El sistema ITK: Modelador Visual para Procesamiento de Imágenes

Nicolás Fernández, Walter Heindl y Claudio Delrieux*

Departamento de Ing. Eléctrica y Computadoras - Universidad Nacional del Sur - claudio@acm.org

Palabras Clave: Procesamiento de Imágenes. Sistemas de Gestión de Información

1. Objetivos de la linea de investigación

El Procesamiento Digital de Imágenes (PDI) tiene como objetivo manipular computacionalmente señales bidimensionales (imágenes) para mejorar la percepción, detección o interpretación, retocar sus características visuales, detectar patrones, etc. Las aplicaciones de PDI consisten mayoritariamente en diseñar una secuencia de procesos, eligiendo los mismos de acuerdo a su efecto sobre las imágenes intermedias, y manipulando los parámetros en forma conveniente. Una aplicación PDI típicamente involucra construir una cadena de procesos entre los cuales es necesario elegir el más conveniente por prueba y error. Además, usualmente estas complejas cadenas de procesos deben aplicarse a más de una imagen, lo cual resulta tedioso pues es necesario aplicar cada uno de los pasos con los mismos parámetros. Estas dificultades llevan a buscar una manera flexible e intuitiva de plantear el diseño de un procesamiento, especialmente para aquellos usuarios que utilizan el PDI como herramienta de trabajo en otras disciplinas. Uno de los sistemas de gestión de información que parece adecuada, por su fácil utilización y su popularidad, es la planilla de cálculo. Esta idea fue explorada en trabajos anteriores. Sin embargo, para muchas tareas, la planilla de cálculo no es la metáfora correcta para interactuar con el usuario, sobre todo en aplicaciones del PDI donde la tubería de procesos no está tan estructurada, o bien hay que alterarla constantemente de manera experimental. Por dicha razón, en este proyecto investigamos otra metáfora para organizar la tubería de procesos PDI, basada en la construcción interactiva de la misma por medio de una biblioteca de procesos. Presentamos la implementación del sistema ITK, el cual permite manejar la tubería de procesos por medio de una interfase gráfica. Esto permite acceder a cada paso del procesamiento y modificar los parámetros de cada uno individualmente, la reutilización de una secuencia de procesos o de sus partes a cualquier otra imagen, guardar las sesiones intermedias, y en general facilita la modificación por prueba y error de los parámetros de cada etapa.

2. PDICalc vs. ITK

Supongamos como ejemplo de PDI la tarea de reconocer un componente electrónico a partir de su identificación. La estrategia usual para reconocer caracteres en este tipo de

*Parcialmente financiado por la SECyT-UNS

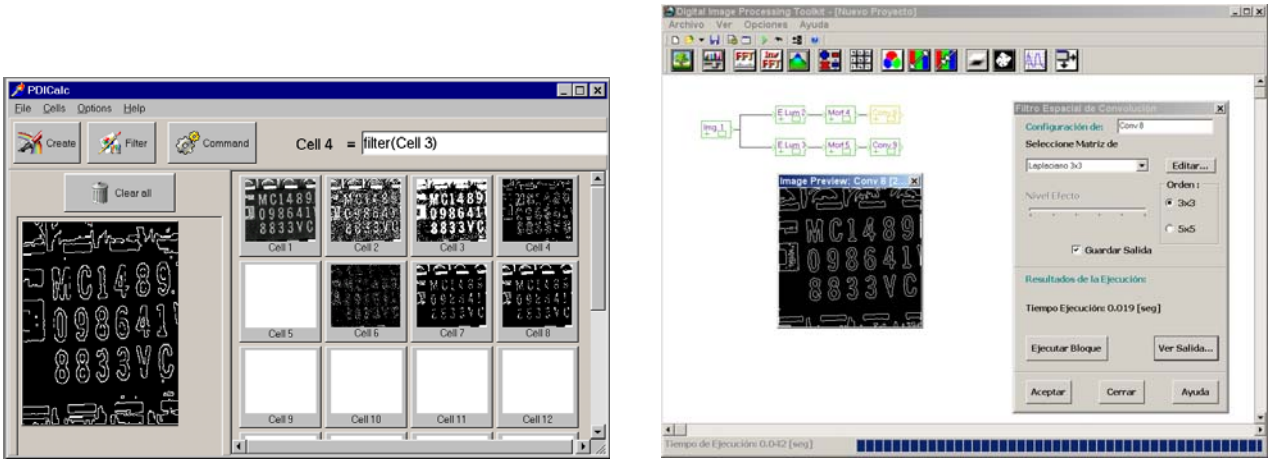

Figura 1: Ejemplo completo programado (a) en el sistema PDICalc, y (b) en el sistema ITK.

contextos consiste en obtener una poligonal a partir de una segmentación de fronteras. Pero las imperfecciones usuales en componentes reales son amplificadas por los operadores pasaaltos, lo cual produce una sobresegmentación que inutiliza el ajuste poligonal a la frontera segmentada. Por esta razón, la tubería de procesamiento es un poco más compleja. Primero se binariza la imagen, y sobre la imagen binarizada se aplican aperturas o cierres morfológicos para recuperar la forma correcta de los caracteres a segmentar. Estos pasos están relacionados entre sí. Una binarización con un valor umbral cercano al blanco va a requerir aperturas para restaurar la forma de los caracteres, mientras que una binarización cercana al negro requerirá cierres. Este es sin duda el punto más importante a tener en cuenta como motivación de este trabajo: *las diferentes etapas de una tubería de procesos PDI están mútuamente realimentadas* y por lo tanto se requiere algún mecanismo adecuado que permita la gestión iterativa y experimental hasta encontrar la secuencia adecuada de procesos, y los valores de los parámetros en cada una de las etapas. Este ejemplo ilustra la naturaleza interactiva, por prueba y error, que tiene el diseño de una tubería de procesos PDI. Muchas veces, además, estas complejas cadenas de procesos deben aplicarse a más de una imagen, lo cual resulta tedioso pues es necesario aplicar cada uno de los pasos con los mismos parámetros. Todas estas dificultades llevaron al desarrollo del sistema PDICalc [3, 2, 1]. PDICalc concibe al PDI como una planilla de cálculo, cuyas celdas están ocupadas por imágenes. El contenido de cada celda puede provenir de un archivo, o bien obtenerse por medio del cómputo de algún tipo de operación sobre otras celdas. La interfase gráfica es semejante en su operación a una planilla común (ver Fig. 1(a)), de manera de lograr que cualquier usuario se sienta cómodo con su uso. Sin embargo, para algunas aplicaciones el uso de la planilla de cálculos termina dificultando la tarea de programar una serie de procesos. Por dicha razón, en este proyecto desarrollamos la otra posible metáfora visual: el Toolkit o Biblioteca de herramientas de PDI, las cuales se vinculan dinámicamente por medio de una interfase gráfica. De esa manera surge el sistema ITK(ver Fig. 1(b)).

3. El sistema ITK

ITK utiliza una interfase basada en los diagramas de bloques, donde cada uno implementa un PDI distinto, los cuales pueden poseer varias imágenes de entrada y salida. Esto conforma una tubería de bloques o procesos los cuales se ejecutan secuencialmente según el diagrama de conexiones entre los mismos. Cada bloque es nombrado automáticamente (aunque el

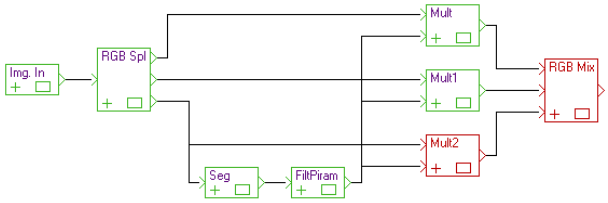

Figura 2: Ejemplo de conexionado entre bloques.

usuario puede cambiar los nombres), y el ícono visual que representa cada bloque contiene su nombre, la o las conexiones de entrada y de salida, un ancla para desplazar el bloque en el área de trabajo, y un área que descuelga el diálogo que permite editar las propiedades del bloque. Cuando el usuario apunta con el mouse a una salida de un bloque, arrastra una conexión desde ese punto a la entrada deseada de otro bloque. Los bloques implementan una abstracción de cada una de las tareas del PDI más usuales, siendo muy fácil la incorporación de nuevos bloques para tareas especiales. El sistema ITK está implementado a través de clases en la plataforma Delphi. Existen 6 clases principales que resumen el funcionamiento de la aplicación:

- *TAbsPDI*: De ésta descienden las clases que implementan las primitivas . Las subclases deben especificar la cantidad de imágenes de entrada y salida.
- *TBloque*: Se relaciona con un descendiente de *TAbsPDI* proporcionándole una interfase gráfica.
- *TBloqueManager*: Se encarga de procesar los eventos que genera el formulario principal.
- *TBloqueCreator*: Es la encargada de crear las distintas instancias de primitivas.
- *TBloqueToDisk*: Permite guardar y abrir las tuberías de PDI.
- *TFormMain*: Especifica la interfase de la aplicación con el usuario.

La ejecución de los procesos de la tubería se realiza ejecutando primero los bloques *TImageSource* y luego los bloques cuyas entradas estén disponibles, así sucesivamente hasta completar la tubería. Para que un bloque se ejecute es necesario que todas sus entradas estén conectadas. Si se modifica la configuración de un filtro PDI intermedio, solo se actualizaran los filtros dependientes de este.

4. Un ejemplo avanzado

Para ilustrar el tipo de aplicaciones para las que ITK está especialmente señalado, utilizaremos como ejemplo el procesamiento de un video. El mismo corresponde a la toma aérea de la ría de Bahía Blanca, realizada para el estudio de las propiedades de la marisma. Lamentablemente el camarógrafo olvidó apagar el fechador, por lo cual el video aparece con fecha y hora, lo cual lo vuelve casi inutilizable. La idea, para recuperar la utilidad del video, consiste en procesar cuadro por cuadro, y eliminar los caracteres generados por el fechador.

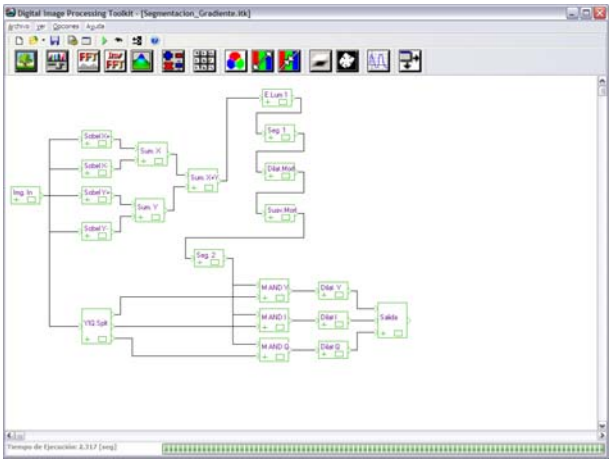

Figura 3: Procesamiento requerido para eliminar caracteres.

Figura 4: Un cuadro procesado con la programación de la Fig. 3.

Una vez encontrado un proceso que realice esta eliminación en forma robusta, simplemente hay que generar un "batch"que se aplique a todos los cuadros del video de entrada, y que genere secuencialmente los cuadros del video procesado.

El proceso que elimina los caracteres se basa en encontrar un área en la imágen en la cual simultáneamente existan altos valores en gradientes direccionales en las cuatro direcciones de los ejes (S-N-O-E). Dichas zonas se segmentan y ecualizan, y se les aplica dilatación y suavizado morfológico para obtener un área convexa. Luego, dicha área convexa se utiliza como máscara sobre la imagen original en el espacio YIQ, produciendo un "blurring"localizado en el área convexa determinada por la zona en la que aparecen los caracteres (ver Fig. 3). En la Fig. 4 se aprecia un cuadro del video procesado por dicha programación. Como se aprecia, la eliminación es bastante eficiente y adecuada, y el deterioro de la imágen es mínimo. Este proceso es lo suficientemente robusto como para eliminar caracteres en una gran cantidad de imágenes totalmente diferentes (ver Fig. 5).

5. Conclusiones y trabajo futuro

Se presentó el sistema ITK para la gestión de imágenes basada en la construcción interactiva por medio de una biblioteca de procesos. Este sistema se basa en manejar la tubería

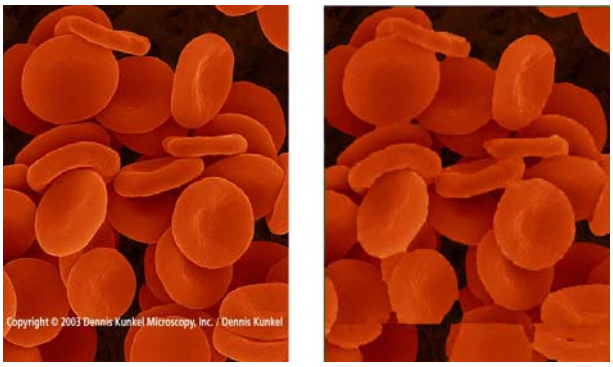

Figura 5: Una imagen cualquiera procesada con la programación de la Fig. 3.

de procesos por medio de una interfase gráfica. Esta forma de implementar el PDI permite manipular con mayor facilidad cada paso del procesamiento, configurar la programación de manera más flexible, reutilizar un procesamiento con varias imágenes, etc. Se mostraron varios ejemplos de procesamiento donde se ilustra la versatilidad y funcionalidad del sistema. Con respecto a trabajos futuros, estamos trabajando actualmente en la posibilidad de elaborar *módulos*, es decir, grupos de bloques interconectados que funcionarían como un único bloque. Esto permitiría un diseño más estructurado de procesos, y facilitaría el reuso y la portabilidad de partes de tuberías.

Referencias

[1] Claudio Delrieux, Gustavo Ramoscelli, Leonardo Arlenghi, and Alejandro Vitale. An Integrated System for Image Processing. In *Argentine Symposium on Computing Technology*, pages 114–125, ISSN 1666-1095, 2002.

[2] Claudio Delrieux, Gustavo Ramoscelli, Leonardo Arlenghi, and Alejandro Vitale. Image Processing Spreadsheet. ISBN: 0-8194-4485-5, 2002.

[3] Claudio Delrieux, Gustavo Ramoscelli, Leonardo Arlenghi, and Alejandro Vitale. The PDICalc Image Processing System. In *Proceedings of the CISST 2002 Conference*, pages 544–551, ISBN 1-892512-94-7, 2002.

[4] B. Dougharte and P. Giardino. *Matrix Structured Image Processing.* Prentice-Hall, Cambridge, MA, 1991.

[5] Andrew Glassner. *Principles of Digital Image Synthesis.* Morgan Kaufman, San Francisco, 1995.

[6] Jonas Gomes and Luiz Velho. *Image Processing for Computer Graphics.* Springer, New York, 1997.

[7] Rafael González and Richard Woods. *Digital Image Processing.* Addison-Wesley, Wilmington, USA, 1996.

[8] Arie Roszenfeld. *Digital Picture Processing.* MIT Press, Cambridge, MA, 1995.