el cual el Sistema de Visión localiza la pelota y los robots, identifica a cada robot de nuestro equipo y calcula su orientación y

finalmente transmite la información por medio de un socket de red. También permite activar y desactivar la búsqueda de un objeto (robots contrarios y robots de nuestro equipo) en las imágenes procesadas. El flujo de eventos de este caso de uso es el siguiente:

CASO DE USO	Procesamiento de Vídeo
Precondiciones	Antes de que este caso de uso comience es necesario haber seleccionado los dispositivos de captura para las dos cámaras, tener valores de umbral registrados en el sistema para los objetos de interés y de manera opcional tener registrados parámetros intrínsecos y extrínsecos de las cámaras para corregir la distorsión geométrica.
Flujo Principal	Este caso de uso comienza cuando el Usuario selecciona la opción EMPEZAR PROCESAMIENTO. El Usuario puede seleccionar que objetos buscar en las imágenes de las cámaras mediante las opciones: ACTIVAR/ DESACTIVAR ROBOT CONTRARIO, ACTIVAR/ DESACTIVAR ROBOT NUESTRO. Además puede seleccionar que color de parche central tiene nuestro equipo mediante la opción PARCHE CENTRAL. Si la opción seleccionada es ACTIVAR/DESACTIVAR ROBOT CONTRARIO es llevado a cabo el subflujo S-1. Si la opción seleccionada es ACTIVAR/DESACTIVAR ROBOT NUESTRO es llevado a cabo el subflujo S-2. Si la opción seleccionada es PARCHE CENTRAL es llevado a cabo el subflujo S-3. El sistema recibe mensajes del Dispositivo de Captura cuando un cuadro de video proveniente de alguna cámara ha sido capturado. Al recibir estos mensajes es llevado a cabo el subflujo S-4. Si se selecciona la opción CERRAR el caso de uso termina.
Subflujos	S-1: Activar/Desactivar robot contrario. El sistema despliega una forma y el Usuario puede seleccionar la cantidad de robots contrarios que desea buscar en los cuadros de vídeo capturados. Esta información es registrada en el sistema. Si se han realizado todas las activaciones/desactivaciones deseadas el subflujo termina (E-1). S-2: Activar/Desactivar robot nuestro El sistema despliega una forma y el Usuario puede elegir que robots

	de nuestro equipo buscar. Puede activar o desactivar la búsqueda de un robot específico, es decir, seleccionar el identificador (del uno al cinco) y realizar la acción deseada. Ésta información es registrada en el sistema. Si se han realizado todas las activaciones/ desactivaciones deseadas el subflujo termina (E-1). S-3: Parche Central El sistema despliega una forma y el Usuario puede elegir que color de parche central tiene nuestro equipo. Las opciones disponibles son AZUL o AMARILLO. La opción activada es registrada en el sistema. S-4: Procesar cuadro de video Un mensaje es enviado por el Dispositivo de Captura cuando un cuadro de video proveniente de alguna cámara ha sido capturado. El sistema procede al procesamiento del cuadro. El procesamiento del cuadro inicia con la segmentación de la imagen, continúa con la generación de la representación de los píxeles (regiones), procede con la identificación o reconocimiento, la localización y finalmente con la transmisión. Para encontrar a nuestros robots se buscan los parches centrales del color seleccionado por el Usuario y se identifica al robot utilizando los parches extras que se encuentran alrededor del parche central. Para encontrar los robots contrarios se buscan los parches centrales del color no seleccionado. Después, se verifica que los identificadores encontrados para nuestros robots estén activos en el registro del sistema y para los robots contrarios se toman en cuenta sólo la cantidad especificada por el Usuario, si existen mas robots contrarios, se desechan aquellos que mas probablemente se hayan reconocido erróneamente. Al calcular la posición de todos los objetos se toma en cuenta la cámara de dónde proviene la imagen. Se quita la distorsión geométrica producida por el lente utilizando los parámetros registrados en el sistema para esa cámara. La información es guardada en una estructura de datos que es transmitida por un socket de red a la dirección IP del Sistema de IA.
Flujos Alternativos	E-1:Se reinicia el caso de uso

3.3.5 Mejoramiento de imagen

Este caso de uso es inicializado por el Usuario cuando desea mejorar la calidad de las imágenes capturadas por las tarjetas de video, es decir, realizar operaciones de preprocesamiento de imagen que ayude a suprimir información que no es relevante para el SV y acentúe las diferencias entre los colores de los objetos de interés.
Específicamente, otorga la capacidad de modificar parámetros como el brillo, contraste, matiz, saturación, etc. El flujo de eventos de este caso de uso es el siguiente:

CASO DE USO	Mejoramiento de imagen
Precondiciones	Antes de que este caso de uso comience es necesario haber seleccionado los dispositivos de captura para ambas cámaras.
Flujo Principal	Como el principal objetivo de este caso de uso es ayudar en el procesamiento del cuadro, este caso de uso puede realizarse en paralelo al caso de uso Procesamiento de Vídeo. Este caso de uso comienza cuando el Usuario selecciona la opción CONFIGURACIÓN DE CÁMARAS El Usuario selecciona la cámara de donde provienen las imágenes que quiere modificar y se despliega una forma con controles gráficos para poder modificar cada uno de los parámetros: - Brillo - Contraste - Matiz - Saturación - Nitidez - Gamma Si el Usuario cambia cualquier valor de los parámetros, el sistema despliega en tiempo real el efecto del cambio en la imagen de la cámara seleccionada. Si el Usuario selecciona la opción ACEPTAR, los valores son registrados para la cámara seleccionada. Si el Usuario selecciona la opción CANCELAR, se descartan los nuevos valores y se restablecen los valores anteriores. Si se han realizado todas las operaciones deseadas el caso de uso termina.

4. DISEÑO DEL SISTEMA

En este capítulo se presenta el diseño del sistema. Empieza describiendo la arquitectura del ***Sistema de Visión*** (**SV**), los módulos que lo componen, el diagrama de clases con una descripción breve de cada una de las clases y los diagramas de secuencia que exponen las interacciones entre los diferentes componentes. Finalmente se presentan los algoritmos empleados en el procesamiento de video.

4.1 ARQUITECTURA

La arquitectura del sistema es fundamental en la organización del sistema visto como un todo. Incluye elementos estáticos y dinámicos, la manera cómo esos elementos trabajan en conjunto y el estilo general del sistema [SCO 01].

Es también importante porque da una idea clara de los componentes lógicos que se persiguieron al diseñar este sistema. La Figura 4.1 muestra el modelo arquitectónico del SV desarrollado en esta tesis.

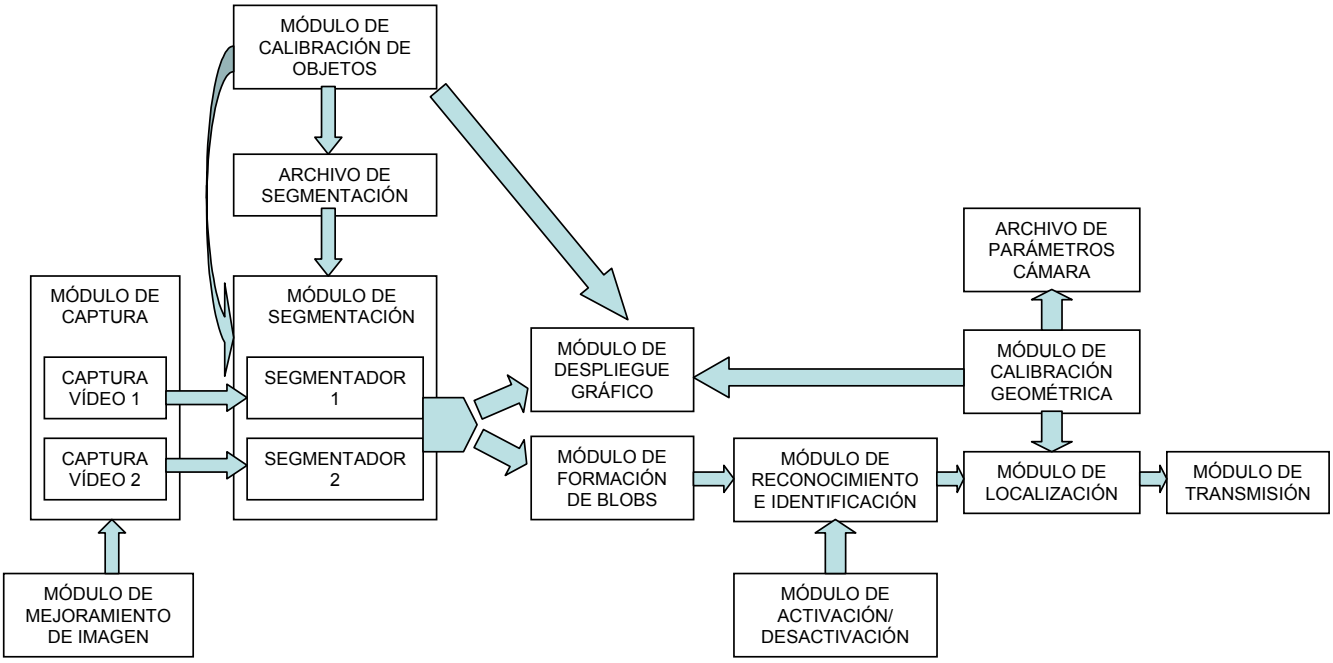


Figura 4.1 Arquitectura del sistema

Es interesante observar que esta arquitectura puede ser utilizada para otros sistemas de visión; cada uno de los módulos lógicos que la conforman pueden ser adaptados a los requerimientos

particulares de diferentes aplicaciones de visión por computadora, incluso es posible agregar o quitar algunos módulos sin necesidad de cambiar los demás. A continuación se describen las responsabilidades de cada uno de los módulos que componen la arquitectura.

4.2 MÓDULOS

Los módulos que componen la arquitectura del SV son:

- **Módulo de Captura**

  El Módulo de Captura tiene la responsabilidad de asignar un dispositivo físico de captura de video (instalado en la computadora) a cada una de las cámaras utilizadas en la obtención de las imágenes del campo de juego. Además provee funcionalidad para establecer:

  1. El tipo de cable utilizado en la conexión de la cámara (video compuesto o S-Video).
  2. La resolución de la imagen capturada (640x480, 320x240, 160x120, etc).
  3. El espacio de color empleado (YCrCb, RGB, etc.).
  4. El tipo de captura del cuadro de video (ambos campos, campo par o campo impar).
  5. La velocidad de captura (cuadros por segundo).

  La salida de este módulo es un flujo de vídeo constante de cada una de las cámaras en su representación digital, es decir, los píxeles de los cuadros de video son ordenados en matrices, la posición dentro de la matriz corresponde a la posición del píxel en la imagen. El valor del píxel se interpreta de acuerdo al espacio de color seleccionado.

- **Módulo de Mejoramiento de Imagen**

  El Módulo de Mejoramiento de Imagen es el encargado de modificar el valor de los píxeles de acuerdo a la operación de pre-procesamiento seleccionada por el usuario. Se puede modificar el brillo, contraste, matiz, saturación, gamma, etc.

  Este módulo esta estrechamente relacionado con la tarjeta de captura de video ya que gran parte de su funcionalidad depende de las interfaces expuestas por el dispositivo.

- **Módulo de Calibración de Objetos**

  Este módulo se encarga del establecimiento de los rangos de valor que definen los colores de todos los objetos de interés para el SV. Es responsable del registro de los rangos de valor usados en la segmentación de las imágenes provenientes de ambas cámaras en el sistema.

Provee una interface donde el Usuario selecciona el objeto que desea calibrar (Pelota, Parche Central Amarillo, Parche Central Azul, Parche Extra 1 y Parche Extra 2) y controles gráficos para modificar los siguientes parámetros asociados al objeto seleccionado:

- Valor mínimos de Y.
- Valor máximo de Y.
- Valor mínimo de U.
- Valor máximo de U.
- Valor mínimo de V.
- Valor máximo de V.

Los rangos válidos para cada componente son del cero al 255. El módulo verifica que los valores mínimos sean menores o iguales que los valores máximos en los tres componentes del espacio de color. Además es el responsable de guardar en un archivo los rangos de valores de cada objeto para ambas cámaras. También provee funcionalidad para leer parámetros de archivos almacenados previamente.

- **Módulo de Segmentación**

  El Módulo de Segmentación es el responsable de separar los píxeles de la imagen que pertenecen a los objetos de interés de los que no. El módulo consiste de dos segmentadores (uno para cada cámara). Cada segmentador utiliza los rangos de valores registrados para su cámara.

  Un píxel tiene tres valores, uno para cada componente del espacio de color. Si el valor de los tres componentes se encuentra dentro de los rangos registrados en algún objeto de interés, el píxel es marcado como perteneciente al objeto. Ésta comparación es repetida en todos los píxeles de la imagen y el resultado obtenido de este proceso es una imagen en donde los píxeles han sido segmentados en los objetos que constituyen la imagen.

- **Módulo de Formación de Blobs**

  El Módulo de Formación de Blobs es el encargado de formar regiones a partir de los píxeles que han sido segmentados. Antes de formar regiones o blobs, la única información que se tiene del píxel segmentado es su posición dentro de la imagen. Al SV no le interesan los píxeles de manera individual sino en su conjunto, es decir, los píxeles segmentados tienen valor en cuanto pertenecen a un grupo que representa un objeto. Todo el conjunto de píxeles que pertenece a un objeto es agrupado en una región mediante un proceso que busca en los píxeles vecinos alguno que pertenezca al mismo objeto, si pertenece, el proceso de búsqueda es repetido hasta llegar al borde o frontera de la región. Este proceso es realizado para todos los objetos de interés para el SV.

La importancia de una región consiste en que posee información útil para el análisis de la imagen como son el área, perímetro, forma y centroide de un objeto.

Este módulo crea listas de regiones de objetos, combinando la información de ambas cámaras en una unidad. Las listas de regiones son actualizadas con cada cuadro procesado.

- **Módulo de Activación/Desactivación**

  El Módulo de Activación/Desactivación es el responsable de proveer una interface en donde el Usuario puede activar o desactivar la búsqueda de un robot en los cuadros de video. El Usuario puede ingresar la cantidad de robots contrarios que se encuentran dentro del campo de juego, facilitando así la búsqueda de éstos. También puede indicar los identificadores de los robots activos de nuestro equipo y así evitar la búsqueda innecesaria de robots que no se encuentran dentro del campo. Finalmente puede seleccionar el color del parche central que ha sido asignado a nuestro equipo.

  El módulo es responsable de registrar esta información y realizar los cambios solicitados por el Usuario.

- **Módulo de Reconocimiento e Identificación**

  El Módulo de Reconocimiento e Identificación tiene como objetivo seleccionar las regiones que mejor se ajusten a los objetos buscados. Para hacer la selección se cuenta con una serie de criterios para cada tipo de objeto. El criterio para la pelota consiste en seleccionar la región naranja que mas se acerque a un área de 85 píxeles. El criterio para los robots contrarios consiste en seleccionar un número de regiones del color del parche central del equipo contrario que mas se aproxime a 115 píxeles. El número de regiones seleccionadas esta determinado por la cantidad introducida por el Usuario.

  El criterio para nuestros robots es un poco mas completo. Incluye al igual que en el criterio para los robots contrarios, la selección de regiones candidatas que mas se acerquen a 115 píxeles, pero además, se buscan parches extras alrededor de la posición del parche central que ayuden a la identificación del robot, si no se encuentran cuatro parches, la región es descartada, si se encuentran cuatro o mas parches son seleccionados los que mas se acerquen a 83 píxeles. Los parches extras son ordenados en sentido contrario a las manecillas del reloj y el código binario codificado en los colores de los parches es extraído. Si el identificador coincide con algún robot buscado, entonces la información es registrada, si no, simplemente se descarta y se continúa con la siguiente región candidata. El proceso continúa hasta que se encuentren todos los identificadores buscados o se terminen las regiones candidatas

- **Módulo de Calibración Geométrica**

  El Módulo de Calibración Geométrica es el responsable de realizar la calibración de las dos cámaras que se utilizan para obtener imágenes del campo de juego. Provee una interface en donde el Usuario puede introducir algunos parámetros de inicialización del proceso como son: el número de cuadrados horizontales en el patrón de ajedrez, el número de cuadros verticales en el patrón de ajedrez, la longitud lineal del cuadrado, el tiempo de espera entre imágenes procesadas, la cantidad de imágenes procesadas, etc. El módulo es responsable de comenzar el proceso de calibración que consiste en la extracción automática de los puntos asociados a las esquinas del patrón de ajedrez, una vez obtenidas todas las esquinas del patrón de calibración (por eso es importante saber cuantos cuadrados tiene el patrón) se puede comenzar con la optimización lineal de mínimos cuadrados de los parámetros extrínsecos e intrínsecos de la cámara. Al concluir esta optimización lineal se refinan los parámetros utilizando una optimización no lineal (Método de Tsai).

  El módulo también provee funcionalidad para guardar los parámetros obtenidos en el proceso de calibración en archivos que después pueden ser abiertos por el mismo módulo y así, obtener los valores previamente salvados.

  Por último, el módulo permite la creación de imágenes libres de distorsión que pueden utilizarse para verificar la validez de los parámetros calculados.

- **Módulo de Localización**

  El Módulo de Localización es el responsable de posicionar a los objetos identificados en un plano de referencia.

  El módulo toma el centroide de la región asociada al objeto encontrado (robots y pelota) y calcula su posición dentro de la imagen eliminando la distorsión causada por los lentes de la cámara. Después calcula su posición dentro del campo de juego mediante una función de translación que depende de la posición de la esquina inferior izquierda del campo de juego (esa es la coordenada (0,0)). La translación se realiza sobre los ejes *x* y *y*.

  Finalmente calcula la orientación de los robots de nuestro equipo utilizando las coordenadas libres de distorsión de los parches extras.

- **Módulo de Transmisión**

  El Módulo de Transmisión se encarga de la creación de un socket de red por donde se transmite una estructura de datos con la información de la posición de la pelota, robots contrarios y nuestros robots, así como también la orientación de nuestros robots. El módulo se encarga de abrir el puerto, establecer la conexión a la dirección IP del Sistema de IA, de la serialización de la estructura y del envío de la información.

- **Módulo de Despliegue Gráfico**
  El Módulo de Despliegue Gráfico se encarga de presentar en la pantalla los cuadros de video provenientes de los dispositivos de captura en tiempo real. Provee además, funciones básicas de dibujo como son líneas rectas, cambio de color de un píxel, etc.

Los módulos anteriores describen los componentes lógicos del sistema de manera general. Sin embargo, para que el diseño del sistema este completo es necesario identificar las clases que integran el sistema y las relaciones que guardan unas con otras. En la siguiente sección se presenta el diagrama de clases.

4.3 DIAGRAMA DE CLASES

El diagrama de clases muestra las clases y sus relaciones de asociación. Una clase es una colección de objetos que comparten las mismas características. El estereotipo de la clase puede ser entidad, control o interface [SCO 01].
Las clases entidad representan objetos en el mundo real. Contienen datos y operaciones que describen al objeto. Las clases de interface son utilizadas para manejar la comunicación entre el sistema y entidades externas como usuarios, operadores y otros sistemas. Las clases de control son creadas para describir comportamientos que no encajan fácilmente en clases de entidad o de interface. La Figura 4.2 muestra el diagrama de clases para el SV.

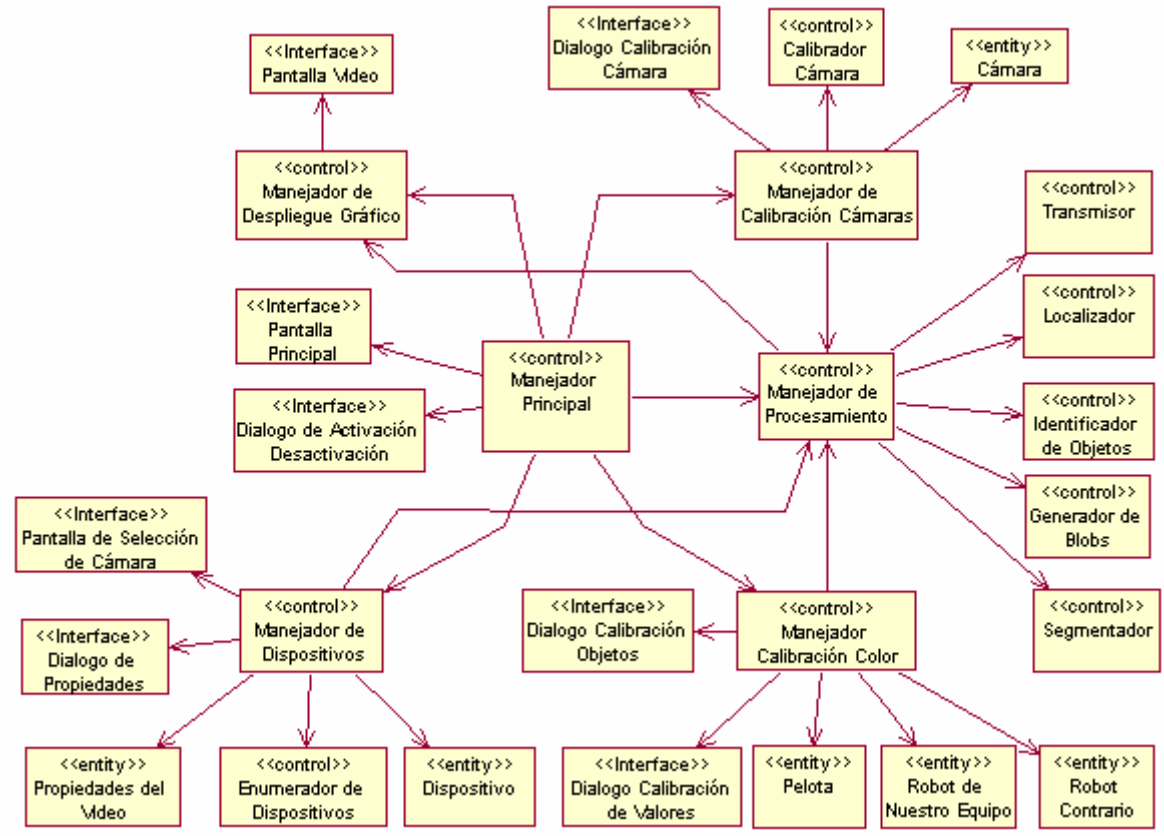


Figura 4.2 Diagrama de clases

Como se puede observar en el diagrama de clases, los manejadores son los encargados de realizar tareas que involucran a varias clases y constituyen la espina dorsal del sistema. Enseguida se describen las responsabilidades de cada clase.

- **Manejador Principal**
  Esta clase inicializa el Sistema de Visión y gestiona todas las operaciones de activación/desactivación de los robots y atiende los mensajes enviados por el Usuario a través de la Pantalla Principal. También es responsable de coordinar la captura, calibración de color, calibración de cámaras, procesamiento y despliegue gráfico.

- **Pantalla Principal**
  Esta clase es la encargada de manejar la comunicación entre el Usuario y el SV. Al recibir un mensaje del Usuario envía la información pertinente al Manejador Principal. Los mensajes que puede procesar son: Selecciona Cámara, Calibra Colores, Calibra Cámara, Activa/Desactiva Objetos e Inicia Procesamiento.

- **Manejador de Dispositivos**
  Se encarga de la gestión de los dispositivos de captura. Interviene en la selección de los dispositivos utilizados. Controla las operaciones de configuración, propiedades y mensajes de los dispositivos.

- **Enumerador de Dispositivos**
  Esta clase es la encargada de inspeccionar los dispositivos instalados en la computadora. Almacena información sobre las capacidades, propiedades e interfaces de cada dispositivo.

- **Pantalla de Selección de Cámara**
  Esta clase es la interface entre el Usuario y el SV para seleccionar los dispositivos (cámaras y tarjetas de captura) que se utilizan. Recibe los nombres de los dispositivos instalados y los despliega en una lista, registra la selección del Usuario y la transmite al Manejador de Dispositivos.

- **Dispositivo**
  Encierra la configuración, interfaces y métodos del dispositivo seleccionado. Es una abstracción del hardware utilizado para la captura de video.

- **Propiedades del Vídeo**
  Esta clase almacena los valores de brillo, contraste, saturación, matiz, etc. del dispositivo. Expone métodos para modificar los parámetros dentro de los rangos válidos.

- **Dialogo de Propiedades**

  Es la interface entre el Usuario y el SV utilizada para modificar las propiedades del video. Registra los cambios realizados por el usuario y expone métodos para su comunicación.

- **Manejador Calibración de Color**

  Esta clase realiza las tareas asociadas a la calibración de color de los objetos de interés (Pelota, Parche Central Amarillo, Parche Central Azul, Parche Extra 1 y Parche Extra 2). Controla el despliegue de las interfaces donde el Usuario puede seleccionar el objeto y modificar los rangos de valores para la segmentación. Además se encarga de guardar y abrir archivos con los rangos.

- **Pelota**

  Esta clase es la abstracción de la pelota en el mundo real. Posee atributos que la describen como su posición, área, perímetro, etc., además aquí se almacenan los rangos de valores en el espacio de color YUV usados para el color naranja.

- **Robot de Nuestro Equipo**

  Esta clase es la abstracción de un robot de nuestro equipo en el mundo real. Tiene los siguientes atributos: color del parche central (azul o amarillo), identificador, posición, orientación y código binario de colores. También se guardan los rangos de valores en el espacio de color YUV usados para el color del parche central y de los parches extras 1 y 2.

- **Robot Contrario**

  Es la abstracción de un robot del equipo contrario en el mundo real. Posee los siguientes atributos: color del parche central (azul o amarillo), posición y los rangos de valores en el espacio de color YUV para el color del parche central.

- **Dialogo Calibración Objetos**

  Esta clase es la interface en donde el Usuario puede seleccionar el objeto que desea calibrar y la cámara en donde los rangos de valor tendrán validez. Tanto el objeto como la cámara seleccionada son registrados y comunicados al Manejador Calibración de Color.

- **Dialogo Calibración de Valores**

  Es la interface en donde el Usuario puede modificar los valores máximos y mínimos de umbral (rangos de valor) para cada uno de los componentes del espacio de color YUV.

- **Manejador de Calibración de Cámaras**

  Esta clase gestiona las operaciones relacionadas con la calibración de las cámaras como son el despliegue del dialogo de calibración para la obtención de los parámetros de calibración iniciales, de iniciar el proceso de calibración, de abrir y guardar archivos con parámetros de calibración de cámaras.

- **Dialogo Calibración Cámara**

  Interface en donde el Usuario puede introducir parámetros de configuración del proceso de calibración. Permite recibir mensajes generados por el Usuario como son: carga de archivos con parámetros, iniciación del proceso y remoción de distorsión geométrica de la imagen.

- **Calibrador Cámara**

  Esta clase realiza todo el proceso de calibración de la cámara, obtiene los parámetros intrínsecos, extrínsecos y coeficientes de distorsión que definen a una cámara particular.

- **Cámara**

  Esta clase es la abstracción de la cámara en el mundo real. Sus atributos son: parámetros intrínsecos y extrínsecos, coeficientes de distorsión y el número que le corresponde a la cámara (uno o dos).

- **Manejador de Despliegue Gráfico**

  Esta clase tiene la responsabilidad de desplegar el video de las dos cámaras en tiempo real. Además expone métodos para poder dibujar líneas y cambiar el color de los píxeles.

- **Pantalla Vídeo**

  Clase de interface en donde el Usuario puede visualizar el video.

- **Dialogo de Activación/Desactivación**

  Esta clase de interface recibe mensajes de activación o desactivación de los robots de nuestro equipo o contrarios. Expone métodos para informar los cambios realizados por el Usuario.

- **Manejador de Procesamiento**

  Esta clase controla el procesamiento de video a partir del momento en donde los cuadros de vídeo son entregados por el dispositivo de captura. Coordina la segmentación, formación de blobs, identificación, localización y transmisión.

- **Segmentador**

  Esta clase realiza el procesamiento de segmentación en los cuadros de video. Al terminar el procesamiento se obtiene una imagen en donde los píxeles han sido segmentados en los objetos que la constituyen.

- **Generador de Blobs**

  Esta clase se encarga de formar regiones a partir de los píxeles que han sido segmentados. Agrupa los píxeles mediante un proceso de búsqueda en la vecindad de cada píxel. Al terminar el proceso se generan listas de regiones de los objetos. La información de las listas contiene los objetos encontrados en los cuadros de video de ambas cámaras.

- **Identificador de Objetos**

  Esta clase es la responsable de aplicar los criterios de selección en las listas de regiones para identificar a la pelota, robots contrarios y robots de nuestro equipo. Verifica además que los robots encontrados estén activados.

- **Localizador**

  Esta clase se encarga de la localización de los objetos identificados. Calcula la orientación de nuestros robots y computa la posición de la pelota y los robots eliminado las distorsiones provocadas por el lente.

- **Transmisor**

  Esta clase se encarga de crear una estructura de datos con la información de la posición de la pelota, robots contrarios y nuestros robots, así como también la orientación de nuestros robots. La clase abre el puerto, establece la conexión a la dirección IP del Sistema de IA y envía la información.

4.4 DIAGRAMAS DE SECUENCIAS

Los diagramas de secuencia permiten exponer el comportamiento dinámico de cada uno de los componentes del SV.
Se enfocan en los mensajes que van de un objeto a otro y el orden con que se fueron generando.

Los diagramas de secuencias presentados a continuación están ligados con los casos de uso expuestos en el capítulo anterior.

4.4.1 Inicialización del Sistema

Este caso de uso fue presentado en la sección 3.3.1. El diagrama de secuencia de este caso de uso esta dividido en tres. La Figura 4.3 muestra la secuencia de selección de cámara, la Figura 4.4 muestra la secuencia para Abrir un Archivo de Segmentación, la Figura 4.5 muestra la secuencia para Abrir un Archivo de Parámetros de Cámaras. Se hizo esta división por claridad y para facilitar la lectura de los pasos en cada secuencia.

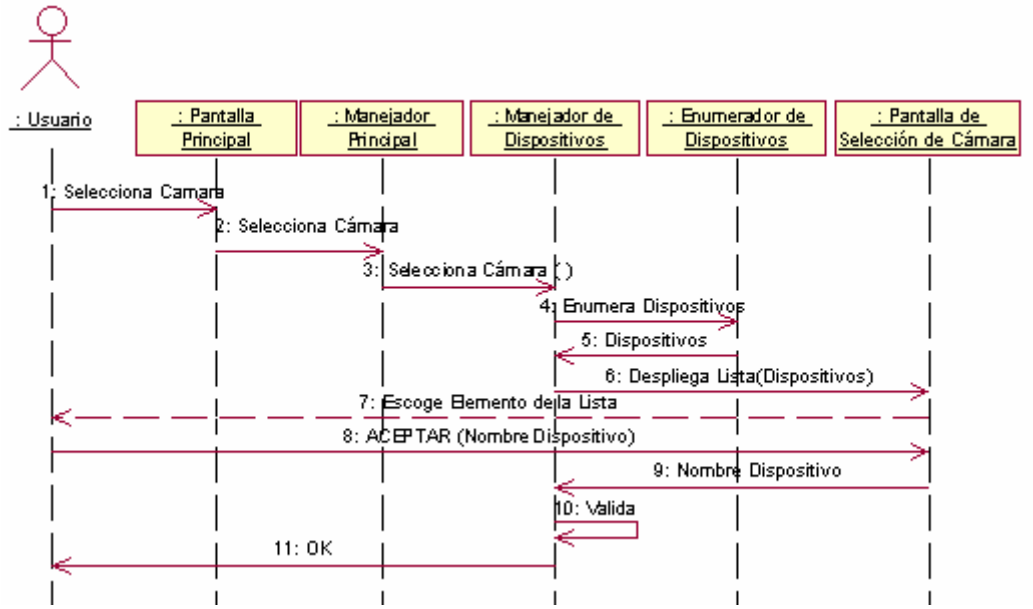


Figura 4.3 Inicialización del Sistema: Seleccionar Cámara

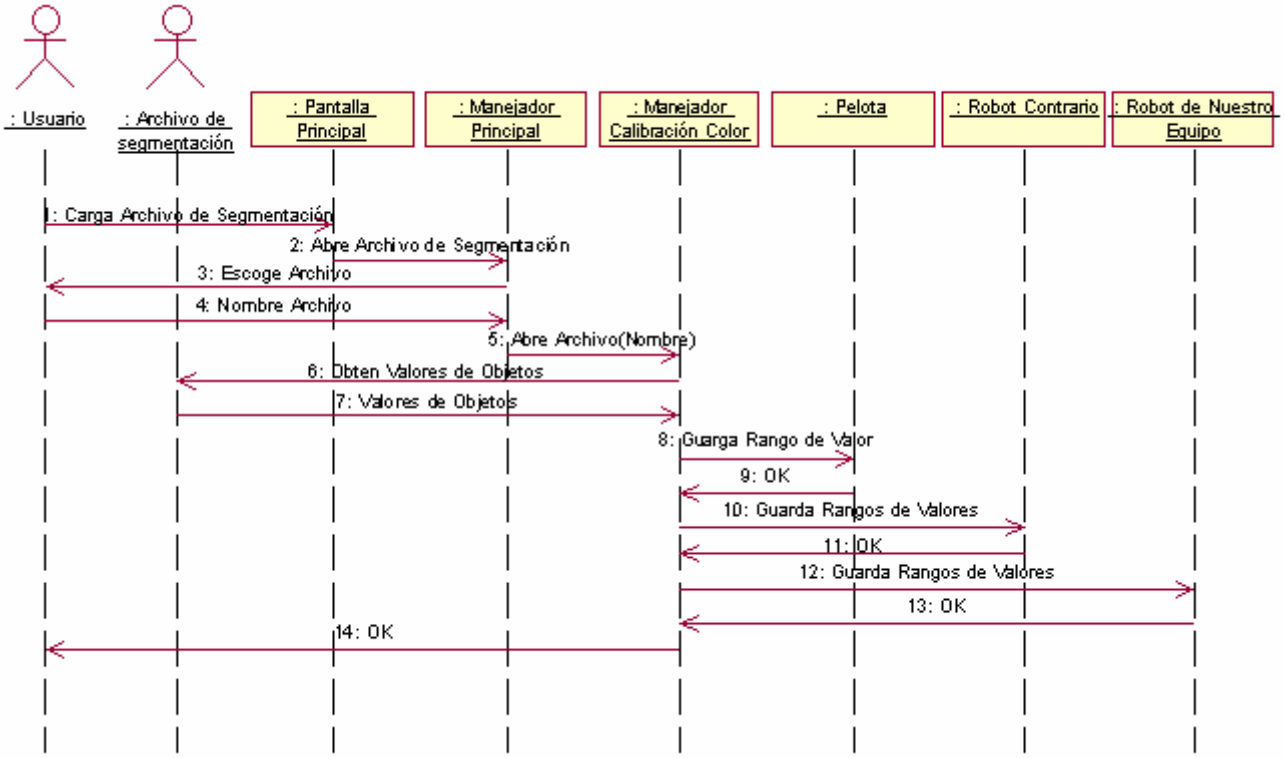


Figura 4.4 Inicialización del Sistema: Carga Archivo de Segmentación

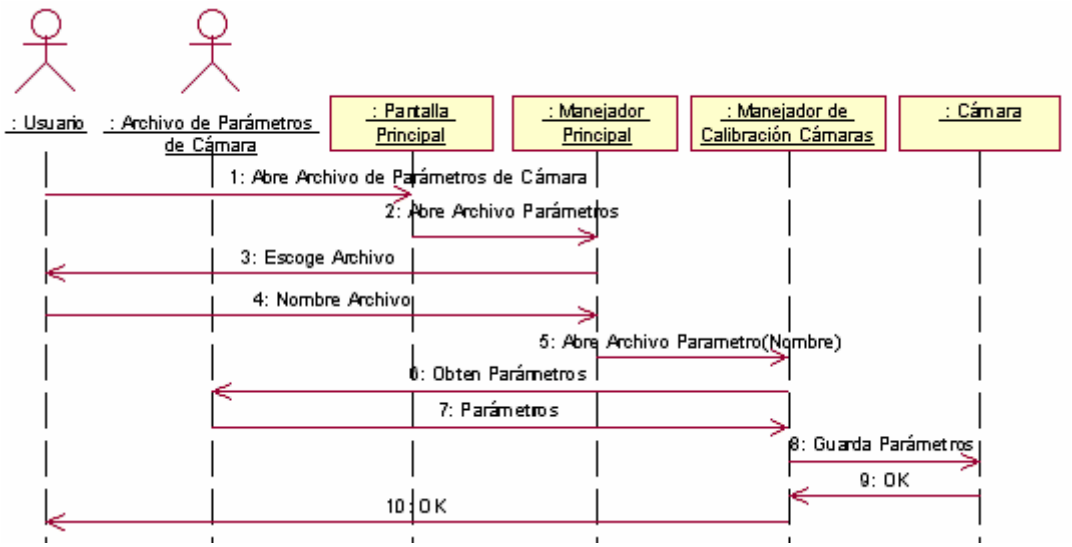


Figura 4.5 Inicialización del Sistema: Abre Archivo Parámetros de Cámara

4.4.2 Calibración de Cámara

Este caso de uso fue presentado en la sección 3.3.2. La Figura 4.6 muestra el diagrama de secuencias que activa el proceso de calibración de una cámara.

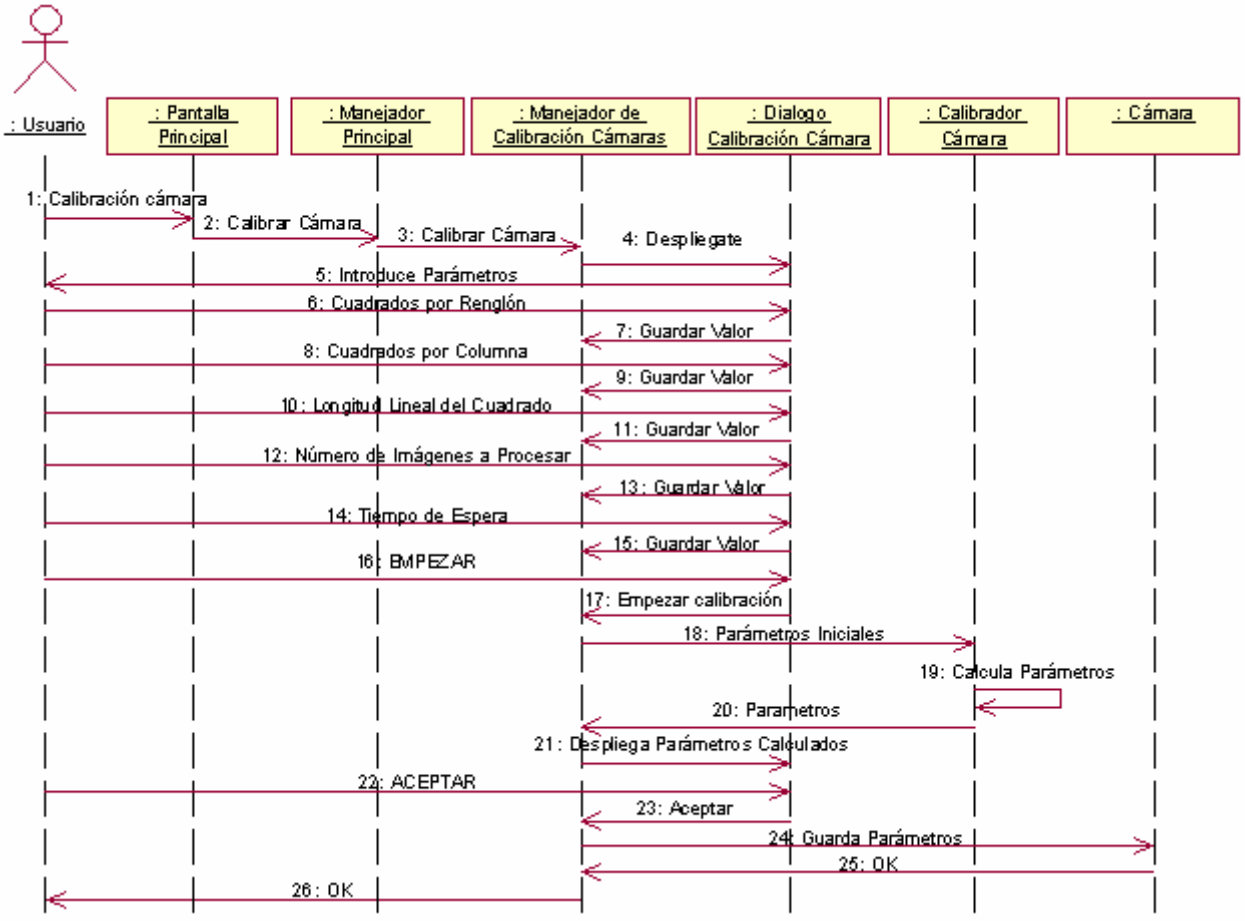


Figura 4.6 Calibración de Cámara

4.4.3 Calibración de Objetos

Este caso de uso fue presentado en la sección 3.3.3. En el diagrama de secuencia se ejemplifica los pasos necesarios para calibrar un objeto (el objeto calibrado es la pelota). Estos pasos son exactamente iguales para cualquier otro objeto, con la única diferencia que en lugar de pedir y registrar los rangos de valores a la clase Pelota se haría en las clases Robot Contrario o Robot de Nuestro Equipo.

La secuencia de los pasos 11 al 16 puede ser repetida cuántas veces quiera el Usuario (se pueden hacer varias modificaciones a los parámetros hasta obtener los deseados). El paso 11: Modifica Valor, se refiere al valor de un parámetro (valor mínimo o máximo para cualquiera de los tres componentes del espacio de color YUV) y la validación del cambio de valor es hecha en el paso 15: Actualiza Valores. La Figura 4.7 muestra el diagrama de secuencias para este caso de uso.

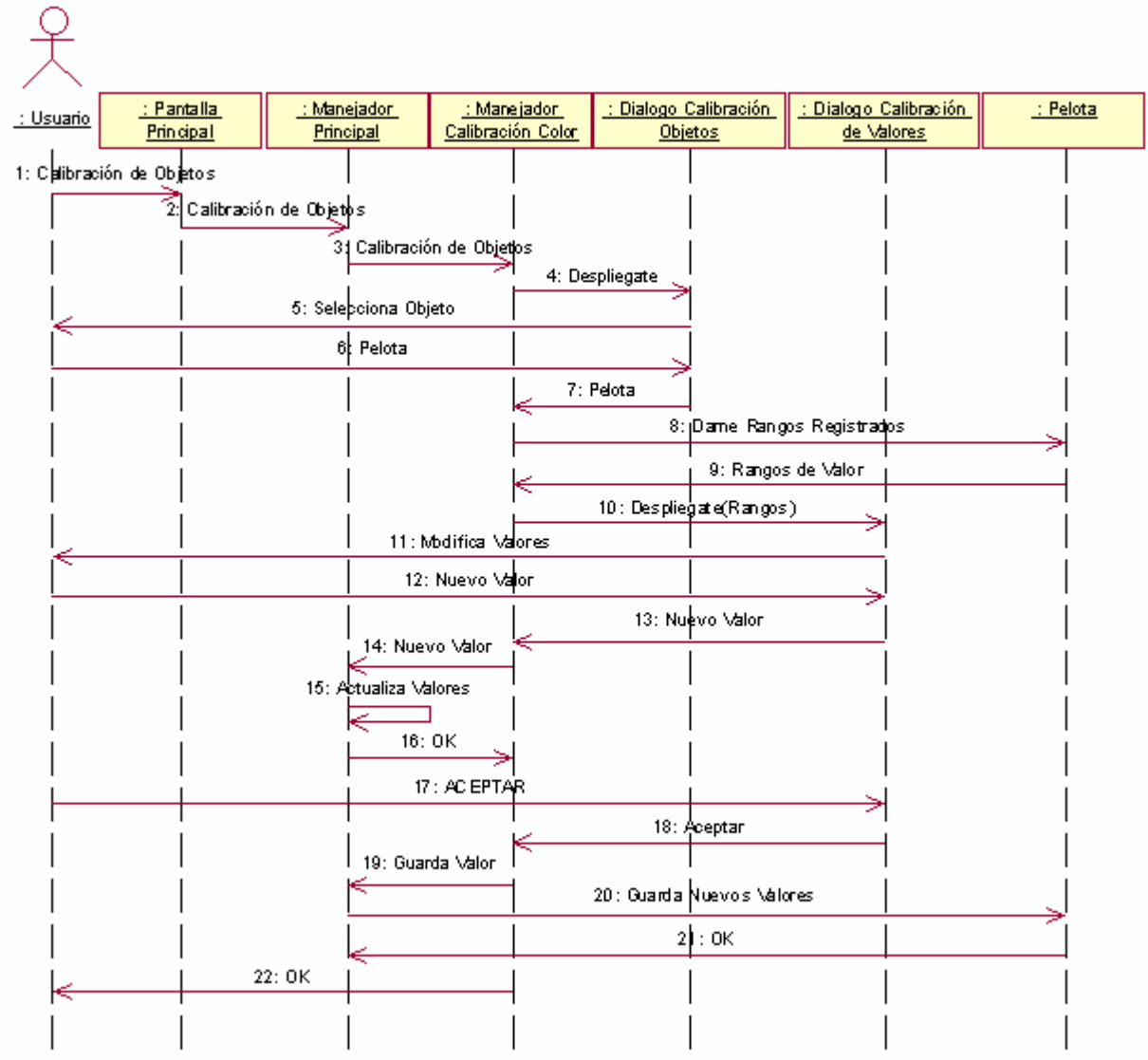


Figura 4.7 Calibración Objetos

4.4.4 Mejoramiento de Imagen

Este caso de uso fue presentado en la sección 3.3.5. La Figura 4.8 muestra el diagrama de secuencias que activa la configuración de las propiedades de una cámara.

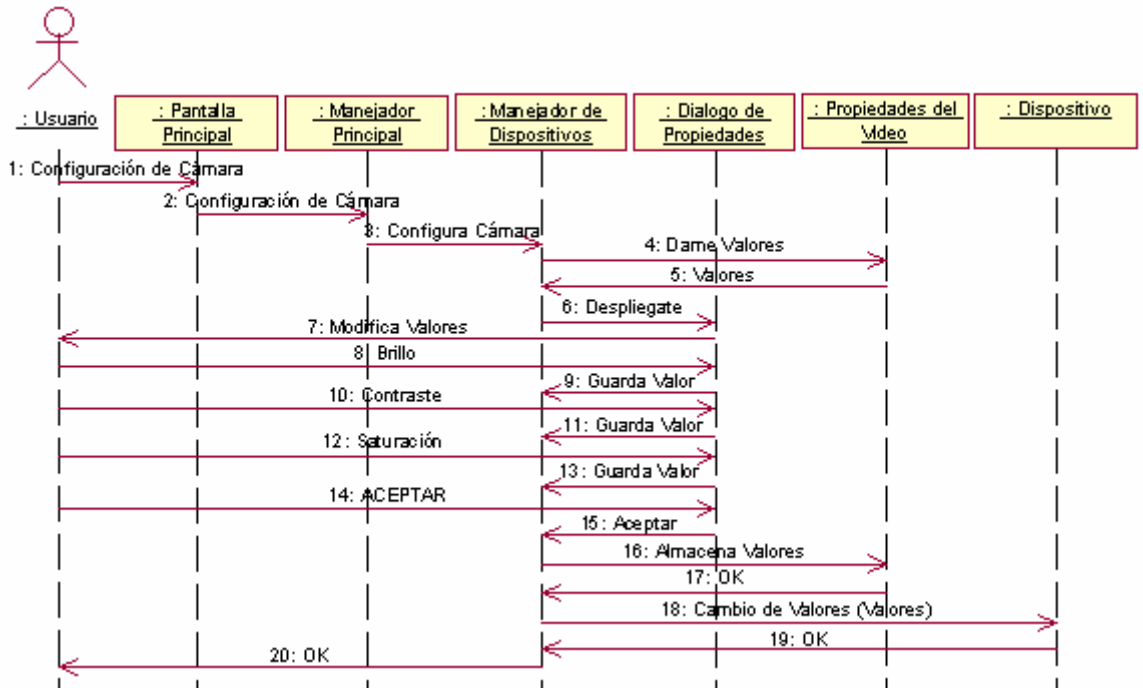


Figura 4.8 Mejoramiento de Imagen

4.4.5 Procesamiento de Vídeo

Este caso de uso fue presentado en la sección 3.3.4. El diagrama de secuencia de este caso de uso esta dividido en dos.

La Figura 4.9 ejemplifica la secuencia de activación/desactivación de los robots.

La Figura 4.10 muestra la secuencia del proceso de un cuadro de video recién capturado por el Dispositivo de Captura.

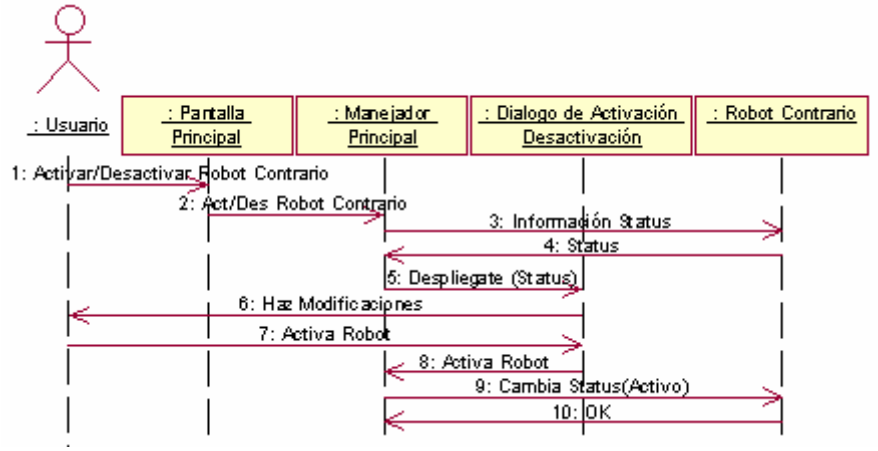


Figura 4.9 Procesamiento de Vídeo: Activación/Desactivación

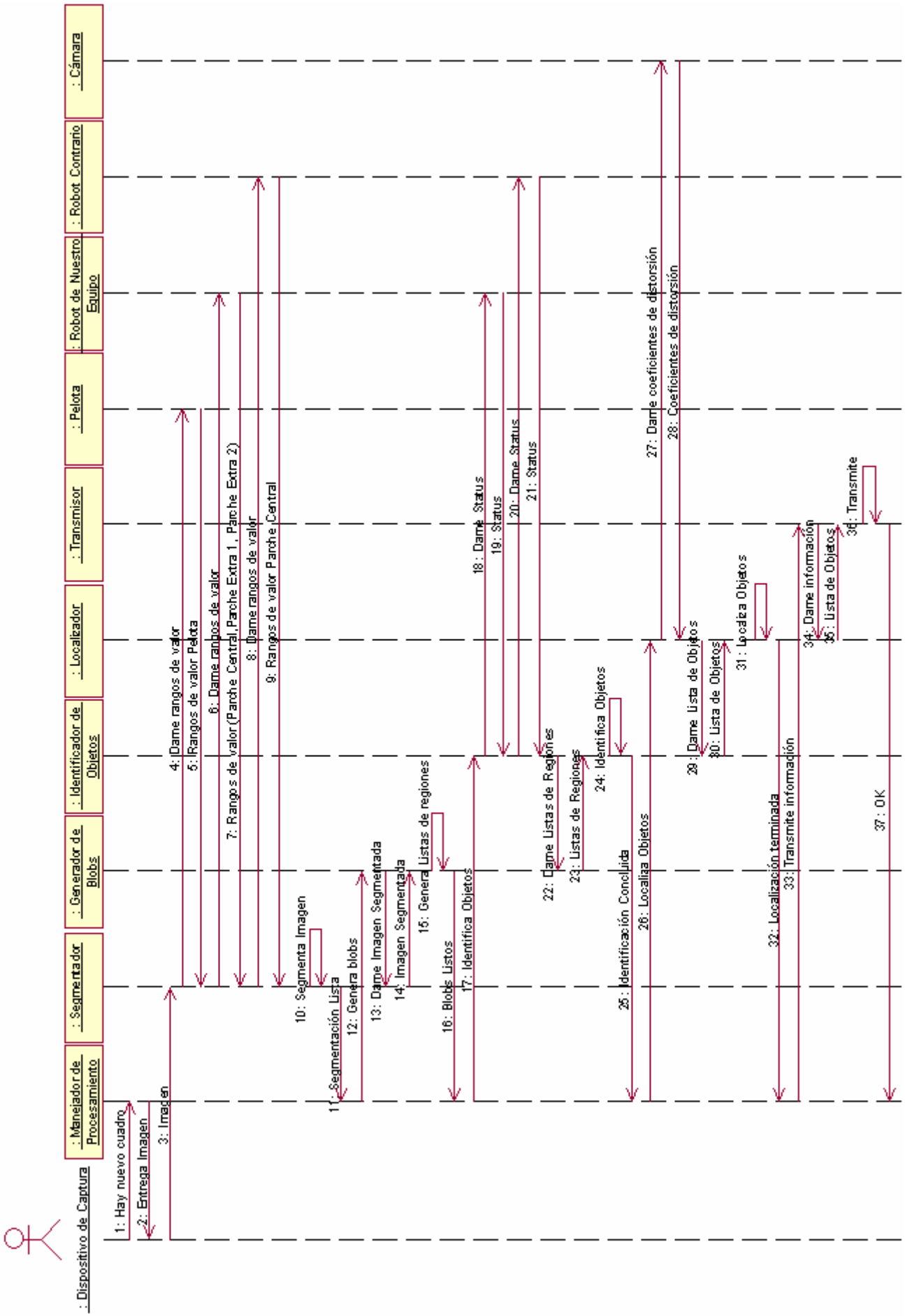


Figura 4.10 Procesamiento Vídeo: Procesa Cuadro

El diagrama de secuencia anterior constituye el núcleo funcional del Sistema de Visión, sin embargo no profundiza en los algoritmos empleados en los puntos claves del procesamiento de video. La siguiente sección ahonda en los detalles de los algoritmos.

4.5 ALGORITMOS

En esta sección se describen a detalle los algoritmos de cada una de las etapas del procesamiento de video. Como se ha explicado anteriormente, las etapas son: Captura, Segmentación, Formación de Regiones, Identificación, Localización y Transmisión.
Todas las etapas, salvo las de Captura y Transmisión, presentan algoritmos que transforman la imagen en información útil. A continuación se explican de manera detallada los algoritmos empleados y las razones por las que se les eligieron. Ésta explicación servirá para aterrizar muchos de los conceptos desarrollados en el Capítulo 2.

4.5.1 Algoritmo de Segmentación

Como se explico en el Capítulo 2, un paso importante en muchas aplicaciones de visión es la clasificación de cada píxel de una imagen en un número discreto de clases.
El objetivo de la segmentación en nuestro sistema es proveer esa clasificación de manera efectiva y en tiempo real. El método empleado puede ser descrito como de umbrales constantes en un espacio de color. El enfoque involucra el uso de rangos de valores de umbral en un espacio de color tridimensional. La elección del espacio de color para la clasificación depende de varios factores incluyendo cual es soportado por los dispositivos de captura y su utilidad en la aplicación.
Se escogió el espacio de color YUV porque es un espacio de color robusto, es decir, es confiable en presencia de variaciones de luz. Además, presenta la ventaja de estar soportado por el dispositivo de captura lo que evita la necesidad de realizar una transformación de espacio de color por software.

El algoritmo de segmentación descrito aquí fue propuesto en [BRU 00] y puede ser usado en cualquier espacio de color multidimensional que tiene componentes de color discretos. Para la explicación se tomará de ejemplo el espacio de color YUV.

Cada clase (objeto de interés definido por un color) es especificado como un conjunto de seis valores de umbral: dos para cada componente del espacio de color. Es decir, los valores máximos y mínimos del rango en cada componente del espacio de color.
La operación de clasificación es la que evalúa si un píxel pertenece a la clase o no y es importante poner atención en su eficiencia porque debe ser repetida para todos los píxel de la

imagen. En una imagen con una resolución de 640x480 píxeles, esta operación deberá ser repetida 307,200 veces.

Una manera de revisar si un píxel con valores Y, U, V pertenece a una clase es usar una comparación similar a esta:

```
if  ( ( Y >= Yminimo )
    AND ( Y <= Ymaximo )
    AND ( U >= Uminimo )
    AND ( U <= Umaximo )
    AND ( V >= Vminimo )
    AND ( V <= Vmaximo) )
    color_pixel = clase_color;
```

Desafortunadamente esta manera es bastante ineficiente, ya que implica 6 comparaciones para determinar la membresía de un píxel a una clase. Esto puede ser especialmente ineficiente en procesadores de pipeline con ejecución especulativa de instrucciones.

El algoritmo empleado utiliza una descomposición booleana de esta comparación. Una clase es representada como el producto de tres funciones, cada una sobre los ejes del espacio de color. Es decir, el píxel pertenece a la clase, si los valores de éste se encuentran dentro de los rangos (especificados por la función) en los tres componentes.

La descomposición del rango de valores es almacenada en arreglos, un elemento del arreglo para cada valor posible del componente del espacio de color. De esta manera la pertenencia a una clase puede ser evaluada con una operación AND de los elementos de cada arreglo indicados por los valores del píxel:

```
pixel_en_clase = Yarreglo[Y]
               AND  Uarreglo[U]
               AND  Varreglo[V];
```

El valor en `pixel_en_clase` indica si el píxel pertenece a la clase o no. El siguiente ejemplo ilustra el concepto:

Supongamos que discretizamos el espacio de color YUV en 10 niveles para cada componente. Entonces, un color cualquiera (digamos naranja) puede representarse asignando los siguientes valores a los elementos de los arreglos:

```
Yarreglo[] = {0,1,1,1,1,1,1,1,1,1};
Uarreglo[] = {0,0,0,0,0,0,0,1,1,1};
Varreglo[] = {0,0,0,0,0,0,0,1,1,1};
```

Para verificar que un píxel con valores `(1,8,9)` pertenezca a la clase naranja lo que necesitamos hacer es evaluar la expresión:

```
Yarreglo[1]AND Uarreglo[8] AND Varreglo[9]
```

que da como resultado `1` o `true` indicando que el píxel pertenece a la clase naranja.

Esta operación es más rápida que la anterior por que un AND tiene significativamente un menor costo que una comparación de enteros en procesadores modernos.

Una las mayores ventajas de este enfoque es que puede determinar la membresía de un píxel en múltiples clases simultáneamente. Explotando el paralelismo en la operación AND de números enteros podemos determinar la membresía de varias clases en una sola operación. Como ejemplo supongamos que el color azul se representa de la siguiente manera:

```
Yarreglo[] = {0,1,1,1,1,1,1,1,1,1};
Uarreglo[] = {1,1,1,0,0,0,0,0,0,0};
Varreglo[] = {0,0,0,1,1,1,0,0,0,0};
```

En lugar de tener un conjunto separado de arreglos para cada color podemos combinar los arreglos usando las posiciones de los bits en los elemento del arreglo para representar los valores de cada color. Por ejemplo, combinando el naranja y azul de los ejemplos anteriores tenemos:

```
Yarreglo[] = {00,11,11,11,11,11,11,11,11,11};
Uarreglo[] = {01,01,01,00,00,00,00,10,10,10};
Varreglo[] = {00,00,00,01,01,01,00,10,10,10};
```

Donde el bit mas significativo en cada elemento del arreglo es usado para representar el naranja y el segundo bit representa el azul. Ahora, volvamos a evaluar el píxel con valores `(1,8,9)` en la expresión:

```
Yarreglo[1]AND Uarreglo[8] AND Varreglo[9]
```

El resultado es `10`, indicando que el color pertenece al naranja pero no al azul. Como un entero tiene 32 bits se pueden segmentar hasta 32 colores en una sola operación AND.

4.5.2 Algoritmo de formación de regiones.

Después de haber segmentado, el siguiente paso consiste en conectar los píxeles formando las regiones o blobs. Éste proceso puede ser bastante exhaustivo y afectar el desempeño en tiempo real. Por eso hay que poner especial atención a su implementación.

El algoritmo empleado en el SV también fue propuesto en [BRU 00] y es implementado en dos pasos.

La primera etapa es computar un RLE de la imagen segmentada. Un RLE (Run Length Encoding) es un algoritmo de compresión sin pérdida [HEL 87]. La idea básica es tomar una secuencia de datos repetidos y reemplazarla por el dato con su cantidad. Esta secuencia de datos repetidos es conocida como corrida (Run), de ahí su nombre.
El siguiente ejemplo muestra el concepto:
Si tenemos los siguientes datos

NNNNNN000AAAA

después de usar el RLE nos queda

6N304A

En aplicaciones de visión por computadora los cambios en píxeles adyacentes son poco frecuentes. Agrupando píxeles de la misma clase en una sola corrida incrementamos la

eficiencia porque procesos subsecuentes pueden operar los datos en las corridas en lugar de hacerlo con píxeles individuales. Además, existe el beneficio práctico de que ahora el proceso de formación de regiones solo necesita buscar la conectividad de forma vertical ya que las componentes horizontales fueron unidas en la transformación de la imagen a su versión RLE.

El método de fusión de regiones tiene como objetivo juntar todas las corridas que pertenecen a una región. La idea, es lograr que todas las corridas de la misma región apunten a la corrida que se encuentre en la parte superior izquierda de la región (padre global). Para esto cada una de ellas cuenta con un apuntador.
Inicialmente, cada corrida se etiqueta a si misma como su padre, es decir se tiene un conjunto de corridas totalmente separadas. El procedimiento de fusión busca en los renglones adyacentes y une las corridas del mismo color que son vecinos en una conectividad de cuatro. Esto resulta en un conjunto de corridas donde los apuntadores apuntan a la corrida superior. Una segunda pasada es necesaria para comprimir los apuntadores a un padre global cuando hayan existido traslapes.
La Figura 4.11 muestra el método de fusión

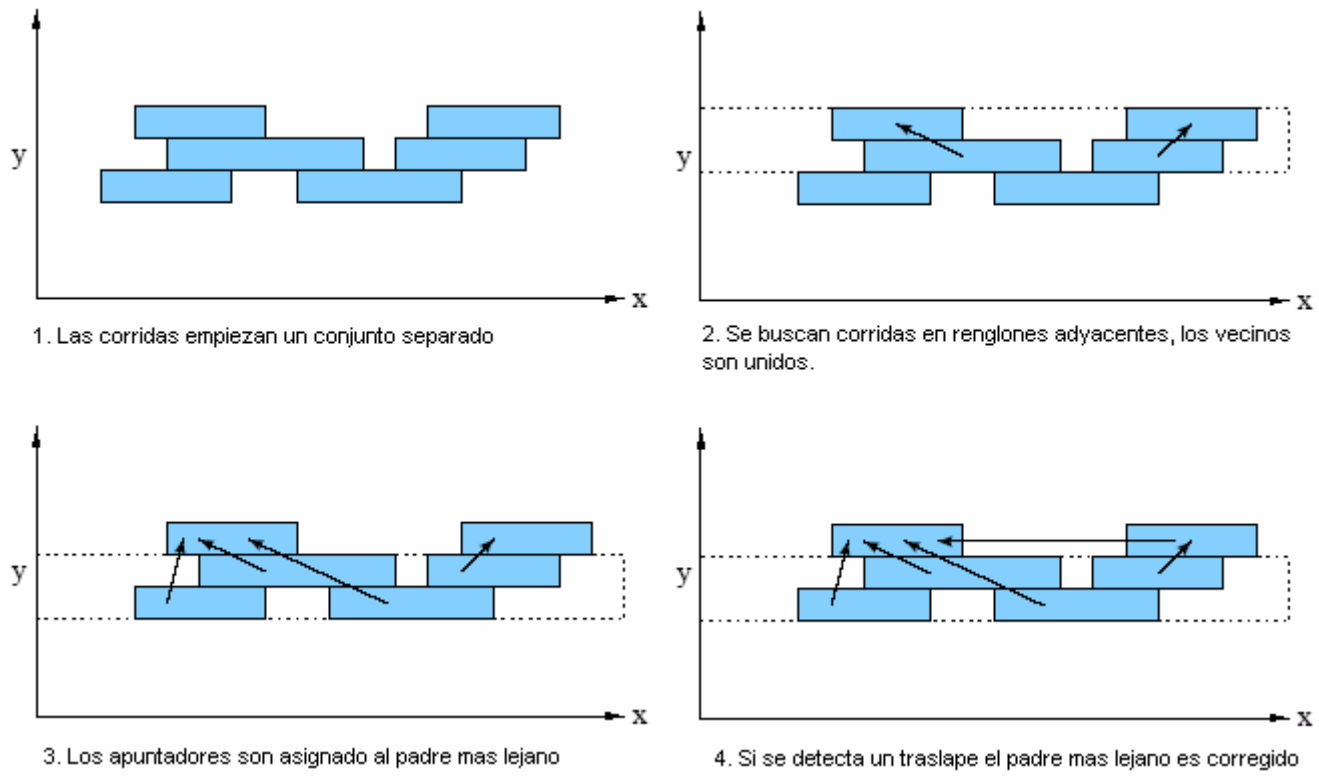


Figura 4.11 Método de fusión de corridas.

Se extrae información de la región recién formada como es: el cuadrado que delimita la región, el centroide, su tamaño, etc. Toda esta información es calculada de manera incremental en la última pasada del método de fusión. Después que las estadísticas han sido calculadas las regiones son separadas por colores y guardadas en su correspondiente lista de región.

4.5.3 Algoritmo de identificación

Una vez que se tienen agrupados los píxeles en regiones empieza el proceso de identificación.

Como se explicó anteriormente, para reconocer a la pelota simplemente se toma la región naranja que tenga el área mas cercana a 85 píxeles.
Primero se calcula el valor absoluto de la diferencia entre el área de la región y 85 (85 píxeles con el área que ocupa la pelota en la imagen). Por ejemplo si el área tiene un valor de 95 la diferencia es de 10, si tiene un valor de 60 la diferencia es 25. Después la lista de regiones es ordenada de menor a mayor diferencia utilizando Quicksort. La pelota es la región que tenga la menor diferencia.

Quicksort [HOA 61] es un algoritmo muy eficiente de ordenamiento. Tiene dos fases, la fase de partición y la fase de ordenamiento.
En la fase de partición se escoge un pivote y se ordenan los elementos de tal forma que los que se encuentran antes del pivote son menores que el pivote y los que se encuentran después deben ser mayores. Esto divide la lista en dos mas pequeñas.
En la fase de ordenamiento simplemente se ordenan las dos listas mas pequeñas llamando recursivamente al procedimiento de quicksort en cada una de ellas. Al concluir las llamadas recursivas se tiene la lista ordenada.

Para reconocer a los robots contrarios se sigue un procedimiento similar, solo que ahora la diferencia es entre el área de la región y 115 (el área del parche central es mayor que el área de la pelota) y la lista utilizada es la del color del parche central asignado al equipo contrario. En este caso los robots son las primeras cinco (o las que haya activado el usuario) regiones del arreglo ordenado.

Para identificar a nuestros robots se repite el procedimiento que ordena la lista de regiones del color del parche central asignado a nuestro equipo.
Los elementos de las listas de regiones de los colores de los parches extras 1 y parches extras 2 son unidos en una sola lista que después es ordenada bajo el mismo criterio utilizado en la lista de los parches centrales.
Después se empieza a recorrer la lista ordenada de los parches centrales; para cada elemento se recorre la lista de los parches extras buscando aquellas regiones cuyo centroide se encuentre dentro de un rectángulo alrededor del centroide del parche central. Se guardan las primeras cuatro regiones que cumplan ese criterio. De esta forma, tenemos 4 parches extras del tamaño adecuado que se encuentran alrededor del parche central. Lo que queda ahora es extraer el código binario de colores para obtener el identificador del robot. Para esto es necesario encontrar el orden correcto de los parches extras.

Lo que queremos es encontrar una trayectoria cerrada simple [GOO 02], es decir conectar los puntos sin que se crucen. La Figura 4.12 ilustra el problema.

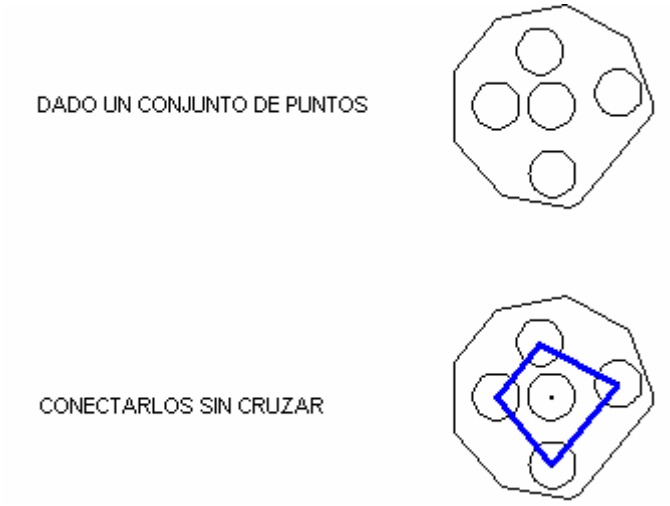


Figura 4.12 Trayectoria cerrada simple

Para generar la trayectoria se toma el punto mas bajo como ancla (A). Se calcula el ángulo del segmento de línea entre este punto ancla y el resto de los puntos. Se ordenan los puntos del menor al mayor ángulo. Como se ilustra en la Figura 4.13, el ángulo del segmento de línea (A,P1) es $\theta 1$ que es menor al ángulo del segmento de línea (A,P2) $\theta 2$.

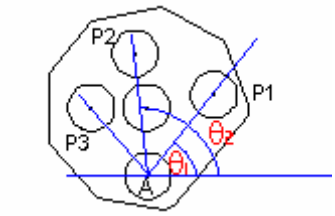


Figura 4.13 Ordenamiento por ángulos

Una vez ordenados, se distingue cuáles son los parches frontales de los traseros calculando la distancia entre los segmentos de línea (A,P1), (P1,P2), (P2,P3) y (P3,A). El segmento que tenga la mayor distancia es el formado por los parches frontales.
De esta manera quedan ordenados los parches y se puede obtener el código binario de colores para encontrar su identificador.

4.5.4 Algoritmo de localización

Una vez que se tiene los objetos identificados es necesario calcular la posición para la pelota, los robots contrarios y los robots de nuestro equipo. También se debe calcular la orientación de nuestros robots.

Para calcular las posiciones simplemente se elimina la distorsión causada por el lente gran angular. Se utiliza la siguiente ecuación:

$$\begin{aligned} \breve{x} &= x + x\left[k_1 r^2 + k_2 r^4\right] + \left[2p_1 xy + p_2\left(r^2 + 2x^2\right)\right] \\ \breve{y} &= y + y\left[k_1 r^2 + k_2 r^4\right] + \left[2p_2 xy + p_2\left(r^2 + 2y^2\right)\right] \end{aligned}$$

donde:

k_1, k_2 son los dos coeficientes de distorsión radial,

p_1, p_2 son los coeficientes de distorsión tangenciales,

(x, y) es la coordenada con distorsión,

$(\breve{x}, \breve{y})$ es la coordenada sin distorsión y

$r^2 = x^2 + y^2$.

Para calcular la orientación de nuestros robots se corrigen las coordenadas de los centroides de los parches frontales, se calcula el punto medio del segmento de línea formado por los parches y se calcula una recta entre ese punto medio y el centroide del parche central, finalmente, se obtiene la orientación del robot calculando el ángulo que forma esa recta con el plano.

5. IMPLEMENTACIÓN DEL SISTEMA

En este capítulo se expone el ***Sistema de Visión*** (**SV**) desde una perspectiva operativa. Se presentan las herramientas de hardware y software utilizadas en su desarrollo, la implementación del prototipo funcional, las pruebas y los resultados que se obtuvieron.

5.1 HERRAMIENTAS

En esta sección se describen las herramientas tanto de software como de hardware sobre las que se construyó el SV. Las herramientas empleadas determinan en gran medida el desempeño del sistema y constituyen uno de los factores claves en el funcionamiento de éste.

Se comienza exponiendo las herramientas de hardware empleadas y se continúa con una explicación detallada de las herramientas de software.

5.1.1 Hardware

El hardware permite situar el entorno físico que se utilizó para crear el sistema y al mismo tiempo permite entender algunas de las limitaciones con las que se tuvieron que lidiar.

Uno de las metas planteadas para el desarrollo del SV era que no tuviera un costo muy elevado. Se decidió que la única forma de lograrlo era utilizando hardware no especializado, es decir utilizar equipo que se puede conseguir prácticamente en cualquier tienda de cómputo.

La plataforma sobre la que corre el SV es una PC estándar:

Procesador	Intel Pentium 4 a 2.00 GHz
Memoria RAM	512 MB
Tarjeta de Vídeo	NVIDIA GeForce4 MX 420, 64MB

Las tarjetas de captura de video empleados fueron dos dispositivos MAXtv de MAXRON. Éstas tarjetas son genéricas y su costo aproximado es de $300 pesos. Comparado con el costo de las tarjetas especializadas de captura de vídeo empleadas por algunos equipos de la Liga de

robots pequeños es significativamente bajo (los costos de las tarjetas especializadas ascienden de $595 a $1095 dólares)[1]

Los dispositivos de captura de vídeo empleados son tarjetas compatibles Plug&Play PCI 2.1 basadas en el chipstet de Brooktree BT878. Pueden capturar video de los estándares NTSC, PAL o SECAM y en los espacios de color RGB y YUV. La velocidad de captura es 30 cuadros por segundo.

Las cámaras utilizadas son dos Sony Digital8 DCR-TRV19. Éstas cámaras tienen 3 tipos de salida de vídeo: IEEE1394, compuesto y S-Vídeo .
A cada cámara se le colocó un lente gran angular Sony VCL-ES06A. El lente gran angular es necesario para conseguir los ángulos de apertura requeridos para obtener una imagen completa de la mitad del campo de juego.

Se emplearon dos cables S-Vídeo de 15.2 m de longitud para conectar las cámaras a las tarjetas de captura de vídeo. La longitud del cable se debe a que debe recorrer los 4m de altura de la barra transversal para el montaje de las cámaras, la longitud del campo de juego (5.5m) y la distancia existente entre el poste de soporte de la barra transversal y nuestra computadora.

5.1.2 Software

El sistema operativo sobre el cual se desarrolló el SV fue Windows 2000 con Service Pack 4. Todo el sistema fue programado utilizando Microsoft Visual C++ 6.0. Se implementaron componentes de DirectShow para realizar todo el procesamiento de video y se utilizaron algunas funciones de OpenCV para la calibración geométrica de las cámaras. En la siguiente sección se explica qué son DirectShow y OpenCV.

5.1.2.1 DirectShow

DirectShow es una arquitectura de Microsoft para audio y video. Provee captura y reproducción de alta calidad para flujos multimedia. Es parte de DirectX y esta integrado con sus tecnologías, además, detecta y utiliza aceleración de video y audio por hardware cuando se encuentra disponible. Esta basado en COM (Component Object Model) que es un modelo de programación orientado a objetos [MSD 04].

DirectShow utiliza una arquitectura modular, donde cada etapa del procesamiento es hecha en un objeto COM llamado filtro. Un filtro es un objeto que realiza alguna operación en el flujo multimedia. Por ejemplo, un filtro puede leer archivos, obtener video de un dispositivo de

[1] Matrox Meteor-II a Matrox Meteor-II/Digital Fuente: http://www.ShopMatrox.com

captura, decodificar video o pasar datos a la tarjeta gráfica. Los filtros reciben una entrada y generan una salida.

En DirectShow una aplicación realiza una tarea conectando los filtros en cadena, de tal forma que la salida de un filtro sea la entrada de otro. Un conjunto de filtros conectados de esta forma es llamado una gráfica.
Los puntos de conexión entre los filtros también son objetos COM y son llamados pines. Los pines se utilizan para mover los datos de un filtro al siguiente.

Los filtros pueden ser divididos en las siguientes categorías [MSD 04]:

- Fuente. Un filtro fuente introduce datos en la gráfica. Los datos pueden venir de un archivo, una red, una cámara, etc.
- Transformación. Un filtro de transformación toma un flujo de entrada, procesa los datos y crea un flujo de salida. Codificadores y decodificadores son ejemplos de este tipo de filtros.
- Despliegue. Un filtro de despliegue se encuentra al final de la cadena. Reciben datos y se los presentan al usuario. Por ejemplo, un filtro de despliegue dibuja los cuadros de video en la pantalla.
- Divisor. Un filtro divisor separa el flujo de entrada en dos o mas salidas.
- Multiplexor. Toma múltiples entradas y las combina en una sola salida..

Los filtros son controlados por un componente de alto nivel llamado Manejador de la Gráfica. La aplicación hace llamadas de alto nivel como “Corre” (para empezar a mover los datos a través de la gráfica) o “Detente” (para detener el flujo de datos). Si se requiere mas control sobre las operaciones en el flujo, se puede acceder directamente a los filtros utilizando interfaces COM. El Manejador de la Gráfica manda mensajes de eventos a la aplicación y también proporciona métodos para construir la gráfica, conectando un filtro con otro.

Para escribir una aplicación de DirectShow hay tres pasos que deben ser realizados [MSD 04]:

1. La aplicación debe instanciar un Manejador de la Gráfica.
2. La aplicación utiliza el Manejador de la Gráfica para construir una gráfica. El conjunto de filtros en la gráfica depende de la aplicación.
3. La aplicación utiliza el Manejador de la Gráfica para controlar el flujo de datos en los filtros. La aplicación también debe responder a los eventos enviados por el Manejador de la Gráfica.

Cuando el procesamiento concluye , la aplicación debe liberar los recursos del Manejador de la Gráfica y de los demás filtros.
La Figura 5.1 muestra los pasos necesarios para escribir una aplicación de DirectShow.

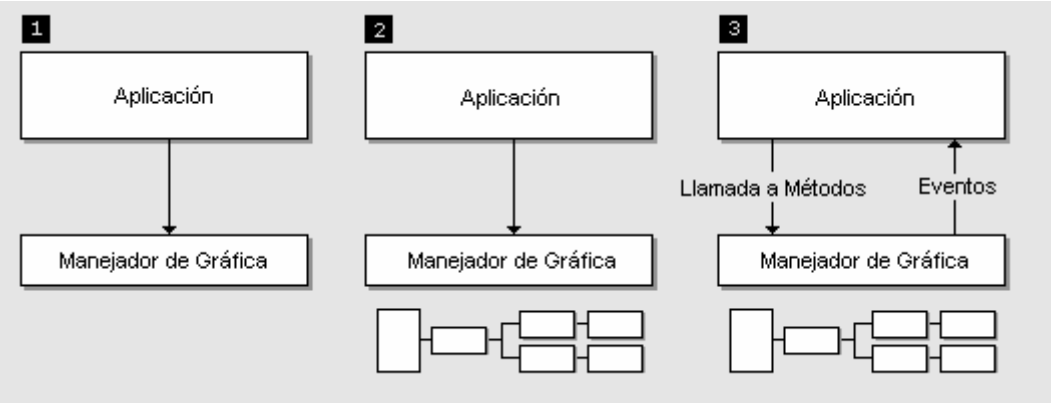


Figura 5.1 Pasos para escribir una aplicación de DirectShow

Como se mencionó, DirectShow está basado en COM. El Manejador de la Gráfica y los filtros son objetos COM. La siguiente sección explica su funcionamiento.

5.1.2.2 COM

Los objetos COM son básicamente cajas negras que pueden ser utilizadas por aplicaciones para realizar una o mas tareas. Generalmente son implementados como librerías ligadas dinámicas (DLL). Como cualquier DLL, los objetos COM exponen métodos que la aplicación puede llamar para realizar alguna función. Las aplicaciones interactúan con los objetos COM de una manera similar a como lo hacen con los objetos de C++. Sin embargo, existen algunas diferencias significativas [MSD 04]:

- Los objetos COM exigen una encapsulación mas estricta que los objetos C++. Es decir, no se puede simplemente crear el objeto y llamar cualquier método público. Los métodos públicos de los objetos COM están agrupados en una o mas interfaces. Para usar un método, se debe crear el objeto y obtener la interface apropiada. Una interface típica contiene un conjunto de métodos relacionados que dan acceso a alguna funcionalidad particular del objeto. Cualquier método que no sea parte de una interface es inaccesible.
- Los objetos COM no son creados de la misma manera que los objetos de C++. Existen diferentes formas de crear un objeto, pero todas ellas involucran técnicas específicas de COM. El API de DirectX incluye una variedad de funciones y métodos auxiliares que simplifican la creación de la mayor parte de los objetos de DirectX.
- Se deben utilizar técnicas específicas de COM para controlar el ciclo de vida del objeto. Los objetos COM no necesitan ser explícitamente cargados. Típicamente están contenidos en una DLL, pero tampoco se necesita cargar explícitamente la DLL o ligar a una librería estática para usarlos. Cada objeto COM tiene un identificador único registrado en el sistema que es usado para crearlo. COM automáticamente carga la DLL correcta.

- Los objetos COM son una especificación binaria. Pueden ser escritos y accesados desde una gran variedad de lenguajes. Por ejemplo, aplicaciones escritas en Visual Basic pueden utilizar objetos COM escritos en C++.

Es importante distinguir la diferencia entre un objeto y una interface. En un objeto se encuentran encapsulados métodos; las interfaces sirven para exponer esos métodos al exterior del objeto. Una interface describe la sintaxis de los métodos que expone y su funcionalidad.

Si un objeto expone una interface, todos los métodos de ésta deben estar soportados por el objeto, es decir, se puede llamar un método con la confianza de que éste existe. Sin embargo, los detalles de implementación de cada método pueden variar de un objeto a otro porque pueden usar diferentes algoritmos para llegar al mismo resultado.

Para crear un objeto COM hay dos formas principales:

- Directamente. Utilizando su identificador único (CLSID) en la función CoCreateInstance. Ésta función crea una instancia pero además entrega un apuntador a la interface que se haya especificado. Al crear el objeto se tiene que solicitar una interface para tener acceso a los métodos. Esto es así porque el objeto no sirve de nada sin la interface. La interface solicitada puede ser cualquiera que este soportada por el objeto.
- Indirectamente. La creación del objeto se hace a través del llamado de un método de DirectX. El método debe regresar un apuntador a una interface, pero generalmente no se puede especificar la interface que es regresada.

Si se desea obtener una interface diferente a la que fue regresada al momento de la creación del objeto, no es necesario instanciar un objeto nuevo, simplemente se puede llamar a la función QueryInterface del objeto creado. La función QueryInterface necesita el IID (identificador de la interface) y la dirección del apuntador que guardará la interface que el método regresa.

Cuando un objeto COM es creado, el sistema asigna los recursos de memoria necesarios. Al terminar de usarlo los recursos deben de ser liberados. Sin embargo, COM no permite la destrucción de un objeto como sucede con los objetos de C++. La razón de esto es que un mismo objeto puede ser utilizado por varias aplicaciones para hacer mas eficiente el uso de recursos. Lo que hace COM es llevar la cuenta del número de veces que una interface de un objeto ha sido solicitada, al liberar una interface la cuenta se reduce en uno; mientras la cuenta se mayor a cero el objeto permanece en memoria, si se llega a cero el objeto es destruido. De esta forma, si una aplicación libera las interfaces de manera adecuada el sistema manejará correctamente el ciclo de vida del objeto.

5.1.2.3 OpenCV

OpenCV (Open Source Computer Vision Library) es una librería desarrollada por Intel. Es una colección de funciones en C y C++ que implementan algunos de los algoritmos más populares en el procesamiento digital de imágenes [INT 01].

Fue originalmente desarrollada para proveer una infraestructura libre y abierta donde los esfuerzos distribuidos de la comunidad de visión por computadora pudieran ser consolidados y optimizados. Algunas áreas de aplicación son interacción computadora-humano (HCI), identificación de objetos, segmentación y reconocimiento, rastreo de movimientos, reconocimiento de rostros, reconocimiento de gestos, monitoreo, entendimiento de movimiento, biométricas, robots móviles, etc.

Es principalmente una librería de alto nivel que implementa algoritmos como técnicas de calibración de cámaras, detección de características y rastreo, análisis de figuras, análisis de movimiento, reconstrucción en 3D, segmentación y reconocimiento de objetos.
La principal característica de la librería además de su funcionalidad es el desempeño. Los algoritmos están basados en estructuras de datos dinámicas y mas de la mitad de las funciones fueron optimizadas a nivel de ensamblador para tomar ventaja de la arquitectura de los procesadores Intel [INT01].

Para poder utilizar muchas de las funciones de OpenCV es necesario tener la imagen en una estructura tipo IplImage. Ésta estructura contiene la siguiente información:

- Ancho y altura de la imagen en píxeles
- La profundidad del píxel en bits. Es decir, con cuantos bits se representa un píxel. Los valores aceptados son 8,16 y 32 bits con signo y sin signo.
- El número de canales. El numero de dimensiones en el espacio de color. RGB tiene 3 canales mientras que una imagen de grises tiene un solo canal.
- Origen. Indica en donde empieza la imagen, puede ser la esquina superior izquierda o la esquina inferior derecha.
- El orden de los datos. Indica si los componentes de una imagen de color vienen entrelazados o separados.
- Un apuntador a los datos de la imagen.

Las funciones que contiene OpenCV son muy variadas, existen desde operaciones sencillas como dibujar líneas y círculos hasta funciones que hallan regiones en la imagen donde se encuentra una mano humana y reconocen el gesto que esta realizando.

5.2 IMPLEMENTACIÓN DEL PROTOTIPO FUNCIONAL

Al utilizar OpenCV en filtros de DirectShow se combina el desempeño de una librería altamente optimizada y la versatilidad de DirectShow que brinda acceso instantáneo a una gran variedad de cámaras, reproductores y formatos de video.

La implementación del SV fue dividida en dos partes. La primera fue el desarrollo de un filtro de transformación en DirectShow donde se realiza la mayor parte del procesamiento del video. La segunda fue la aplicación propiamente dicha donde se configura el sistema y se construye la gráfica en que se utiliza el filtro desarrollado.

5.2.1 Filtro de Transformación

En este filtro de transformación se realiza prácticamente todo el procesamiento de video. Se encarga de la segmentación, formación de regiones, reconocimiento e identificación y de la transmisión.
La captura de video y el despliegue en pantalla se hace en otros filtros especializados.
El filtro es un conjunto de clases que interactúan entre si para obtener los datos de video del dispositivo de captura, procesarlos y entregarlos para que sean desplegados.
El filtro tiene dos pines de entrada (para los dispositivos de captura) y dos pines de salida (para mostrar el video en pantalla). La Figura 5.2 muestra el filtro de transformación.

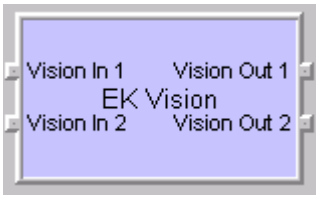


Figura 5.2 Filtro de Transformación

El SDK de DirectShow incluye un conjunto de clases de C++ para escribir un filtro. La clase CBaseFilter define la clase raíz de donde heredan todos los filtros. Para crear un nuevo filtro se debe escribir un clase que herede de ella. La definición de nuestro filtro es la siguiente:

```
class CVisionFilter : public CBaseFilter,
                      public IIPParam,
```

Como se puede observar nuestro filtro hereda de la clase CBaseFilter pero también expone la interface IIPParam.

La interface IIPParam es la interace que utiliza el filtro para comunicarse con el exterior. Los métodos que expone son únicamente dos: get_IPParam y put_IPParam.
Como se puede adivinar por sus nombres son métodos para obtener y poner los parámetros utilizados por el filtro. Los parámetros están contenidos en la siguiente estructura:

```
typedef struct
{
        //Segmenter array for camera input 1
        BYTE    Yarreglo1[255];
        BYTE    Uarreglo1[255];
        BYTE    Varreglo1[255];
        //Segmenter array for camera input 2
        BYTE    Yarreglo2 [255];
        BYTE    Uarreglo2[255];
        BYTE    Varreglo2[255];
        //Color and search for robots
        int  color;
        BOOL EK[5];           //Eagle Knights Robots TRUE->active
        BOOL OT[5];           //Opposite Team TRUE->active
        CvCameraParams paramCamera1;
        CvCameraParams paramCamera2;
} Param;
```

Esta estructura contiene toda la información que es necesaria para el procesamiento de video: los arreglos utilizados en el algoritmo de segmentación explicados en la sección 4.5.1, el color del parche central de los robots de nuestro equipo, un par de arreglos indicando que robots están activados y que por tanto hay que buscar, y finalmente una estructura de OpenCV con los parámetros intrínsecos y extrínsecos de las dos cámaras.

El filtro debe crear los pines de salida y de entrada. Los pines generalmente heredan de la clase CBasePin o de alguna clase hija de ésta como CBaseInputPin o CBaseOutputPin. Los pines del filtro deben ser declarados como variables miembros. En nuestro filtro tenemos 4 pines:

```
CVisionInputPin1 *m_pInput1;
CVisionInputPin2 *m_pInput2;
CVisionOutputPin1 *m_pOutput1;
CVisionOutputPin2 *m_pOutput2;
```

Las clases CVisionInputPin1 y CVisionOutput1, heredan de CBaseInputPin y CBaseOutputPin.

El filtro debe tomar en consideración que cuando el Manejador de la Gráfica intenta conectar dos filtros los pines de entrada y salida involucrados en la conexión deben ponerse de acuerdo en varias cosas. Si no pueden, el intento fracasa. Los pines negocian lo siguiente:

- Transporte. El transporte es el mecanismo que los filtros utilizan para mover las muestras de datos de un pin de salida a un pin de entrada. Por ejemplo, pueden utilizar la interface IMemInputPin (modelo empujar) o IAsyncReader (modelo jalar).
- El tipo de datos. Se utiliza para describir el formato de los datos (espacio de color) que pueden entregan o recibir.
- Recursos. Los pines deben acordar cual de ellos proveerá los recursos para manejar los datos. También deben acordar el tamaño del buffer, el número de buffers y sus propiedades.

Los pines del filtro desarrollado trabajan con el modelo empujar. En este modelo un filtro tipo fuente (como un dispositivo de captura de video) genera datos y los envía al filtro conectado a su pin de salida. Éste filtro recibe pasivamente los datos, los procesa y los envía al siguiente filtro. En el modelo jalar, en lugar de recibir los datos pasivamente, el filtro los solicita a la fuente y ésta responde enviando los datos.

El tipo de datos que puede procesar el filtro es YUV, la siguiente función verifica que el tipo de datos entregado por el filtro fuente sea el correcto. Primero verifica que el tipo de datos sea de video y después verifica que el subtipo (formato o espacio de color) sea YUV.

```
if (IsEqualGUID(*pMediaType->Type(), MEDIATYPE_Video))
        if (IsEqualGUID(*pMediaType->Subtype(), MEDIASUBTYPE_YUY2))
                return TRUE;
return FALSE;
```

Una vez que las conexiones entre los pines se han completado, el pin de entrada esta listo para recibir los datos del video. El filtro fuente (dispositivo de captura) invoca el método Receive del pin de entrada. El pin debe entonces enviar estos datos al filtro de transformación para que realice el procesamiento.

Es importante señalar que a pesar de todos estos detalles técnicos propios de DirectShow, el diseño expuesto en el capítulo anterior se conserva. Todo el procesamiento de video está contenido en este objeto COM y las clases que lo realizan como el Segmentador, Generador de Blobs, Identificador de Objetos, Localizador y Transmisor se encuentran dentro de él, en realidad lo que sucede es que el filtro ha encapsulado todo la funcionalidad y le ha otorgado ventajas que se pueden explotar como son:

- Los dispositivos de captura son independientes del procesamiento de video. Se puede conectar al filtro de transformación cualquier dispositivo de captura que sea capaz de entregar video en formato YUV.
- El despliegue en pantalla es también independiente. Esto permite cambiar la tarjeta de video sin tener un impacto desfavorable en el desempeño del procesamiento de video,

o bien, si no se cuenta con suficiente capacidad para desplegar el video se puede optar por no hacerlo.

- La aplicación que construye la gráfica puede realizarse en cualquier lenguaje de programación que soporte COM, esto da una gran libertad para la realización de actualizaciones.

5.2.2 Aplicación

La aplicación es la encargada de construir la gráfica de procesamiento, es decir, es la responsable de conectar en el orden adecuado a los filtros de los dispositivos de captura, el filtro de transformación y los filtros de despliegue. También se encarga de construir la gráfica de calibración geométrica para las cámaras y de desplegar las pantallas de interface con el Usuario.

Para construir la gráfica de filtros es necesario tener un Manejador de la Gráfica. La siguiente función crea el objeto y obtiene la interface IGraphBuilder:

```
HRESULT CManager::InitGraphBuilder(IGraphBuilder **ppGraph)
{
  IGraphBuilder *pGraph = NULL;
  // Create the Filter Graph Manager.
  HRESULT hr = CoCreateInstance(CLSID_FilterGraph, 0,
                 CLSCTX_INPROC_SERVER,IID_IGraphBuilder,(void**)&pGraph);
  if (SUCCEEDED(hr)){
    // Return interface pointer to the caller.
    *ppGraph = pGraph; // The caller must release both interfaces.
    return S_OK;
  }
  return hr; // Failed
}
```

Una vez que se tiene el Manejador de la Gráfica se puede empezar a agregar filtros a la gráfica. Los primeros que se agregan en la aplicación del SV son los dispositivos de captura.

En DirectShow cualquier dispositivo de captura de video con un driver WDM (Windows Driver Model) puede ser agregado a la gráfica como un filtro fuente. Éste filtro se configura a si mismo basado en las características del driver.
Algunos dispositivos de captura viejos utilizan drivers de VFW (Video for Windows). Aunque estos drivers son obsoletos están soportados por DirectShow a través de un wrapper.
Como se describió en los casos de uso y diagramas de secuencia de la inicialización del sistema, el Usuario debe seleccionar un dispositivo para cada una de las cámaras. Para eso, es necesario obtener los dispositivos instalados en el sistema a través de un enumerador, el

siguiente fragmento de código muestra como se obtiene el enumerador, se especifica la categoría de filtros que se desea recorrer, se obtienen sus propiedades y se agregan a una lista:

```
ICreateDevEnum* pDevEnum = NULL;     //Device enumerator
IEnumMoniker*   pEnum    = NULL;     //Moniker
int             countDev = 0;        //Counter for devices
// Create the System Device Enumerator.
HRESULT hr = CoCreateInstance(CLSID_SystemDeviceEnum,NULL,CLSCTX_INPROC_SERVER,
                              IID_ICreateDevEnum, reinterpret_cast<void**>(&pDevEnum));
if ( SUCCEEDED(hr) ) // Create enumerator for the video capture category.
   hr = pDevEnum->CreateClassEnumerator(CLSID_VideoInputDeviceCategory,&pEnum, 0);
//Atach a moniker to the device enumerator
IMoniker* pMoniker = NULL;
while (pEnum->Next(1, &pMoniker, NULL) == S_OK)
{
   //Obtain the properties from the moniker
   IPropertyBag* pPropBag;
   hr = pMoniker->BindToStorage(0, 0, IID_IPropertyBag,(void**)(&pPropBag));
   if ( FAILED(hr) ){
      SafeRelease(pMoniker);
       continue;  // Skip this one, maybe the next one will work.
   }
   // Find the description or friendly name.
   VARIANT varName;
   VariantInit(&varName);
   hr = pPropBag->Read(L"Description", &varName, 0);
   if ( FAILED(hr) )
      hr = pPropBag->Read(L"FriendlyName", &varName, 0);
   if ( SUCCEEDED(hr) ){
      // Add it to the application's list box.
       USES_CONVERSION;
       hList->AddString(OLE2T(varName.bstrVal));
       countDev++;
   }
}
```

Es ahora cuando el Usuario debe seleccionar un dispositivo de la lista que ha sido desplegada en un dialogo como se muestra en la Figura 5.3. Cuando presiona OK, se guarda el identificador del dispositivo y se liga al Moniker (objeto COM que contiene información de otro objeto) para agregarlo a la gráfica:

```
IBaseFilter *baseFilter;
//Check if number of device is the one selected for the user
if (num == number) {

  found= true;
  //Select it
```

```
  hr= pMoniker->BindToObject(NULL, NULL, IID_IBaseFilter, (void**)&baseFilter);
  if (FAILED(hr)){
     printf("Error");
     return hr;
  }
  else {
     //Add it to the graph
     hr = pGB->AddFilter(baseFilter, varName.bstrVal);
  }
}
```

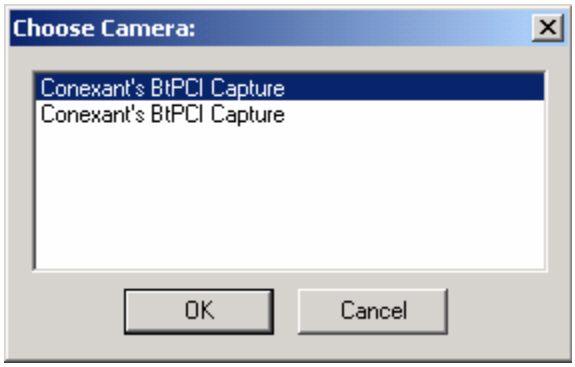


Figura 5.3 Pantalla de Selección de Cámara

Ahora es momento de configurar el formato de datos (espacio de color) que va a entregar el dispositivos de captura. Primero se obtiene el primer pin de salida del filtro de captura y la interface de configuración del flujo:

```
//Configure the format for capture pin
IAMStreamConfig *pConfig = NULL;
//Get first output pin from capture filter
pPinCap = GetPin(pCap,PINDIR_OUTPUT);
//Get interface to configure stream
hr = pPinCap->QueryInterface(IID_IAMStreamConfig, (void**)&pConfig);
```

Después se interroga por el número de formatos que soporta y se examinan todos hasta encontrar uno que sea del tipo YUV, cuando lo encontramos se lo asignamos al pin de salida. Este procedimiento es repetido para las dos cámaras.

```
   int iCount = 0, iSize = 0;
   //Get number of capabilities
   hr = pConfig->GetNumberOfCapabilities(&iCount, &iSize);
   // Check the size to make sure we pass in the correct structure.
   if (iSize == sizeof(VIDEO_STREAM_CONFIG_CAPS)) {
     // Use the video capabilities structure.
     for (int iFormat = 0; iFormat < iCount; iFormat++) {
       VIDEO_STREAM_CONFIG_CAPS scc;
       //Obtain capabilities
       hr = pConfig->GetStreamCaps(iFormat, &pmtConfig, (BYTE*)&scc);
```

```
        if (SUCCEEDED(hr)) {
           /* Examine the format, and possibly use it. */
           if ((pmtConfig->majortype == MEDIATYPE_Video) &&
                (pmtConfig->subtype == MEDIASUBTYPE_YUY2) &&
                (pmtConfig->formattype == FORMAT_VideoInfo) &&
                (pmtConfig->cbFormat >= sizeof (VIDEOINFOHEADER)) &&
                (pmtConfig->pbFormat != NULL))
                {
                  //We find a YUV video type, set custom size to the format
                  pVih->bmiHeader.biWidth = WIDTH;
                  pVih->bmiHeader.biHeight = HEIGHT;
                  pVih->bmiHeader.biSizeImage = DIBSIZE(pVih->bmiHeader);
                 //Set format
                 hr = pConfig->SetFormat(pmtConfig);
               break;
             }
           }
        }
    }
```

El siguiente paso es crear el filtro de transformación y los filtros de despliegue:

```
IBaseFilter* pVision = NULL;
// Create the Vision filter and add it to the filter graph. CLSID_Vision
HRESULT hr = CoCreateInstance(CLSID_Vision, NULL,CLSCTX_INPROC, IID_IBaseFilter,
                              (void**)&pVision);
if (SUCCEEDED(hr)) {
   //Add it to the graph
   hr = pGB->AddFilter(pVision, L"EK Vision");
}
```

Para crear el filtro de despliegue:

```
IBaseFilter* pVmr = NULL;
// Create the VMR and add it to the filter graph.
HRESULT hr = CoCreateInstance(CLSID_VideoMixingRenderer9, NULL,CLSCTX_INPROC,
                              IID_IBaseFilter, (void**)&pVmr);
if (SUCCEEDED(hr)){
    //Add it to the filter
    hr = pGB->AddFilter(pVmr, L"Video Mixing Renderer 9");
}
```

Finalmente se tienen que conectar todos los filtros. Para conectar cualquier par de filtros se necesita un pin de salida y el filtro con el que se desea hacer la conexión. La siguiente función obtiene un pin de entrada no utilizado del filtro destino e intenta conectar los pines de manera directa, si falla, utiliza “Smart Connect”, esta funcionalidad de DirectShow examina los filtros

registrados en el sistema e intenta encontrar alguno que pueda utilizarse entre los filtros para lograr la conexión (generalmente el filtro insertado es un filtro que cambia el formato de datos).

```
HRESULT CManager::ConnectFilters(IGraphBuilder *pGraph, IPin *pOut, IBaseFilter *pDest,
                                 AM_MEDIA_TYPE *pmtConfig)
{
  // Find an input pin on the downstream filter.
   IPin *pIn = 0;
   //Get a pin that hasn´t been connected
   HRESULT hr = GetUnconnectedPin(pDest, PINDIR_INPUT, &pIn);
   if (FAILED(hr))
        return hr;
   // Try to connect them directly.
   hr= pGraph->ConnectDirect(pOut,pIn,pmtConfig);
   if( FAILED(hr))     //Can´t do it
      hr = pGraph->Connect(pOut, pIn); //Use smart connect
   SafeRelease(pIn);
   return hr;
}
```

La gráfica completa con todos los filtros involucrados en el procesamiento de video es mostrada en la Figura 5.4.

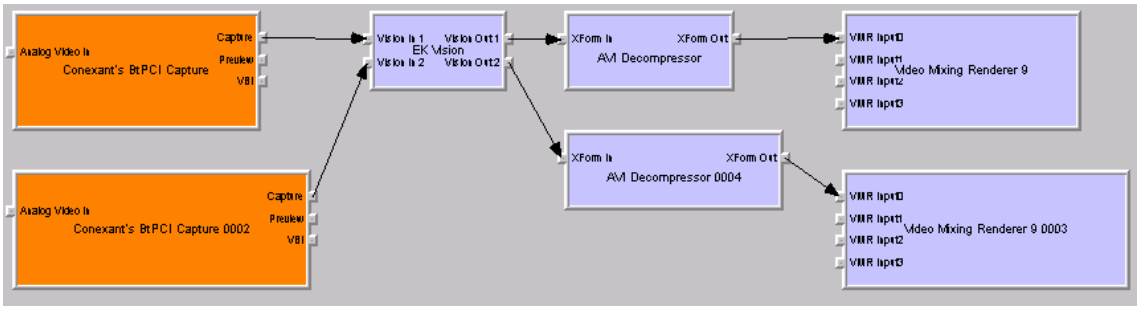


Figura 5.4 Gráfica de procesamiento

Para la calibración geométrica de las cámaras se utiliza un filtro de DirectShow que utiliza funciones de OpenCV. La creación del filtro se hace de la siguiente manera:

```
IBaseFilter* pCalib = NULL;
// Create the Calibration filter and add it to the filter graph. CLSID_Vision
HRESULT hr = CoCreateInstance(CLSID_CCalibFilter, NULL,CLSCTX_INPROC, IID_IBaseFilter,
                              (void**)&pCalib);
if (SUCCEEDED(hr))  //Add it to the graph
    hr = pGB->AddFilter(pCalib, L"EK Calibration");
```

Internamente el filtro realiza el siguiente procesamiento:

1. Crea una imagen IplImage

```
IplImage*         m_rgb_img;
AM_MEDIA_TYPE* pType = &m_pInput->CurrentMediaType();
VIDEOINFOHEADER *pvi = (VIDEOINFOHEADER *) pType->pbFormat;
uchar*  rgb_data;
CvSize size = cvSize( pvi->bmiHeader.biWidth, pvi->bmiHeader.biHeight );
int  step = (size.width*3 + 3) & -4;
pSample->GetPointer(&rgb_data);
assert( pvi->bmiHeader.biBitCount == 24 );
cvInitImageHeader(  m_rgb_img, size, IPL_DEPTH_8U, 3, IPL_ORIGIN_TL, 4 );
cvSetImageData( m_rgb_img, rgb_data, step );
```

2. Convierte la imagen a blanco y negro y encuentra esquinas de los cuadrados.

```
/* Begin of find etalon points */
int count = etalon_points;// + 10;
cvCvtColor( rgb_img, m_gray_img, CV_BGR2GRAY );
/*****************************************************/
//////////////   FIND CHECKERBOARD CORNERS   //////////////
/////////////////////////////////////////////////////////
chess_found = cvFindChessBoardCornerGuesses( m_gray_img, m_thresh_img, 0,
                                             etalon_size, pt_ptr, &count ) != 0;
if( count != 0 )
  cvFindCornerSubPix(m_gray_img, pt_ptr, count, cvSize(5,5), cvSize(-1,-1),
                cvTermCriteria( CV_TERMCRIT_ITER|CV_TERMCRIT_EPS, 10, 0.01f ));
DrawEtalon( rgb_img, pt_ptr, count, etalon_size, chess_found );
```

3. Al terminar de colectar el número de imágenes solicitadas por el usuario, calcula los parámetros de la cámara.

```
if( m_params.frames_collected == m_params.frames_to_collect )
{
    /* all frames are collected. Now will calibrate */
    /* Calibrate camera */
    cvCalibrateCamera( m_params.frames_collected, m_numsPoints,
                       size, m_imagePoints, m_objectPoints,
                       m_camera.distortion, m_camera.matrix,
                       (float*)m_transVects, m_rotMatrs, 0 );
    /* Copy some camera parameters */
    m_camera.focalLength[0] = m_camera.matrix[0];
    m_camera.focalLength[1] = m_camera.matrix[4];

    m_camera.principalPoint[0] = m_camera.matrix[2];
    m_camera.principalPoint[1] = m_camera.matrix[5];
    m_params.calib_state = CalibState_Calibrated;
    SetDirty(TRUE);
}/* End calibration */
```

La Figura 5.5 muestra una imagen del procedimiento de calibración de la cámara.

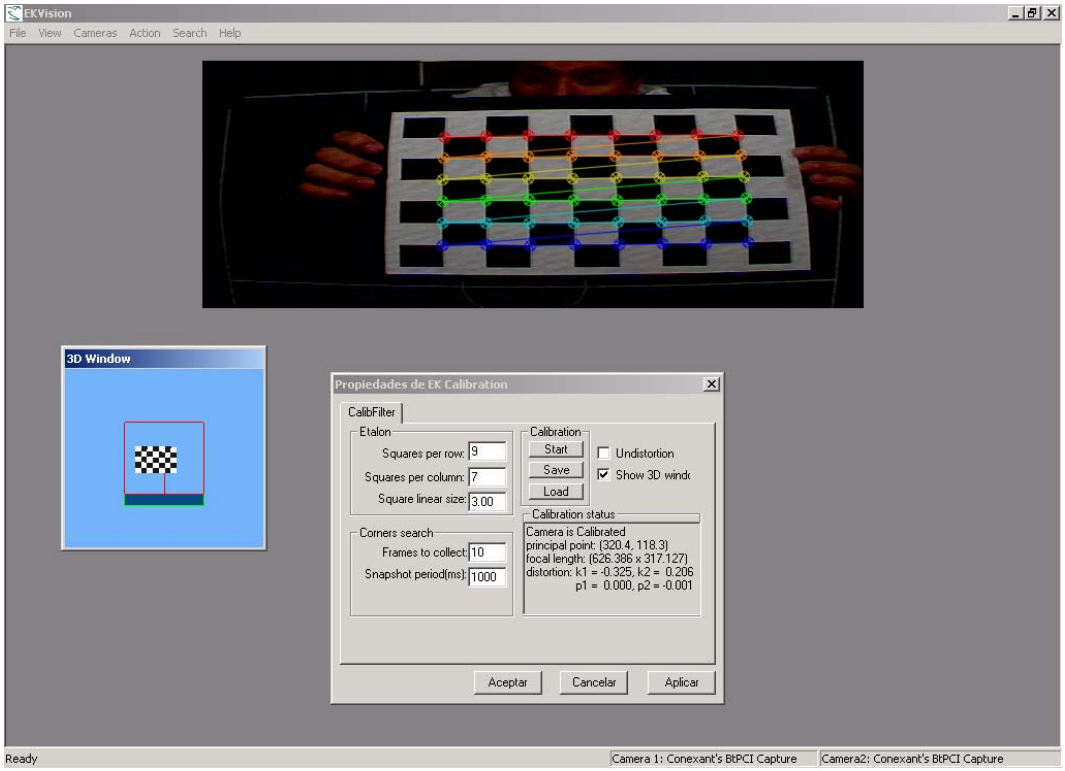


Figura 5.5 Procedimiento de calibración de la cámara.

La Figura 5.6 muestra la pantalla principal del SV localizando a cinco robots, la Figura 5.7 muestra la interface del Sistema de IA recibiendo la información.

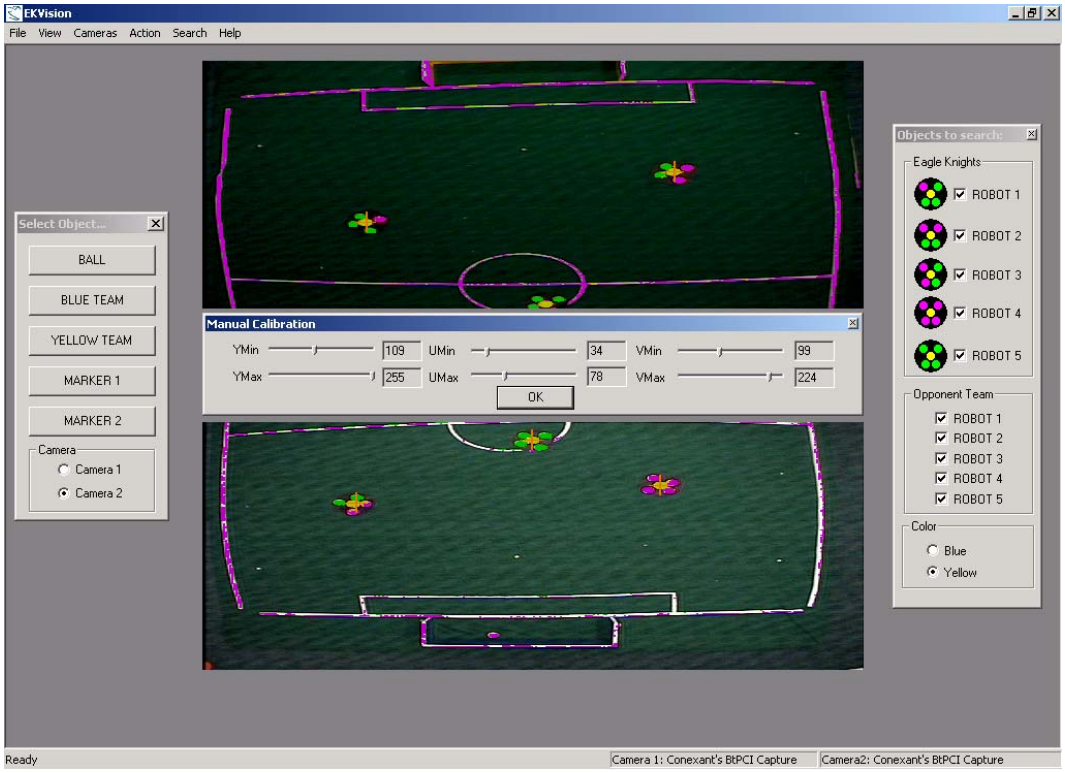


Figura 5.6 Pantalla Principal del Sistema de Visión

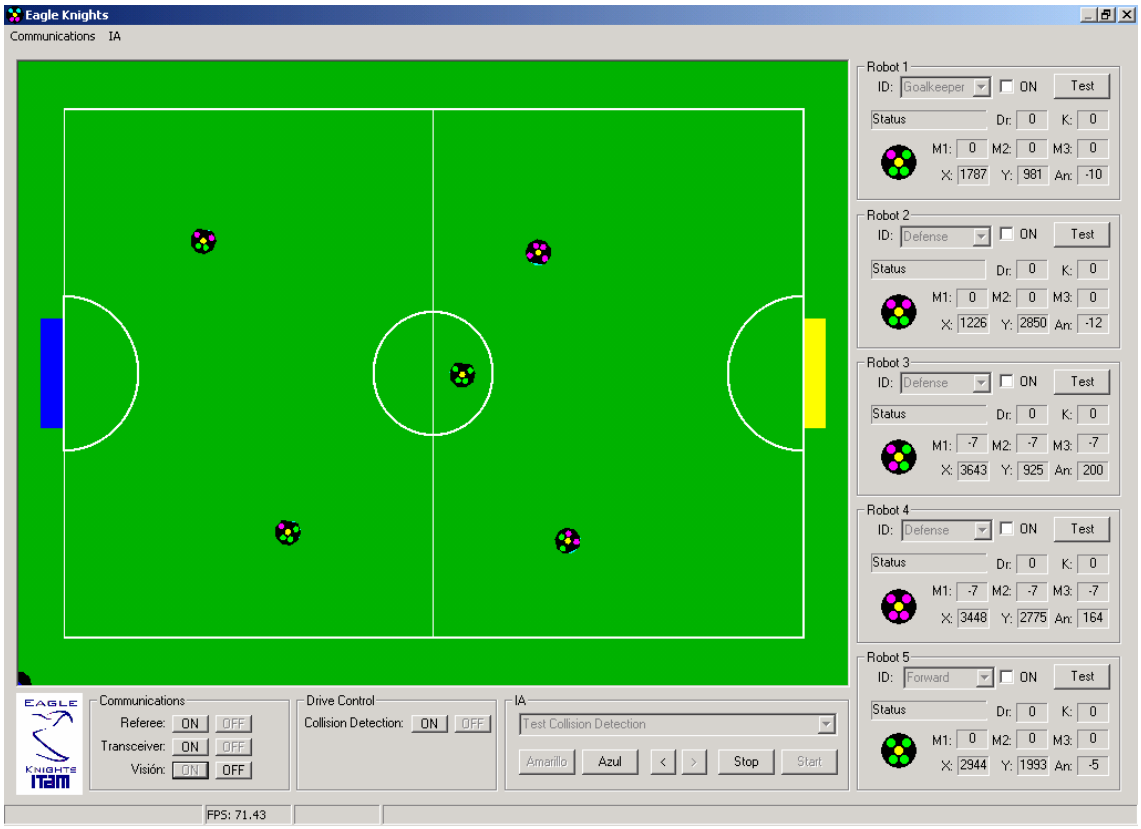


Figura 5.7 Interface del Sistema IA

6. PRUEBAS Y RESULTADOS

En este capítulo se presentan las pruebas realizadas al Sistema de Visión y los resultados obtenidos. Las pruebas estuvieron orientadas a medir la confiabilidad del sistema en la localización de la pelota y los robots en tiempo real.

Se realizaron siete pruebas de desempeño:

- Velocidad de procesamiento.
- Porcentaje de pérdida o localización errónea de la pelota.
- Porcentaje de pérdida de los robots contrarios.
- Porcentaje de pérdida de nuestros robots.
- Porcentaje de identificación errónea.
- Variación en el cálculo de posición.
- Variación en el cálculo de orientación.

Las pruebas de porcentajes y variación se realizaron en el módulo de comunicación del Sistema de IA el cual recibe los paquetes de información enviados por el SV. Cada paquete de información contiene la posición de la pelota, la posición de cada robot (nuestro o contrario) y la orientación de nuestros robots.

6.1 VELOCIDAD DE PROCESAMIENTO

La primer prueba realizada es obtener la velocidad de procesamiento del SV. Se calcula el tiempo que tarda el sistema en capturar, procesar y desplegar un cuadro de video. Para tomar los tiempos se utilizó el reloj de la computadora. La medición se realizó en dos puntos distintos:

1. El filtro de despliegue. Este filtro es el último de la cadena de procesamiento (ver Figura 5.4). Se encarga de presentar el video en pantalla. Para calcular la velocidad de procesamiento se mide el tiempo que transcurrió entre el despliegue de dos cuadros consecutivos de video. Al llegar un cuadro se toma el tiempo del sistema y se guarda en una variable t_0, se espera la llegada de un segundo cuadro y nuevamente el tiempo del sistema es guardado en una variable t_1. El tiempo que tomó procesar un cuadro es la diferencia entre los tiempos (t_1-t_0).
2. El Sistema de IA. Éste sistema recibe los paquetes de información enviados por el SV. Calcula el tiempo que tarda el SV en entregar dos paquetes consecutivos de información sobre el estado del juego. El tiempo t_0 se toma cuando llega el primer paquete. Al llegar el segundo paquete de información se toma el tiempo t_1. La

diferencia entre los tiempos (t_1-t_0) es el tiempo requerido por el SV para procesar y transmitir la información.

Para calcular los cuadros por segundo se utiliza la siguiente fórmula:

$$fps = \frac{1}{t_1 - t_0}$$

La Tabla 6.1 muestra los resultados obtenidos en las mediciones realizadas en cada uno de estos puntos.

Punto de Medición	Cuadros por segundo
Filtro de Despliegue Cámara 1	59.04
Filtro de Despliegue Cámara 2	59.04
Sistema de IA	58.82

Tabla 6.1 Mediciones de velocidad de procesamiento

La velocidad de procesamiento promedio obtenida de 58.96 fps se encuentra dentro del rango establecido en los requerimientos los cuales señalaban que debía estar entre 57 y 60 fps.

6.2 PORCENTAJE DE PÉRDIDA O LOCALIZACIÓN ERRÓNEA DE LA PELOTA

La siguiente prueba realizada estuvo orientada a obtener el porcentaje de pérdida o localización errónea de la pelota. La configuración del escenario consistió en colocar en el campo de juego a cinco robots de nuestro equipo y una pelota. El procedimiento fue el siguiente:

1. Colocar la pelota de manera aleatoria en el campo de juego.
2. Calcular su posición correcta.
3. Contar los paquetes equivocados recibidos por el Sistema de IA. Un paquete equivocado es cuando no hay información de la pelota (el SV no la localizó dentro del campo de juego) o la posición de la pelota fuera errónea (la posición de la pelota tuvo una variación de mas de 15 píxeles de la posición correcta calculada).
4. Contar el total de paquetes recibidos.

Los resultados son mostrados en la Tabla 6.2:

Paquetes Recibidos	Paquetes equivocados	Porcentaje equivocados
13874	120	0.86%
12921	1360	10.52%
14423	30	0.20%
14586	216	1.48%
14094	18	0.12%

Tabla 6.2 Resultados prueba localización pelota

Éstos resultados son contrastantes, en algunas ocasiones el porcentaje es muy alto (el 10.52% implica que 6.32 veces por segundo el SV se equivoca) y en otras es muy bajo (el 0.12% implica que el SV se equivoca una vez cada 13.88 segundos). Las razones de esto es que la iluminación no es uniforme en todas las zonas de la cancha y existen zonas en que los valores utilizados en la segmentación no funcionan de la misma manera que en otras zonas mejor iluminadas. La Figura 6.1 ilustra la diferencia de iluminación en diferentes secciones del campo de juego. Cada número es la cantidad de luxes medidos utilizando un luxómetro digital. Un lux es la unidad del Sistema Métrico Decimal para medir la cantidad de luz.

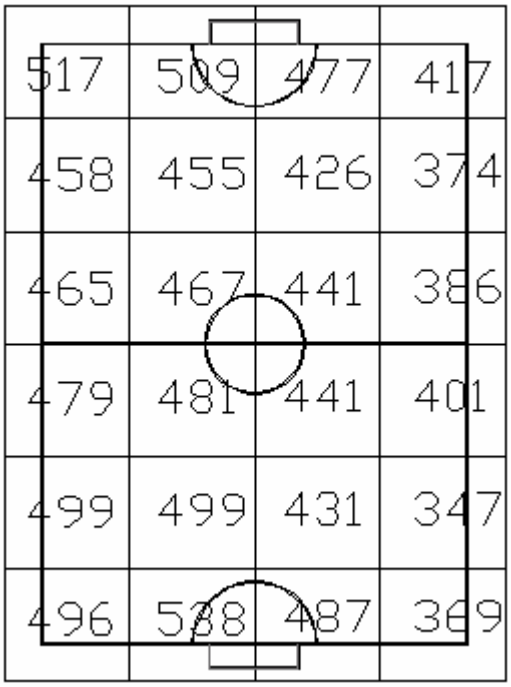


Figura 6.1 Iluminación del campo de juego

Como puede observarse, la diferencia entre el valor mínimo y máximo es de 191 luxes lo que representa una variación del 42.2% sobre el promedio de iluminación que es de 452.5 luxes. Esta variación es excesiva y es la causante de las identificaciones erróneas. La mayor cantidad de errores estuvo en las secciones cuya cantidad de iluminación estaba mas alejada del promedio. Esto implica que el SV es muy sensible a cambios de iluminación y una posible solución es tener valores de segmentación para cada una de las secciones del campo de juego.

6.3 PORCENTAJE DE PÉRDIDA O LOCALIZACIÓN ERRÓNEA DE LA ROBOTS CONTRARIOS

Esta prueba es muy similar a la de la pelota. Se colocaron los robots contrarios de forma aleatoria en el campo de juego, se calculó su posición correcta y se contaron los paquetes con información faltante o errónea. Los resultados se muestran en la Tabla 6.3.

Paquetes Recibidos	Paquetes equivocados	Porcentaje equivocados
14598	19	0.10%
13969	9	0.06%
14788	11	0.07%
14738	13	0.08%
14397	10	0.06%

Tabla 6.3 Resultados de prueba localización robots contrarios

Éstos resultados son razonablemente buenos porque señalan que en promedio cada 22 segundos hay una localización errónea de los robots contrarios. Esto no es nada grave porque como los robots contrarios son considerados como obstáculos un error ocasional solo implica que en un caso extremo exista una colisión. En la mayor parte de las ocasiones el error puede ser corregido casi de inmediato ya que el siguiente paquete de información (con los datos correctos) se recibe solo 16 ms después.

Los errores se deben también a la iluminación no uniforme del campo de juego, sin embargo, son menores que con la pelota porque al ser los parches planos (un pedazo de cartulina de color) la luz que incide en ellos es reflejada de manera uniforme y el color presenta menos variación. Esto no sucede con la pelota que al ser esférica refleja la luz hacia diferentes direcciones provocando que se vea mas clara en su parte superior y mas obscura en la parte inferior.

Una posible solución para evitar los errores al localizar los robots del equipo contrario es suponer que el otro equipo va a utilizar parches extras (lo cual es casi seguro porque todos los equipos utilizan por lo menos un parche extra para calcular su orientación) y de esta forma utilizar ambos parches para tener un criterio adicional para hacer el reconocimiento de los robots contrarios.

6.4 PORCENTAJE DE PÉRDIDA O LOCALIZACIÓN ERRÓNEA DE NUESTROS ROBOTS

Ésta prueba es prácticamente idéntica a la anterior pero ahora se realizó con los robots de nuestro equipo, los cuales poseen mas información para su localización adecuada (parche central y parches extras). Se colocaron los robots de nuestro equipo en forma aleatoria en el campo de juego, se calculó su posición correcta y se contaron los paquetes con información faltante o errónea,

Los resultados obtenidos son mostrados en la Tabla 6.4

Paquetes Recibidos	Paquetes equivocados	Porcentaje equivocados
14698	20	0.13%
14987	26	0.17%
14393	18	0.12%
14654	10	0.06%
14216	19	0.13%

Tabla 6.4 Resultados de prueba localización robots de nuestro equipo

Contrario a lo supuesto, el porcentaje de paquetes equivocados es mayor que con los robots del equipo contrario, sin embargo, la mayor parte de los paquetes equivocados son por pérdida del robot (el robot no fue encontrado por el SV) y muy pocos por localización errónea. Esto es porque que al tener mas elementos para localizar al robot de nuestro equipo el SV casi no confunde regiones causadas por ruido con robots verdaderos. En ese sentido el SV es robusto porque no envía información equivocada. Sin embargo, lo deseable sería que el SV fuera capaz de localizar a los robots todo el tiempo por lo que una posible solución es utilizar un estimador de posiciones futuras (como un filtro Kalman o una red neuronal). Al tener un estimado de la posición futura de un robot se puede utilizar la información de dos maneras:

1. Simplemente enviar el estimado de la posición y orientación del robot cuando el SV no encuentre el robot.
2. Buscar de manera mas intensiva al robot en la zona donde se ha estimado que puede localizarse.

6.5 PORCENTAJE DE IDENTIFICACIÓN ERRÓNEA

El porcentaje de identificación errónea sirve para cuantificar las veces que el SV reconoce a un robot de nuestro equipo de forma correcta pero comete equivocaciones al identificarlo, en otras palabras, el SV acertó en encontrar un robot de nuestro equipo pero falló al identificar de que robot se trataba, por ejemplo, confundió al robot número 1 con el robot número 3.

Para realizar esta prueba se volvieron a colocar los robots en posiciones aleatorias sobre el campo de juego y se contaron los paquetes en que la identificación fuera equivocada. Es importante señalar que solo se consideran los paquetes en donde se localizó exitosamente a un robot de nuestro equipo, es decir, esta prueba se realizó solo cuando el SV no cometió errores al encontrar al robot. Se decidió hacerlo de esta manera porque así se puede aislar los errores cometidos en una etapa anterior de los errores que provocan una identificación errónea.
Los resultados se resumen en la Tabla 6.5.

Paquetes Recibidos	Paquetes equivocados	Porcentaje equivocados
15357	0	0.0%
15912	0	0.0%
15937	1	0.006%
15631	0	0.0%
15965	0	0.0%

Tabla 6.5 Resultados de prueba de identificación errónea

Éstos resultados señalan que la identificación del robot es extremadamente confiable porque en todas las pruebas realizadas solo hubo una ocasión en que se identificó erróneamente al robot. Sin embargo, esto no quiere decir que el SV no se equivoque al identificar a un robot, de hecho cuando existe un error al encontrar un robot de nuestro equipo (localización errónea de nuestros robots) la identificación es errónea también, es decir, el SV se equivoca al identificar al robot porque se equivocó al localizarlo. Esos casos no fueron considerados para la obtención de estos porcentajes porque se quería saber que tan confiable es el SV cuando se tiene una localización exitosa.

6.6 VARIACIÓN EN LA POSICIÓN

Para medir la variación de la posición del robot, se coloco al robot en posiciones aleatorias dentro del campo de juego y se calculó el cambio de la posición calculada por el SV. Se registró todas las posiciones enviadas al Sistema de IA y se calculó su máximo, su mínimo y su promedio en ambos ejes coordenados. La variación máxima se obtuvo calculando la diferencia entre el máximo y el mínimo registrado. Los resultados indican que la variación máxima de la posición de un robot es de 2 píxeles en cualquier dirección. Dos píxeles equivale a una distancia de 8.43 milímetros. En una cancha de 5.4 metros de largo por 4 metros de ancho esta variación de la posición del robot es despreciable.

La causa de esta variación reside en el cálculo del centroide de la región ya que considera todos los píxeles pertenecientes a la región para realizar el cálculo. Si algún píxel es segmentado de mas o de menos o existe una variación de los píxeles segmentados el centroide calculado varía de igual forma.

6.7 VARIACIÓN DE LA ORIENTACIÓN

Para medir la variación de la orientación del robot se colocó un robot en 13 orientaciones diferentes (ángulos diferentes) y se obtuvo el máximo y mínimo ángulo calculado por el sistema. Los valores se obtuvieron registrando 1000 paquetes de datos para cada orientación y

calculando el máximo, mínimo y promedio. Una gráfica con las mediciones es mostrada en la Figura 6.2.

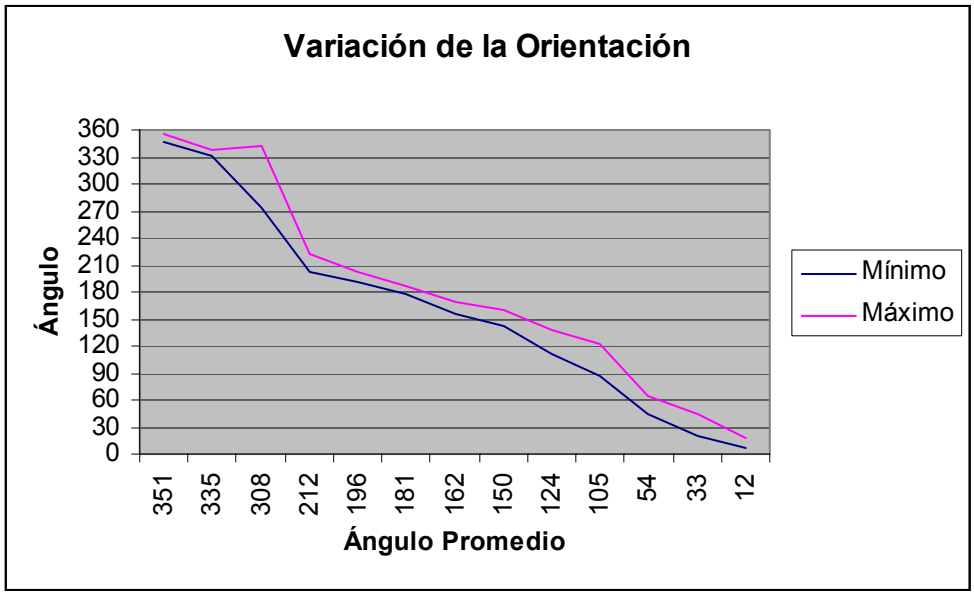


Figura 6.2 Máximos y mínimos de la orientación

El promedio de variación es de +/- 10 grados. Sin embargo hay algunas orientaciones en que la variación es mayor como por ejemplo alrededor de los 270º y 90º provocadas porque son los valores en que la función tangente (utilizada para el cálculo de la orientación) esta indefinida.

7. CONCLUSIONES

En este capítulo se presentan las conclusiones del trabajo desarrollado en esta tesis. En la primera sección se exponen las conclusiones particulares del sistema. La segunda sección está dedicada a las líneas futuras. Continúa con las limitaciones del sistema y por último, se concluye con reflexiones generales sobre el proyecto RoboCup del ITAM.

7.1 CONCLUSIONES PARTICULARES DEL SISTEMA

El Sistema de Visión desarrollado cumplió exitosamente con los objetivos y requerimientos planteados. Es capaz de localizar en tiempo real a la pelota, los robots contrarios, los robots de nuestro equipo y calcular su orientación. Además, puede transmitir toda la información recabada de manera sencilla utilizando un enlace de red. La aplicación permite al Usuario configurar de manera sencilla los dispositivos de captura empleados, los valores de segmentación para cada uno de los objetos de interés, la activación y desactivación de los robots de nuestro equipo y de los robots contrarios, y finalmente, la visualización de los resultados.

La arquitectura del sistema puede ser modificada y ampliada sin alterar de manera significativa el funcionamiento de los demás componentes ya que su estructura modular permite una fácil reconfiguración basada en la funcionalidad requerida para un objetivo particular, es decir, los módulos pueden ser modificados internamente para conseguir la extracción de información relevante para la aplicación. De esta forma, esta arquitectura puede ser utilizada para futuros desarrollos de visión por computadora relacionados con RoboCup o con cualquier aplicación cuya fuente principal de información sea la visual.

Se puso especial atención en la elección de los algoritmos empleados. Se utilizaron aquellos que combinaran la eficiencia con la confiabilidad. Por eso se logro la segmentación de hasta 32 colores con dos operaciones AND en lugar de 192 comparaciones de enteros. La utilización de regiones o blobs permitió el fácil reconocimiento de los objetos porque éste se basó en características físicas como el área y las relaciones espaciales de los parches de colores de los robots.

Las herramientas de hardware empleadas respondieron a su disponibilidad en el mercado. Sin embargo, uno de los factores que pueden ser modificados para mejorar el desempeño y la eficiencia del SV es precisamente éste. La utilización de un digitalizador introduce ruido en la

imagen que puede ser evitado si se emplean medios digitales de transmisión y recepción (como el IEEE1394). Un cambio obligado para futuras versiones del SV debe ser éste.

Las herramientas de software han sobrepasado las expectativas que se tenían de ellas. El desempeño, la modularidad y hasta su disponibilidad de manera gratuita (lo cual es extraño en Microsoft) han permitido el desarrollo de este sistema de manera exitosa. Además, permite a futuros desarrolladores del sistema utilizar un ambiente al que seguramente estarán acostumbrados (dada la popularidad de Windows) y facilitará el realizar mantenimiento y actualizaciones que puedan ser requeridas.

7.2 VERSIONES DEL SISTEMA

Durante el tiempo que llevó la realización de esta tesis hubo un cambio significativo en el tamaño del campo de juego (se aumentó en mas de 200% su tamaño), lo que llevó a la elaboración de una segunda versión para el SV (la tesis está basada en la segunda versión del sistema). A continuación se enumeran las principales modificaciones realizadas en el cambio de versión:

1. Se tuvo que replantear los requerimientos. Fue obvio que una sola cámara no iba a ser suficiente para obtener una imagen de todo el campo de juego lo que obligó a utilizar dos cámaras.
2. Se puso especial atención en la eficiencia de los algoritmos empleados porque resultaba crítico ahora que se tenía que procesar el doble de información de la primera versión.
3. Se utilizó una herramienta de software diferente: DirectShow. La versión anterior utilizaba la tecnología de Video for Windows que no permitía el funcionamiento simultáneo de dos dispositivos de captura.
4. Se cambió la interface con el Usuario y se modificó mucha funcionalidad para permitir la configuración de un sistema que utiliza dos señales de video de forma simultánea.

La segunda versión del SV presenta muchas mejoras con respecto a la primera. La velocidad de procesamiento es significativamente mayor (la primera versión trabajaba a 13 fps), es mas robusta en la identificación de los robots y tiene un porcentaje de pérdida menor. La primera versión fue utilizada en el American Open 2003.

7.3 LÍNEAS FUTURAS

RoboCup es una iniciativa en continuo cambio debido a su ímpetu de llevar la tecnología al límite. Es por eso que este trabajo estaría inconcluso si no se delinearan propuestas de innovaciones para el futuro.

La visión por computadora en la Liga de Robots Pequeños tiene dos tendencias principales: los sistemas de visión global y los sistemas de visión local. A continuación se examinan los posibles retos que pueden surgir en cada una de estas tendencias.

7.3.1 Sistemas de Visión Global

Los sistemas de visión global constituyen la tendencia mas tradicional y popular. Esta tesis se basó en esa tendencia.
Existe un potencial de desarrollo enorme en los sistemas de este tipo. Por poner un ejemplo, se han desarrollado dispositivos mecánicos capaces de patear la pelota y elevarla lo suficiente para que pase por encima de otros robots. Los sistemas de visión que actualmente trabajan en dos dimensiones (en el plano de la cancha) tendrán que ser modificados para poder funcionar en tres dimensiones y obtener información sobre la altura a la que viaja la pelota y poder reaccionar acorde a eso.

Otro problema que debe ser resuelto es la calibración autónoma de los colores. En la actualidad todos los equipos dedican gran parte de su tiempo de configuración a la obtención de parámetros para segmentar los colores utilizados por sus sistemas. Un proceso que sea capaz de realizar esto de forma autónoma permitiría a los equipos ocupar ese tiempo en otras actividades. Además, las variaciones de luz entre un partido y otro podrían ser fácilmente sobrellevadas con un proceso de este tipo.

Finalmente, los robots son cada año mas rápidos y es necesario incrementar la velocidad de procesamiento para estar a la par de los avances en mecánica. Si los sistemas de visión se quedan atrás pueden convertirse en la causa de estancamiento dentro de una liga que se ha caracterizado por su evolución continua. Para incrementar la velocidad de procesamiento es necesario optimizar la cantidad de información procesada. Se pueden utilizar estimadores de posiciones futuras (Kalman y redes neuronales) para procesar solo las partes de la imagen en donde se encuentran los objetos de interés y así no realizar un procesamiento innecesario.

7.3.2 Sistemas de Visión Local

Los sistemas de visión local son aquellos en los que cada robot tiene integrada una cámara y el procesamiento del video se realiza abordo. Ésta tendencia presenta desafíos tanto en visión como en IA por lo que es necesario desarrollar algoritmos eficientes y confiables para que en un futuro este tipo de sistemas se vuelvan tan populares como los sistemas de visión global.

Algunos de los desafíos incluyen la visión omnidireccional (utilizar espejos cónicos para poder tener una imagen de toda la cancha a la altura del robot), la obtención de un marco de referencia constante aún con el robot en movimiento, etc.

Otro aspecto interesante en los sistemas de visión local es la manera en que se puede compartir la información obtenida por los diferentes robots. Por ejemplo, si un robot localizó la pelota puede comunicar esa información a los demás y así evitarles un trabajo innecesario. Para lograr esto se necesita desarrollar estructuras de comunicación dinámicas.

7.4 LIMITACIONES

El SV puede ser mejorado en muchos aspectos en donde las pruebas indican porcentajes de fallas considerables. Esto consistirá en un desafío para aquellos que continúen el trabajo de esta tesis y requerirá el análisis profundo de los motivos de las fallas. Sin embargo, esta sección pretende indicar los aspectos en donde se debe poner especial atención y que constituyen las limitaciones del sistema:

1. El paso fundamental para el éxito o fracaso del SV es la segmentación. Todo depende de que cada uno de los colores pueda ser identificado y separado de manera correcta. Las fallas que surgieron en este trabajo estuvieron directamente relacionadas con una mala segmentación. El SV es poco tolerante a cambios de iluminación debido a que los valores de umbral utilizados en la segmentación permanecen fijos una vez que han sido establecidos por el usuario. Los valores deben actualizarse de manera automática utilizando estadísticas calculadas durante el procesamiento de cada imagen (como puede ser la media y varianza para cada color) para así tener unos valores mas acordes con la situación de iluminación del campo de juego y mas adaptables a cambios graduales presentes a lo largo del dia.

2. Otra limitante es el espacio de color, se utilizó YUV para no afectar el desempeño del sistema, sin embargo HSV es en términos generales el mas indicado para esta aplicación ya que separa el color en un solo canal y sería posible tener un proceso de segmentación que aprovechará mejor las características físicas del ambiente de la liga de robots pequeños.

3. Durante el desarrollo del SV se notó que al utilizar una tarjeta de captura de video se perdía mucha de la calidad original de la imagen. Sin embargo se desechó la utilización de una señal digital IEEE 1394 (que no degrada la calidad de la imagen) porque se observó a simple vista que existía un tiempo de latencia considerable (el video estaba atrasado mas de 1 segundo). Esto puede corregirse rescribiendo los drivers para el

puerto IEEE1394 pero esta solución se encontraba fuera del alcance establecido para el SV. Esto puede ser retomado en futuros desarrollos ya que mejoraría considerablemente los resultados obtenidos.

4. El proceso de calibración manual que se utilizó en el sistema presenta una posible fuente de error, ya que al depender completamente del usuario los resultados obtenidos varían de acuerdo a la habilidad y experiencia de la persona encargada de realizar esta actividad. Éste proceso puede ser reemplazado por un método automático que extraiga los valores para la segmentación basados en información previa o bien en características preestablecidas que sean optimizadas de acuerdo a métricas de éxito o fracaso en la localización de los objetos de interés dentro del campo de juego.

5. La pelota puede ser ocluida por un robot (es “tapada” por el robot y no aparece en el cuadro de video). El SV no tiene ningún mecanismo para compensar esta falla. Una posible solución es la utilización de mas cámaras (por lo menos 2 mas) o bien un estimador de posiciones futuras que prediga la posición de la pelota cuando ocurre la oclusión.

7.5 CONCLUSIONES GENERALES

El Sistema de Visión (SV) desarrollado en esta tesis ha contribuido de forma significativa en el éxito del equipo de robots autónomos del ITAM (Eagle Knights) en competencias internacionales. Nuestro equipo, obtuvo el tercer lugar en el RoboCup American Open 2003 y segundo lugar en el RoboCup USOpen 2004. Éstos resultados son el producto de poco mas de año y medio de trabajo continuo en el que se han resuelto una cantidad innumerable de problemas y desafíos. La complejidad de un proyecto de esta envergadura en donde una gran cantidad de tecnología debe integrarse y funcionar al unísono solo puede ser superada proponiendo soluciones prácticas e innovadoras. En ese sentido este trabajo de tesis es el primero de muchos por venir, todos ellos motivados por el proyecto RoboCup del ITAM que tiene un futuro prometedor y se ha constituido en pionero del desarrollo de la robótica en México y ha sentado las bases para que proyectos similares puedan surgir por todo el país.

BIBLIOGRAFÍA

[BRO 02] Brown, David William
"An Introduction to OBJECT-ORIENTED ANALYSIS Objects and UML in Plain English" John Wiley & Sons, Inc, 2002.

[BRU 00] Bruce, Balch, Veloso
"Fast and inexpensive color image segmentation for interactive robots"
Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '00), 2000.

[CHA 96] Chazelle et al.
"Application challenges to computational geometry: CG impact task force report"
Technical Report TR-521-96, Princeton University, 1996.
http://ncstrl.cs.princeton.edu/expand.php?id=TR-521-96

[GON 93] González, Woods
"Digital Image Processing"
Adison-Wesley Publishing Company, 1993.

[GOO 02] Goodrich, Michael
"Data Structures and Algorithms in Java"
Sams Publishing, 2002.

[HEI 97] Heikkila, Janne
"A Four-step Camera Calibration Procedure with Implicit Image Correction"
Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, 1997.

[HEL 87] Held, Gilbert
"Data Compresiion: Techniques and Applications, Hardware and Software Considerations"
John Wiley & Sons, 1987

[HOA 61] Hoare, C.A.R
"Communications of the ACM"
ACM Press, 1961
http://portal.acm.org/citation.cfm?id=366644&dl=ACM&coll=portal

[INT 01] Intel Corporation
"Open Source Computer Vision Library, Reference Manual"
Intel Corporation, 2001
http://www.cs.unc.edu/Research/stc/FAQs/OpenCV/OpenCVReferenceManual.pdf

[IOC 98] Iocchi, Luca
"Calibration of Stereo Cameras for Mobile Robots"
Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza",1998
http://www.dis.uniroma1.it/~iocchi/stereo/calib.html

[KAR 99] Karpati,D´ Andrea Lee
"Real Time Visual Perception for Autonomous Robotic Soccer Agents"
Cornell University RoboCup Team: Big Red Bots, 1999.
http://robocup.mae.cornell.edu/documentation/robocup/1999/Vision_Manual.pdf

[KEI 01] Keith, J.
"Video Demystified"
LLH Publications, 2001.

[KIT 95] Kitano,H
"RoboCup: The robot world initiative."
Proceedings of the IJCAI-95 Workshop on Entretainment and AI/ A Life, 1995.

[LAU 04] Laurin Publishing
"The Photonics Directory"
Photonics.com , 2004

[LAW 01] Lawrence,Jim
"Video Signal Primer"
AtariLabs.com , 2001
http://atarilabs.com/meat/2000/1201_videoprimer.shtml

[LI 00] Li, Ze-Nian
"Notes: Multimedia Systems"
Simon Fraser University, 2000.
http://www.cs.sfu.ca/CC/365/li/material/notes/Chap3/Chap3.3/Chap3.3.html

[MEL 94] Melen, T.
"Geometrical modelling and calibration of video cameras for underwater navigation"
Tésis de doctorado, Institutt for teknisk kybernetikk, 1994.

[MSD 04] MSDN Library
"Introduction to Direct Show"
DirectShow SDK Documentation, 2004
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/directx9_c/directX/htm/directshow.asp

[PAR 97] Parker, J.R.
"Algorithms for Image Processing and Computer Vision"
Wiley Computer Publishing, 1997.

[ROB 98] The RoboCup Federation
"What is RoboCup"
http://www.robocup.org/02.html , 1998.

[RIP 96] Ripley, B.D.
"Pattern recognition and Neural Networks"
Cambridge University Press, 1996.

[SLA 80] Slama, C.C
"Manual of Photogrammetry"
American Society of Virgina, 1980

[SCO 01] Scott, Kendall
"UML Explained"
Addison-Wesley,2001.

[SON 99] Sonka Hlavac & Boyle
"Image Processing, Analysis, and Machine Vision"
PWC Publishings, 1999.

[TSA 87] Tsai, R. Y.
"A versatile camera calibration technique for high accuracy 3D machine vision metrology using oof-the-shelf TV cameras and lenses"
IEEE Journal of Robotics and Automation, 1987

[WEI 03] Weitzenfeld, Gutierrez-Nolasco & Venkatasubramanian
"Controlling Mobile Robots with Distributed Neuro-Biological Systems"
Proc. AINS 2003, 2nd Annual Symposium on Autonomous Intelligent Networks and Systems, 2003.

[WEI 93] Wei & Ma
"A complete two-plane camera calibration method and experimental comparisons"
Proc. 4th International Conference on Computer Vision, 1993.

[WEN 92] Weng, Cohen & Herniou
"Camera calibration with distortion models and accuracy evaluation"
IEEE Transactions on Pattern Analysis and Machine Intelligence, 1992.

APÉNDICE A

Reglas completas de la Liga F180

En éste apéndice se presentan las reglas que regulan la liga F180 de RoboCup para el 2004. Las reglas son redactadas y revisadas por un Comité Técnico cada año y son publicadas en la dirección electrónica de la Federación RoboCup (http://www.robocup.org).

A.1 EL CAMPO DE JUEGO

A.1.1 Dimensiones

El campo de juego debe ser rectangular. Las dimensiones incluyen las líneas limítrofes.
Longitud: 4900 mm
Ancho: 3400 mm

A.1.2 Superficie del campo de juego

La superficie del campo de juego es alfombra o fieltro verde. El piso debajo de la alfombra es nivelado, plano y sólido.
La superficie del campo de juego continúa 300 mm después de las líneas limítrofes en todos los lados. En el borde del campo de juego una pared evitará que los robots se salgan del borde.

A.1.3 Marcas del campo de juego

El campo de juego esta marcado con líneas. Las líneas pertenecen a las áreas de las cuales son límites.
Los dos lados mas largos son llamados límites de toque. Los dos lados cortos son llamados límites de meta.
Todas las líneas tienen un grosor de 10 mm y son pintadas de blanco.
El campo de juego es dividido en dos por una línea de medio campo.
El punto central se encuentra a la mitad de la línea de medio campo. Un círculo con un diámetro de 800 mm es marcado alrededor de él.

A.1.4 El área defensiva.

El área defensiva esta definida en cada extremo del campo de juego como sigue:

Un arco semicircular es dibujado en el campo de juego con su centro en el punto medio entre los postes de la portería y tiene un radio de 500 mm.
El área delimitada por éste arco y la línea de gol es el área defensiva.

A.1.5 El punto de penalti

Dentro de cada área defensiva un punto de penalti es marcado a una distancia de 450 mm del punto medio entre las paredes laterales de la portería y equidistante a ellas. El punto es un círculo con un diámetro de 10 mm pintado de blanco.

A.1.6 Porterías

Las porterías deben ser colocadas en el centro de cada límite de meta.
Consisten de dos paredes verticales de 150 mm unidas por la parte posterior por una pared de 150 mm. Las caras de la portería que dan hacia el campo de juego son pintadas de azul en un extremo del campo y de amarillo en el otro. Los bordes y la parte superior de la portería son pintados de blanco.
Hay un cable que descansa sobre las paredes laterales y que corre a través de la línea de meta. El cable debe ser lo suficientemente fuerte para que la pelota rebote. El cable debe ser transparente o de un material translucido como el de una línea de pescar.
La distancia entre las paredes laterales es de 700 mm. La profundidad de la portería es de 180 mm. La distancia del borde inferior del cable transversal a la superficie de juego es de 150 mm.
El piso dentro de la portería es del mismo color que el de la superficie de juego.
Las paredes y la barra tienen el mismo grosor que las líneas, 10 mm.
Las porterías deben ser sujetadas fuertemente a la superficie de juego.

A.1.7 Barra de montaje de equipo

Una barra de montaje de equipo será provista a 4 m de altura sobre el campo de juego. La barra corre de portería a portería en el punto medio del campo de juego.

A.1.8 Decisiones del Comité Técnico sobre el campo de juego.

Decisión 1. El organizador del evento debe proveer una iluminación uniforme y difusa de aproximadamente 500 lux o mas brillante. A partir del 2004 y en adelante, no habrá equipo de iluminación especial para proveer esas condiciones. El comité organizador distribuirá los detalles de las condiciones de iluminación a los competidores a la brevedad posible.
Decisión 2. Ninguna clase de publicidad real o virtual es permitida dentro del campo de juego (incluyendo las redes de las porterías y el área que ellas conforman), y tampoco en los robots

desde el momento en que los equipos entran al campo de juego hasta el momento en que lo dejan al medio tiempo y desde que vuelven a entrar al campo de juego al inicio de la segunda mitad hasta que termina el juego. En particular, ningún material publicitario de ninguna clase puede ser desplegado dentro de las porterías o paredes. Ningún equipo extraño (cámaras, micrófonos, etc.) puede ser colocado en dichos elementos.

Decisión 3. El color específico y la textura de la superficie no esta determinada y puede variar de competencia a competencia (de la misma forma en que las canchas de fútbol varían). La superficie debajo de la alfombra debe estar nivelada y ser sólida. Ejemplos de superficies aprobadas incluyen: cemento, linoleum, piso de madera, plywood, mesas de ping pong y tablas; superficies suaves no son permitidas. Se hará todo lo posible para asegurar que la superficie sea lisa, sin embargo, queda a los equipos diseñar sus robots para que funcionen con ligeras curvaturas en la superficie.

Decisión 4. En los juegos en donde un equipo utilice visión local (los robots tienen su propia cámara) los organizadores proveerán de una pared blanca de 100 mm de altura a una distancia de 300 mm de los límites del campo de juego para prevenir interferencia con el sistema de visión.

A.2 LA PELOTA

A.2.1 Características y medidas

La pelota es una pelota de golf naranja estándar. Es:

- Esférica.
- De color naranja.
- De aproximadamente 46 g de masa.
- De aproximadamente 43 mm de diámetro.

A.2.2 Reemplazo de una pelota defectuosa

Si la pelota se vuelve defectuosa cuando el encuentro se este jugando:

- El encuentro es detenido.
- El encuentro es reiniciado colocando la pelota de repuesto en el punto donde la pelota anterior se volvió defectuosa.

Si la pelota se vuelve defectuosa mientras no esta en juego, es decir en un saque inicial, saque de meta, tiro de esquina, tiro libre, penalti o saque de banda:

- El juego es reiniciado según corresponda.

La pelota no debe cambiarse sin la autorización del árbitro.

A.3 EL NÚMERO DE ROBOTS

A.3.1 Robots

Un encuentro es jugado por dos equipos, cada uno con no más de 5 robots, uno de los cuales es el portero. Cada robot debe estar claramente numerado para que el árbitro pueda identificarlo durante el encuentro. El portero debe ser designado antes de iniciar el juego. Un juego no puede iniciar sin que los dos equipos tengan por lo menos un robot.

A.3.2 Intercambio

Los robots pueden ser intercambiados. No existe límite en el número de intercambios.

A.3.3 Procedimiento de intercambio

Para intercambiar un robot las siguientes condiciones deben de cumplirse:

- Un intercambio solo puede realizarse mientras el encuentro esta detenido.
- El árbitro es informado antes de realizar el intercambio.
- El robot entrante es colocado en el campo de juego después de que el robot que abandona haya sido removido.
- El robot entrante es colocado en la línea de media cancha.

A.3.4 Cambiando al portero

Cualquiera de los otros robots pueden intercambiar con el portero, dado que:

- El árbitro es informado antes de realizar el intercambio.
- El cambio es realizado cuando el encuentro se detenga.

A.3.5 Robots expulsados

Un robot expulsado puede ser intercambiado por otro robot que deje el campo.

A.3.6 Decisiones del Comité Técnico sobre el número de robots.

Decisión 1. Cada equipo debe designar una persona encargada de manejar los robots para realizar los intercambios y colocar los robots en posición cuando sea necesario. Movimientos de los robots por el manejador no son permitidos.

A.4 EL EQUIPAMIENTO ROBÓTICO

A.4.1 Seguridad

El robot no debe tener nada en su construcción que pueda ser peligroso para él mismo, para otro robot o para humanos.

A.4.2 Forma

Un robot debe caber dentro de un cilindro de 18 cm de diámetro. Si el equipo esta utilizando visión global, cada robot del equipo debe tener una altura máxima de 150 mm. En cualquier otro caso, el robot debe tener una altura máxima de 225 mm.

A.4.3 Colores y parches

Antes de un encuentro, cada uno de los dos equipos tiene un color asignado: amarillo o azul. Cada equipo debe ser capaz de utilizar los parches azules o amarillos. Parches circulares del color asignado deben ser montados en la parte superior del robot. El centro del parche debe estar colocado en el centro visual del robot visto desde arriba. Los parches deben tener un diámetro de 50 mm.
Los robots pueden usar coloración negra o blanca sin restricción. Los robots también pueden utilizar parches color verde claro, rosa claro y cian, donde los colores exactos son definidos por cartulinas distribuidas por los organizadores antes de la competencia.

A.4.4 Locomoción

Las ruedas de los robots (o cualquier otra superficie que haga contacto con el campo de juego) debe estar fabricado de algún material que no dañe la superficie de juego.

A.4.5 Comunicación inalámbrica

Los robots pueden usar comunicación inalámbrica con computadoras o redes localizadas fuera del campo de juego.

A.4.6 Sistema de visión global

El uso de un sistema de visión global o sistemas de visión distribuidos está permitido, pero no requerido, para identificar y rastrear la posición de los robots y de la pelota. Esto es logrado

utilizando una o mas cámaras. Las cámaras no deben sobrepasar 150 mm por debajo de la barra de montaje provista sobre el campo de juego.

A.4.7 Autonomía

El equipo robótico debe ser completamente autónomo. Los operadores humanos no tienen permitido introducir ninguna información durante un encuentro, excepto en el medio tiempo o durante un tiempo fuera.

A.4.8 Infracciones / Sanciones 1

Para cualquier infracción de ésta regla:

- El encuentro no es detenido
- El robot infractor es instruido por el árbitro para dejar el campo de juego para corregir su equipamiento.
- El robot deja el campo de juego cuando la pelota deje de estar en juego.
- Cualquier robot requerido para dejar el campo de juego para corregir su equipamiento no puede ingresar sin la autorización del árbitro.
- El árbitro verifica que el equipamiento sea el correcto antes de permitir el reingreso del robot al campo de juego.
- El robot solo es autorizado a ingresar al campo de juego cuando la pelota esta fuera de juego.

Un robot que ha sido requerido para abandonar el campo de juego debido a una infracción a ésta regla y que reingrese sin la autorización del árbitro es sancionado y se le muestra la tarjeta amarilla.

A.4.9 Reinicio del encuentro

Si el encuentro es detenido por el árbitro para administrar una sanción:

- El encuentro es reiniciado por medio de un tiro libre indirecto otorgado al equipo no infractor en lugar donde la pelota estaba colocada antes de que el árbitro detuviera el encuentro.

A.4.10 Decisiones del Comité Técnico sobre el equipamiento robótico

Decisión 1. Participantes utilizando comunicación inalámbrica deben notificar al comité organizador el método de comunicación inalámbrica, potencia y frecuencia. El comité

organizador de la competencia debe ser notificado de cualquier cambio posterior al registro lo antes posible.

Para evitar interferencia, los equipos deben ser capaces de seleccionar dos frecuencias portadoras antes del inicio del encuentro. El tipo de comunicación inalámbrica debe cumplir con las normas del país en donde la competencia se realiza. El cumplimiento de las normas locales es responsabilidad de los equipos competidores y no de la Federación de RoboCup.

Decisión 2. Los dispositivos de pateo son permitidos.

Decisión 3. Puntas de metal y velcro están prohibidos para la locomoción.

A.5 EL ÁRBITRO

A.5.1 La autoridad del árbitro

Cada encuentro es dirigido por un árbitro que tiene completa autoridad para hacer cumplir las reglas del juego en el partido que ha sido asignado.

A.5.2 Poderes y deberes

- Hacer cumplir las reglas del juego.
- Controlar el encuentro en cooperación con el árbitro asistente.
- Asegurarse de que la pelota cumpla con la regla A.2
- Asegurar que el equipamiento robótico cumpla con la regla A.4
- Informar al árbitro asistente cuando comienzan y terminan los periodos de tiempo perdido de acuerdo a la regla A.7
- Detener, suspender o terminar el encuentro a su discreción por cualquier infracción a las reglas.
- Detener, suspender o terminar el encuentro debido a interferencia externa de cualquier tipo.
- Detener el encuentro, si en su opinión, un robot puede causar serio daño a humanos, otros robots o a él mismo y asegurarse que es removido del campo de juego.
- Reposicionar la pelota en posición neutral si se queda atascada durante el encuentro.
- Permitir que el encuentro continúe cuando el equipo que recibe una infracción se beneficie de tal ventaja, y penalizar la infracción original cuando la ventaja anticipada no se asegure en ese momento.
- Castigar la infracción mas grave cuando un robot comete mas de una infracción al mismo tiempo.
- Tomar acciones disciplinarias en contra de los robots culpables de cometer infracciones merecedoras de amonestación o expulsión. El árbitro no esta obligado a tomar esta acción de forma inmediata pero debe hacerlo cuando la pelota este fuera de juego.

- Tomar acciones en contra de miembros de equipos que fallen en conducirse de manera responsable y puede a su discreción, expulsarlos del campo de juego y sus alrededores inmediatos.
- Actuar por el consejo del árbitro asistente con relación a incidentes que no haya observado.
- Asegurarse que personas no autorizadas rodeen el campo de juego.
- Reiniciar el encuentro cuando haya sido detenido.
- Entregar al comité organizador un reporte del encuentro que incluya información de las acciones disciplinarias tomadas en contra de miembros del equipo y de cualquier otro incidente que haya ocurrido antes, durante y después del encuentro.

A.5.3 Decisiones del árbitro

Las decisiones del árbitro relacionadas con hechos del encuentro son finales.
El árbitro solo puede cambiar de decisión cuando considere que ha sido incorrecta, o a su discreción por el consejo del árbitro asistente, siempre y cuando no se haya reiniciado el encuentro.

A.5.4 Equipo de señalización del árbitro

Un dispositivo será provisto para convertir las señales del árbitro en una comunicación serial que será transmitida a ambos equipos. El dispositivo será operado por el árbitro asistente. Detalles del dispositivo serán provistos por el comité organizador antes de la competencia.

A.5.5 Señales del árbitro

Durante el encuentro el árbitro señalizará el inicio y detención del encuentro de la manera acostumbrada. El árbitro asistente enviará una señal acorde a la indicación del árbitro mediante un enlace de comunicación serial a cada uno de los equipos. No esta permitido la interpretación de las señales del árbitro por operadores humanos.
La señal del silbato indica que el árbitro ha detenido el juego, y todos los robots deben moverse a 400 mm de la pelota para permitirle al árbitro colocar la pelota para reiniciar el encuentro. Todos los robots deben permanecer a 400 mm de la pelota mientras esta es movida a su posición de reinicio.
Para un gol (regla A.10), amonestación o expulsión (regla A.12), una señal informativa será enviada para indicar la decisión del árbitro.
La señal de reinicio indicará el tipo de reinicio. Los robots deben moverse a posiciones validas hasta la recepción de la señal. Para reiniciar el encuentro, excepto en saque inicial (regla A.8) o

tiro penalti (regla A.14), el cobrador puede pegarle a la pelota cuando este listo sin necesidad de recibir señales adicionales del árbitro.
Para un saque inicial (regla A.8) o un tiro penalti (regla A.14), una señal de inicio será enviada para que el cobrador pueda proceder. Esta señal no será enviada para otras formas de reiniciar del encuentro.
Las señales indicando periodos de tiempo fuera y tiempo perdido serán enviadas cuando sean requeridas.
Se entenderá que el árbitro ha dado una señal cuando el árbitro asistente haya mandado la señal por el enlace serial de comunicación.

A.5.6 Decisiones del Comité Técnico sobre el árbitro

Decisión 1. Un árbitro (o cuando sea aplicable el árbitro asistente) no es considerado responsable por:

- Cualquier herida sufrida por un oficial o espectador.
- Daño a propiedad de cualquier tipo.
- Cualquier otra perdida sufrida por un individuo, club, compañía, asociación o cualquier otra entidad, que sea debida o pueda ser debida a cualquier decisión que haya hecho bajo los términos de las reglas del juego o al respecto de los procedimientos normales requeridos para sostener, jugar o controlar un encuentro.

Esto puede incluir:

- Una decisión de la condición del campo de juego o sus alrededores para permitir o no la realización de un encuentro.
- La decisión de abandonar el encuentro por cualquier motivo.
- La decisión con respecto a las condiciones o reparaciones del equipamiento usado durante un encuentro incluyendo el campo de juego y la pelota.
- La decisión de detener o no el encuentro debido a la interferencia de un espectador o a cualquier problema en el área de espectadores.
- La decisión de detener o no el encuentro para permitir a un robot dañado que sea removido del campo de juego para su reparación.
- La decisión de solicitar o insistir que un robot dañado sea removido del campo de juego para su reparación.
- La decisión de permitir o no permitir que un robot tenga determinados colores.
- La decisión (incluso cuando esto sea su responsabilidad) de permitir o no permitir a alguna persona (incluyendo miembros del equipo o oficiales del estadio, oficiales de seguridad, fotógrafos u otro representantes de la prensa) estar en la vecindad del campo de juego.
- Cualquier otra decisión que pueda tomar en concordancia con las Reglas del Juego o en conformidad con sus deberes bajo los términos de la Federación de RoboCup o reglas de la liga o regulaciones bajo los cuales un encuentro es jugado.

Decisión 2 Hechos conectados con el encuentro incluyen si un gol es anotado o no y el resultado del encuentro.
Decisión 3. El árbitro debe usar un palo negro o algún otro dispositivo para reposicionar la pelota para reducir la interferencia con el sistema de visión.

A.6 EL ÁRBITRO ASISTENTE

A.6.1 Deberes

El árbitro asistente tiene los siguientes deberes sujetos a la decisión final del árbitro:

- Llevar el tiempo y marcador del encuentro.
- Operar el dispositivo de comunicación serial para transmitir las señales del árbitro por medio de los enlaces de comunicación.
- Monitorear a los operadores de los robots para que no envíen señales ilegales a los robots.
- Indicar cuando un intercambio es solicitado.
- Indicar cualquier incidente o conducta inapropiada que ocurra sin ser observada por el árbitro.
- Indicar cualquier falta que haya sido cometida cuando se encuentre mas cerca que el árbitro (esto incluye, en particular faltas cometidas en el área defensiva).
- Indicar si en un tiro de penalti el portero se movió hacia delante antes de que el tiro haya sido cobrado y también indicar si la pelota cruzó la línea de gol.

A.6.2 Asistencia

El árbitro asistente también asiste al árbitro en el control del encuentro en concordancia a las reglas del juego. En el evento de una conducta inapropiada o interferencia, el árbitro puede destituir de sus deberes al árbitro asistente y reportarlo al comité organizador.

A.7 LA DURACIÓN DEL ENCUENTRO

A.7.1 Períodos de juego

El encuentro dura dos períodos de 10 minutos, a menos que se haya llegado a un acuerdo entre el árbitro y los dos equipos participantes. Cualquier acuerdo de modificar el tiempo de los períodos de juego (por ejemplo, reducir cada mitad a 7 minutos debido a un horario apretado) debe ser hecho antes de empezar el encuentro y debe cumplir con las reglas de la competencia.

A.7.2 Intervalo de medio tiempo

Los equipos disponen de un intervalo de medio tiempo. El intervalo de medio tiempo no debe exceder los 10 minutos. Las reglas de la competencia deben establecer la duración del intervalo de medio tiempo. La duración del intervalo solo puede ser alterada con el consentimiento de ambos equipos y del árbitro.

A.7.3 Tiempos fuera

Cada equipo dispone de cuatro tiempos fuera al inicio del encuentro. Un total de 10 minutos es permitido para todos los tiempos fuera. Por ejemplo, un equipo puede tomar tres tiempos fuera de un minuto y tener entonces un tiempo fuera restante de hasta siete minutos. Los tiempos fuera solo pueden pedirse cuando el encuentro este detenido. El tiempo es llevado por el árbitro asistente.

A.7.4 Reposición de tiempo perdido

La reposición del tiempo perdido se realiza en el periodo correspondiente por:

- substitución
- arreglo de robots dañados
- remoción de robots dañados del campo de juego
- perdida de tiempo
- cualquier otra causa

La reposición del tiempo perdido es a discreción del árbitro.

A.7.5 Tiempo extra

Las reglas de la competencia pueden proveer dos períodos extras de juego de igual duración. Las condiciones de la regla A.8 se aplican.

A.7.6 Encuentro abandonado

Un encuentro abandonado es vuelto a jugar a menos que las reglas de la competencia indiquen algo contrario.

A.7.7 Decisiones del Comité Técnico sobre la duración del encuentro

Decisión 1. Los organizadores harán todo lo posible para dar acceso a ambos equipos al área de competencia al menos dos horas antes de que empiece la competencia. También darán acceso una hora antes del encuentro para preparación del equipo. Los participantes, sin embargo, deben estar concientes de que existan circunstancias que impidan esto.

A.8 EL INICIO Y REINICIO DEL ENCUENTRO

A.8.1 Preliminares

Si los dos equipos utilizan la misma frecuencia de comunicación inalámbrica, el comité organizador asignará esa frecuencia a un equipo para la primera mitad del encuentro. Si ambos equipos desean el mismo color de parches, el comité organizador asignará el color a un equipo para la primera mitad del encuentro.
Una moneda es lanzada y el equipo que gane decide a que portería atacar durante la primera mitad del encuentro.
El otro equipo hace el saque inicial del encuentro.
El equipo ganador del volado hace el saque inicial para la segunda mitad del encuentro.
En la segunda mitad del encuentro los equipos cambian de lado y atacan la portería opuesta. Los equipos pueden acordar no cambiar de lado con el consentimiento del árbitro.
Si los equipos utilizan la misma frecuencia de comunicación inalámbrica, deben cambiar la asignación de esa frecuencia para el segundo tiempo. Los equipos pueden acordar no cambiar la frecuencia con el consentimiento del árbitro.
Si ambos equipos desean el mismo color de parches, los equipos deben intercambiar el color del parche para la segunda mitad. Los equipos pueden acordar no cambiar el color del parche con el consentimiento del árbitro.

A.8.2 Saque inicial

El saque inicial es la forma de iniciar y reiniciar un encuentro:

- Al inicio del encuentro.
- Después que un gol ha sido anotado.
- Al inicio de la segunda mitad.
- Al inicio de los tiempos extras, donde aplique.

Un gol puede ser anotado directamente de un saque inicial.

A.8.2.1 Procedimiento

- Todos los robots están en su mitad del campo de juego.

- Los robots del equipo oponente al que realiza el saque se encuentran al menos 400 mm de la pelota hasta que esta se pone en juego.
- La pelota esta quieta en el punto central.
- El árbitro da la señal.
- La pelota esta en juego cuando es pateada y se mueve hacia delante.
- El cobrador no toca la pelota una segunda ocasión hasta que ha sido tocada por otro robot.

Después que un equipo anota un gol, el saque inicial lo realiza el equipo que recibió la anotación.

A.8.2.2 Infracciones / sanciones 2

Si el cobrador toca la pelota una segunda ocasión antes de que haya sido tocada por oro robot:

- Un tiro libre indirecto es otorgado al equipo oponente en el lugar en que ocurrió la infracción.

Para cualquier otra infracción:

- El saque inicial es realizado nuevamente.

A.8.5 Pelota colocada

Una pelota colocada es la manera de reiniciar el encuentro después de una pausa temporal que es necesaria mientras la pelota esté en juego, y por cualquier otra razón no mencionada en las reglas del juego.

A.8.5.1 Procedimiento 2

El árbitro coloca la pelota en el lugar en que se detuvo el encuentro. Por la regla A.9, los robots deben permanecer a 400 mm de la pelota mientras ésta es colocada. El encuentro se reinicia cuando el árbitro da la señal.

A.8.5.2 Infracciones / Sanciones 3

La pelota es colocada nuevamente:

- Si un robot esta a menos de 400 mm de la pelota antes de que el árbitro dé la señal.

A.8.6 Circunstancias especiales

Si un tiro libre es ganado por un equipo es su propia área defensiva es cobrado en el punto de tiro libre mas cercano al lugar de la infracción.

Si un tiro libre es ganado por un equipo en el área defensiva del equipo contrario es cobrado en el punto de tiro libre mas cercano al lugar de la infracción.
Una pelota colocada para reiniciar el encuentro después de una pausa temporal dentro del área defensiva se realiza en el punto de tiro libre mas cercano al lugar donde se encontraba la pelota antes de detenerse el encuentro.

A.9 LA PELOTA DENTRO Y FUERA DE JUEGO

A.9.1 La pelota fuera de juego

La pelota esta fuera de juego cuando:

- Haya rebasado completamente el límite de meta o de toque en el suelo o en el aire.
- El juego se haya detenido por una señal del árbitro.

Cuando la pelota este fuera de juego, los robots deben permanecer a 400 mm de la pelota mientras es colocada y hasta que el árbitro dé la señal.

A.9.2 La pelota en juego

La pelota esta en juego en cualquier otra ocasión.

A.9.3 Decisiones del Comité Técnico sobre la pelota dentro y fuera de juego

Decisión 1. Para todos los reinicios donde las reglas estipulen que la pelota se encuentra en juego cuando es golpeada y se mueve, el robot debe claramente empujar o patear la pelota para hacerla moverse. Se tiene entendido que la pelota puede permanecer en contacto con el robot por una distancia corta mientras el golpe es hecho, pero bajo ninguna circunstancia el robot debe recorrer mas de 50 mm con la pelota. No debe utilizar un mecanismo de acarreo para darle un giro contrario a la pelota.

A.10 EL MÉTODO DE ANOTAR

A.10.1 Gol anotado

Un gol es anotado cuando toda la pelota rebasa la línea de meta entre las paredes laterales de la portería debajo del cable transversal, dada la condición de que no exista ninguna infracción a las reglas del juego previa anotación del gol.

A.10.2 Equipo ganador

El equipo que anote más goles gana. Si los dos equipos anotan el mismo número de goles o si no se anotan goles, el encuentro termina en empate.

A.10.3 Reglas de la competencia

Para encuentros terminados en empate, las reglas de la competencia pueden extender el encuentro mediante tiempos extras, u otros procedimientos aprobados por la Federación RoboCup para determinar el ganador del encuentro.

A.11 FUERA DE LUGAR

El fuera de lugar no es adoptado.

A.12 FALTAS Y CONDUCTA INAPROPIADA

Las faltas y conducta inapropiada se castiga como sigue:

A.12.1 Tiro libre directo

Un tiro libre directo es ganado por el equipo opositor si un robot comete alguna de las siguientes cinco faltas:

- Hace contacto substancial con su oponente.
- Detiene a un oponente.
- Detiene la pelota deliberadamente (excepto el portero en su propia área defensiva).
- Es el segundo robot del equipo defensor en entrar simultáneamente en su propia área defensiva de tal manera que afecte substancialmente el juego.
- Es el segundo robot del equipo atacante en entrar al área defensiva del equipo contrario.

Un tiro libre es cobrado desde el lugar en que se cometió la falta.

A.12.2 Tiro penalti

Un tiro penalti es ganado si cualquiera de las cinco faltas mencionadas es cometida por un robot dentro de su propia área defensiva, sin importar la posición de la pelota y con la condición de que este en juego.

A.12.3 Tiro libre indirecto

Un tiro libre indirecto es ganado si el portero del equipo contrario dentro de su propia área defensiva comete cualquiera de las siguientes faltas:

- Detiene por mas de 15 segundos la pelota antes de liberarla.
- Detiene la pelota nuevamente después de haberla liberado sin que haya tocado a otro robot.
- Suelta la pelota y esta rebasa la línea de media cancha sin que ningún robot la haya tocado.

También se gana un tiro libre indirecto si algún jugador del equipo contrario:

- Toca al portero y el punto de contacto es dentro del área defensiva.
- Acarrea la pelota una distancia superior a 300 mm.
- Comete cualquier otra falta, no previamente mencionada en A.12, por la que se detiene el encuentro para amonestar o expulsar a un jugador.

El tiro libre indirecto es cobrado en el lugar donde se cometió la falta.

A.12.4 Sanciones disciplinarias

A.12.4.1 Infracciones de amonestación

Un equipo es amonestado y se le muestra la tarjeta amarilla, si un robot de su equipo comete alguna de las siguientes seis faltas:

1. Es culpable de conducta anti-deportiva
2. Es culpable de serio y violento contacto.
3. Persistentemente viola las reglas del juego.
4. Retrasa el reinicio del encuentro.
5. No respeta la distancia requerida cuando el encuentro es reiniciado en un saque de meta, tiro de esquina o tiro libre.
6. Modifica o daña el campo de juego o la pelota.

A.12.4.2 Infracciones de expulsión

Un robot es expulsado y se le muestra la tarjeta roja si su equipo recibe una segunda amonestación. El número de robots del equipo es reducido en uno después de la expulsión.

A.12.5 Decisiones del Comité Técnico sobre faltas y conducta inapropiada

Decisión 1. Contacto substancial es un contacto suficientemente fuerte para desviar a un robot de su orientación, posición o movimiento actual. Cuando dos robots se mueven a velocidades similares y la causa del contacto no es obvia, el árbitro permitirá la continuación del encuentro. Esta regla esta diseñada para proteger a los robots que se mueven lentamente o que están estacionarios al tiempo de contacto y por tanto deben ser detectados por sistemas de evasión de obstáculos.

Decisión 2. Amonestaciones por serio y violento contacto son una manera de desalentar a los equipos de ignorar el espíritu de no contacto. Ejemplos de infracciones de amonestación incluyen movimiento incontrolado, pobre evasión de obstáculos, empujar, o giros rápidos cerca de un oponente. En un escenario típico, el árbitro advertirá al equipo y esperará a que modifiquen sus sistemas para reducir la violencia de su juego. Si el árbitro no está satisfecho se hará la amonestación.

Decisión 3. Un robot en el campo de juego que claramente sea incapaz de movimiento será amonestado por conducta anti-deportiva.

Decisión 4. Un robot esta deteniendo la pelota si toma total control de ella removiendo todos sus grados de libertad; típicamente, encerrando la pelota y evitando el acceso a los demás; 80% del área de la pelota vista desde arriba debe estar libre. Otro robot debe ser capaz de quitarle la pelota a otro jugador.

Decisión 5. Acarreo esta definido como el movimiento activo de giro hacia atrás de la pelota que mantiene la pelota en contacto con el robot (ver la regla A.4). La distancia de restricción comienza cuando un robot empieza a acarrear y continua hasta que el robot pierde posesión de la pelota, incluso si el robot para de acarrear después. La pérdida de posesión esta definida como una separación observable de la pelota y el robot.

La restricción de la distancia de acarreo fue añadida para prevenir que un robot con un mecanismo superior de acarreo tenga control ilimitado de la pelota. La distancia de restricción aún permite que los mecanismos de acarreo sean usados para recibir pases, girar y detenerse con la pelota. Los mecanismos de acarreo pueden ser usados para acarrear la pelota grandes distancias siempre y cuando el robot pierda posesión de la pelota periódicamente, similar a como los humanos patean la pelota hacia delante para acarrearla. El comité técnico espera que la distancia sea auto-impuesta, ejemplo, los equipos asegurarán que su software cumple de antemano con la regla y se les puede pedir que lo demuestren antes de la competencia. Los árbitros aún pueden marcar faltas por violaciones y pueden amonestar (tarjeta amarilla) por violaciones repetidas.

A.13 TIROS LIBRES

A.13.1 Tipos de tiros libres.

Los tiros libres son directos o indirectos.
Para ambos tipos de tiro libre, la pelota debe esta estacionaria cuando el tiro es cobrado y el cobrador no debe tocar la pelota una segunda ocasión antes de que otro robot lo haga.

A.13.2 Tiro libre directo

- Si un tiro libre directo es pateado y entra a la portería del equipo contrario un gol es anotado.
- Si un tiro libre directo es pateado y entra a la portería del equipo cobrador un gol es anotado en contra.

A.13.2 Tiro libre indirecto

A.13.2.1 Señal

El arbitro indica un tiro libre indirecto alzando su brazo sobre su cabeza. Mantiene esta posición hasta que el tiro se ha realizado y la pelota ha tocado a otro robot o esta fuera de juego.

A.13.2.2 Pelota entrando a la portería

Un gol puede ser anotado solo si subsecuentemente toca otro robot antes de entrar a la portería.

- Si un tiro libre indirecto es pateado y entra a la portería del equipo contrario un gol es anotado.
- Si un tiro libre indirecto es pateado y entra a la portería del equipo cobrador un tiro de esquina es ganado por el equipo contrario.

A.13.3 Procedimiento de tiro libre

Si el tiro libre es ganado dentro del área defensiva, el tiro libre es tomado en el punto de la línea de toque a una distancia de 500 mm de la línea de meta mas cercana a lugar de la infracción.
Si el tiro libre es ganado por el equipo atacante dentro de 600 mm del área defensiva, la pelota se mueve al punto mas cercano a una distancia de 600 mm del área defensiva.

De cualquier otro modo, el tiro es cobrado en el punto en donde se cometió la falta.
Todos los robots deben permanecer al menos a 400 mm de la pelota.
La pelota esta en juego cuando es golpeada y se mueve.

A.13.4 Infracciones / Sanciones 4

Si cuando un tiro libre es cobrado, un oponente se encuentra a una distancia menor de la requerida:

- El tiro es vuelto a cobrar.

A.13.5 Tiro libre cobrado por un robot que no sea el portero

Si después de que la pelota esta en juego, el cobrador vuelva a tocar la pelota una segunda ocasión (sin detener la pelota) antes de que toque a otro robot:

- Un tiro libre indirecto es ganado por el equipo oponente, el tiro se cobra en el lugar donde ocurrió la infracción.

Si después de que la pelota esta en juego, el cobrador deliberadamente detiene la pelota antes de que toque a otro robot:

- Un tiro libre directo es ganado por el equipo oponente, el tiro se cobra en el lugar donde ocurrió la infracción.
- Un tiro de penalti es ganado si la infracción ocurre en el área defensiva del cobrador.

A.13.6 Tiro libre cobrado por el portero

Si después de que la pelota esta en juego, el portero vuelva a tocar la pelota una segunda ocasión (sin detener la pelota) antes de que toque a otro robot:

- un tiro libre indirecto es ganado por el equipo oponente, el tiro se cobra en el lugar donde ocurrió la infracción.

Si después de que la pelota esta en juego, el portero deliberadamente detiene la pelota antes de que toque a otro robot:

- un tiro libre directo es ganado por el equipo oponente si la infracción ocurre fuera del área defensiva del portero, el tiro se cobra en el lugar donde ocurrió la infracción.
- un tiro libre indirecto es ganado por el equipo oponente si la infracción ocurre en el área defensiva del portero, el tiro se cobra en el lugar donde ocurrió la infracción.

A.14 EL TIRO PENALTI

Un tiro penalti es ganado si cualquiera de las faltas por las cuales un tiro libre directo puede ser ganado es cometida por un robot dentro de su propia área defensiva cuando la pelota esta en juego.
Un gol puede ser anotado directamente de un penalti.
Tiempo adicional es permitido para que se cobre un tiro penalti al final de cada mitad o tiempos extras.

A.14.1 Posición de la pelota y de los robots

La pelota:

- Es colocada en el punto de penalti.

El robot que cobra el penalti:

- Es debidamente identificado.

El portero:

- Permanece sobre la línea de meta, entre las paredes laterales de la portería hasta que la pelota es golpeada.

Los demás robots:

- Dentro del campo de juego.
- Al menos a 400 mm detrás del punto de penalti.

A.14.2 El árbitro

- No da autorización para cobrar un tiro penalti hasta que los robots han tomado posición de acuerdo a esta regla.
- Decide cuando un penalti ha sido completado.

A.14.3 Procedimiento para el tiro penalti

- El robot cobrando el penalti golpea la pelota hacia delante
- No toca la pelota una segunda ocasión antes de que haya tocado a otro robot.
- La pelota esta en juego cuando es golpeada y se mueve hacia delante.

Cuando un penalti es cobrado durante el curso normal del partido, o el tiempo ha sido extendido a medio tiempo o al final del partido para cobrar un penalti, un gol es ganado si antes de pasar entre las paredes de la portería y debajo del cable transversal:

- la pelota toca uno o ambos postes y/o el cable transversal y/o al portero.

A.14.4 Infracciones / Sanciones 5

Si el árbitro da la señal para que un tiro penalti sea cobrado y antes de que la pelota este en juego ocurre alguna de las siguientes situaciones:

El jugador cobrador infringe una de las reglas del juego:

- El árbitro permite que el tiro continúe.
- Si la pelota entra, el tiro se repite.
- Si la pelota no entra, el tiro no se repite.

El portero infringe alguna regla del juego:

- El árbitro permite que el tiro continúe.
- Si la pelota entra, el gol es anotado.
- Si la pelota no entra, el tiro se repite.

Un compañero del robot cobrador entra en el área de 400 mm después del punto de penalti.

- El árbitro permite que el tiro continúe.
- Si la pelota entra, el tiro se repite.
- Si la pelota no entra, el tiro no se repite.
- Si la pelota rebota del portero, el cable transversal o un poste y es tocado por este robot, el árbitro detiene el encuentro y reinicia el juego con un tiro libre indirecto a favor del equipo defensivo.

Un compañero del portero entra en el área de 400 mm después del punto de penalti.

- El árbitro permite que el tiro continúe.
- Si la pelota entra, el gol es anotado.
- Si la pelota no entra, el tiro se repite

Un jugador de cualquier equipo infringe una de las reglas del juego:

- El tiro se repite.

Si después de que el tiro ha sido cobrado:

El cobrador toca la pelota una segunda ocasión (si detenerla) antes de que toque a otro robot:

- Un tiro libre indirecto es ganado por el equipo oponente, el tiro se cobra donde ocurrió la infracción.

El cobrador detiene la pelota antes de que toque a otro robot:

- Un tiro libre directo es ganado por el equipo oponente, el tiro se cobra donde ocurrió la infracción.

La pelota es tocada por un agente externo:

- El tiro se repite.

La pelota rebota dentro del campo de juego después de haber tocado primero al portero, el cable transversal, los postes y después es tocada por un agente externo:

- El árbitro detiene el encuentro.
- El encuentro es reiniciado con una pelota colocada en el punto donde toco al agente externo.

A.15 EL SAQUE DE BANDA

El saque es una manera de reiniciar el encuentro.
Un saque de banda es ganado:

- Cuando toda la pelota rebase el límite de toque, ya sea por tierra o por aire.
- Del punto donde cruzo el límite de toque.
- Al equipo oponente del robot que tocó por última vez la pelota.

A.15.1 Procedimiento del saque de banda

El árbitro coloca la pelota en la posición designada.
Todos los robots oponentes deben estar a 400 mm de la pelota.
La pelota esta en juego cuando es golpeada y se mueve.

A.15.2 Infracciones / Sanciones 6

Si cuando un saque de banda es cobrado, un oponente esta mas cerca de la pelota que lo requerido:

- El saque se repite

A.15.3 Saque de banda cobrado por otro robot que no sea el portero

Si después que la pelota este en juego, el cobrador toca la pelota una segunda ocasión (sin detenerla) antes de que toque a otro robot:

- Un tiro libre indirecto es ganado por el equipo oponente, el tiro es cobrado en el lugar donde ocurrió la infracción.

Si después que la pelota este en juego, el cobrador detiene deliberadamente la pelota antes de que toque a otro robot:

- Un tiro libre directo es ganado por el equipo oponente, el tiro es cobrado en el lugar donde ocurrió la infracción.
- Un tiro penalti es ganado si la infracción ocurrió dentro del área defensiva del cobrador.

A.15.4 Saque de banda cobrado por el portero

Si después que la pelota este en juego, el portero toca la pelota una segunda ocasión (sin detenerla) antes de que toque a otro robot:

- Un tiro libre indirecto es ganado por el equipo oponente, el tiro es cobrado en el lugar donde ocurrió la infracción.

Si después que la pelota este en juego, el portero detiene deliberadamente la pelota antes de que toque a otro robot:

- Un tiro libre directo es ganado por el equipo oponente si la infracción ocurrió fuera del área defensiva del portero, el tiro es cobrado en el lugar donde ocurrió la infracción.
- Un tiro libre indirecto es ganado por el equipo oponente si la infracción ocurrió dentro del área defensiva del portero, el tiro es cobrado en el lugar donde ocurrió la infracción.

A.16 SAQUE DE META

El saque de meta es una manera de reiniciar el encuentro.
Un gol puede ser anotado directamente de un saque de meta.
Un saque de meta es ganado cuando:

- Toda la pelota, habiendo tocado por último a un robot atacante, rebase el límite de meta sobre la superficie o por el aire y no sea gol.

A.16.1 Procedimiento de saque de meta

- La pelota es colocada en el punto de la línea de toque a una distancia de 500 mm de la línea de meta mas cercana a donde se rebasó la línea de meta.
- Los robots oponentes permanecen a 400 mm de la pelota hasta que se encuentre en juego.
- El cobrador no debe tocar la pelota una segunda ocasión hasta que otro robot la haya tocado.
- La pelota esta en juego cuando es golpeada y se mueve.

A.16.2 Saque de meta cobrado por otro robot que no sea el portero

Si después que la pelota este en juego, el cobrador toca la pelota una segunda ocasión (sin detenerla) antes de que toque a otro robot:

- Un tiro libre indirecto es ganado por el equipo oponente, el tiro es cobrado en el lugar donde ocurrió la infracción.

Si después que la pelota este en juego, el cobrador detiene deliberadamente la pelota antes de que toque a otro robot:

- Un tiro libre directo es ganado por el equipo oponente, el tiro es cobrado en el lugar donde ocurrió la infracción.
- Un tiro penalti es ganado si la infracción ocurrió dentro del área defensiva del cobrador.

A.16.3 Saque de meta cobrado por el portero

Si después que la pelota este en juego, el portero toca la pelota una segunda ocasión (sin detenerla) antes de que toque a otro robot:

- Un tiro libre indirecto es ganado por el equipo oponente, el tiro es cobrado en el lugar donde ocurrió la infracción.

Si después que la pelota este en juego, el portero detiene deliberadamente la pelota antes de que toque a otro robot:

- Un tiro libre directo es ganado por el equipo oponente si la infracción ocurrió fuera del área defensiva del portero, el tiro es cobrado en el lugar donde ocurrió la infracción.
- Un tiro libre indirecto es ganado por el equipo oponente si la infracción ocurrió dentro del área defensiva del portero, el tiro es cobrado en el lugar donde ocurrió la infracción.

Para cualquier otra infracción de esta regla:

- El saque de meta se repite.

A.17 TIRO DE ESQUINA

El tiro de esquina es una manera de reiniciar el encuentro.
Un gol puede ser anotado directamente de un tiro de esquina, pero solamente a la portería del equipo oponente.
Un tiro de esquina es ganado cuando:

- Toda la pelota, habiendo tocado por último a un robot defensivo, rebase el límite de meta sobre la superficie o por el aire y no sea gol.

A.17.1 Procedimiento de tiro de esquina

- La pelota es colocada en la esquina mas cercana.
- Los robots oponentes permanecen a 400 mm de la pelota hasta que se encuentre en juego.
- El cobrador no debe tocar la pelota una segunda ocasión hasta que otro robot la haya tocado.
- La pelota esta en juego cuando es golpeada y se mueve.

A.17.2 Tiro de esquina cobrado por otro robot que no sea el portero

Si después que la pelota este en juego, el cobrador toca la pelota una segunda ocasión (sin detenerla) antes de que toque a otro robot:

- Un tiro libre indirecto es ganado por el equipo oponente, el tiro es cobrado en el lugar donde ocurrió la infracción.

Si después que la pelota este en juego, el cobrador detiene deliberadamente la pelota antes de que toque a otro robot:

- Un tiro libre directo es ganado por el equipo oponente, el tiro es cobrado en el lugar donde ocurrió la infracción.
- Un tiro penalti es ganado si la infracción ocurrió dentro del área defensiva del cobrador.

A.17.3 Tiro de esquina cobrado por el portero

Si después que la pelota este en juego, el portero toca la pelota una segunda ocasión (sin detenerla) antes de que toque a otro robot:

- Un tiro libre indirecto es ganado por el equipo oponente, el tiro es cobrado en el lugar donde ocurrió la infracción.

Si después que la pelota este en juego, el portero detiene deliberadamente la pelota antes de que toque a otro robot:

- Un tiro libre directo es ganado por el equipo oponente si la infracción ocurrió fuera del área defensiva del portero, el tiro es cobrado en el lugar donde ocurrió la infracción.
- Un tiro libre indirecto es ganado por el equipo oponente si la infracción ocurrió dentro del área defensiva del portero, el tiro es cobrado en el lugar donde ocurrió la infracción.

Para cualquier otra infracción de esta regla:

- El tiro de esquina se repite.