A D I

AMBIENTE DE DESARROLLO DE APLICACIONES PARA PROCESAMIENTO DE IMAGENES DIGITALES.

Instituto Tecnológico y de Estudios Superiores de Monterrey
Campus Querétaro

Autores :

Idalia C. Solis Castro.
Ma. Luisa Herrera López.
Edgar Gerardo Mondrágon Martinez.

Alumnos de 9º semestre de la carrera de Ingeniería en Sistemas Computacionales, en el ITESM CQ.

Coordinación del Proyecto :

Ing. Carlos E. Romero Ortiz.
Profesor Depto. Sistemas Computacionales

Ing. Salvador Galindo Hernández.
Director del Depto. de Electrónica y Comunicaciones.

Dirección :

Departamento de sistemas Computacionales del
ITESM Campús Querétaro:
Henry Ford No. 10 Parques Industriales
CP 76130 Querétaro, Qro.
México.
Tel. (42) 12 60 90 Ext 160.
Fax: (42) 12 82 50
Bitnet : HMORELOS at VMTECQRO

A D I
AMBIENTE DE DESARROLLO DE APLICACIONES PARA PROCESAMIENTO DE IMAGENES DIGITALES.

RESUMEN

El presente trabajo consiste en el desarollo de una herramienta para generación de aplicaciones en el área de procesamiento de imágenes digitales. La idea del proyecto es el poder contar con una herramienta de alto nivel para el desarrollo de aplicaciones en el área de tratamiento de imagenes digitales.

El término de procesamiento o tratamiento de una imagen se refiere a la utilización de ciertos algoritmos y herramientas para cambiar la apariencia de la imagen original, utilizando para esto información de la misma, o realizando una transformación y aplicando los algoritmos o funciones matemáticas sobre la transformada de la imagen.

Actualmente muchas áreas en la industria y medicina dependen de algún tipo de imágen para diagnòstico o toma de decisiones en ciertos procesos. Estas van desde fotografía normal hasta radiografías, termografías y tomografías. La calidad de la imágen es un factor de grán importancia y en algunos casos es complicado lograr algo óptimo. Las técnicas antes mencionadas dan una alternativa hacia éste problema, el poder tratar de cierta forma la imágen y lograr resaltar o hacer más claros ciertos objetos o áreas de interes por medio de una computadora. Esto implica beneficios en cuanto al costo del proceso, debido a que en algunos casos el material para volver a tomar la imágen son demasiado caros o por la situación es casi imposible volver a hacerlo.

La herramienta que se está desarrollando provee un puente de alto nivel entre los lenguajes de programación convencionales y el usuario o desarrollador de la aplicación en una área específica. Una de las partes importantes del proyecto es la inclusión de un lenguaje orientado a objetos con características de alto nivel. Esto permitirá que las aplicaciones utilizando las técnicas de tratamiento de imágenes sean más comunes y aprovecharlas en distintos campos de México y América Latina.

Esto es como parte de los proyectos del Departamento de Sistemas Computacionales del ITESM Campús Querétaro.

DESARROLLO

INTRODUCCION

El uso de técnicas de procesamiento nos permite utilizar de una manera más completa la información que pudiera contener una imagen tomada de alguna forma (fotografía normal, termografía, radiografía, tomografía, etc.). AL tener una mejor información con respecto a la imagen, podemos hacer un uso más eficiente de los recursos, ya sea tiempo para tomar una fotografía mejor, disponibilidad de la fotografía, y de alguna manera mejorar los resultados que se obtendrían con la imagen original.

El término de procesamiento o tratamiento de una imagen se refiere a la utilización de ciertos algoritmos y herramientas para cambiar la apariencia de la imagen original, utilizando para esto información de la misma, o realizando una transformación y aplicando los algoritmos o funciones matemáticas sobre la transformada de la imagen.

La información que tenemos de la imagen recién digitalizada y sobre la que se trabaja inicialmente, es de intensidad luminosa. Si consideramos que se encuentra discretizada en un arreglo de **i** renglones y **j** columnas, tenemos una función de mapeo **f(i,j)** que nos representa la intensidad luminosa o nivel de gris en el punto **(i,j)** de la imagen. Esta intensidad luminosa es discreta y generalmente por facilidad de almacenamiento es un número potencia de 2.

Hay ciertos casos en que la información tal cual de la imágen está mezclada con ruido de tal manera que la combinación de técnicas espaciales no es ideal para mejorarla. Para esto se utiliza la información de frecuencia de la imágen, obteniendola por medio de la Transformada de Fourier. Este espectro de frecuencia puede filtrarse y obtener resultados mejores que en espacial.

HERRAMIENTA DE DESARROLLO

El uso de éstas técnicas permite que muchos campos, donde las imágenes son importantes, se vean beneficiados en la calidad final de sus procesos.
El problema de introducir tecnología nueva es la complejidad de ésta para aprovecharla al máximo. La búsqueda de la facilidad de uso utilizando al máximo el poder de la computadora fué la razón de crear un ambiente completo para el desarrollo de éstas aplicaciones para procesamiento de imagenes digitales. Este provee un puente entre los usuarios o desarrolladores y un sistema sofisticado de tratamiento.

La herramienta fue dividida en varias partes relacionadas, con las cuales se trabaja de forma simultánea :

I. lenguaje orientado a objetos
II. rutinas de soporte para el S.O.(Memoria, E/S).
III. funciones para procesamiento espacial
IV. funciones para procesamiento en frecuencia (Fourier).
V. funciones de soporte para visualización, impresión, compactación y comunicación con dispositivos.
VI. interfase del ambiente final.

Estructuralmente la herramienta se divide en tres partes principales :

◊ Ambiente Externo.
◊ Ambiente Interno.
◊ Ambiente de Usuario (Desarrollo).

Ambiente Externo :
Incluye todas las declaraciones y el codigo objeto de las funciones de soporte y procesamiento. Este codigo es la base para la aplicación final, éstas rutinas van totalmente relacionadas con las declaraciones de objetos del lenguaje del ambiente.
Las rutinas para dispositivos consideradas en esta parte por grupos son:

1) De digitalización.
Las rutinas para estos dispositivos incluyen la toma interactiva de imágenes. El equipo con que se cuenta para esto consiste en :

◊ Cámara para digitalización WINCAM.
◊ Tarjeta para digitalización de video.

2) De entrada interactiva.
Estos dispositivos permiten una comunicación con el usuario de forma interactiva, actualmente se soportan sólo dos :

◊ teclado.
◊ ratón.

La interfase en el lenguaje se hace por medio de mensajes por parte de los dispositivos a la aplicación.

3) De salida impresa.
Estos dispositivos son basicamente impresoras, diferenciandolas en dos tipos principales :

◊ Matriz de puntos.
◊ Impresora Laser.

Para el manejo de la última, se ésta desarrollando un módulo de comunicación al equipo Apple Macintosh para aprovechar completamente la capacidad gráfica del equipo con que se cuenta. Este módulo incluye compactación, decompactación y manejo de comunicación por puerto serial.

4) De salida para visualización.
Estas rutinas dan soporte para la visualización en video de la imágen a procesar y la procesada. Se soporta los manejadores de video para los subsistemas MCGA y VGA, además de manejo de un monitor externo.
Las imágenes con las cuales se ésta trabajando actualmente tienen una resolución hasta de 220 por 256 pixels con 256 niveles de gris.

5) De salida a almacenamiento.
Estas rutinas tienen como función el almacenamiento en disco de las imágenes, así como su lectura y compactación. Para estas tareas se utilizan formatos generales, para asegurar compatibilidad con otros equipos y sistemas de software. Los formatos utilizados incluyen : TIFF, BUF y PCX.

6) De salida a comunicaciones.
La función de éste grupo de rutinas es la de proveer una salida del sistema hacia otras máquinas, incluyendo comunicación con equipos de procesamiento mayores (mainframes). Esto se incluye debido a ciertos paquetes de procesamiento para imagenes muy grandes que existen para mainframes.

I. LENGUAJE ORIENTADO A OBJETOS

La idea de proveer un lenguaje orientado a objetos es la facilidad que dá este tipo de programación en cuanto a visualizar como objetos independientes y completos a los distintos dispositivos y entidades logicas, en lugar de manipular directamente un conjunto de datos y funciones como ocurre en la programación normal.
Esto le dará al usuario un ambiente amigable y sencillo sin restar sofisticación y calidad a las aplicaciones finales que se deseen realizar.

El desarrollo del lenguaje se dividió en las siguientes partes :

- ◊ Especificación de los alcances y capacidad del lenguaje.
- ◊ Evaluación y Elección del lenguaje para la aplicación final.
- ◊ Diseño de la gramática.
- ◊ Elaboración de la tabla de parse.
- ◊ Elaboración del manejador de la tabla (chequeo de sintaxis, chequeo de tipos y referencias a objetos).
- ◊ Elaboración del Generador de Codigo Final.

La idea principal del lenguaje es la de dar al usuario una manera más sencilla de referirse a los dispositivos y a la imágen en sí para procesarla. La base del diseño del lenguaje es la programación orientada a objetos, de acuerdo a esta filosofía de programación, tendremos ahora representados a cada una de las entidades lógicas y físicas por medio de objetos independientes y relacionados con los demás que comparten el medio. La definición para un objeto básico de una imágen sería la siguiente :

```
imagen = objeto
metodos
        invierte.
        desplaza ( x, y ).
        escala ( x, y ).
        ...
atributos
        info (resx,resy).
        resx.
        resy.
        grises.
fin
```

Donde los métodos son las acciones que un objeto dado, en este caso una imágen, puede llevar a cabo consigo misma, y los atributos son las caracteristicas que la definen. Esta definición es para una **clase** general de imágen. Una **instancia** sería la definición específica para una imágen con valores conocidos, un ejemplo de instancia es el siguiente :

```
i1 : imagen ( resx = 100.
              resy = 100.
              grises = 256.
            ).
```

Aqui el objeto i1 está definido como imágen con una resolución de 100 puntos horizontales por 100 verticales y 256 niveles de muestra.
Para usarlo como imágen en sí faltaría darle los datos sobre sí misma, la información en sí de la imágen. En éste punto entra la interacción con los demás objetos definidos. Con los objetos le damos al desarrollador una visión no de datos aislados, sino información interrelacionada y con capacidad de manipularse.

El ambiente al principio contempla una serie de definiciones básicas de objetos, a partir de los cuales se pueden crear subclases con nuevos métodos y atributos para formar una biblioteca y permitir que el desarrollador haga crecer el sistema. En sí estos objetos con métodos y características nuevas son representaciones de las funciones y el código externo definido que se habrá de ligar con el código generado para la aplicación final.

La comunicación entre los objetos puede definirse como un mapeo entre unidades lógicas y dispositivos físicos. Las unidades lógicas definidas inicialmente son :

- ◊ imágen.
- ◊ archivo.
- ◊ Path.
- ◊ histograma.
- ◊ Cada modo de video soportado.
- ◊ Nombre de cada dispositivo externo de digitalización.

Las unidades físicas iniciales son las siguientes:

- ◊ de digitalización.
- ◊ de entrada interactiva.
- ◊ de entrada/salida para almacenamiento.
- ◊ de despliegue.

El mapeo entre unidades lógicas y fisicas o de lógicas a lógicas define las acciones o información que se puede compartir entre ellas. Para una imágen el mapeo sería el siguiente :

- ◊ imágen <-> archivo. : Mapeamos un archivo a una imágen o viceversa.
- ◊ imágen -> despliegue. : Una imágen puede ser mandada a un dispositivo físico de despliegue.
- ◊ imágen <- digitalizador. : Una imágen puede ser digitalizada.

Aqui el mapa de relaciones puede hacerse más complejo al definir objetos nuevos, el compilador antes de generar codigo se encargará de checar y resolver éstas relaciones para evitar conflictos de tipos en la aplicación final.

Los objetos definidos por el usuario incluiran un objeto tipo programa con código para manejo de instancias. El conjunto de éstos programas es el que definirá la aplicación final. Un programa que lea una imágen , invierta sus niveles y la despliegue sería :

```
p : programa (

      i : imagen (   xres=100.
                     yres=100.
                     grises = 256.
                 ).
      d  : camara.
      v  : monitor.

      camara.digitaliza(i).
      i.invierte.
      v.despliega(i).
 ).
```

A partir de éste objeto programa, el compilador checa los errores de sintaxis , de tipos y de referencias, para generar una aplicación final en algún lenguaje de alto nivel.

Utilizando programación orientada a objetos nos permite poder crear una aplicación desde alto nivle, especificando sólo un mínimo de codigo en nuestro objeto programa. Si quisieramos mandar a cualquier dispositivo nuestra imágen, la línea de despliega puede cambiarse a algo más general, aprovechando las propiedades de **herencia** y **polimorfismo** de cada clase de objeto :

```
p : programa (

        i : imagen (    xres=100.
                        yres=100.
                        grises = 256.
                    ).
        d   : camara.
        v   : dispositivo.

        camara.digitaliza(i).
        v = que_dispositivo?
        i.invierte.
        v.despliega(i).
    ).
```

Con esto podemos asegurar que nuestra aplicación sea lo más general posible en cuanto al hardware que se puede utilizar. El elnguaje cuenta con variables especiales para guardar información sobre el ambiente de usuario, cuantos objetos existen, cuantos de algúna clase, etc.

Para generar las tablas del compilador y de codigo final se utilizará una herramienta del sistema operativo UNIX : YACC. Esta herramienta recibe una gramática de entrada y genera una tabla de parse tipo LALR.

El lenguaje fuente de la aplicación final puede ser C++ o C. Para el primero el codigo es inmediato, debido a la orientación de objetos, para el segundo se requiere de una traducción utilizando rutinas de soporte para simular un ambiente de objetos. La aplicación final ejecutable será generada utilizando un compilador para cualquiera de los lenguajes.

II. RUTINAS DE SOPORTE PARA EL S.O.(MEMORIA, E/S).

El ambiente requiere de ciertas rutinas de soporte interno para poder manipular los objetos, el compilador, etc. Estas funciones se dividen en tres grupos :

◊ Manejo de Memoria.
◊ Manejo de I/O (archivos, impresora, etc...).
◊ Interface para llamadas al S.O.

En éstas rutinas se incluyen manejadores de memoria, de disco y llamadas a interrupciones.
El ambiente inicial para todas las partes del proyecto es el sistema operativo MS DOS 3.3, el lenguaje para desarrollo es C, sin embargo se ha comenzado a evaluar otras alternativas sobre el sistema operativo y diferentes implantaciones del lenguaje C.

El problema principal en la manipulación de imágenes son los requerimientos de memoria, para resolverlo se proponen tres soluciones :

◊ **Manejador de Memoria virtual.**
Este permite mover entre memoria y disco los objetos a manipular, permitiendo así de cierta manera la disposición de memoria mayor que la física. El equipo que se utiliza cuenta con discos duros desde 30 hasta 60 megabytes de capacidad, por lo que es bueno considerar la posibilidad de usarlos como extensión de memoria principal.

◊ **Sistemas Operativos Alternos.**
El ambiente de desarrollo actual es DOS 3.3, las limitaciones de memoria y la disponibilidad de maquinas con procesadores más sofisticados y de mayor capacidad (i286,i386) permiten pensar en utilizar un sistema operativo con mayor capacidad. El sistema que se considero para evaluación es MS OS/2 principalmente por su compatibilidad con MS DOS, actualmente se está revisando bibliografía para determinar si su uso puede ser considerado. Otra alternativa la constituyen UNIX y AIX por ser un sistema operativo de gran difusión para estaciones de trabajo, sin embargo se han tenido problemas en su instalación, aunque se toma en consideración para la evaluación.

◊ **Manejador de Memoria utilizando i386.**
En el proyecto se dispone de computadoras IBM modelo PS55sx, éstas cuentan con un procesador i386. Este cuenta con ciertas caracteristicas avanzadas para manejo de memoria y procesos, la posiblidad que existe es el poder desarrollar un manejador que aproveche éstas caracteristicas. El manejador sería un shell pequeño, que permitiera activar procesos, asignar memoria y accesar interrupciones de DOS.

III. FUNCIONES PARA PROCESAMIENTO ESPACIAL

Una de las partes básicas del proyecto lo constituyen en sí una serie de rutinas para procesamiento, juntas forman una biblioteca completa para ser utilizadas en la parte de la aplicación final.

La biblioteca ésta divida en dos tipos de funciones, las primeras son para procesamiento espacial , estas se basan en algoritmos que actúan sobre los puntos originales de la imágen. Los principales algoritmos utilizados en estas funciones son los que engloban las siguientes técnicas:

◊ **Reforzamiento y detección de bordes.**
Esta técnica se basa en la utilización del gradiente y el Laplaciano de una función discreta. Aplicando éstos operadores, las discontinuidades entre los bordes de la imágen se marcan de tal manera que es más sencillo diferenciar diferentes objetos por sus contornos.

◊ **Manipulación de histogramas.**
La distribución de probabilidad de los niveles de gris en una imágen nos dan una forma de medir su constraste. Al manipularlo por medio de herramientas estadisticas es posible hacer más clara un área dada o oscurecer y dar contraste para diferenciar objetos en una imágen.

◊ **Suavizamiento.**
La idea de suavizamiento es el contrario de reforzamiento de bordes, en ésta técnica las discontinuidades son consideradas como ruido en forma de puntos brillantes de mayor magnitud que el promedio de la imágen, para reducirlo se utilizan métodos de promedios y media entre puntos vecinos. La característica principal que tiene este uso es que al reducir las discontinuidades la imágen aparece borrosa, y dependiendo de la consideración de tamaño para puntos vecinos se genera un efecto de cuadricula sobre la imágen.

◊ **Aplicación de mascarillas (esta incluye algunas de las técnicas anteriores).**
Las técnicas anteriores involucran algoritmos especiales para cada una, utilizando el principio del teorema de convolución podemos crear una matriz de n x n para lograr el mismo efecto que la detección de bordes, reforzamiento y suavizamiento.
El problema con el uso de éstas matrices o mascarillas es que en algunos casos el efecto local se marca demasiado en imágenes con resolución baja, por ejemplo para una mascarilla de tamaño 3 x 3 que suavize una imágen por promedios, el efecto de cuadricula (tablero de ajedrez) que se mencionó anteriormente es muy marcado.

◊ **Pseudocolor.**
En algunas aplicaciones se plantea como opción el coloreado de ciertas áreas para resaltar algunos detalles de interes! Esto puede ser hecho de dos formas principalmente :
por nivel, esto es buscar un nivel de gris y cambiar lo que se encuentre en ese nivel a un valor asociado de color, el otro es punto a punto, similar a los paquetes de edición de gráficos comerciales que existen. En éstas rutinas se incluye soporte para manejar toda la capacidad gráfica de los subsistemas VGA y MCGA en el equipo IBM PS.

◊ **Fotomontaje.**

El fotomontaje engloba la edición y restauración de fotografías principalmente. Se desarrollo un algoritmo para recorte de objetos en una imágen, moverlos y posicionarlos en otra diferente.

Este tipo de edición es de utilidad en arquitectura, si se piensa en la posibilidad de unir un sistema de imágenes con un sistema de diseño CAD, para editar modelos reales con esquemas de diseño por computadora.

La biblioteca se encuentra actualmente en formación. Las ampliaciones que se harán serán en cuanto a capacidad para procesamiento y manejo de imágenes más grandes o conjuntos de ellas. Se espera poder utilizar al máximo al capacidad del hardware disponible, tanto de procesamiento como de digitalización y salida.

IV. FUNCIONES PARA PROCESAMIENTO EN FRECUENCIA (FOURIER).

El procesamiento en frecuencia es más complicado que el espacial, debido a que implica una transformación de la imágen como fase previa. La transformación utilizada es por medio del algoritmo de transformada rápida de Fourier con el algoritmo de Cooley Tukey referido en [GONZ87] y [LIM 90]. La codificación de éste algoritmo exige requerimientos de memoria altos, ésta fué la principal razón para considerar otros sistemas operativos y manejadores de memoria.
Una vez procesada la imágen se utilizan filtros para lograr enfatizar ciertas zonas, los filtros ha utilizarse son los siguientes :

- ◊ Pasa Altos.
- ◊ Pasa Bajos.
- ◊ Butterworth.
- ◊ Homomórfico.

Algunos de éstos tienen relación con los procesos espaciales aunque los resultados logrados en frecuencia son mejores. Es posible también crear filtros en frecuencia y realizar una convolución en espacio, es similar a la técnica con mascarillas.

V. FUNCIONES DE SOPORTE PARA VISUALIZACION, IMPRESION, COMPACTACION Y COMUNICACION CON DISPOSITIVOS.

Una vez que se obtienen resultados al procesar una imágen es necesario visualizarlos de alguna forma, se cuenta con dos dispositivos principales de salida para visualización : video e impresora.

En video se incluyen los subsistemas VGA y MCGA para el equipo IBM PS2 y monitores externos con mayor resolución y niveles de gris.

La impresión se realiza por medio de una técnica de sustitución de patrones de puntos por niveles de gris, en éste caso se asigna para cada valor de gris un patron de 8 por 8 pixels a imprimirse.
Los mejores resultados fueron utilizando una impresora Apple Laserwriter II NTX, éste equipo es capáz de imprimir con una resolución de 300 puntos por pulgada.
Se comtempló también la implantación de esquemas de codificación para compactación de imágenes, ya sea para almacenamiento o para transmisión por puertos. Los esquemas utlizados son los siguientes :

- ◊ Codificación Huffman.
- ◊ Codigos B.
- ◊ Codigos de Shift.
- ◊ Codificación Aritmética.

Los primeros códigos son llamados de longitud variable porque se asigna a cada caracter un numero variable de bits dependiendo de su frecuencia de ocurrencia. La última codificación utiliza un enfoque de geometría fractal para compactar una imágen completa en un número de punto flotante.

VI. INTERFASE DEL AMBIENTE FINAL.

El ambiente integrado final, se piensa con una interfase interactiva con ventanas múltiples y ratón como dispositivo de selección. Esta tiene que ser muy amigable para seguir con la filosofía orientada al usuario del proyecto.
La primera versión será orientada a texto, después se pensará en migrar a un ambiente gráfico, la razón es dar más énfasis a otros puntos base del proyecto.

CONCLUSIONES

El uso de técnicas de procesamiento abre una herramienta nueva para diferentes campos que dependen de una imágen de algún tipo como parte importante de los procesos a realizarse, ya sean de diagnóstico como es la medicina o de reconocimiento de formas para la industria.

Esto permite pensar en procesos con una calidad mayor y un control más estricto llevado a cabo con tecnología no importada para America Latina. Esto implica no depender de otros países para utilizar tecnología de vanguardia en procesos industriales y de medicina.

AUTORES y COAUTORES DEL PROYECTO

A continuación se lista a los autores y coautores del proyecto por módulo, todos son estudiantes de 8º y 9º semestres de la carrera de Ingenieria en Sistemas Computacionales en el ITESM Campus Querétaro.

Diseño del Lenguaje y Compilador :
Idalia Constantina Solis Castro.
Francisco Espinola García.

Asesoria en el área de Compiladores y Lenguajes:
Ing. Alfonso Rodriguez.

Manejo de Memoria, Sistema Operativo y Transformada de Fourier :
Ma. Luisa Herrera López.
Concepción Loyola Martínez.
Eva del Carmén de Aquino Rizo.
Alejandro Garza.

Rutinas de Soporte :
Edgar Gerardo Mondrágon Martinez.
Alejandro Fernández Fraga.
Oscar León Espinosa.
Salvador Ugalde Shimano.
Damián Vazquez Soriano.
Gerardo Salazar Rocha.
Mauricio Mejía Alvarez.
Edgar Ortega M.

REFERENCIAS Y BIBLIOGRAFIA

[GONZ87] Digital Image Processing , Gonzalez, C. Rafael. Addison Wesley, 1987.

[BORL88] Turbo C Versión 2.0 Programmer's Guide. Borland International, 1988.

[ELEC89] EDC - 1000 Computer Camera Technical Manual. Electrim Corp., 1989.

[APPL89] Laser Writer II NT NTX Owner's Guide. Apple Computers, 1989.

[DUNC89] Advanced MS DOS, Duncan Ray. Microsoft Press, 1989.

[NORT89] Programmer's guide to the IBM PC & PS/2. Norton, Peter. Microsoft Press, 1989.

[LIM 90] Two Dimensional Signal and Image Processing Lim, S. Jae. Prentice Hall, 1990.