UNIVERSIDAD TECNOLÓGICA DE LA MIXTECA

SISTEMA PARA LA DIFUSIÓN DE PARTIDAS DE AJEDREZ EN INTERNET

TESIS:
PARA OBTENER EL TÍTULO DE

INGENIERO EN ELECTRÓNICA

PRESENTA:

JOSÉ CARLOS LÓPEZ VALENCIA

DIRECTOR DE TESIS:

M.C. FELIPE SANTIAGO ESPINOSA

HUAJUAPAN DE LEÓN OAXACA. JUNIO DEL 2005

Tesis presentada el 22 de Junio del 2005 ante los siguientes sinodales:

M. C. Enrique Espinosa Justo
M. C. Hugo Suárez Onofre
M. C. José Antonio Moreno Espinosa

Director de Tesis:
M. C. Felipe Santiago Espinosa

Dedicatorias

Dedico este trabajo a mis queridos padres por su amor, comprensión y apoyo incondicional en todo momento. Gracias por haberme inculcado el espíritu de superación, esto representa la cosecha de todo su esfuerzo.

A mis hermanas por su amor y optimismo mostrado en todo momento, muchas gracias por su apoyo.

José Carlos

Agradecimientos

Agradezco a DIOS por permitirme realizar este trabajo de tesis, por darme la oportunidad de vivir y compartir este gran logro con mis seres queridos.

A mi asesor, M. C. Felipe Santiago Espinosa, por su apoyo y orientación en el desarrollo de este trabajo de tesis.

A mis compañeros y amigos, que siempre estuvieron conmigo en los momentos buenos y malos, Selene Alvarado Legaria, Procopio Gómez Martínez, Alberto Ángel Sandoval López, porque gracias a ustedes estoy culminando una etapa muy importante de mi vida.

A mis profesores por haber compartido sus conocimientos y experiencias Enrique Guzmán Ramírez, Hugo Ramírez Leyva, Ramón Maldonado Basilio, Esteban Guerrero, Víctor Manuel.

A mis grandes amigos Cesar, Homero, Eric, Ricardo, José Luís gracias por su amistad y apoyo incondicional.

A todos mil gracias.

ÍNDICE

Índice de tablas

Índice de figuras

Capítulo 1 Introducción

El ajedrez es uno de los deportes que más contribuyen al desarrollo de las habilidades mentales, debido al gran número de combinaciones ajedrecísticas que se pueden realizar en cada partida. Este deporte se ha venido practicando desde hace siglos en todo el mundo y su interés se ha incrementado intensamente.

Debido a que la práctica del ajedrez fortalece el pensamiento lógico, se ha instituido en algunas escuelas como materia optativa. Por otro lado, los clubes de ajedrez así como el Consejo Nacional para la Cultura y las Artes [Mussot] han fomentado su difusión; sin embargo existe poca difusión de partidas en tiempo real como suele suceder con la mayoría de otros deportes; esto debido a su estructura y complejidad.

La televisión es un medio de difusión masiva que podría trasmitir partidas de ajedrez en vivo, sin embargo esto no se realiza por los altos costos que implica; además que la duración de una partida se desconoce, puede durar unos cuantos minutos o incluso horas; lo que imposibilita la programación de un partido para su difusión. Otro medio de difusión es la Internet; una opción es el envío de video, pero requiere de un gran ancho de banda e implica la conexión de pocos usuarios. Por otro lado Internet cuenta con aplicaciones con la capacidad de animación con lo que se logra representar gráficamente sucesos reales.

Con base en los problemas observados para difundir el ajedrez, el presente trabajo tiene el objetivo de realizar un sistema para la difusión de partidas de ajedrez en tiempo real; manipulando los recursos de una computadora personal (PC) por medio de software para dar difusión a las partidas de ajedrez de una forma sencilla y a un bajo costo a través de la Internet. La base de este sistema es el procesamiento digital de imágenes, obtenidas de una cámara digital de la cual se obtiene la secuencia de movimientos de las piezas, se verifican a través de software y hace la difusión por medio de la Internet, haciendo uso de animaciones realizadas en Macromedia Flash;

así como su difusión en forma local por medio de un monitor, pantallas de TV o el empleo de proyectores.

1.1 Objetivo general

El objetivo general es implementar un sistema para la difusión de partidas de ajedrez por Internet. El sistema se desarrollará para un ambiente cliente-servidor; donde el servidor tendrá las funciones de adquisición de información, procesamiento de la información y difusión de la partida. Para la adquisición de imagen se empleará una cámara digital para determinar las posiciones en el tablero de la pieza que se ha movido, el software de dicho sistema validará las jugadas y obtendrá la notación de los movimientos desarrollados durante el juego. Cada vez que se genere un movimiento, éste será enviado a través de Internet a cada usuario, para lo cual se implementará una aplicación del lado del cliente para simular en forma gráfica el movimiento de la pieza; de esta forma se envían pequeños paquetes de información que contienen únicamente la posición del último movimiento, generando poco tráfico en el flujo de información en la red.

1.2 Objetivos específicos

Se implementará un sistema para la difusión de partidas de ajedrez en Internet el cual constará principalmente de dos módulos: el servidor y el cliente. Para el servidor se realizará un programa para las plataformas Windows NT, 2000 y XP; la implementación de dicho programa requerirá de lo siguiente:

- Generar una clase para la manipulación de los controladores de una cámara digital para la captura de imágenes en tiempo real.
- Generar una clase con los algoritmos básicos de procesamiento digital de imágenes para determinar las coordenadas de los movimientos realizados en el tablero.
- Generar una clase para la lectura de eventos mediante el puerto paralelo en Windows XP, para controlar la toma de las imágenes y llevar el control de tiempo de cada jugador.
- Manejar la validación automática de los movimientos ajedrecísticos realizados.
- Implementar un módulo para la narración de cada uno de los movimientos realizados.
- Guardar las partidas jugadas automáticamente sin necesidad de tener que anotarlas.
- Llevar un historial de partidas jugadas para su posterior análisis.

- Montar un servidor de http dentro del programa para difundir las partidas sin necesidad de instalar otro módulo servidor.

En lo que respecta al cliente, se realizará un programa que permita observar la partida por medio de animaciones, así como también escuchar la narración por medio de un motor de voz; en la implementación de esta aplicación se realizará lo siguiente:

- Implementar una interfaz con Macromedia Flash para la animación por medio de ActionScripts, con lo cual el motor de cálculo correrá del lado del cliente, quedando el servidor libre de procesamiento.
- Generar una aplicación independiente de la plataforma.
- Realizar un canal de comunicación con el servidor basada en sockets.
- Implementar la narración de las partidas por medio de JavaScripts para hacer uso de los agentes de Microsoft.

1.3 Antecedentes

1.3.1 Tablero DGT

En el mercado existe el *tablero electrónico de la compañía DGTprojects [DGT],* el cual es un tablero de madera con sensores magnéticos incorporados que se encargan de detectar los movimientos realizados sobre el tablero (figura 1.1), y la información de los movimientos es transmitida automáticamente a una computadora.

Figura 1.1 Tablero comercial

La difusión de los movimientos realizados en el tablero se hace mediante el protocolo de transferencia de archivos (ftp). El funcionamiento de este tablero está basado en sensores magnéticos colocados dentro del tablero, y cada pieza contiene un imán. Conjuntamente estos dispositivos proporcionan la posición de la pieza en el tablero. La desventaja que presenta este tablero es su elevado costo que es

aproximadamente $5,000.00. Así como también se requiere de la compra de licencias para el uso del software de difusión, lo cual incrementa el costo del sistema.

1.3.2 Sistemas de visión artificial

En el campo de la inteligencia artificial, la visión por computadora puede considerarse como el conjunto de técnicas que permiten el procesamiento, análisis e interpretación de imágenes digitales. La visión por computadora estudia los métodos que permiten recibir información a través de emular la percepción visual humana, procesarla y tomar decisiones, produciendo resultados finales que son de utilidad para el hombre, seguir un objeto en movimiento, determinar la distancia de un objeto, identificar patrones, por mencionar algunas.

Dentro de los métodos de visión por computadora, se encuentra la segmentación, la cual consiste en separar una imagen digital en entidades significativas, un adecuado método de segmentación es la clave para el correcto análisis y procesamiento de imágenes.

El método comúnmente utilizado para las aplicaciones reales es el llamado de segmentación global basado en umbrales fijos. Éste ha sido empleado en sistemas para el conteo automático de huevecillos de moscas [Cruz], donde por las características de contraste entre los huevecillos y el fondo se aplica un umbral de valor fijo. Otro de los trabajos emplea un umbral global, el cual es propuesto en [Wang et.al.], donde se usa la información contenida en el histograma para determinar de manera automática los umbrales, haciendo uso del algoritmo Lloyd-Max.

Con base en esto, el presente trabajo pretende realizar la obtención de la posición de una pieza de ajedrez en el tablero utilizando una cámara digital como sensor de movimiento, lo que conduzca a una solución más económica utilizando los recursos de una computadora personal.

1.4 Aspectos generales

Para determinar la secuencia de movimientos de las piezas se optó por utilizar una cámara digital como sensor de movimiento, con la cual se tomarán imágenes de dos movimientos consecutivos y al obtener la diferencia de estas imágenes se generará otra, con la información de las piezas que se han movido. Al analizar la imagen resultante se obtendrán las coordenadas de la posición que ocupa la pieza en el tablero. La captura de estas imágenes se realizará al momento que el jugador detenga su reloj; para determinar esto se colocará un sensor de contacto que generará una interrupción en el puerto paralelo, la cual ejecutará el evento de captura de imagen y el cambio de corrida del reloj entre cada jugador.

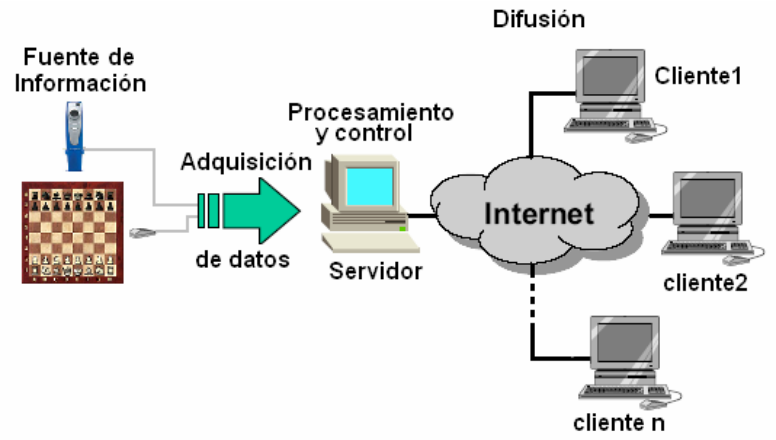


Figura 1.2 Esquema general del sistema.

La figura 1.2 muestra un esquema general del sistema; se puede observar que la etapa de obtención de información consta básicamente de una cámara digital y un botón para capturar la imagen y detener el reloj. La información generada por estos elementos se envía a una PC (Servidor), la cual realizará el control de la cámara y la lectura del botón. Para ello se desarrollará un programa que realizará el procesamiento de la imagen, para determinar los movimientos y verificar que éstos sean válidos; contará con un motor de voz que narrará el movimiento de las piezas. El sistema llevará un historial de los movimientos, el cual se guardará para su posterior análisis, también contendrá el módulo servidor para dar difusión a la partida.

Dentro de este programa se realizará una interfaz del juego para poder enviar la partida a un monitor u otro dispositivo de salida para su difusión en forma local.

Para la implementación cliente-servidor del sistema, es necesario determinar el lugar dónde sea más conveniente ubicar al motor de cálculo (local o remoto). En principio, si la interfaz y el motor de cálculo se ejecutan en el computador del cliente, se reduce el uso del canal de comunicación, la comunicación con el servidor se realizará con sockets, recibiendo únicamente las posiciones del último movimiento realizado, para el caso de una partida en vivo; la partida se simula del lado del cliente haciendo uso de animaciones en flash las cuales se manipulan por medio de ActionScript. Si el usuario desea ver partidas finalizadas, la aplicación del cliente recibirá el historial completo. En este programa se desplegará el tablero con las correspondientes jugadas llevadas a cabo, así también el programa contará con la posibilidad de narrar los movimientos.

1.5 Delimitaciones

El trabajo estará delimitado por los siguientes aspectos:

- Las piezas del ajedrez deberán de ser de un color contrastante con el color de los cuadros del tablero, esto debido a que el sistema manipulará las imágenes en escala de grises.
- La colocación de la cámara deberá ser tal que la visión de esta cubra todo el tablero.
- La iluminación deberá ser uniforme para evitar la proyección de sombra.
- El tablero y la cámara deberán estar fijas.

1.6 Justificación

Los Ingenieros en Electrónica deben saber explotar los recursos de hardware con que cuenta una computadora; como lo es una cámara CCD, la cual captura la luz sobre una pequeña rejilla de píxeles en su superficie y luego pasa esta información a un convertidor analógico a digital, teniendo así una información completa de los sucesos que ocurren en el medio ambiente. Otro recurso que se va a manejar es el puerto paralelo, el cual en sistemas como Windows NT, 2000 y XP no permiten acceder al hardware de forma tan sencilla como lo hacen las versiones 95, 98 y ME.

Con el desarrollo del presente trabajo se sientan las bases para el futuro desarrollo de sistemas de control, que utilicen una cámara digital como medio de adquisición de información del entorno, así como también el acceso al puerto paralelo para la interacción con hardware externo. Esto debido a que las clases desarrolladas pueden reutilizarse en otras aplicaciones.

Cabe resaltar el uso de un servidor Web, ya que es el principio de funcionamiento de sistemas controlados a distancia mediante Internet.

1.7 Software/hardware a utilizar

Para el desarrollo del proyecto se utiliza Builder C++ Versión 6.0; debido a que es una herramienta de desarrollo rápido, simple de utilizar, la cual cuenta con un gran

número de componentes que facilitan de forma notable la creación de aplicaciones [Charter].

Para el desarrollo de la aplicación del cliente se emplea Macromedia Flash MX; ya que es una herramienta que permite realizar interfaces gráficas de usuario de manera fácil y rápida, utilizando el lenguaje ActionScripts para la programación de animaciones.

En la conversión de texto a voz se utiliza el componente MsAgente el cual permite la manipulación de los agentes de Microsoft.

La síntesis de voz se realiza a través de los TtS(Text to Speech) que son sistemas que transforman texto en sonido que se puede reconocer como voz [Msagent].

Como hardware se requiere una computadora personal con sistema operativo Windows XP. Una cámara Digital Cool@Cam, u otra con características similares, un conector DB25 con un botón que se conectará entre los pines ocupado (busy) y tierra, dos leds conectados a los pines 2 y 4 (Ver manual).

Para el acceso a Internet se emplea un ruteador con soporte DDNS(Sistema Dinámico de Nombres de Dominio) y un modem para realizar la conexión, si el proveedor de Internet asigna una dirección IP dinámica y también es necesario estar actualizando dicha dirección en el proveedor de nombres de dominio. En caso de que el acceso a Internet sea por medio de una LAN y se cuente con una IP fija, el ruteador y el servicio del nombre no son necesarios.

1.8 Organización de la tesis

En esta sección se proporciona un panorama general de la organización del documento, dando una breve descripción de los temas a tratar en cada uno de los capítulos.

En el capítulo 2, “Fundamentos teóricos” se expone el estudio de los fundamentos sobre los sistemas operativos multitarea, procesamiento digital de imágenes (PDI) y los protocolos de Internet, que son la base del presente trabajo.

En el capítulo 3, “Diseño e implementación del sistema” se describe la solución propuesta para el problema. Se presenta el diseño, implementación y diagrama de flujo de cada etapa del sistema.

En el capítulo 4, llamado “Resultados”, presenta los resultados experimentales del procesamiento de imágenes y muestra las interfaces de los programas que se desarrollaron.

En el capítulo 5, “Conclusiones y trabajos futuros”, se dan a conocer las conclusiones del trabajo, así como los trabajos futuros.

Finalmente en los apéndices se muestran las clases que se han desarrollado y el manual para el manejo de los programas tanto del servidor como del cliente.

Capítulo 2 Fundamentos teóricos

En este capítulo se describen los fundamentos teóricos que son la base del presente trabajo de tesis, para la captura y difusión de una partida de ajedrez por Internet. Se revisarán los siguientes temas:

- Sistemas operativos multitarea
- Visión
- Internet

2.1 Sistemas operativos multitarea

Se distinguen por sus habilidades para poder soportar la ejecución de dos o más trabajos activos al mismo tiempo. Esto trae como resultado que la Unidad Central de Procesamiento (UCP) siempre tenga alguna tarea que ejecutar, aprovechando al máximo su utilización.

Su funcionamiento consiste en dividir las aplicaciones en varios hilos (*Threads)* de ejecución. Cada uno de estos hilos se puede ver como un programa absolutamente independiente de los demás, trabajando todos en conjunto para formar el programa completo.

En esta sección se revisan las características y modos de trabajo de la plataforma Windows, para su posterior aplicación en la lectura de eventos en el puerto paralelo, implementando un hilo como proceso de verificación del estado de dicho puerto.

2.1.1 Introducción a los procesos

Un proceso es una instancia de ejecución de un programa, caracterizado por su contador de programa, su palabra de estado, su segmento de texto, pila y datos. [Dekey].

Los procesos pueden ser cooperantes o independientes, en el primer caso se entiende que los procesos interactúan entre sí y pertenecen a una misma aplicación. En el segundo caso, los procesos independientes, son aquellos que no interactúan y no requieren información de otros procesos.

Un proceso puede estar en cualquiera de los siguientes tres estados: en ejecución, listo y bloqueado.

En ejecución: El proceso ocupa la CPU actualmente, es decir, se está ejecutando.

Listo: El proceso dispone de todos los recursos para su ejecución, sólo le falta la CPU.

Bloqueado: Al proceso le falta algún recurso para poder seguir ejecutándose, además de la CPU. Por recurso se pueden entender un dispositivo, un dato, etc. El proceso necesita que ocurra algún evento que le permita poder proseguir su ejecución.

2.1.2 Hilos

Un hilo es cada una de las subtareas que debe ejecutar un proceso [Vegas].

Un hilo tiene un área privada de almacenamiento llamada CONTEXTO DEL HILO, con un único identificador (Id _ cliente), un área de almacenamiento para subsistemas y librerías, una pila en modo usuario y otra en modo kernel.

Cada hilo se ejecuta en un espacio virtual separado para cada proceso. Cuando un proceso consta de varios hilos, éstos se ejecutan en el mismo espacio virtual.

Para la ejecución de los hilos, el sistema operativo se ayuda del multiprocesamiento en el cual se ejecutan varios programas aparentemente en forma simultánea. Como se observa el la figura 2.1, los procesos o tareas se paralelizan, lo cual se logra asignando tiempos de procesador independiente de cada hilo

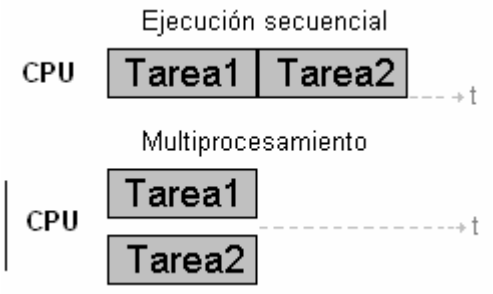


Figura 2.1 Multiprocesamiento

2.1.3 Acción en tiempo real

Este proyecto se define como sistema de respuesta en tiempo real, debido a la noción de tiempo de respuesta o tiempo que tarda el sistema en generar una salida asociada a una entrada.

Un sistema de tiempo real se debe distinguir de otros en donde el tiempo de respuesta es importante pero no crítico. Controlando eventos que ocurren en el mundo real (planta), actuando con un tiempo de respuesta restringido.

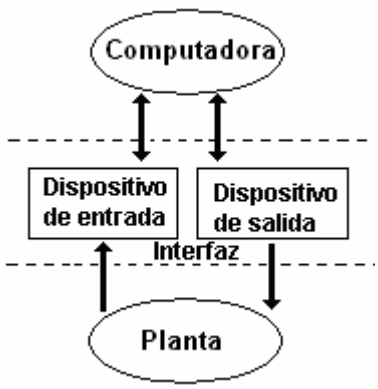


Figura 2.2 Esquema general de un sistema de control por computadora

En la figura 2.2 se representa el esquema general de un sistema de control por computadora, el cual es un claro ejemplo de un sistema de respuesta en tiempo real. Una característica común en los sistemas de tiempo real es que la computadora se encuentra conectada en un entorno dentro del cual está trabajando para dispositivos de los que recibe y a los que da una amplia variedad de estímulos. Las entradas y salidas son señales de comunicación que tienen una característica en común. Éstas están conectadas a dispositivos, que se comunican con la computadora mediante procesos que operan en tiempos diferentes, y la computadora por medio de software se encarga de producir las salidas correspondientes.

En la actualidad los costos de una computadora son relativamente bajos, por lo que es factible realizar con ellas sistemas dedicados al control o automatización, lo cual hace necesario entender la estructura de los sistemas operativos actuales, para poder explotar los periféricos de hoy en día.

2.2 Visión

Un sistema de procesamiento de imágenes se utiliza para desarrollar diferentes tareas como la inspección de procesos, visión artificial, control de calidad, entre otras.

El presente trabajo empleará las técnicas de procesamiento de imágenes para el desarrollo de un sistema de visión artificial que permite la detección de cambios en la colocación de las piezas en un tablero de ajedrez.

2.2.1 Componentes de un sistema de visión

Un sistema de visión depende en gran parte de sus componentes, existiendo cinco etapas básicas [González et. al.] (figura 2.3) para su correcto funcionamiento. A continuación se describen brevemente estas etapas:

Adquisición de la imagen: En esta etapa se obtiene la imagen con formato digital. Para ello se necesita un sensor de imágenes y la posibilidad de digitalizar la señal producida por el sensor.

Preprocesado: En esta etapa se realzan las características de la imagen, como aumento de contraste o disminución de ruido.

Segmentación: En un sistema de visión artificial, la segmentación es una etapa determinante en el procesamiento de imágenes. Su objetivo es el aislamiento de los objetos de interés, para realizar el análisis de sus características. Existe una gran cantidad de algoritmos para segmentar objetos en una imagen, la elección de algunos de ellos depende de la aplicación.

Representación y Descripción: Es el proceso mediante el cual se obtienen las características relevantes para diferenciar un objeto de otro, como puede ser su ubicación en el ambiente.

Reconocimiento e Interpretación: El reconocimiento es el proceso que asigna una etiqueta a un objeto y la interpretación implica asignar un significado a un conjunto de objetos reconocidos.

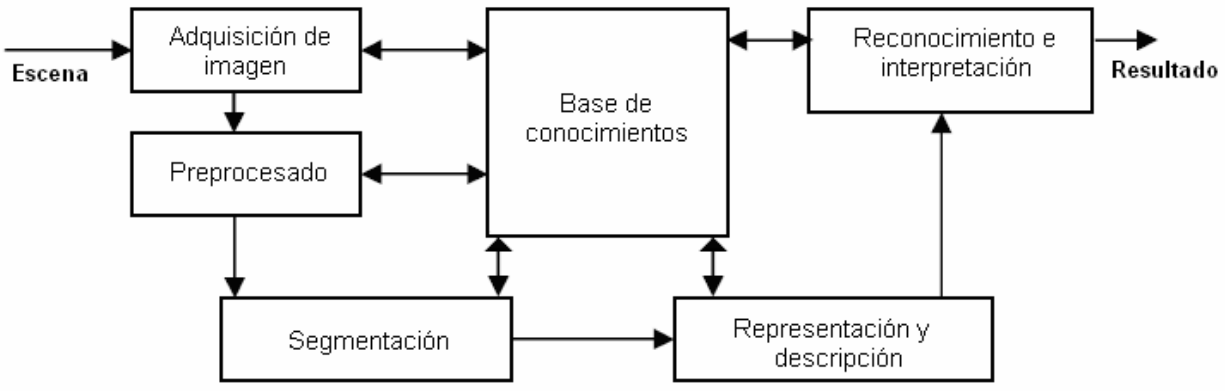


Figura 2.3 Etapas de un sistema de visión artificial.

Es necesario un conocimiento previo sobre los objetos a reconocer, esto es parte de una base de conocimientos. La interacción entre la base de conocimiento y los módulos de procesamiento puede ser muy compleja. La base de conocimientos puede contener una lista de todas las posibles características de los objetos o bien, las reglas para el correcto procesamiento en cada etapa.

2.2.2 Imagen digital

Una imagen básicamente es una colección de puntos en un arreglo bidimensional donde cada punto o píxel representa una serie de parámetros de la imagen [Gerhard et. al.]. En la figura 2.4 se muestra la representación más simple de una imagen donde n es el número de columnas y m el número de renglones.

$$I = \begin{Bmatrix} M_{11} & M_{12} & \cdots & M_{1n} \\ M_{21} & M_{22} & \cdots & M_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ M_{m1} & M_{m2} & \cdots & M_{mn} \end{Bmatrix}$$

Figura 2.4 Representación de una imagen

Cada uno de los parámetros $M_{m,n}$ están dados por el modelo de color sobre el que se esté trabajando; los modelos más utilizados son el RGB, YIQ, CMY. Cuando se trabaja con la intensidad del modelo YIQ, se dice que la imagen se encuentra en tonos de gris. Para poder representar una imagen en forma digital se emplean diferentes formatos, los cuales están relacionados con el número de bits necesarios (Profundidad de color) para su representación. En el presente trabajo se trabajará con imágenes bitmap de 24 bits. En la tabla 2.1 se muestra una clasificación de acuerdo a la profundidad de color de la imagen.

Tabla 2.1 Profundidad de color

Número de bits	Descripción
1	Imagen monocromática Blanco/Negro
4	Imagen de 16 niveles
8	Imagen de 256 niveles, escala de grises
15	Imagen de 32768 niveles
16	Imagen de 65536 niveles
24	Imagen a color (RGB)
32	Modo MCYK

La mayoría de los dispositivos actuales para la captura de imagen entregan una imagen con una profundidad de color de 24 bits, la cual representa un vector formado por la combinación de los colores básicos utilizados en los dispositivos electrónicos de video. Estos colores son los del modelo RGB (Red, Green, Blue), rojo, verde y azul respectivamente. Con la combinación lineal de estos colores se genera cualquier otro color. Básicamente el modelo RGB se puede representar con un espacio tridimensional como se muestra en la figura 2.5.

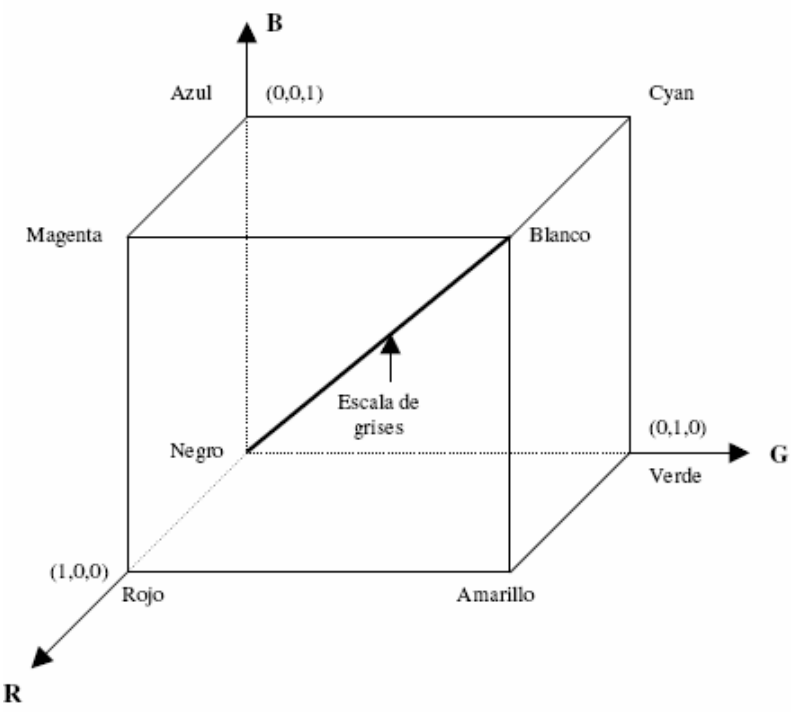


Figura 2.5 Cubo de valores RGB.

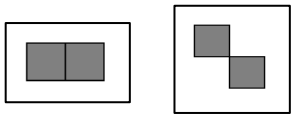
Donde (r,g,b) son las coordenadas del color y (R,G,B) los vectores unitarios a su color correspondiente. Una representación de una imagen en gris en este modelo (RGB) se puede ver como el píxel donde sus tres componentes son iguales. Es decir la recta que se ubica del origen (0,0,0) al punto (255,255,255).

2.2.2.1 Definición de vecindades

En esta sección se considerarán algunas relaciones básicas pero importantes entre los píxeles de una imagen digital.

Adyacencia.

Dos píxeles son adyacentes si y solo si tienen en común una de sus fronteras, o al menos una de sus esquinas [González et. al.]. La figura 2.6 muestra píxeles adyacentes.



a) b)

Figura 2.6 Píxeles adyacentes.
(a) adyacentes por frontera. (b) adyacente por esquina.

Píxel Vecino

Dos píxeles son vecinos, si cumplen con la definición de adyacencia. Si los píxeles comparten una de sus fronteras se dice que los mismo son "Vecinos directos", si por el contrario solo se tocan en una de sus esquinas se llaman "Vecinos indirectos".

Vecindad

Algunas de las vecindades mas comunes son la de conectividad 4 y la de conectividad 8; las cuales se pueden observar en figura 2.7, donde la primera está formada por píxeles que son vecinos directos, mientras que la vecindad de 8 está formada tanto por vecinos directos como por indirectos. [Young et. al.]

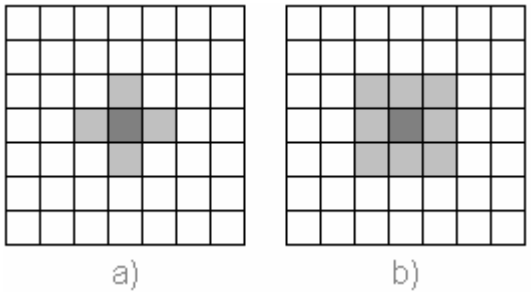


Figura 2.7 Vecindades
(a) vecindad de 4, (b) vecindad de 8

En el desarrollo de técnicas de procesamiento de imágenes, que involucren el análisis de una determinada región de la escena digital, es posible encontrar vecindades de 5x5 y hasta de 9x9; básicamente la definición de las dimensiones de la vecindad, depende de la técnica que se este desarrollando.

2.2.2.2 Operaciones sobre imágenes

Las operaciones que se realizan sobre imágenes y que tienen como resultado otra imagen se pueden clasificar en tres grupos [Vicens et. al.]:

- **Operaciones puntuales:** Son aquellas en que el valor de cada píxel *O[i,j]* de la imagen resultante se obtiene a partir del valor del píxel *I[i,j]* de la imagen original, sin involucrar ningún otro píxel. Como ejemplos se pueden señalar la umbralización, el realce de contraste, la modificación del histograma, incremento del brillo entre otras.

- **Operaciones locales:** Son aquellas en las que el valor de cada píxel *O[i,j]* de la imagen resultante se obtiene a partir de los valores del píxel *I[i,j]* y de sus vecinos en la imagen original. Entre operaciones locales básicas se pueden mencionar las convoluciones, las operaciones morfológicas y las operaciones booleanas locales. Como ejemplos se pueden señalar los filtros por convolución, los detectores de bordes, etc.

. **Operaciones globales:** Son aquellas en las que el valor de cada píxel *O[i,j]* de la imagen resultante se obtiene de los valores del píxel *I[i,j]* y de todos los demás píxeles en la imagen original. Como ejemplos de las operaciones globales se citan la transformada de Fourier, la del coseno, y sus respectivas antitransformadas.

Las diferentes operaciones se muestran gráficamente en la figura 2.8

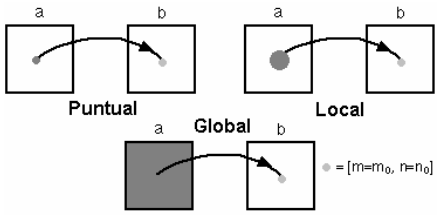


Figura 2.8 Tipos de operaciones en las imágenes

2.2.3 Operaciones puntuales

Algunas de las operaciones aritméticas son las siguientes:

Resta o diferencia (-): La resta de píxeles tiene como entrada dos imágenes y produce una tercera imagen como salida; también se puede realizar la resta con un valor constante.
Básicamente la resta consiste en ir haciendo la operación píxel a píxel con las dos imágenes de entrada es decir:

$$O[x,y]= I_1[i,j] - I_2[i,j]$$

También se puede calcular el valor absoluto de la resta de las dos imágenes:

$$O[x,y]= | I_1[i,j] - I_2[i,j] |$$

O se puede realizar la resta con un valor constante:

$$O[x,y]= | I_1[i,j] – C |$$

Comúnmente la resta de imágenes se emplea para detectar cambios o el movimiento de objetos; si en las dos imágenes no hay cambios la imagen resultante será una imagen en blanco de lo contrario se obtendrá una imagen con el objeto que se ha movido.

De este mismo tipo son las operaciones de suma, multiplicación y división cuyo estudio no se realizará debido a que cae fuera del contexto del presente trabajo. Así también se encuentran las operaciones lógicas: or, nor, not, and, nand, xor, xnor.

2.2.4 Filtros digitales

La aplicación de estos filtros se basa en la inspección de los vecinos de un píxel P[i,j] y una máscara, dada por la vecindad del píxel. El acceso a cada uno de estos valores queda definido como en la ecuación 2.1.

$$P[i,j] = \begin{bmatrix} P[i-1,j-1] & P[i-1,j] & P[i-1,j+1] \\ P[i,j-1] & P[i,j] & P[i,j+1] \\ P[i+1,j+1] & P[i+1,j] & P[i+1,j+1] \end{bmatrix} \quad (2.1)$$

Donde los valores P[i,j] representan el tono de gris en la imagen o bien el tono de cada componente RGB en una imagen a color.

A continuación se explicarán brevemente algunos de los filtros que se implementaron y cuyo código se encuentra en el disco anexo.

2.2.4.1 Filtro de mediana

El empleo de un filtro de mediana es una buena alternativa cuando se desea la reducción de ruido, sin difuminar los bordes. Este filtro actúa sobre el entorno de un píxel reemplazando el nivel de gris del píxel por la mediana del entorno. Para calcular la mediana de un conjunto de valores, primero se deben arreglar los valores del conjunto en orden ascendente y escoger el valor que se sitúe en la mitad de dicho arreglo. Un ejemplo de este proceso se muestra en la figura 2.9

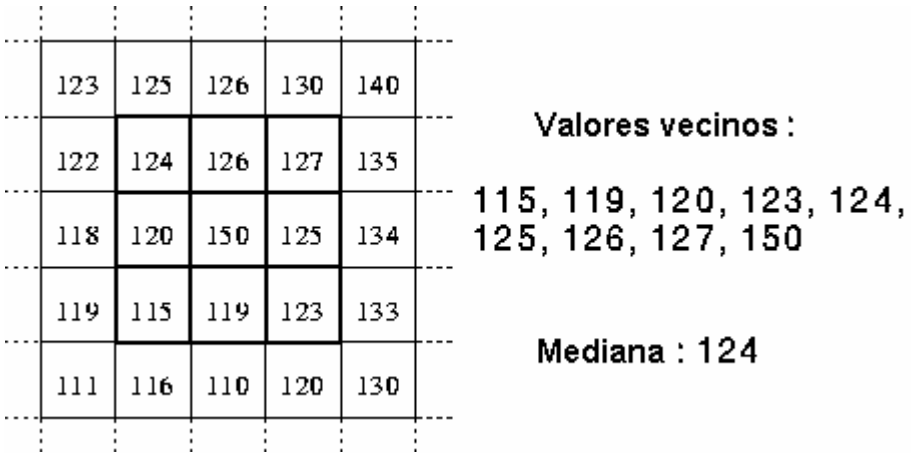


Figura 2.9 Cálculo de la mediana de los píxeles vecinos en una matriz de 3X3

2.2.4.2 Filtro media aritmética (Promedio)

El efecto de la media aritmética es el de suavizar la imagen, ya que los píxeles vecinos se parecerán debido a la mezcla que se produce entre ellos. Una observación que se puede hacer es que la imagen se hará borrosa y las transiciones fuertes (bordes) se disolverán parcialmente.

El modelo más simple corresponde a la media aritmética, éste considera la media de los píxeles en un entorno de 3x3 centrado en un píxel (x,y).

Analíticamente queda representada por:

$$\text{PixelSalida}[x,y] = \frac{1}{9}\sum_{i=-1}^{1}\sum_{j=-1}^{1}\text{PixelEntrada}[x+i, y+j] \tag{2.2}$$

Esto se puede reescribir posicionalmente de la siguiente manera:

$$I'[x,y] = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} \otimes \begin{bmatrix} I[x-1,y-1] & I[x,y-1] & I[x+1,y-1] \\ I[x-1,y] & I[x,y] & I[x+1,y] \\ I[x-1,y+1] & I[x,y+1] & I[x+1,y+1] \end{bmatrix} \tag{2.3}$$

De donde la matriz de convolución que modela a la media aritmética resulta:

$$\tilde{M}_{MA} = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} \tag{2.4}$$

Lógicamente este filtro se puede aplicar en el cuerpo de la imagen, excepto en las orillas, donde se deben usar estructuras reducidas debido a la carencia de todos los vecinos.

2.2.4.3 Distorsión radial

La distorsión radial es una consecuencia de los defectos del sistema óptico, que hacen que se produzcan variaciones en la geometría de la imagen, es decir, diferencias en la posición relativa de sus puntos con respecto a la imagen ideal que se desearía obtener [González et. al.]. Estas variaciones son producidas por la curvatura de las lentes de la cámara, que generan una imagen con distintos factores de ampliación a lo largo del campo de la imagen. Como consecuencia, puede resultar tener mas ampliación el centro de la imagen que la periferia (distorsión negativa, de barril) o al contrario, tener menos ampliación en el centro de la imagen que en la periferia (distorsión positiva, de cojín). En la distorsión negativa (de barril) las líneas se estiran hacia fuera como si se inflara la imagen, mientras que en la distorsión positiva (de cojín) las líneas caen hacia dentro como si se desinflara la imagen. Este efecto es ilustrado en la figura 2.10, donde se muestra el efecto que produce sobre la imagen.

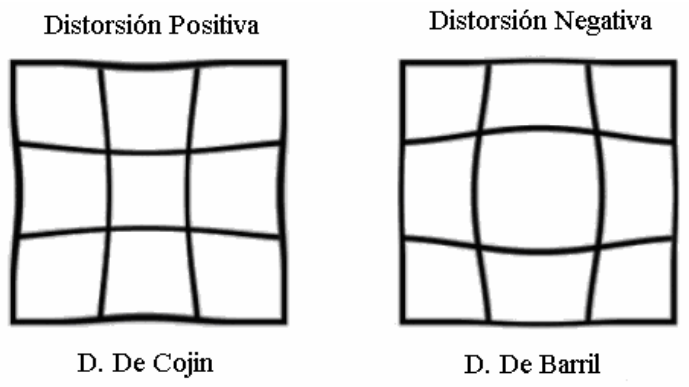


Figura 2.10 Efectos de la distorsión radial

La ecuación 2.5 corrige aproximadamente la curvatura. Para la mayoría de lentes la proyección ax y ay podría ser la misma o estar relacionada con el ancho y alto de la imagen. Típicamente estos valores están entre 0 (No hay corrección) y 0.1 (ángulo de curvatura del lente). [Bourke].

$$P'_x = P_x(1 - a_x \parallel P \parallel^2)$$
$$P'_y = Py(1 - a_y \parallel P \parallel^2) \qquad (2.5)$$

2.3 Internet

En esta sección se estudian los protocolos de Internet, TCP[1]/IP[2], que trabajan en la transmisión de datos; clasificándolos para su estudio en un modelo de tres capas: acceso a la red, transporte y aplicación [Stallings 2002].

La Internet es una red global en la cual, cada computadora actúa como un cliente y un servidor; dando como resultado aplicaciones que comunican a miles de computadoras alrededor del mundo.

Internet no depende de la máquina ni del sistema operativo utilizado. Lo que permite transmitir información entre un servidor Unix y una pc con Windows, gracias a la familia de protocolos TCP/IP que permiten que la Internet sea una Red de redes.

El modelo básico en Internet es el modelo Cliente/Servidor. El cliente es un programa que le solicita un servicio, el cual es proporcionado por el servidor.

[1] Transmission Control Protocol
[2] Internet Protocol

2.3.1 Capa de acceso a la red

Esta capa está relacionada con el intercambio de datos entre el computador y la red a la que pertenece. El computador emisor debe proporcionar a la red la dirección del destino, de tal forma que la red pueda encaminar los datos al destino apropiado. [Stallings 2002].

2.3.1.1 Clases de redes

Las direcciones se dividen conceptualmente en dos partes: el *identificador de red* e *identificador del servidor (host)*. El tamaño del identificador del servidor puede ser de 8,16 o 24 bits, por lo que se generan tres clases principales de redes: Clase A, Clase B, Clase C, como se muestra en la figura 2.11 y las redes cuyo primer byte es superior a 223 corresponden a otras clases especiales, la clase D está reservada para difusión de tablas de enrutamiento y canales de videoconferencia; la clase E aún no está definida.

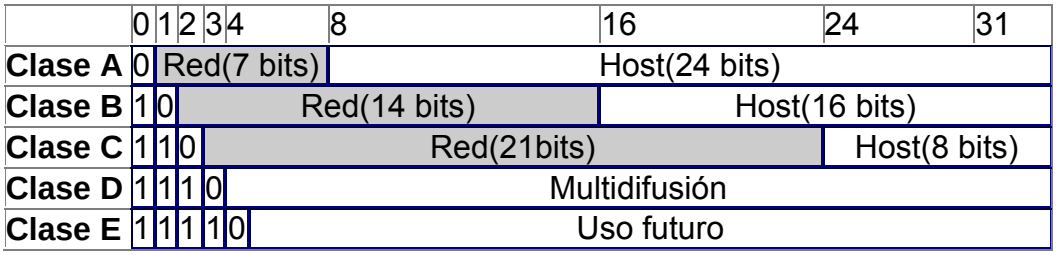


Figura 2.11 Formatos de dirección IP

Este esquema proporciona flexibilidad al asignar direcciones IP a una red, donde se considera el tamaño y las necesidades de ésta, permitiendo una mezcla de tamaños de red en un conjunto de redes.

2.3.1.2 Direcciones IP

La dirección IP es el identificador de cada computadora dentro de una red. Las direcciones IP están formadas por 4 bytes (32 bits). Se suelen representar de la forma a.b.c.d donde cada letra es un número comprendido entre el 0 y el 255.

Las direcciones IP se clasifican en:

Direcciones IP públicas. Son visibles en todo Internet. Una computadora con una IP pública es accesible desde cualquier otra conectada a Internet. Para conectarse a Internet es necesario tener una dirección IP pública. Las direcciones IP públicas son únicas en la Internet.

Direcciones IP privadas. Son visibles únicamente por computadoras que pertenecen a su misma red o a otras redes privadas interconectadas por un ruteador (router). Se

utilizan para crear subredes. Las computadoras con direcciones IP privadas pueden conectarse a Internet por medio de un ruteador que tenga una IP pública.

Otra clasificación para las direcciones IP es la siguiente:

Direcciones IP estáticas. Una pc servidor que se conecte a la red con dirección IP estática siempre utilizará la misma IP. Las direcciones IP públicas estáticas son las que utilizan los servidores de Internet con objeto de que estén siempre localizables por los usuarios de Internet.

Direcciones IP dinámicas. Una computadora servidor que se conecte a la red mediante dirección IP dinámica, cada vez que se conecte a Internet tendrá una dirección IP distinta. Las direcciones IP públicas dinámicas son las que se utilizan en las conexiones a Internet mediante un módem.

2.3.1.3 Protocolo IP

El protocolo IP es el principal protocolo del modelo OSI [Noli et. al.], así como parte integral del TCP/IP. Las tareas principales del IP son el direccionamiento de los datagramas de información y la administración del proceso de fragmentación de dichos datagramas.

El datagrama es la unidad de transferencia que el IP utiliza, algunas veces identificada en forma más específica como datagrama Internet o datagrama IP.

Las principales características de este protocolo son:

- **N*o orientado a conexión,*** en el que cada paquete puede seguir una ruta distinta entre el origen y el destino. Por lo que pueden llegar duplicados o desordenados.

- **N*o fiable,*** porque los paquetes pueden perderse, dañarse o llegar retrasados.

- Transmisión en unidades denominadas datagramas.

2.3.2 Capa de transporte

La capa de red transfiere datagramas entre dos computadoras a través de la red utilizando como identificadores las direcciones IP. La capa de transporte añade el número de *puerto* para distinguir el destino del paquete dentro de un mismo servidor. El empleo de puertos también se utiliza para el envío de mensajes; al momento de enviar un paquete el sistema debe buscar un puerto libre para transmitir los mensajes.

2.3.2.1 Puertos

Para distinguir las distintas conexiones dentro de una misma pc se utilizan los puertos, los cuales definen el tipo de servicio que atenderán.

Un puerto es un número de 16 bits, por lo que existen 65536 puertos en cada computadora.

Los números de puerto de las aplicaciones cliente son asignados dinámicamente y generalmente son superiores al 1024. Cuando una aplicación cliente quiere comunicarse con un servidor, busca un número de puerto libre y lo utiliza. En cambio, las aplicaciones servidoras utilizan unos números de puerto prefijados: llamados puertos bien conocidos, aunque se puede definir un servidor en cualquier otro puerto. Los puertos bien conocidos están definidos en la RFC 1700 y se pueden consultar en http://www.ietf.org/rfc/rfc1700.txt.

2.3.2.2 Protocolo UDP

El protocolo UDP (User Datagram Protocol, protocolo de datagrama de usuario) proporciona una comunicación sencilla entre las aplicaciones de dos computadoras. Al igual que el protocolo IP, UDP es:

- ***No orientado a conexión***. No establece una conexión previa con el otro extremo para transmitir un mensaje UDP. Lo cual implica que los datos pueden duplicarse o llegar desordenados al destino.
- ***No fiable.*** Los mensajes UDP se pueden perder o llegar dañados.

UDP utiliza el protocolo IP para transportar sus mensajes. UDP, no añade ninguna mejora en la calidad de la transferencia; aunque sí incorpora los puertos origen y destino en su formato de mensaje. Las aplicaciones (y no el protocolo UDP) deberán programarse teniendo en cuenta que la información puede no llegar de forma correcta.

2.3.2.3 Protocolo TCP

El protocolo TCP (*Transmission Control Protocol*, protocolo de control de transmisión) está basado en IP que es no fiable y no orientado a conexión, y sin embargo es:

- ***Orientado a conexió****n*. Es necesario establecer una conexión previa entre las dos computadoras antes de poder transmitir algún dato. A través de esta conexión los datos llegarán siempre a la aplicación destino de forma ordenada y sin duplicarse. Al finalizar la transmisión, es necesario cerrar la conexión.
- ***Fiable***. La información que envía el emisor llega de forma correcta al destino.

Para que se pueda establecer una comunicación es necesario abrir previamente una conexión. Esta conexión garantiza que todos los datos lleguen en forma correcta, ordenada y sin duplicados.

Cada vez que se abre una conexión, se crea un canal de comunicación bidireccional en el que ambas aplicaciones pueden enviar y recibir información, es decir, una conexión *full-dúplex*.

2.3.2.4 Conexiones

Una de las alternativas para que dos programas intercambien datos entre sí es utilizando sockets. Un socket es un canal de comunicación entre dos programas que corren sobre computadoras diferentes o en la misma pc, utilizando el protocolo TCP/IP.

Existen básicamente dos tipos de canales de comunicación o sockets, los orientados a conexión y los no orientados a conexión.

En el primer caso ambos programas deben conectarse entre ellos con un socket y hasta que no esté establecida correctamente la conexión, ninguno de los dos puede transmitir datos, esta es la forma de operar del protocolo TCP.

En el segundo caso, no es necesario que los programas se conecten. Cualquiera de ellos puede transmitir datos en cualquier momento, independientemente de que el otro programa esté escuchando o no. Este es el modo de operar del protocolo UDP.

Para poder realizar la conexión entre dos programas se necesitan los siguientes datos:
• Nombre o dirección IP del servidor (host)
• Nombre del servicio o puerto del servidor (port)

<Dirección IP : Puerto>

En el siguiente ejemplo (figura 2.12) se crean tres conexiones. Las dos primeras son al mismo servidor Web (puerto 80) y la tercera a un servidor de FTP (puerto 21).

Host 1	Host 2
194.35.133.5:1256	135.22.8.165:80
184.42.15.16:1305	135.22.8.165:80
184.42.15.16:1323	135.22.10.15:21

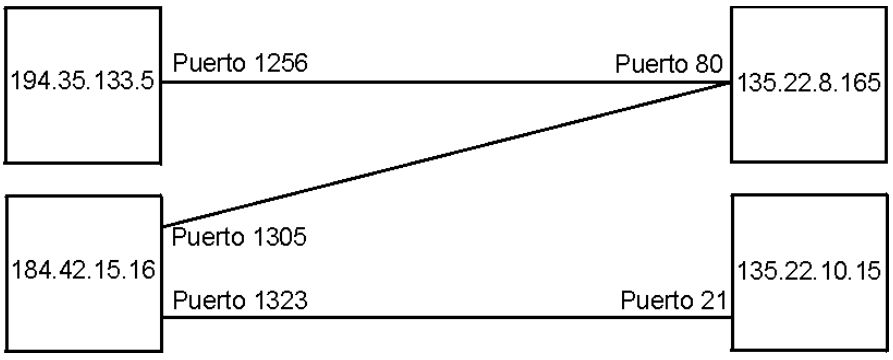


Figura 2.12 Ejemplo de conexión.

2.3.3 Capa de aplicación

Ofrece a las aplicaciones la posibilidad de acceder a los servicios de las demás capas y define los protocolos que utilizan las aplicaciones para intercambiar datos, como correo electrónico, gestores de bases de datos y servidor de ficheros. Los servicios más conocidos son los siguientes:

- HTTP (HyperText Transfer Protocol)
- FTP (File Transfer Protocol)
- SMTP (Simple Mail Transfer Protocol)
- POP (Post Office Protocol)
- SSH (Secure Shell)
- Telnet

Otros protocolos de nivel de aplicación que facilitan el uso y administración de la red son los siguientes:

- SNMP (Simple Network Management Protocol)
- DNS (Domain Name *System*)

2.3.3.1 HTTP

Es el protocolo base de la Web (WWW[2]), usado en cada transacción. Las letras significan Hyper Text Transfer Protocol, es decir, protocolo de transferencia de hipertexto. El hipertexto es el contenido de las páginas web, y el protocolo de transferencia es el sistema mediante el cual se envían las peticiones y respuestas para el acceso a una página web.

HTTP es un protocolo basado en TCP sin estados cada transacción se trata independientemente. Por lo cual cada que se realiza una petición se creará una nueva conexión entre el cliente y el servidor; al finalizar la transacción de la página la conexión termina.

2.3.3.2 DNS

Un **DNS** (*Domain Name System*) es un conjunto de protocolos y servicios funcionando como una base de datos distribuida, que permite a los usuarios utilizar nombres en lugar de direcciones IP numéricas. Cuando la dirección asignada es una dirección IP dinámica que está en constante cambio, es conveniente tener un nombre.

[2] World Wide Web

2.3.3.3 Resolución de nombres

La resolución de nombres es uno de los pilares de Internet, el cual consiste en encontrar la dirección IP correspondiente a un nombre de domino. Esta resolución de nombres de dominio se realiza de arriba hacia abajo, comenzando con el servidor de nombres raíz y siguiendo luego hacia los servidores localizados en las ramas del árbol de la red. Esto se puede ver gráficamente en la figura 2.13.

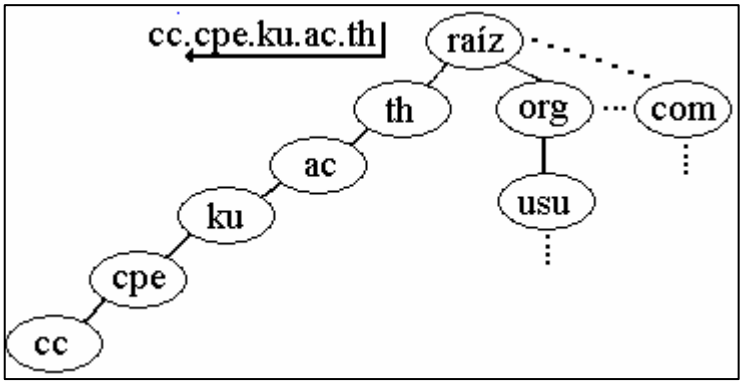


Figura 2.13 Resolución de nombres

Hay dos formas de utilizar un sistema de nombres de dominio: contactar un servidor de nombres cada vez o solicitar al sistema de servidores de nombres que realice la traducción completa. En este caso, el software cliente forma una solicitud de nombres de dominio que contiene el nombre a resolver, una declaración sobre la clase del nombre, el tipo de respuesta deseada y un código que especifica si el servidor de nombres debe traducir el nombre completamente. Se envía la solicitud a un servidor de nombres para su resolución.

Cuando un servidor de nombres de dominio recibe una solicitud, verifica si el nombre señala un subdominio sobre el cual tenga autoridad. Si es así, traduce el nombre a una dirección de acuerdo con su base de datos y anexa una repuesta a la solicitud, antes de enviarla de regreso al cliente. Si el DNS no puede resolver el nombre completamente, verifica que tipo de interacción especificó el cliente. Si el cliente solicita una traducción completa (una resolución recursiva en la terminología DNS), el servidor se pone en contacto con un servidor de nombres de dominio que pueda resolver el problema del nombre y devuelve la respuesta al cliente.

Capítulo 3 Diseño e implementación del sistema

3.1 Metodología de desarrollo

Para lograr el éxito en el desarrollo de software es fundamental establecer una metodología que permita tener un control sobre cada etapa a desarrollar. Es por esto que, el sistema se desarrolló a partir de un modelo TOP-DOWN, donde se divide el problema principal en módulos menos complejos que darán solución a tareas específicas del sistema y en conjunto darán solución a éste.

Cada módulo se realizó con base al modelo clásico o de cascada (figura 3.1), debido a que se desarrollan a partir de los requerimientos de la entrada o salida de cada módulo y se refinan hasta llegar a su versión final, que satisface dichos requerimientos.

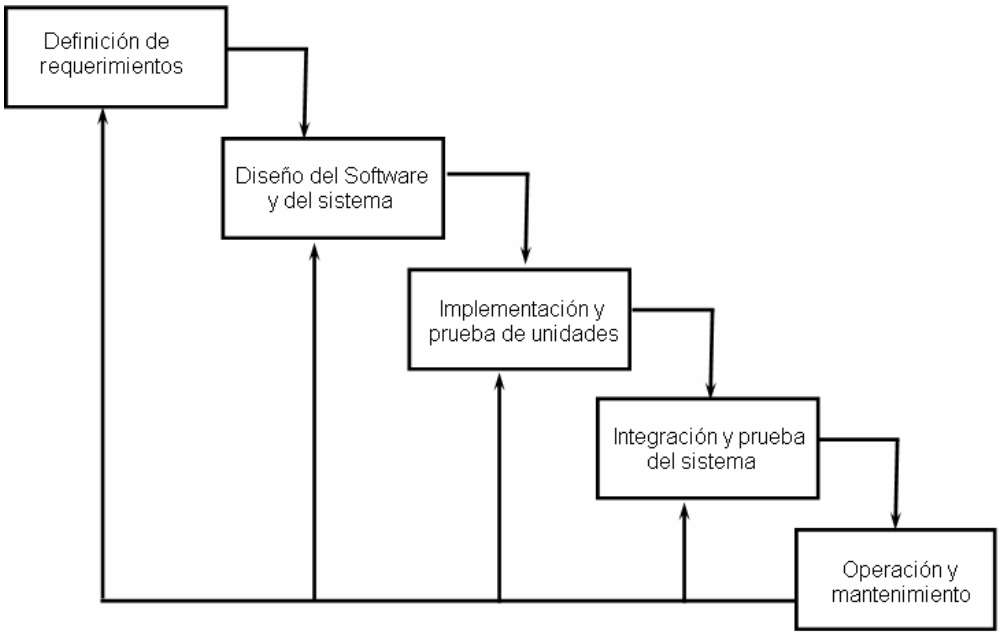


Figura 3.1 Modelo de cascada

El modelo en cascada cuenta con las siguientes etapas:

- *Definición de requerimientos*: etapa en que se establecen alcances y delimitaciones del sistema.

- *Diseño del software y del sistema*: se hace una descripción TOP-DOWN para el diseño y desarrollo del sistema visto como un sistema completo, hasta especificar cada uno de los módulos que lo constituyen.

- *Implementación y prueba de unidades*: describe la implementación de cada módulo desarrollado y se evalúa cada unidad de manera independiente.

- *Integración y prueba del sistema*: etapa que integra cada módulo desarrollado para obtener el funcionamiento del sistema completo.

- *Operación y mantenimiento*: se evalúa el funcionamiento del sistema para detectar errores o realizar modificaciones que mejoren su funcionamiento.

3.2 Diseño del sistema

El sistema para la difusión de partidas de ajedrez en Internet, denominado WebChess, es dividido en dos partes: el módulo servidor y el módulo cliente. El módulo servidor se encarga de obtener, procesar y difundir la partida de ajedrez; mientras que el módulo cliente recibe, procesa y muestra dicha partida.

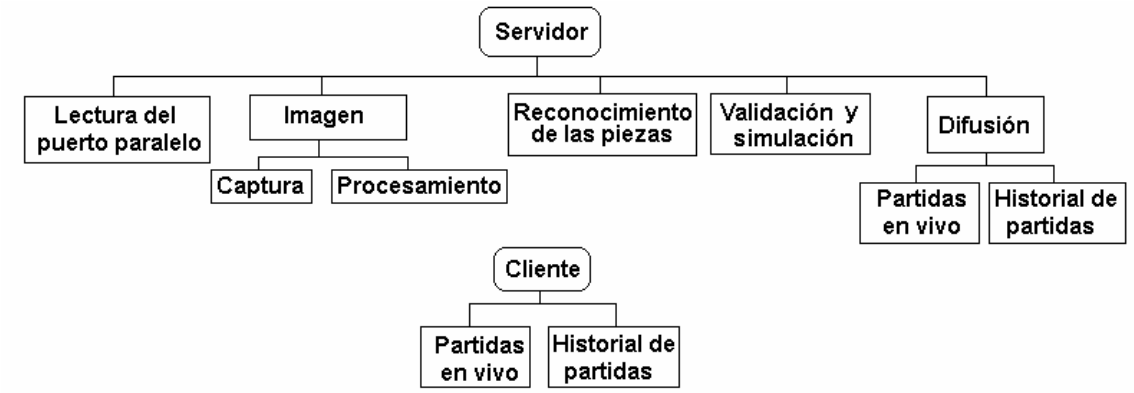


Figura 3.2 Diagrama a bloques de la implementación

Como puede observarse en la figura 3.2 se desarrollaron sub-módulos para cada módulo principal. El módulo servidor consta de 5 sub-módulos:

- El sub-módulo de lectura del puerto paralelo verifica el estado de dicho puerto y cuando ocurra un cambio generará una señal para la captura de la imagen.

- El sub-módulo de imagen, consta de dos etapas; la captura de la imagen, la cual se encarga de la apertura de los controladores e inicialización de la cámara; y la de procesamiento de la imagen; donde se implementan los algoritmos del procesamiento de imágenes para la identificación de las coordenadas de la pieza movida.
- Reconocimiento de piezas, este sub-módulo identifica a que pieza corresponde el movimiento detectado.
- Validación y simulación, una vez identificada la pieza, se valida su movimiento; de ser correcto se genera la simulación en pantalla y la reproducción en forma de sonido de las posiciones de partida y fin de la ficha identificada.
- Difusión, este sub-módulo cuenta con dos etapas: partidas en vivo e historial de partidas. La etapa de partidas en vivo implementará la comunicación vía TCP/IP cliente/servidor para el envío de las jugadas detectadas por el servidor. En el historial de partidas se muestra la lista de partidas previamente jugadas, las cuales pueden ser vistas paso a paso de la misma forma que las partidas en vivo.

Para el módulo cliente se requirieron 2 sub-módulos:

- El sub-módulo de partidas en vivo se encarga de realizar la conexión vía TCP para recibir las posiciones de los últimos movimientos detectados, para después simularlos en una interfaz realizada con Macromedia Flash. Este sub-módulo cuenta también de una etapa de reproducción en forma de voz de la notación del movimiento recibido; esto se realiza a través de comunicar Macromedia Flash y JavaScripts.
- El sub-módulo del historial de partidas mostrará una lista de las partidas previamente jugadas, las cuales están almacenadas en un archivo XML que es leído cada que se entra al historial. A partir de esta lista se puede seleccionar una partida para ser visualizada del mismo modo que una partida en vivo con la diferencia que ya se cuenta con todas los movimientos realizados.

3.3 Implementación

La implementación del sistema para difundir partidas de ajedrez por Internet, se lleva a cabo mediante el desarrollo de los bloques mostrados en la figura 3.2.

3.3.1 Lectura del puerto paralelo

Para la lectura del puerto paralelo se utiliza una librería de enlace dinámico (dll, dynamic Link Library) denominada inpout32, la cual es un controlador que permite hacer uso del puerto paralelo sobre plataformas Windows 95/98/NT. Se emplea esta dll debido a que windows NT restringe el uso de los puertos para brindar mayor seguridad

y solo los procesos en modo Kernel tienen permiso de ejecutar instrucciones de lectura y escritura. El kernel es el responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora. Esta librería (inpout32.dll) puede ser adquirida gratuitamente en Internet y puede ser empleada en lenguajes de programación como son Visual Basic y C++.

Como entrada al sistema se requiere saber cuándo ha ocurrido un evento en el puerto paralelo; para lo cual se realizó una clase denominada ParallelThread_Unit. Esta clase se describe en la figura 3.3:

TParallelThread
Datos miembro
inpfuncPtr inp32; outfuncPtr out32; HINSTANCE hLib; bool Connected; AnsiString DeviceName; int ReceiveQueue; bool Disconnecting; int ReceiveInterval;
Funciones miembro

TParallelThread(void(__closure *NewReturnMethod)(AnsiString));
TParallelThread();
void AfterReceiveData();
int BytesAvailable(void);
void Close();
void Open();
void(__closure *ReturnMethod)(AnsiString);
bool Connect(void);
short Read();
void Write(short datos);
bool Disconnect(void);
bool GetConnected(void);
AnsiString GetDeviceName(void);
HANDLE GetDeviceHandle(void);
AnsiString GetAvailableData(void);
void SetReceiveInterval(int NewReceiveInterval);
int GetReceiveInterval(void);
TStringList* TParallelThread::GetAvailableDevicesNames(bool IncludeParallel,
TStringList * AvaiableDeviceNames);

Figura 3.3 Diagrama de la clase TParallelThread

Dentro de la clase TParallelThread es posible resaltar las siguientes funciones:

- **Open**() Carga la librería inpout32 e instancia las funciones de lectura y escritura (inpfuncPtr inp32, oupfuncPtr outp32). Así como también instancia el hilo que se encargará de verificar si hay cambios en el puerto.
- **Close**() Libera la librería y suspende la ejecución del hilo.
- **GetAvailableDevicesNames**(AvaiableDevicesNames) Retorna un listado con los periféricos instalados en la computadora (LPT1,LPT2).

- **Read**() Lee los datos presentes en el puerto.
- **Write**(dato) Escribe el dato enviado al puerto.
- **BytesAvailable**() Verifica constantemente el estado del pin ocupado(busy)
- **AfterReceiveData**() Retorna los datos leídos.
- **SetReturnMethod**() Establece el evento que será generado después de haber detectado un cambio en el pin ocupado.
- **SetReceiveInterval**() establece el tiempo en que será verificado el estado del puerto.

En el apéndice C se encuentra una descripción de todos los atributos y funciones.

Básicamente, al instanciar esta clase se crea un objeto "ParallelThread" el cual genera un hilo o subproceso que se encarga de conocer el estado del puerto paralelo; cuando ha ocurrido un cambio en la línea de datos de entrada éste genera un evento, que se encargará de la captura de la imagen.

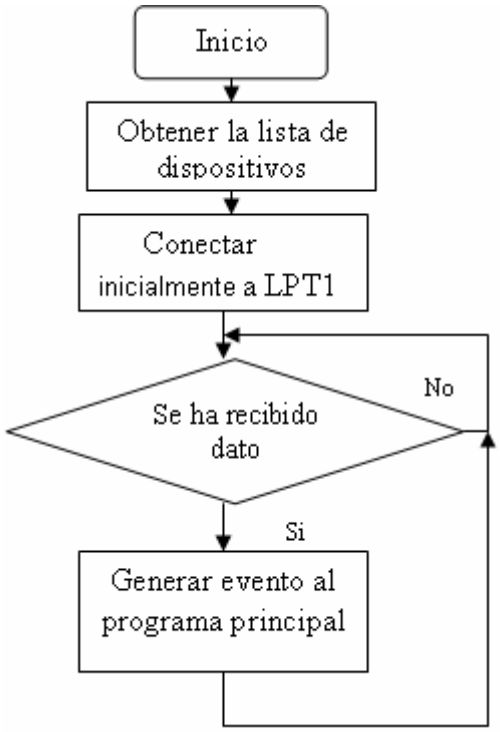


Figura 3.4 Diagrama de la lectura del puerto paralelo

En la figura 3.4 se muestra el diagrama del procedimiento a realizar para la generación del evento en el puerto paralelo; puede observase que el primer paso, una vez creado el objeto TparallelThread, es obtener y asignar el dispositivo a utilizar, encontrado en la lista de dispositivos del registro de Windows *"hardware\\devicemap\\parallel ports"*. Esta lista puede contener los dispositivos LPT1 y LPT2 que por lo general se encuentran en la dirección 0X378 y 0X279 respectivamente. Al iniciar el programa, se conectará el objeto TparallelThread al primer dispositivo encontrado (LPT1).

3.3.2 Imagen

El funcionamiento de este sub-módulo consiste básicamente en la resta de dos imágenes, las cuales son obtenidas con una cámara digital. Se toma una imagen cada que se genera un evento en el puerto paralelo. Como salida este sub-módulo retorna una estructura que contiene el número de movimientos encontrados y las posiciones de estos.

El trabajo de este sub-módulo es distribuido en dos etapas (figura 3.2); la captura y el procesamiento de la imagen, cuyo objetivo se describe a continuación.

3.3.2.1 Captura de imagen

Como sistema de adquisición de imágenes se utiliza una cámara digital Cool@Cam. La cámara obtiene una imagen cada que ocurre un evento en el puerto paralelo, generado cuando un jugador ha detenido su reloj de juego. Esta cámara proporciona una imagen del tipo Bitmap (BMP) con el Formato Común de Imagen (Common Image Format, CIF) de 352 x 288 píxeles y con una profundidad de color de 24 bits.

La cámara está colocada sobre el tablero de juego a una distancia a la cual pueda cubrir exactamente todo el tablero (figura 3.5), cabe resaltar que el tablero con el cual se hicieron las pruebas tiene las medidas oficiales que establece la Federación Internacional de Ajedrez (FIDE) que son de 5 x 5 cm por cuadro.

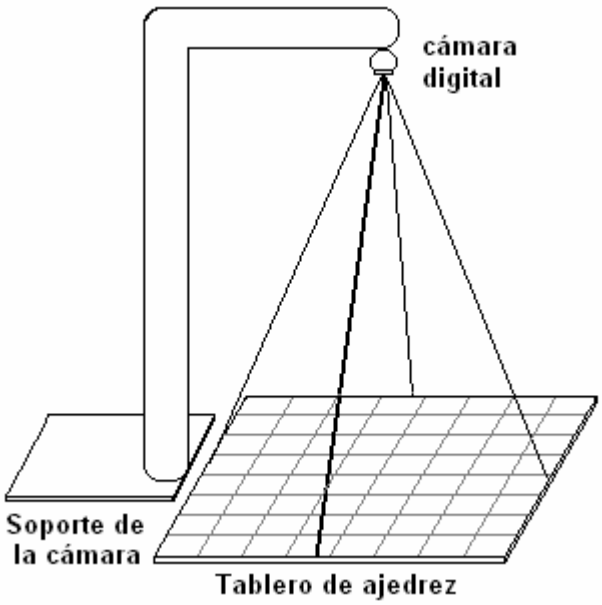


Figura 3.5 Colocación de la cámara

Para poder acceder a la cámara se hizo uso de la librería vfw.h (Video for Windows), que permite capturar y procesar video; dando acceso directo a los controladores de los dispositivos de video que estén registrados en la computadora.

Estos dispositivos se encuentran registrados en la ruta MiP*c\HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\VIDEO de los registros de Windows.*

Para llevar acabo el proceso de inicialización de la cámara de video se implementó la clase c_cap, la cual hereda de la clase vfw de Windows; que es descrita en la figura 3.6.

TCap
Datos miembro
HWND ParentHandle; HWND hwndVideo; CAPDRIVERCAPS CapDrvCaps; int SelectedDevice; bool active;
Funciones miembro

TCap(HWND Handle);
~TCap();
TStringList *pStringCapDrivers;
int EnumCapDrv();
bool Connect (int Selected);
bool IsConnect();
void Format ();
void Source ();
void CaptureFrame (char *FileName);
void CaptureToClip();
void SetNewWindowPos(const int,const int,bool);

Figura 3.6 Diagrama de la clase TCap

Mediante la clase TCap se inicializará la cámara y permitirá acceder a los parámetros de esta, así como a las funciones heredadas de la clase vwf (video for windows).

Dentro de la clase TCap se resaltan los siguientes atributos:

-**ParentHandle** Manejador para crear la ventana de video donde se mostrará la imagen de la cámara Web.
- **hwndVideo** Manejador de la ventana de video.
- **CapDrvCaps** capacidades de los controladores.
- **SelectedDevice** Índice para identificar el dispositivo seleccionado.
- **active** Bandera para saber el estado del objeto (activa/inactiva).

Y las siguientes funciones:

- **TCap** Constructor del objeto.
- **pStringCapDrivers** Contiene la lista de controladores de video instalados.
- **EnumCapDrv**() enumerar los controladores de video instalados.
- **Connect** () Realiza las funciones de inicialización de la cámara.
- **IsConnect**() Retorna el estado del objeto (activo/inactivo)
- **Format** () Muestra la ventana de configuración del formato de la imagen.

- **Source** () Muestra la ventana de configuración de los parámetros de la imagen, como son brillo, contraste y color.
- **CaptureFrame** (FileName) Captura y guarda una imagen a archivo.
- **CaptureToClip**() Captura y almacena una imagen en memoria (Porta papeles).
- **SetNewWindowPos**(const int,const int,bool) Establece la posición y el estado (visible/oculto) de la ventana de video.

En el apéndice C se encuentra una descripción completa de los atributos y funciones de esta clase.

Con esta clase se crea un objeto Tcap, se creará una ventana para la captura de video, una vez creado el objeto se procede a obtener una lista de controladores de dispositivos de video instalados en la PC; también se informará si no se encontró algún dispositivo de video. De haber encontrado algún dispositivo de video se intentará conectar al primer dispositivo obtenido en la lista. Para llevar acabo la conexión con dicho dispositivo se verificará si se encuentra conectado a la PC, de no estar conectado se dará el aviso de “Error al iniciar la cámara”; si el dispositivo de video se encuentra conectado se cargan los controladores para este dispositivo; a continuación se configura la velocidad o cuadros por segundo con los que se desea que trabaje la cámara, este valor se dejó en 33.3 milisegundos o 30 cuadros por segundo. Ahora queda mandar la imagen a la ventana de captura de video. Este proceso se ilustra en el diagrama de flujo mostrado en la figura 3.7.

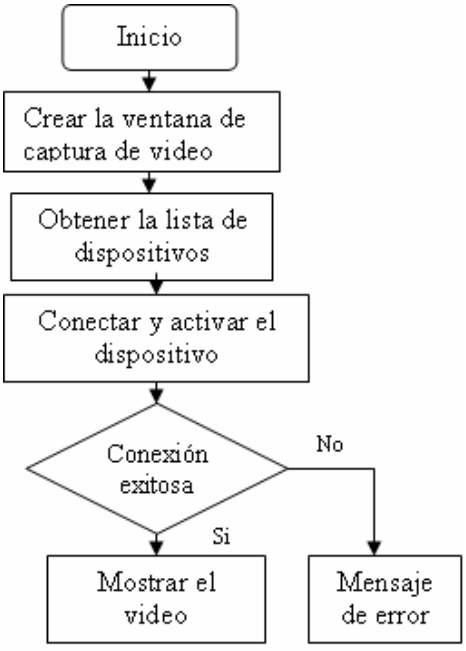


Figura 3.7 Diagrama de inicialización de la cámara de video

Las características de las imágenes con las que se trabaja son las siguientes:

Tamaño de la imagen: 352 X 288 píxeles

Formato de la imagen: RGB de 24 bits
Calidad: Color de Alta calidad

Cabe resaltar que la configuración de las propiedades de la imagen, como son tamaño, brillo, contraste, etc. solamente se realiza una vez y la próxima vez que se active la cámara se cargará la última configuración realizada.

3.3.2.2 Procesamiento de la imagen

Una vez que se tienen las imágenes a color, se procede a realizar el procesamiento digital de éstas. El cual se lleva a cabo mediante los pasos que se muestran en la figura 3.8 y que son explicados en las líneas posteriores.

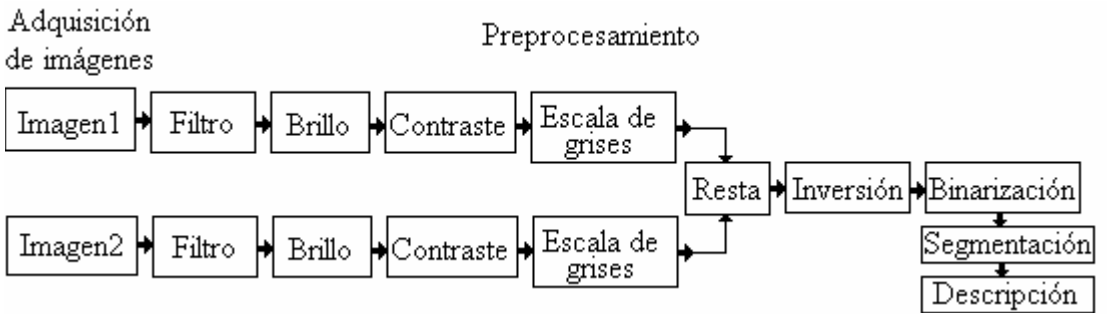


Figura 3.8 Diagrama a bloques del procesamiento de la imagen

3.3.2.2.1 Filtro suavizante pasa bajo

Estos filtros atenúan los componentes de alta frecuencia presentes en la imagen, responsables de los bordes y detalles finos, por lo cual son utilizados para reducir el ruido [González et. al.]

Para lograr esto en la imagen, se aplica un filtro pasa bajas, cuya máscara se muestra en la figura 3.9

$$\begin{bmatrix} 1/8 & 1/4 & 1/8 \\ 1/4 & 1/2 & 1/4 \\ 1/8 & 1/4 & 1/8 \end{bmatrix}$$

Figura 3.9 Máscara del filtro pasa-bajas

El efecto de aplicar el filtro suavizante sobre la imagen, es el de hacerla borrosa y eliminar detalles de intensidad de color y bordes.

3.3.2.2.2 Aumento de brillo

Aumentar el brillo de la imagen consiste en sumar un valor constante al del tono de cada uno de los píxeles de la misma. Si este valor es positivo estamos aumentando el brillo de la imagen, si por el contrario es negativo, el brillo disminuye. La rutina usada para aumentar o disminuir el brillo queda representada de la siguiente forma:

PíxelSalida=PíxelEntrada+brillo;

Si (PíxelSalida >255) ***Entonces*** PíxelSalida =255;

Si (PíxelSalida <0) ***Entonces*** PíxelSalida =0;

Donde brillo es un valor positivo o negativo, dependiendo de si se quiere aumentar o disminuir el mismo. Sin embargo, las modificaciones de brillo corren el riesgo de perder información de la imagen.

3.3.2.2.3 Cambio del contraste

Consiste en incrementar el rango dinámico de los niveles de gris de la imagen procesada. Esta transformación permite de alguna manera distinguir con más facilidad a un objeto de otro [González et. al.].

Existen diferentes funciones de transferencia que permiten aumentar el contraste, algunas pueden aproximarse con una función que permita distribuir los píxeles sobre una curva que haga que los colores claros se vuelvan más claros y los oscuros más oscuros. Teniendo en cuenta que los valores de entrada que no quedan dentro de este rango (0-255) son convertidos al limite más cercano (0 o 255).

Las funciones de transferencia que se implementaron fueron las siguientes:

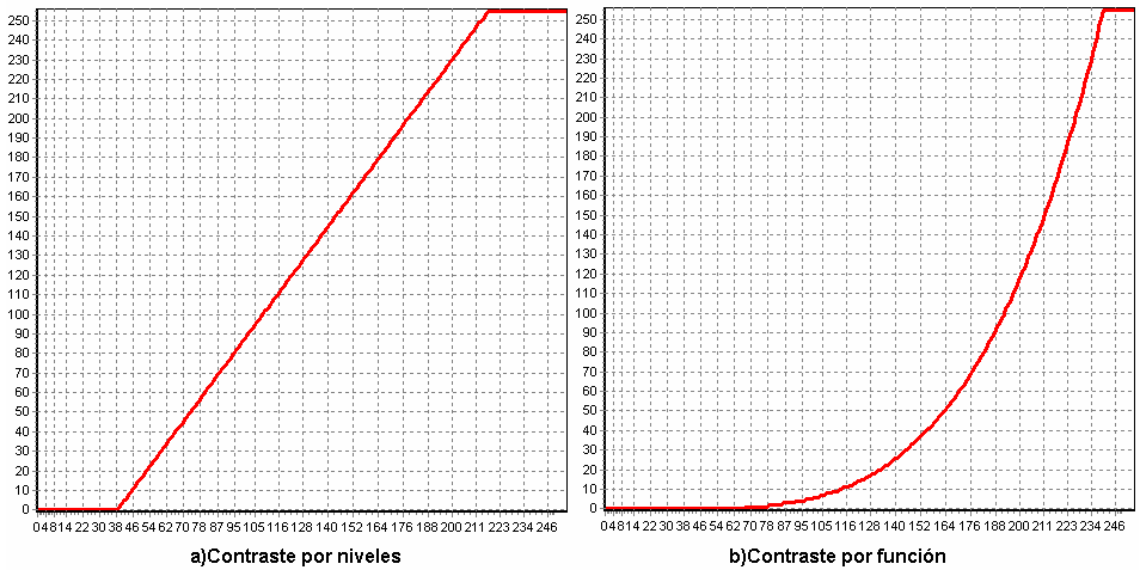


Figura 3.10 Funciones para el cambio de contraste

Dando mejores resultados la función mostrada en la figura 3.10 a.

3.3.2.2.4 Conversión a escala de grises

En el formato de color RGB, las imágenes a color se almacenan en 3 canales independientes (rojo, verde y azul) que toman valores de 0 a 255 dependiendo de la intensidad [Gonzáles et. al.].

El formato de color YIQ representa una división entre la luminosidad (Y) y el color (I, Q). La relación entre RGB e YIQ está dada por la ecuación 3.1.

$$\begin{pmatrix} Y \\ I \\ Q \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \tag{3.1}$$

Tomando solo la componente Y de la imagen, se obtiene una imagen en escala de grises. La forma de obtener una imagen en escala de grises a partir de una imagen en RGB es aplicando a cada valor RGB la ecuación 3.2.

$$Y = 0.299*R + 0.587*G + 0.114*B. \tag{3.2}$$

3.3.2.2.5 Resta de imágenes

Una vez que se tienen dos imágenes subsecuentes y su correspondiente preprocesamiento, se procede a realizar la resta, cuya operación se lleva acabo obteniendo el valor absoluto de la diferencia de las dos imágenes píxel a píxel [Fisher et. al.] Como lo muestra la ecuación 3.3

$$PixelSalida[i,j]=| PixelImag1[i,j]- PixelImag2[i,j] | \tag{3.3}$$

3.3.2.2.6 Inversión o negativo

Este efecto consiste en obtener el negativo de una imagen. Para conseguirlo solo es necesario aplicar la ecuación 3.4 sobre cada píxel de la imagen en escala de grises.

$$PixelSalida =255- PixelEntrada \tag{3.4}$$

Entonces, el negativo de la imagen se obtiene restando el valor de 255 al valor del píxel, en ese mismo punto [González et. al.]

3.3.2.2.7 Binarización ó umbralización

Este proceso convierte una imagen en tonos de gris en una imagen binaria (blanco y negro), asignando el valor 255 (blanco) a los valores de píxel por encima de un cierto umbral, el cual es un parámetro, o 0 si está por debajo de éste umbral. La operación a realizar es la siguiente:

Si (PixelEntrada >umbral) *entonces* PixelSalida =255
Si no PixelSalida =0

Para el cálculo del umbral se empleó la binarización media la cual consiste en obtener el histograma de la imagen para posteriormente calcular el valor medio; y este valor es el que se emplea como umbral.

3.3.2.2.8 Segmentación

Proceso en que se divide una escena en partes, para extraer objetos para su posterior reconocimiento y análisis.

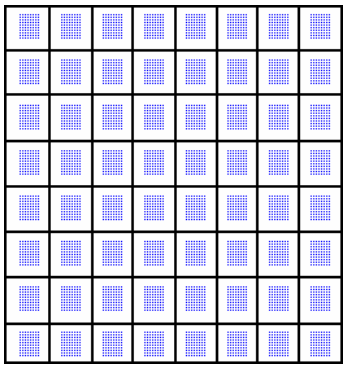


Figura 3.11 Matriz para la segmentación

La imagen es dividida en una matriz de 8X8 como se muestra en la figura 3.11. Se desarrolla este tipo de segmentación para disminuir problemas de confusión de piezas por el efecto de la sombra. Estas rejillas pueden ser modificadas según se necesite. Normalmente a este tipo de segmentación se le llama segmentación general.

3.3.2.2.9 Representación y descripción

En este proceso se cuenta el número de píxeles negros existentes en la imagen, estos píxeles son únicamente los que se encuentran en la máscara que se creó en la etapa de la segmentación (figura 3.11), una vez calculado el número de píxeles

existentes en una rejilla, se compara con un nivel para determinar si ha habido movimiento o no; de haber pasado este umbral se actualiza una estructura que contiene un campo para el número de movimientos encontrados, un arreglo que contiene la posición vertical y otro arreglo con la posición horizontal de la pieza que se halla detectado.

Al finalizar el análisis de la imagen se compara el valor del contador de número de movimientos y de haber detectado más de 4, se retorna un valor de 0 identificado como error; de lo contrario se retorna la estructura con los datos de cuantos movimientos se encontraron y sus correspondientes posiciones. De no encontrar movimientos válidos se notificará a los jugadores, quienes tendrán que regresar a la posición de partida, presionar el botón para la toma de la imagen, realizar el movimiento de la ficha a su posición final y nuevamente presionar el botón.

3.3.2.2.10 Salida del módulo de imagen

El módulo de procesamiento de imagen retorna una estructura con el número de movimientos encontrados y la colocación de estos. Con esta información se va a generar la simulación tanto en el servidor como en el cliente de la pieza movida; así como también se agrega dicho movimiento al historial de la partida, utilizando la notación algebraica.

Para el procesamiento de las imágenes se implementó la clase imagen, para la cual se implementaron las funciones mostradas en la figura 3.12.

Esta clase permite acceder a los atributos de una imagen y encapsular las operaciones realizadas sobre un mapa de bits (bitmap). Estas operaciones pueden ser simples, como acceder al nivel de gris de cada píxel, o bien complejas, como el proceso de umbralización, pasando por la aplicación de filtros o corrección del efecto de distorsión radial.

El atributo principal de esta clase es el siguiente:

- **BMP**. Representa un puntero que indica el comienzo de los píxeles de la imagen. Utilizando este atributo se puede obtener valor de cualquier píxel o actualizar cualquier píxel con cierto nivel de gris.

Imagen
Datos miembro

Graphics::TBitmap *BMP;
void FiltroVertical();
void FiltroHorizontal();

Funciones miembro

Imagen();//Constructor
~Imagen();//Destructor
Imagen(const Imagen &der);//Constructor de copia
Graphics::TBitmap * Bitmap();
//Sobre carga de operadores
const Imagen &operator =(const Imagen &der);
const Imagen &operator =(Graphics::TBitmap *Bmp);
const Imagen operator -(const Imagen &der);
//Funciones miembro
void CargarDeClipboard(Word AFormat, unsigned AData,
HPALETTE APalette);
void BinarizacionUmbral(const TColor umbral);
void BinarizacionMedia();
void Brillo(const int Value);
void Contraste(const int Value);
void Invertir();
// Operaciones Morfológicas básicas
void ErosionGray(const int N=1);
void DilationGray(const int NN=1);
//Transformación entre modelos de color
void EscalaDeGrises (const bool F32bits=false);
//Filtros
void EqualizacionConHistograma();
void RangoDinamico();//Filtro Logarítmico
void FiltroSuavizante();
void FiltroPromedio();
void FiltroMedia();
void Gaussian();
void FiltroSuavizado
//Operaciones
info Descripcion(const int H,const int W);
//Correctores de distorsión
void DistorsionPincushion(const double alphax,const double alphay,
const double antialiasing,const int nyout,const int nxout);
void DistorsionBarrel(const double alphax,const double alphay,
const double antialiasing,const int nyout,const int nxout);

Figura 3.12 Diagrama de la clase Imagen

En cuanto a las funciones miembro, se destacan las siguientes:

- **CargarDeClipboard** se encarga de asignar la imagen que se encuentra en memoria (el portapapeles) a la variable Imagen creada con anterioridad.
- **FiltroSuavizante** Aplica el filtro suavizante a la imagen contenida en el mapa de bits(BMP)

- **Contraste, Brillo, BinarizacionMedia, Invertir** desempeñan las funciones ya explicadas en este documento, realizando transformaciones a la imagen.
- **EscalaDeGrises** Realiza la conversión de la imagen a color a escala de grises.
- **DistorsionPincushion** y **DistorsionBarrel** funciones encargadas de corregir el efecto de distorsión provocada por la curvatura de la lente de la cámara.
- **Descripcion** mediante esta función se obtienen las posiciones X y Y de las piezas obtenidas después de realizar la resta de las dos imágenes, esta función regresa una estructura la cual contiene el número de movimientos encontrados y las correspondientes posiciones.

En cuanto a las operaciones implementadas, se sobrecarga al operador de resta, para poder realizar la resta entre dos variables tipo Imagen, regresando como resultado una tercera la cual puede ser asignada a otra variable tipo Imagen por medio del operador de asignación (=), el cual se sobrecargo para permitir la asignación entre dos variables del tipo imagen y aceptar imágenes del tipo bitmap.

También se encuentra una descripción completa de los atributos y funciones en el apéndice C.

3.3.3 Reconocimiento

En esta etapa se identifica a que pieza corresponde el movimiento encontrado y si es válido. Esto se logra con apoyo de una matriz de 8X8, la cual originalmente contiene la colocación inicial de las piezas del juego de ajedrez (figura 3.13).

Una vez que se tiene la estructura con el número de movimientos y el arreglo con sus posiciones, se procede a verificar que pieza se encuentra en la posición donde hubo algún cambio.

T	C	A	D	R	A	C	T
P	P	P	P	P	P	P	P
p	p	p	p	p	p	p	p
t	c	a	d	r	a	c	t

Figura 3.13 Representación de las posiciones iniciales en el tablero

Para distinguir las piezas de cada uno de los jugadores, se denota con minúsculas a las piezas blancas y con mayúsculas a las piezas negras Para mayor información consultar el apéndice A, el cual contiene un resumen de la notación y reglas del ajedrez.

Dado el número de cambios entre una imagen y la subsecuente, los movimientos de las piezas se clasifican de la siguiente forma:

- 2 cambios pueden representar el movimiento normal de una pieza, la captura de una pieza, o la promoción de un peón.
- 3 cambios pueden representar la captura al paso de un peón.
- 4 cambios el enroque corto o enroque largo.

3.3.4 Validación y simulación

Una vez identificado el posible movimiento se procede a identificar a que pieza corresponde, se valida dicho movimiento y si éste cumple las reglas del ajedrez se realiza la simulación del movimiento, se agrega al historial y se envía por medio de sockets a los clientes que estén conectados. En el apéndice A se explican las reglas básicas del movimiento correspondiente a cada pieza del ajedrez, las cuales fueron validadas mediante software.

La simulación es realizada por el programa con la representación gráfica de las piezas y el tablero, esto se realiza con la finalidad de poder difundir la partida en forma local por medio de proyectores o simplemente en el monitor de la pc. Esta simulación sirve también para comprobar si los movimientos realizados en el tablero corresponden con los detectados por el sistema.

Al realizar el movimiento correspondiente, el sistema lo narrará empleando para esto el motor de voz TtS(Text to Speech) de Microsoft, el cual transforma texto en sonido(voz). Un ejemplo de esto se muestra en la figura 3.14.

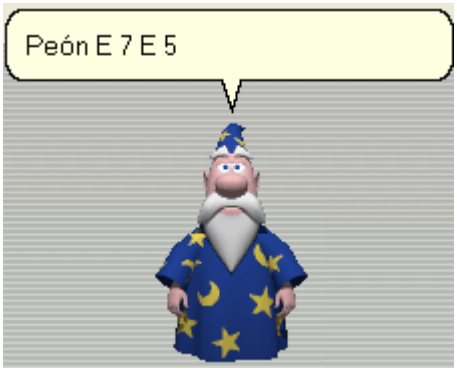


Figura 3.14 Agente de Microsoft.

3.3.5 Difusión

La página Web del sistema WebChess proporciona acceso a las partidas en vivo y al historial de partidas; esta página se puede acceder desde la dirección http://webchess.dyndns.org, la cual está montada sobre un servidor web ubicado dentro

del programa principal del servidor, dicha implementación se creó usando el componente IdHTTPServer; este componente da respuesta a los comandos get de http, cabe resaltar que estas peticiones implementan el protocolo UDP, por lo que al responder con el documento solicitado la conexión se cierra y ya no existe conexión Cliente-Servidor.

Otro punto importante es que el IdHTTPServer implementa hilos por lo que cada que un usuario se conecta al servidor se genera un nuevo proceso que atiende las peticiones; con lo cual se puede atender a varios usuarios aunque estos se conecten al mismo tiempo.

3.3.5.1 Partidas en vivo

Una vez que el cliente hace la petición de la página de partidas en vivo, la aplicación del cliente realiza una conexión por medio de sockets en un puerto entre el 1024 y el 65000 según se halla especificado; esta conexión es atendida por un servidor de sockets que se encuentra en el programa del servidor, es en este punto donde se implementó un contador de visitas y de usuarios conectados para tener una referencia del número de visitas a la página. Cada que un usuario se conecta a las partidas en vivo se incrementa un contador de usuarios en línea y un contador de usuarios que han visitado la página. Y cada que un usuario deja la página se decrementa el contador de usuarios en línea o conectados.

Ya que se ha establecido la conexión entre el cliente y el servidor, se procede a enviar las jugadas presentes en el historial; si no hubiese jugadas en el historial se comienza a enviar las jugadas detectadas por el sistema, sólo se manda la notación algebraica del último movimiento detectado; por ejemplo, si el último movimiento fue el de el avance del peón del rey blanco dos cuadros hacia delante la notación sería E2-E4.

Con estos datos es posible saber perfectamente de que cuadro a que cuadro se movió la ficha y generar dicho movimiento del lado del cliente.

Para que una cadena pueda enviarse se debe convertir al protocolo usado por el objeto XMLSocket [Macromedia1] el cual tiene las siguientes características:

- Los mensajes son enviados por una conexión full-duplex TCP/IP
- Cada mensaje es un documento XML completo. Terminado por un byte cero.
- El número de mensajes enviados y recibidos es ilimitado sobre una sola conexión.

3.3.5.2 Historial de partidas

Además de las partidas en vivo, el servidor contiene un historial de partidas previamente desarrolladas para que puedan ser visualizadas por los clientes.

Cada que se hace una petición a la página del historial de partidas, se envía un documento XML el cual contiene una lista del historial de partidas.

El uso principal de XML es estructurar datos, recibirlos y/o enviarlos, pero también es posible guardar datos en documentos locales para que posteriormente sean tratados con X lenguaje. XML se convierte en una muy buena posibilidad ya que los datos que se envían están estructurados como se muestra continuación.

```
<?xml version="1.0" encoding="windows-874" ?>
<Historial>
        <OPCION> 26/10/2004Hr10min42
                <JUGADA>E2 E4</JUGADA>
                <JUGADA>F7 F5</JUGADA>
                <JUGADA>E4 X F5</JUGADA>
        </OPCION>
</Historial >
```

Este archivo o documento contendrá como máximo 10 partidas, esto para evitar que su transferencia se haga lenta.

3.4 Integración

Una vez finalizada la implementación y prueba de los módulos, éstos se integraron en el programa principal; cuyo comportamiento se ilustra en el diagrama de flujo de la figura 3.15; Puede observarse que el módulo del puerto paralelo es el que da inicio al proceso; al inicio del juego se deberá presionar el botón para que se capture la imagen con la disposición inicial de las piezas en el tablero y comience a correr el tiempo el primer jugador. Al siguiente evento producido se respalda la imagen anterior, la cual ha sido procesada previamente; y se toma una segunda imagen, se procesa y se realiza la resta de estas dos imágenes, lo cual genera otra imagen que será procesada para obtener los arreglos de posiciones de las piezas detectadas. Teniendo los arreglos de las posiciones se procede a identificar a que piezas corresponden dichos movimientos y si estos son válidos. De ser válidos se procede con la simulación y difusión por medio de Internet, a los clientes que se encuentren conectados. De haber encontrado errores el programa se regresará al inicio, lo que implica retornar las piezas a la posición original antes de realizar el último movimiento.

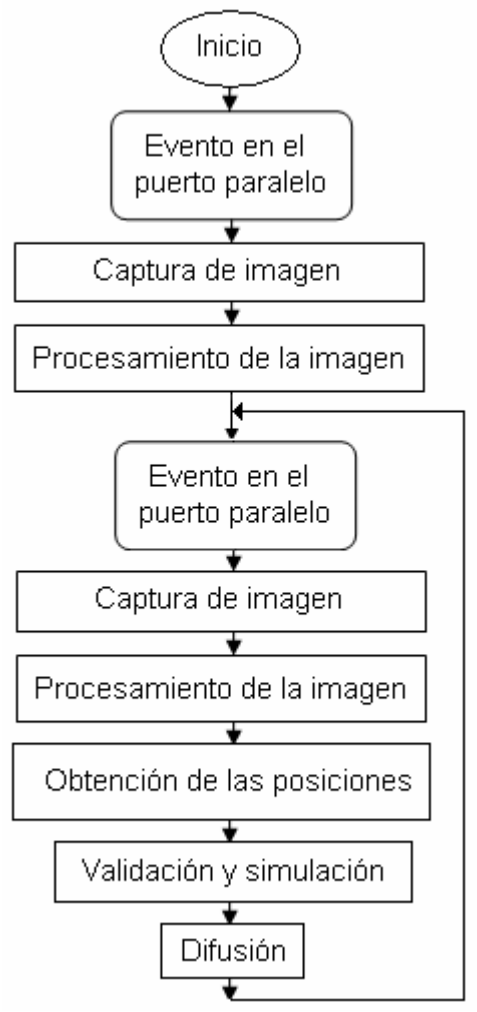


Figura 3.15 Diagrama de interacción entre módulos.

3.5 Pruebas del software

La prueba es un elemento crítico para la calidad del software. Las pruebas permiten validar y verificar el software, entendiendo como validación del software el proceso que determina si el software satisface los requisitos, y verificación como el proceso que determina si los módulos de una fase satisfacen las condiciones de dicha fase.

Las pruebas implementadas en el desarrollo del sistema fueron de caja negra, cuyo objetivo es encontrar errores de las siguientes categorías:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en la estructura de datos.

- Errores de rendimiento.
- Errores de inicialización y de terminación.

A continuación se describen las pruebas que se realizaron en el desarrollo del sistema WebChess.

Pruebas al sub-módulo lectura del puerto paralelo:

- Detección de posibles errores al inicializar y finalizar el puerto.

- Evitar que el hilo consumiera mucho tiempo de CPU por lo que se agregó un retardo en el tiempo de chequeo del estado del puerto.

Pruebas al sub-módulo imagen:

- Detección de posibles errores al inicializar la cámara.

- Se realizaron pruebas con diferentes tipos de filtros dejando el que arrojara mejores resultados.

Pruebas al sub-módulo reconocimiento de las piezas:

- Validar que en las coordenadas regresadas por el sub-módulo imagen tuvieran una pieza en la matriz de juego (figura 3.13) y así determinar la posición inicial y final correspondiente.

Pruebas al sub-módulo validación y simulación:

- Se probaron todos los posibles errores en el movimiento de las piezas, con lo que el programa manda el mensaje de “Movimiento inválido” por medio del agente de Microsoft.
- Se validaron los movimientos especiales como son el enroque corto, largo y la captura al paso.
- Se verificó la correcta simulación de cada uno de los movimientos, así como la correcta sustitución de la pieza en el caso de promover un peón por una reina.

Pruebas al sub-módulo difusión:

- Que los clientes al momento de estar viendo una partida, pudieran navegar en el historial de movimientos.
- Se verificó la correcta simulación de cada uno de los movimientos así como la correcta sustitución de la pieza en el caso de promover un peón por una reina.

Pruebas del sistema completo:

- Se puso en funcionamiento el sistema completo; contando con cinco clientes para las partidas en vivo cuyos resultados se muestran en capítulo cuatro, el máximo número de usuarios que se probaron fue de 32.

- Se verificó que al momento de transmitir una partida en vivo se atendieran las peticiones del historial de partidas, logrando un resultado satisfactorio.

- Se agregó la opción de realizar manualmente el movimiento de las piezas, esto para que si por algún factor externo, el sistema no fuera capaz de detectar el movimiento realizado.

Capítulo 4 Resultados

En este capítulo se muestran y analizan los resultados obtenidos de la implementación del sistema WebChess. También se realiza una comparación de la eficiencia del sistema en cuanto al tráfico en la red, y se muestran los resultados obtenidos en la transmisión de una partida real.

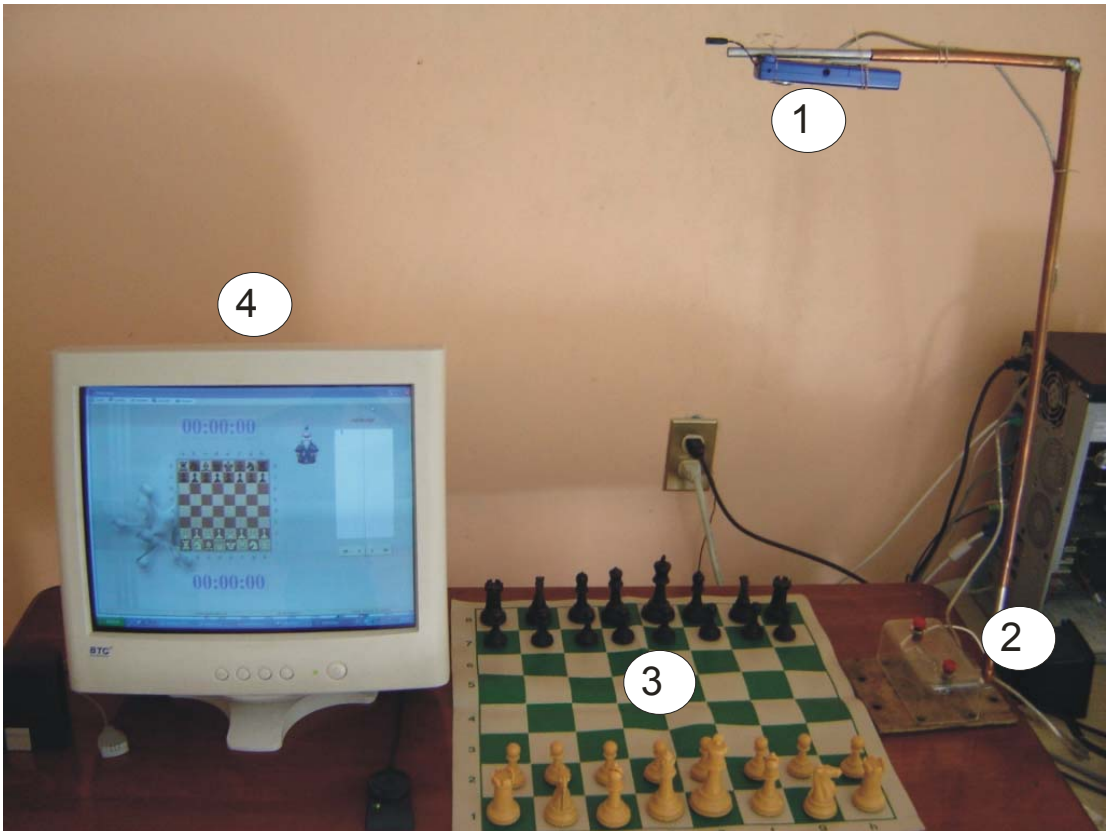


Figura 4.1 Vista de la implementación del sistema

En la figura 4.1 se puede observar el módulo servidor del sistema, donde se encuentra la cámara (1), el botón para el cambio de turno (2), el tablero (3), y el programa que realizará el procesamiento y difusión (4).

4.1 Lectura de datos

Se logró implementar la lectura y escritura del puerto paralelo, empleando la librería de enlace dinámico input32 con lo que se da lectura a los botones mostrados en la figura 4.2 y se activan los leds mostrados en la misma figura, con los que se indica el turno correspondiente a cada jugador.

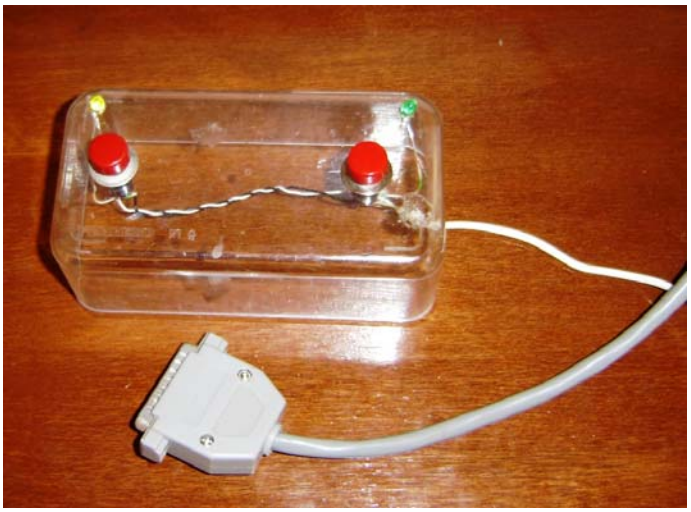


Figura 4.2 Vista del botón con el conector DB25

Al instanciar la clase TParallelThread, se crea un hilo que trabaja en forma simultánea a otros procesos dentro del ambiente multitareas del sistema operativo Windows XP. Con lo cual, el monitoreo del puerto paralelo es continuo y sin descuidar otros procesos.

Así mismo, con la lectura del puerto paralelo se pudo realizar un disparador de eventos para la toma de la imagen y el control de tiempo de cada jugador.

En la figura 4.3 se muestra una vista superior de la colocación de cámara sobre el tablero, desde donde serán tomadas las imágenes que darán la posición de los movimientos realizados.

Para la cámara cool@cam y el tablero estándar de la FIDE usados en el desarrollo de este trabajo, la cámara se situó a una distancia de 111 cm desde el centro del tablero, a modo de que las imágenes capturadas cubran completamente sus dimensiones. Es posible usar una cámara o tablero diferentes, sólo habrá que determinar la distancia adecuada.

Figura 4.3 Vista superior de la colocación de la cámara

4.2 Procesamiento de las imágenes

A continuación, se muestran los resultados obtenidos del procesamiento de las imágenes.

Como primer paso se capturan dos imágenes que forman la secuencia del movimiento de una pieza; posteriormente se filtra la imagen con su media aritmética. El resultado de aplicar este filtro a una imagen se muestra en la figura 4.4; para este paso se implementaron y probaron los filtros: filtro logarítmico, filtro suavizante, filtro media, gaussiano, filtro suavizado (para imágenes a color); obteniendo mejores resultados con el filtro suavizante. El código de dichos filtros se encuentra en el CD-ROM.

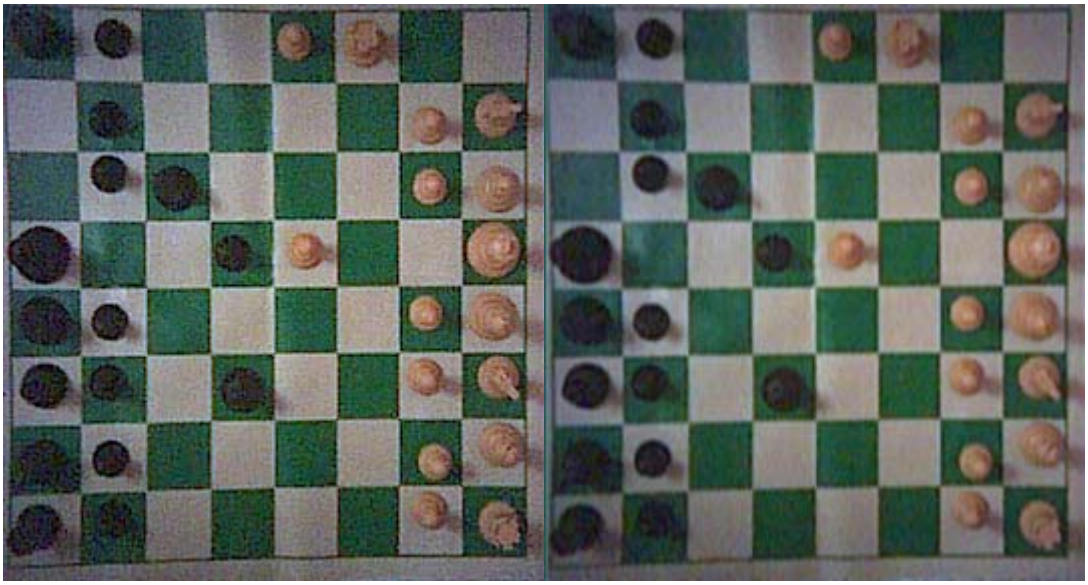


Figura 4.4 Imagen filtrada 2 veces

Para mejorar la imagen se aumentó el nivel de brillo, experimentalmente se buscó el nivel adecuado para dicho aumento, encontrando resultados adecuados empleando un nivel de 50; sin embargo, este nivel puede ser modificado directamente en el programa. La aplicación de dicho proceso genera la imagen que se muestra en la figura 4.5.

Figura 4.5 Imagen con brillo aumentado.

A la imagen mostrada en la figura 4.5 se le aplicó el algoritmo de cambio de contraste, para resaltar las partes más oscuras de las más claras de la imagen, el resultado se muestra en la figura 4.6.

Figura 4.6 Imagen con contraste modificado.

Después de haber aplicado el algoritmo de contraste, la imagen es convertida a escala de grises, esto para llevar a cabo la resta entre las dos imágenes. Este proceso se realiza únicamente para la última imagen capturada ya que la anterior ya ha pasado por este proceso y se encuentra guardada en memoria.

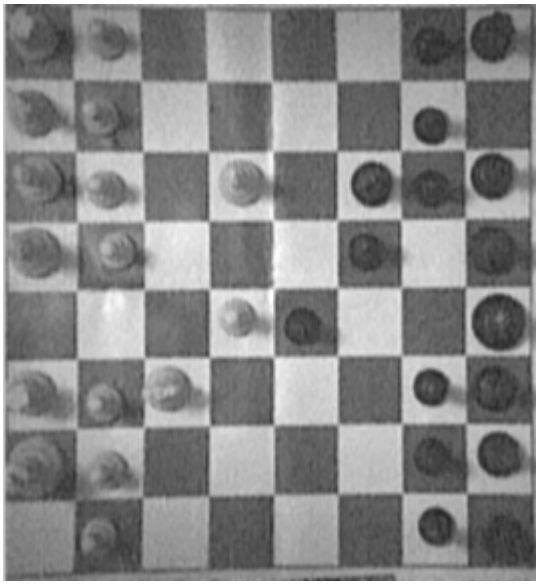


Figura 4.7 Imagen en escala de grises.

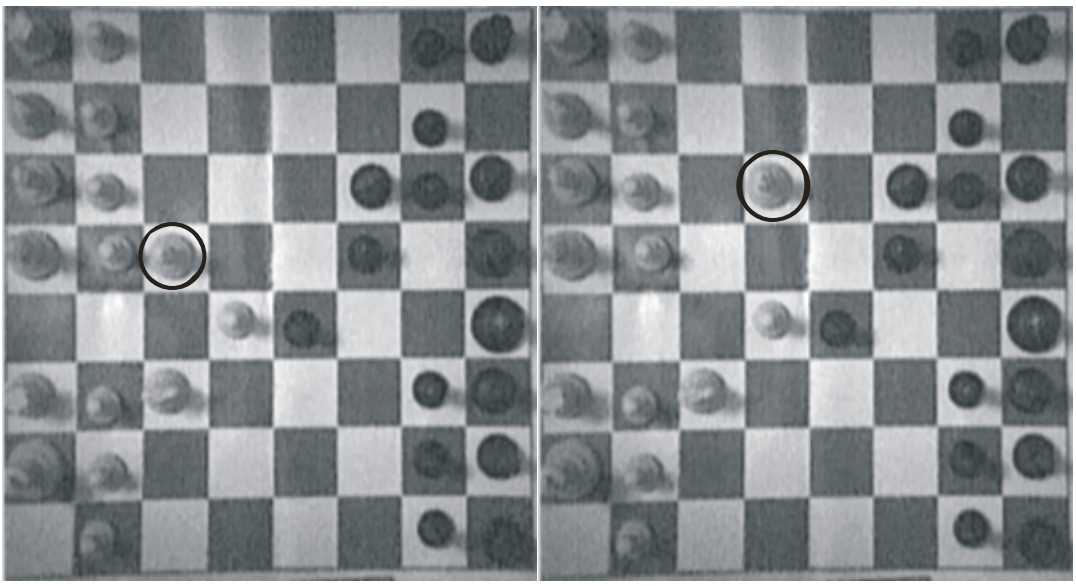


Figura 4.8 Secuencia de movimiento de una pieza.

Una vez que se tienen las dos imágenes con su respectivo preprocesamiento se procede a realizar la resta, esto se realiza píxel a píxel.

En la figura 4.8 se muestran dos imágenes consecutivas, donde se aprecia el movimiento de una pieza. El resultado de la resta de ambas imágenes se muestra en la figura 4.9a; y en la figura 4.9b se muestra el inverso de la diferencia, para visualizar mejor el movimiento de las piezas.

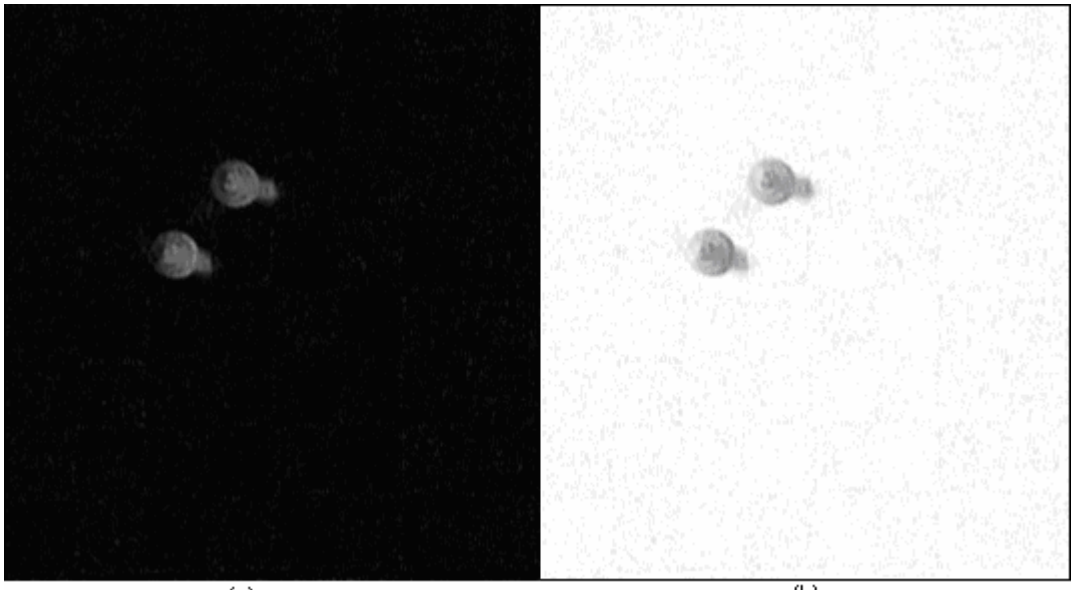


Figura 4.9 Resta de las imágenes a) y su inverso b)

Debido a que los valores de los píxeles están todavía en el rango de 0 a 255, es necesario aplicar el algoritmo de umbralización para obtener una imagen binarizada, con este algoritmo termina el preprocesamiento, sus resultados se pueden apreciar en la figura 4.10. El valor del umbral usado en las imágenes procesadas fue obtenido del

valor medio de su histograma. Ya que la aplicación de otros valores para el umbral arrojaba imágenes con mucha basura.

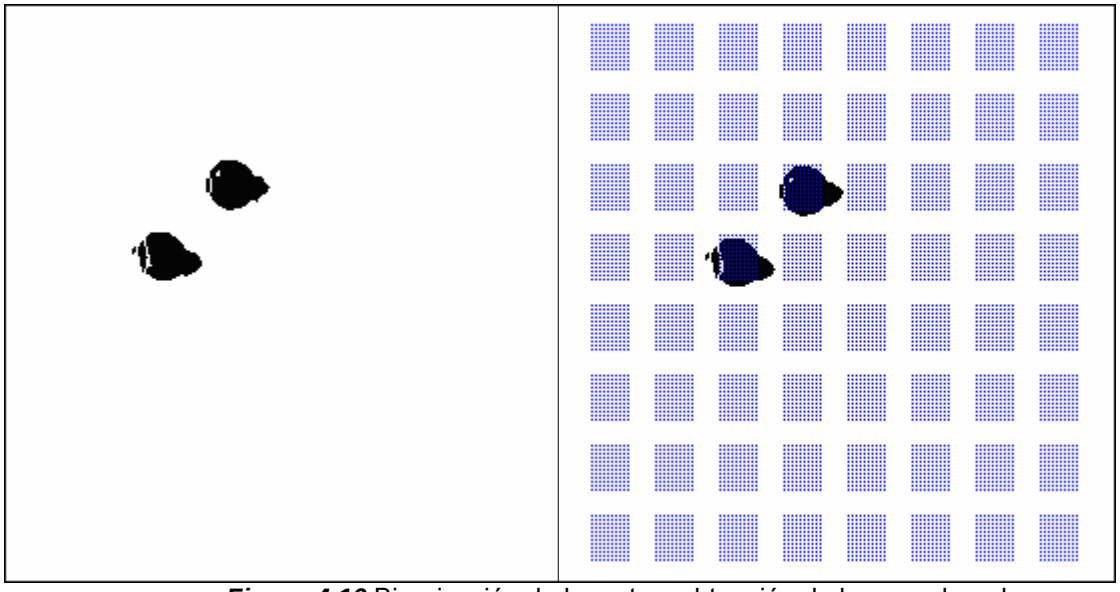


Figura 4.10 Binarización de la resta y obtención de las coordenadas

4.3 Corrección radial

En el procesamiento de la imagen se añadió la corrección radial obteniendo como resultado la reducción del efecto causado por las lentes. En la figura 4.11 se muestra el resultado de aplicar el algoritmo para dicha corrección.

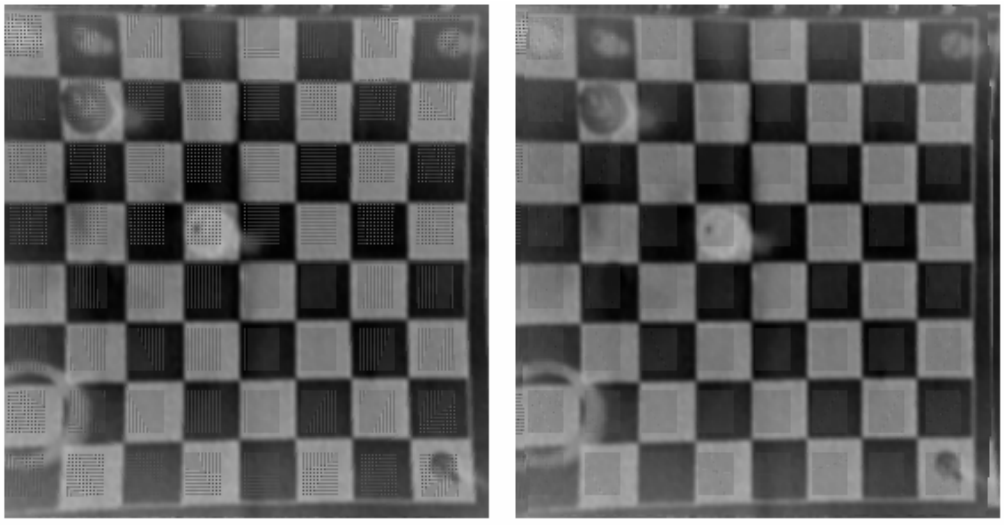


Figura 4.11 Corrección radial

4.4 Tiempo de procesamiento

El ambiente de desarrollo usado permite tratar a las imágenes acensándolas como: Tipo Imagen,Tipo Bitmap y Scanline.

- **Imagen** se emplea para desplegar una imagen. Usa el objeto Tpicture en la propiedad Picture para especificar el mapa de bits, por lo cual su acceso se hace lento.
- **Bitmap** es un objeto gráfico usado para crear, manipular y poner imágenes en memoria o en archivos. Es un encapsulado de una matriz de píxeles que definen la imagen esto hace más rápido el acceso a cada píxel.
- **Scanline** Provee un apuntador para el acceso a cada línea de píxeles, con lo cual se leen línea por línea los píxeles y evita el cálculo de salto que se realiza en una matriz.

La implementación de los algoritmos para el procesamiento digital de las imágenes se realizó haciendo uso del acceso Scanline debido a que mejora notablemente el tiempo de procesamiento. Esto se puede ver en la tabla 4.1 donde se muestra una comparación entre los diferentes métodos de acceso a imágenes.

Tabla 4.1 Comparación del procesamiento

Método de acceso	Kpíxeles/s procesados	Tiempo de procesamiento
Imagen	**202.75**	**0.5 seg.**
Bitmap	**405.504**	**0.25 seg.**
Scanline	**48000**	**0.016 seg.**

Con dicha implementación se obtuvo un tiempo de procesamiento para la obtención de las posiciones de las piezas de 0.094 segundos, lo cual es satisfactorio si se toma en cuenta que el tiempo mínimo que se tarda en tirar un jugador es de 2 seg. en partidas rápidas.

En el caso de no encontrar cambios o sólo encontrar 1 entre 2 imágenes consecutivas se intenta aumentar el resultado de la resta haciendo la dilatación de la imagen en cuyo caso el tiempo total es de 0.25 seg. Esto ocurrirá sólo si la iluminación es muy baja o deficiente.

Por otro lado, si se encuentran más de cuatro cambios se intenta corregir con erosión, este caso se presenta algunas veces cuando la sombra de la pieza es muy grande y abarca otro cuadro del tablero lo cual sobrepasa el nivel de comparación y se identifica como una pieza. El tiempo en hacer dicho proceso es de 0.438 segundos. Esto ocurrirá si la fuente de luz no está por encima del centro del tablero, si no en uno de sus lados.

En la sección 3.3 se describió la cantidad de cambios requeridos por los movimientos válidos.

Una fuente de errores para el sistema es la iluminación, ésta debe ser lo más uniforme posible. También pueden presentarse errores en el reconocimiento si el tablero o la cámara se mueven, estos elementos del sistema deben permanecer fijos desde el comienzo, hasta el final de la partida. Ya que de tener movimientos, la resta presentará datos incorrectos.

4.5 Comparación del tráfico en la red WebChess &Video

A continuación se muestran la imágenes de tráfico en la red para el envío de video y el envío de los paquetes que realiza el sistema WebChess; como puede observarse en la figura 4.12, el flujo de datos para el envío de video es constante, mientras que en la figura 4.13 el flujo de información es menor y solo durante periodos cortos; cabe resaltar que el envió de video se realizó solo con una computadora como cliente mientras que el envío de paquetes se realizó con cinco clientes conectados al sistema.

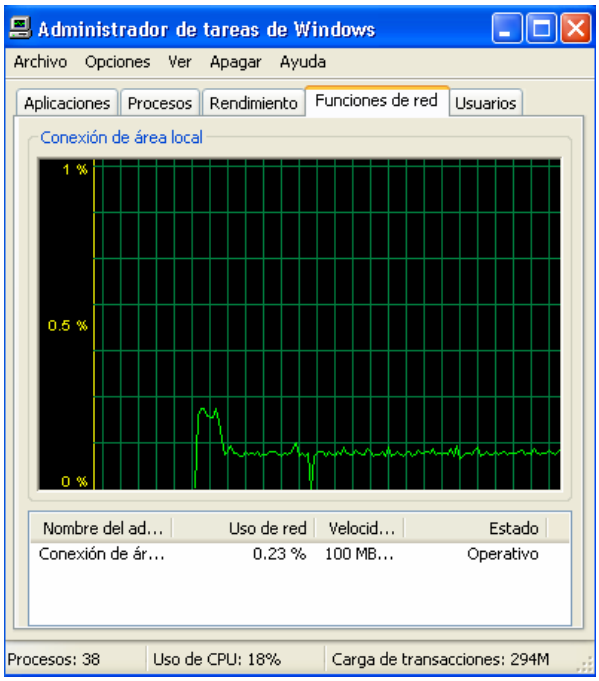


Figura 4.12 Monitoreo del tráfico en el envío de video

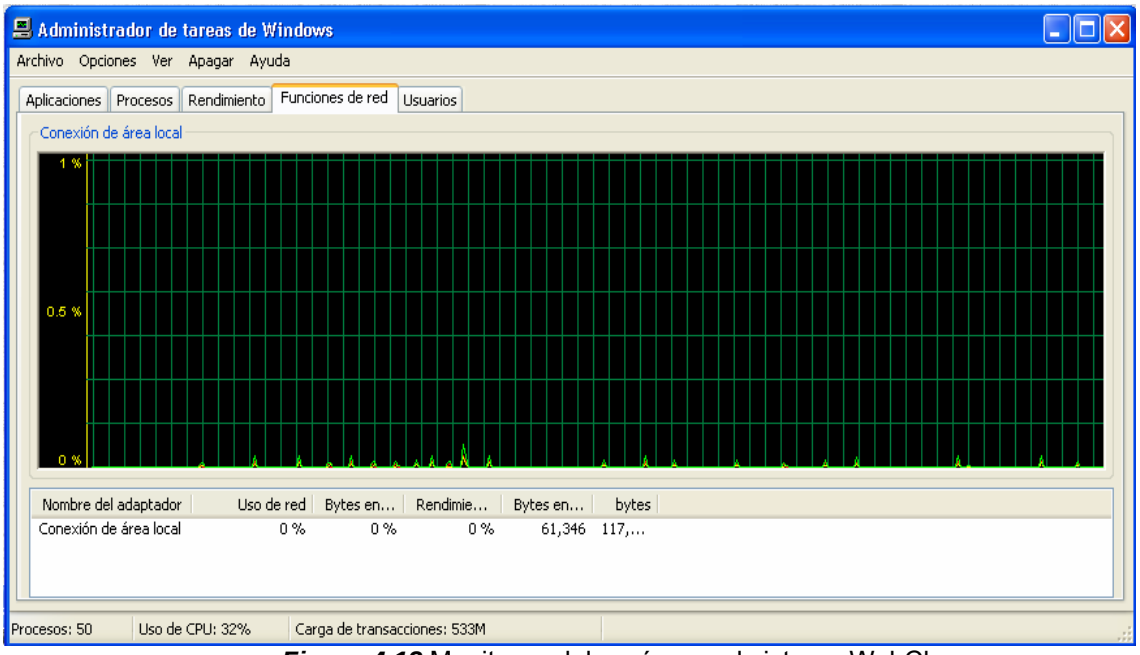


Figura 4.13 Monitoreo del envío con el sistema WebChess

4.6 Interfaz del servidor

En la figura 4.14 se presenta la ventana principal del sistema desarrollado, en la cual se observan los siguientes elementos:

- Un historial que muestra las jugadas en notación algebraica realizadas por los jugadores.
- Agente de Microsoft el cual se encarga de reproducir en forma de voz las jugadas realizadas o posibles errores.
- El tiempo de juego de cada jugador.
- El tiempo total de la partida.
- El número de usuarios conectados a la transmisión de partidas en vivo.
- El número de usuarios que han visitado la página de la transmisión de partidas en vivo.
- Estado de la cámara o posibles errores en el sistema.

Figura 4.14 Ventana principal del sistema WebChess

En la figura 4.15 se muestra la ventana de configuración de los parámetros para el procesamiento de la imagen. Esta ventana cuenta con los siguientes parámetros:

- **Brillo**.- Se emplea para incrementar o disminuir el nivel de brillo aplicado a la imagen.
- **Contraste**.- Se emplea para incrementar o disminuir el nivel de contraste aplicado a la imagen.
- **Tamaño de rejilla**. Determina el tamaño de las rejillas las cuales se emplean para la segmentación.
- **Nivel de comparación**. Establece el nivel de umbral de píxeles en negro dentro de las rejillas para identificar un objeto como pieza.
- **Distorsión**.- Permite activar o desactivar la corrección de la distorsión barrel o pincushion; una vez seleccionada una de estas dos se activan los parámetros de la curvatura de la lente.
- **Guardar imagen**.- Da la opción de tomar una imagen actual de la cámara y guardarla a disco.
- **Captura de la imagen**.- Toma una imagen y realiza el procesamiento de esta para obtener la resta con una segunda imagen. Esto es empleado para poder calibrar de una manera más eficiente los parámetros de brillo y contraste, ya que permite ver la imagen obtenida al realizar la resta y el número de movimientos detectados.
- **Cerrar**.- Cierra la ventana actual y regresa a la ventana principal.

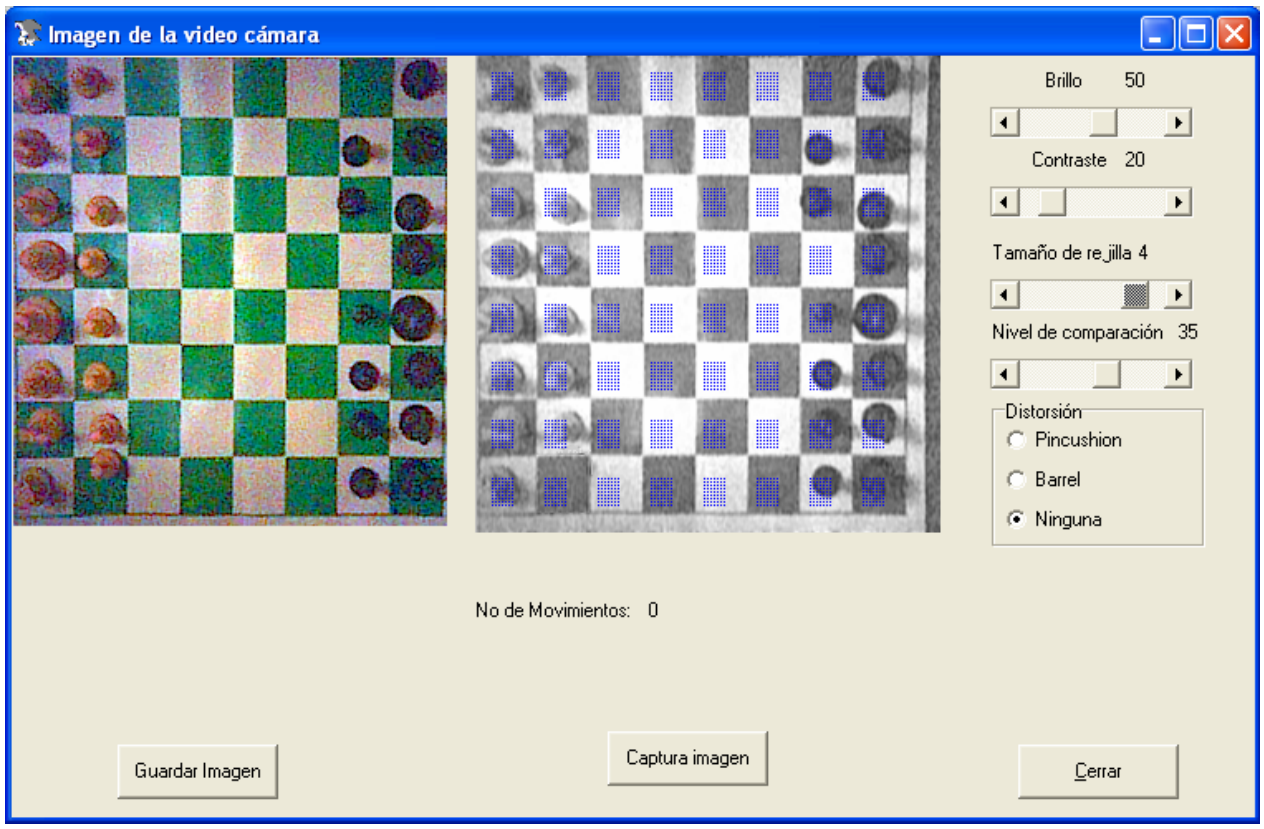


Figura 4.15 Ventana de ajustes de la imagen

En la figura 4.16 se muestra la ventana para la configuración del servidor; básicamente se configuran dos parámetros: la dirección URL donde se encuentra el servidor y el número de puerto por el que se realizará la conexión; este número deberá estar entre el rango de 1024 y 65500 que son puertos libres. Al configurar estos parámetros se modificarán automáticamente en las aplicaciones en flash y las páginas de Internet, por lo que el usuario podrá establecer su servidor en cualquier pc.

Figura 4.16 Ventana para la configuración del servidor

Otro de los servicios manejados por el programa es el historial de partidas. Para la edición de este historial se agregó una ventana al programa principal en la cual el administrador puede agregar o eliminar las partidas que desee mostrar en el historial.

Esta ventana es mostrada en la figura 4.17 la cual muestra las partidas y sus correspondientes jugadas en forma de árboles.

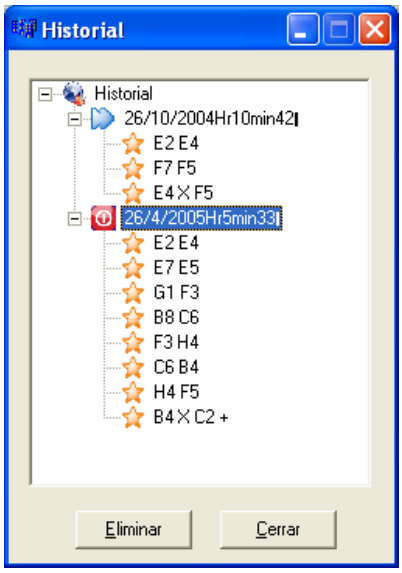


Figura 4.17 Editor de historial

4.7 Interfaz del cliente

Para el lado del cliente se realizó una animación con Macromedia Flash la cual se muestra en la figura 4.18. Por medio de ella, los usuarios conectados al sistema WebChess pueden ver la partida en vivo y analizar cada uno de los movimientos desde su inicio hasta su fin. Haciendo uso de esta misma interfaz, se realizó un historial de partidas, el cual cuenta con partidas previamente jugadas. Para poder acceder a este historial se realizó otra interfaz que presenta al usuario un listado de las partidas, pudiendo seleccionar una para su posterior análisis, dicha interfaz es presentada en la figura 4.19.

Las partidas pueden ser exploradas con los botones de avanzar, retroceder, inicio y fin. Esta interfaz cuenta también con el agente de Microsoft (Merlín) que narrará los movimientos realizados por los jugadores.

Figura 4.18 Vista de la aplicación en el cliente

En la figura 4.19 se muestra la ventana del historial en la cual se presenta un listado de las partidas que ya han finalizado. Una vez que se selecciona una partida y se da click en el botón siguiente, se verá la partida en una ventana igual a la de la figura 4.18

Figura 4.19 Ventana del historial

Capítulo 5 Conclusiones y trabajos futuros

Tras el desarrollo de esta tesis se cumplió con el objetivo de realizar un sistema capaz de difundir partidas de ajedrez por Internet. Empleando como fuente de información una cámara digital y el puerto paralelo de una computadora; como medio de difusión se empleó la Internet mediante el envío de pequeños paquetes generando poco tráfico en la red.

En este contexto se resalta la característica de explotar los recursos (hardware) de una computadora por medio de herramientas de desarrollo de software para dar soluciones a problemas de la vida diaria.

5.1 Características principales del sistema WebChess

La característica principal de este proyecto es la extracción de información de un evento, como lo es la posición de las piezas movidas en un tablero de ajedrez; es por ello que este trabajo pretende ejemplificar la funcionalidad de explotar los recursos de una computadora para facilitar la interpretación y difusión de eventos.

El desarrollo del presente trabajo requirió del estudio de diferentes áreas, como el procesamiento digital de imágenes, protocolos de comunicación (TCP/IP), sistemas operativos y el hardware de una computadora. Pueden mencionarse las siguientes aportaciones:

- La captura de los movimientos realizados, es decir, la notación del juego se realiza automáticamente, lo que hace posible generar un historial de partidas.
- El envió de información se reduce, ya que solo se envían paquetes de información con las posiciones de los últimos movimientos.
- Se cuenta con una clase para el acceso a los dispositivos de video, la cual puede ser reutilizada.
- Se puede tener acceso al puerto paralelo, lo cual está restringido en los sistemas operativos Windows NT y XP; gracias al desarrollo de la clase TParallelThread, siendo reutilizable su código.

- Con el desarrollo de la clase Imagen se cuenta con una herramienta para el procesamiento digital de imágenes.

El conjunto de protocolos TCP/IP ha sido de vital importancia para el desarrollo de las redes de comunicación, sobre todo para Internet. El ritmo de expansión de Internet también es una consecuencia de estos protocolos, sin los cuales, conectar redes de distintas naturalezas (diferente Hardware, sistema operativo, etc.), hubiera sido mas difícil. Así pues, se puede decir que los protocolos TCP/IP fueron y son el motor necesario para que las redes en general, e Internet en particular, se mejoren y se pueda lograr un mejor medio para la difusión de información.

El procesamiento de imágenes es un campo generalmente de investigación, ya que hay muchos métodos para solucionar un mismo problema, pero estos métodos varían según las necesidades, por lo que no hay una única manera de solucionar un problema de visión artificial. Por lo tanto, es muy importante contribuir con aplicaciones reales al campo de visión artificial.

5.2 Mejoras al sistema WebChess

Las principales ventajas que tiene el desarrollo de software es que se puede reutilizar y evolucionar, permitiendo que ciertos módulos den solución a otros problemas, o que ciertos módulos puedan ser modificados sin alterar todo el sistema. Por lo consiguiente se puede decir que puede hacerse mucho para la expansión de este sistema. A continuación se enlistan algunas de las posibilidades de mejoras, y se hace una breve discusión al respecto de cada una de ellas.

- **La iluminación:** Es un punto clave ya que cuando se hace una variación se presentan problemas en la entrada de datos, como son el no definir el valor de umbral a la hora de segmentar la imagen, esto puede ser producido por una iluminación no uniforme o por que la iluminación no es la misma en las dos imágenes, o por ser demasiada, generando poco rango para la identificación de las piezas.

- **Bases de datos:** El historial puede implementarse sobre una plataforma mas concernientemente a bases de datos, como un servidor de SQL, CORBA, o cualquier otra dedicada a este tipo de aplicaciones. Con el fin de albergar un número mayor de partidas, mejorar la organización y búsquedas de éstas.

- **Dispositivos de adquisición de la imagen:** Obtener las imágenes desde cámaras IP con el fin de poder trasmitir más de una partida a la vez.

- **Sistema tutor:** Implementar un módulo tutor para la explicación de los movimientos ajedrecísticos y su corrección en el caso de realizar movimientos inválidos. Así mismo, dentro del historial se pueden cargar algunas aperturas interesantes, con el fin de poder analizarlas paso a paso.

- **DDNS:** La actualización de la dirección, en el servidor de nombres de dominio, podría implementarse dentro del mismo sistema.

5.3 Trabajos futuros

Las clases implementadas en el desarrollo del presente trabajo pueden resultar útiles en diferentes aplicaciones y situaciones. Así como también se pueden probar otros métodos para la identificación de la posición de las piezas, algunas de estas propuestas podrían ser las siguientes:

- El empleo de heurísticas o métodos para el reconocimiento de objetos, como sería la aplicación de redes neuronales o el uso de la trasformada Hough, la cual consiste en la detección de formas, pero con altos costos computacionales.

- Detección de las piezas a partir de imágenes digitales en color.

- Detección de píxeles o grupos de píxeles de un determinado tono de color, sobretodo en el caso de regiones de apariencia homogénea para una mejor segmentación.

- El principio de funcionamiento podría dar difusión a juegos similares como son damas chinas, damas inglesas, etc. Que presenten el movimiento de piezas en un tablero o ambientes con poco cambio.

- La clase para la captura de imagen puede ser agregada a sistemas simples, como bases de datos para los clientes de algún departamento, en el cual se quiera agregar la foto del cliente. O bien, en sistemas donde la adquisición de la imagen sea la base para el chequeo de ciertas piezas o monitoreo de sucesos.

- En el área electrónica se pueden realizar muchos experimentos con el puerto paralelo desde la adquisición de datos para su graficación, hasta la escritura con lo que se pueden hacer generadores de funciones.

- Reutilizando los elementos que en este trabajo se emplean, se puede dar lugar a sistemas para monitoreo y control a distancia. Explotando los recursos de una computadora personal y la Internet, como medio de comunicación en el sistema.

Referencias

[Charter] CHARTER OJEDA, F. *Programación con C++ Builder*. Ed. Anaya Multimedia, 1997.

[Cruz] CRUZ, J. *Sistemas para el conteo automático de huevecillos de moscas del Mediterraneo*, Tesis de maestría. Instituto Nacional de Astrofísica, Óptica y Electrónica. MECAS, 2001.

[Deitel] DEITEL, H.M. y Deitel, P.J., *Como programar en C/C++,* México, Ed. Prentice-Hall Hispanoamericana, 2003.

[Escalera] DE LA ESCALERA Hueso, Arturo, *Visión por computadora: Fundamentos y Métodos*, Universidad Carlos III de Madrid, Ed. Prentice Hall, España, 2001.

[Gerhard et. al.] RITTER, Gerhard X.; Joseph N. Wilson *Handbook of Computer Vision Algorithms in Image Algebra*. CRC Press, 1996.

[González et. al.] GONZÁLEZ, Rafael y Woods, Richard, *Tratamiento Digital de Imágenes*, USA, 1996.

[Infaimon] INFAIMON, *Especialistas en Visión Artificial, Software Fundamentos Teóricos*, 2003.

[Mussot] DEL CASTILLO Mussot, Marcelo, *Cómo acercarse al ajedrez*, Limusa, México, 1993.

[Reisdorph et. al.] REISDORPH Kent y Henderson Ken *Teach Yourself Borland C++Builder in 21 Days*, Sam library, 1997.

[Sahlin] SAHLIN, Doug, *Macromedia Flash MX ActionScripts for designers*. Wiley Publishing 2002.

[Schildt] SCHILDT, Herbert, *Borland® C++ Builder™: The Complete Reference* .McGraw-Hill Companies,inc. 2001.

[Simon] SIMON, Richard *Windows NT Win32 API SuperBible*. Macmillan Computer Publishing, 1998.

[Stallings 2002] STALLINGS, W. *Comunicaciones y redes de computadoras*. Prentice Hall, 2002.

[Stallings 1995] STALLINGS, W. *Sistemas Operativos.* Prentice Hall, 1995.

[Stevens] STEVENS W., Richard, *TCP/IP illustrated: The Protocols.* Addison-Wesley, 2002.

[Sucar] SUCAR, Luis Enrique y Gómez Giovanni, *Procesamiento de imágenes y Visión computacional*: Libro de Texto, Laboratorio de Sistemas Inteligente, Instituto Tecnológico y de Estudios Superiores de Monterrey campus Cuernavaca. México, 2003.

[Valera] VALERA, Gómez Jesús, *Metodología de la corrección digital de la distorsión en sistemas de captación de imágenes basados en lentes*. 2002.

[Vegas] VEGAS Hernández, Jesús M. *Introducción a los Hilos*, Dpto. Informática Universidad de Valladolid.

[Vicens et. al] VICENS, M., et. al. *Tratamiento digital de imágenes. Técnicas básicas.* Mundo Electrónico, May. 1990.

[Wang et. al.] WANG, Y. et. al. *Automatic threshold selection using histogram quatization*. SPIE Journal of Biomedical Optics, 1997.

Sitios de Internet

[Bourke] BOURKE, Paul 2002 *Lens Correction and Distortion* http://astronomy.swin.edu.au/~pbourke/projection/lenscorrection/ [Consulta: 25 Mayo 2005].

[Dekey] DEKEY, Sam, *Soluciones en Tiempo Real con Windows NT*. Revista de Universidades y Radiorama, Marzo 2001.Dirección URL: <http://www.redeweb.com>. [Consulta: 15 Noviembre 2004].

[DGT] Tablero DGT. *DGT projects*, Editor-in-Chief, Tufts University. http://www.dgtprojects.com. [Consulta: 15 Abril 2005].

[Dyndns] Dynamic Network Services https://www.dyndns.org/ [Consulta: 25 Mayo 2005]

[Fisher et. al.] FISHER, Robert, *Hypermedia image processing reference* http://homepages.inf.ed.ac.uk/rbf/HIPR2/pixsub.htm [Consulta: 25 Mayo 2005]

[Macromedia1]http://www.macromedia.com/support/flash/action_scripts/actionscript_dictionary/actionscript_dictionary860.html [Consulta: 25 Mayo 2005]

[Msagent] http://www.microsoft.com/msagent/downloads/user.asp#intro [Consulta: 25 Mayo 2005]

[Noli et. al.] NOLI Aldo et. al. *Arquitectura TCP/IP*
http://web.frm.utn.edu.ar/comunicaciones/tcp_ip.html#1.2 [Consulta: 9 Febrero 2005]
[NWO] Netherlands Organization for Scientific Research (NWO) Image Processing Fundamentals
http://www.ph.tn.tudelft.nl/Courses/FIP/frames/fip.html [Consulta: 16 Marzo 2005]

[Young et. al.] YOUNG, I.T. *Image Processing Fundamentals*
http://www.ph.tn.tudelft.nl/Courses/FIP/noframes/fip.html [Consulta: 6 Marzo 2005]