PDICalc: Una Planilla de Cálculo para el Procesamiento Digital de Imágenes

L. Arlenghi, A. Vitale, C. Delrieux y G. Ramoscelli
Departamento de Ingeniería Eléctrica
Universidad Nacional del Sur
Alem 1253 - (8000) Bahía Blanca - ARGENTINA - e-mail:claudio@acm.org

Resumen

El procesamiento digital de imágenes permite gestionar datos e información valiosa a partir de imágenes obtenidas en los contextos más diversos. Uno de los problemas más comunes que ocurre cuando se utiliza el procesamiento de imágenes, es que éste incluye un conjunto heterogéneo de técnicas (histogramas, filtrado, morfología, etc.) entre las cuales es necesario elegir el más conveniente por prueba y error y/o manipulando los parámetros del procesamiento. A ésto se agrega que ciertas gestiones de imágenes requieren la aplicación sucesiva de varios pasos diferentes, lo cual complica aún más el proceso iterativo de diseñar un procesamiento específico. Muchas veces estas complejas cadenas de procesos deben aplicarse a más de una imagen, lo cual resulta tedioso pues es necesario repetir cada uno de los pasos con los mismos parámetros.

En este trabajo presentamos el sistema PDICalc*, el cual permite manejar imágenes como si fuesen celdas de una planilla de cálculo, agrupando y secuenciando las distintas etapas del procesamiento de una forma intuitiva, flexible y reutilizable. Esta forma de proceder simplifica enormemente la gestión de imágenes procesadas, facilita el diseño iterativo de procesamiento, y permite reutilizar una cadena completa de procesos o cualquiera de sus partes.*

Palabras Clave: Procesamiento de Imágenes. Visualización. Planilla de Cálculo.

1 Introducción

El Procesamiento Digital de Imágenes (PDI) es un área de creciente importancia tecnológica. Su objetivo general consiste en manipular señales bidimensionales (imágenes) para mejorar alguna de sus características. Por ejemplo, procesar datos adquiridos satelitalmente para mejorar la percepción, detección o interpretación de algún patrón específico [6, 7]; aplicar filtrados a imágenes fotográficas para reconstruir o retocar sus características

visuales [4, 5]; o comprimir información gráfica para facilitar su transporte por las redes de comunicaciones [8, 2]. El auge de las técnicas del PDI se debe, precisamente, a que actualmente todos estos medios tecnológicos están experimentando una rápida popularidad. El PDI se realiza por medio de procedimientos expresados en la forma de algoritmos. Por dicha razón es que la mayor parte de las funciones de procesamiento pueden implementarse en software [3, 9].

Uno de los problemas más comunes que ocurre cuando se utiliza el PDI, es que éste incluye un conjunto heterogéneo de técnicas (histogramas, filtrado, morfología, etc.) entre las cuales es necesario elegir el más conveniente por prueba y error. Podemos decir que aplicar PDI consiste en diseñar en forma iterativa una secuencia de procesos, eligiendo los mismos de acuerdo a su efecto sobre las imágenes intermedias, y manipulando los parámetros en forma conveniente. De esta manera se trabaja con la mayoría de los sistemas de procesamiento que acompañan a diversos aparatos o instrumentos científicos (microscopios electrónicos, gestores de imágenes satelitales, procesadores de imágenes médicas, etc.). Muchas veces, sin embargo, 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 llevan a pensar en 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. Una de los sistemas de gestión de información que parece adecuada, por su fácil utilización y su popularidad, es la planilla de cálculo. La idea distintiva de la planilla de cálculo es que el área de trabajo está compuesta por varias celdas, cada una de las cuales porta un elemento de información, y el contenido de cada celda es estático o bien se computa a partir de la información contenida en otras celdas. En este trabajo presentamos el sistema PDICalc, el cual permite manejar imágenes como si fuesen celdas de una planilla de cálculo. Esto permite acceder a cada paso del procesamiento y modificar los parámetros de cada paso individualmente, sin tener que repetir todo el proceso. También permite 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 Un ejemplo particular

Para ilustrar la versatilidad de nuestro sistema, supongamos por ejemplo que se requiere el diseño de un filtro de segmentación para un hipotético sistema robótico. El mismo recibe como entrada imágenes de una *motherboard*, dentro de la cual debe ubicar la posición de un determinado circuito integrado, identificado por medio de su código, para realizar algún tipo de determinación o manipulación. Una imagen de entrada para este sistema puede tener la apariencia de la Fig. 1. La imagen presenta dos tipos diferentes de dificultades superpuestas. Por un lado tiene ruido, el cual se puede originar por varias causas imposibles de evitar (ruido de cuantización, ruido térmico inherente a la cámara digital, malas condiciones de iluminación, etc.) [10]. Por otro lado, la impresión de los caracteres

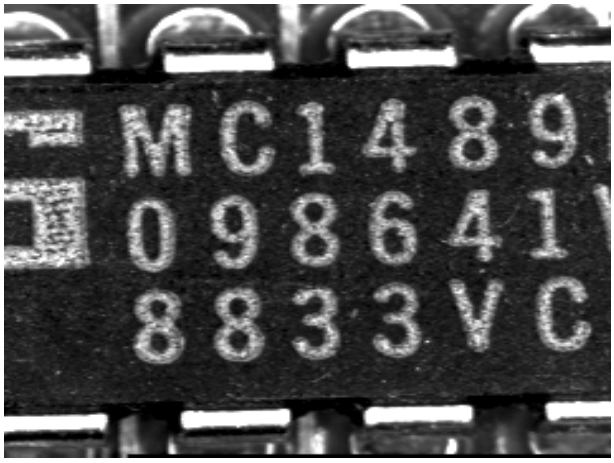

Figura 1: Imagen de entrada para un hipotético sistema robótico de reconocimiento (tomada de [1]).

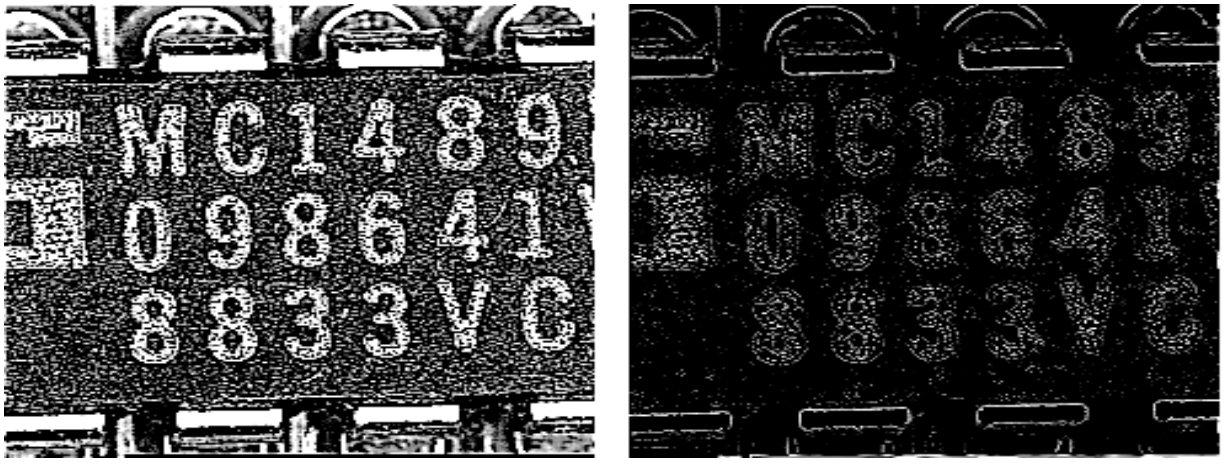

Figura 2: Imagen de la Fig. 1 (a) con filtrado de gradiente de Roberts y (b) con filtrado Laplaciano.

que identifican al integrado ha sido imperfecta, lo cual también debe considerarse para obtener un sistema robusto.

La estrategia para reconocer el integrado consiste en segmentar una representación poligonizada de su identificación (en este caso las letras `MC1489`), de manera que un sistema basado en reglas pueda reconocer los caracteres e identificar al componente. Un filtrado pasaaltos (tanto de gradiente como de segundo orden) previo al algoritmo de segmentación, sería de escasa utilidad, porque las imperfecciones en la pintura y el ruido en la imagen producirían resultados inadecuados (ver Fig. 2). Por lo tanto, es necesario plantear una estrategia de procesamiento más compleja.

Aplicar filtrado pasabajos para reducir el ruido queda descartado, porque la supresión de ruido producirá como efecto indeseado el borroneo de los bordes de los caracteres que queremos reconocer. Otra posibilidad para explorar es aplicar algún filtrado morfológico para restaurar la forma correcta de los caracteres. Esto implica un paso previo de bi-

Figura 3: Imagen de la Fig. 1 binarizada (a) con umbral demasiado cercano al blanco y (b) umbral demasiado cercano al negro.

Figura 4: Imágenes de la Fig. 3 aplicándole respectivamente apertura y cierre morfológico.

narización de la imagen, con un valor umbral adecuado. Si nuestra estrategia es aplicar aperturas (opening) a la imagen binarizada, es preferible umbralizar con un valor cercano al blanco. A la inversa, si buscamos corregir los defectos por medio de la aplicación de cierres (closing), entonces se prefiere umbralizar cercano al negro (ver Fig. 3). Sin duda, como no podemos anticipar el efecto del procesamiento morfológico, lo ideal será explorar ambas posibilidades en paralelo.

En la Fig. 4 vemos el efecto de aplicar (respectivamente) apertura morfológica a la imagen umbralizada con valor cercano al blanco, y cierre morfológico a la umbralizada cerca del negro. Aparentemente el primer caso produce mejores resultados durante la segmentación de aristas (ver Fig. 5). Todos estos pasos ilustran la naturaleza interactiva y por prueba y error del PDI. Cada caso particular implica una exploración previa de posibilidades, y una elección inadecuada en un paso previo implica tener que recorrer todo el camino nuevamente.

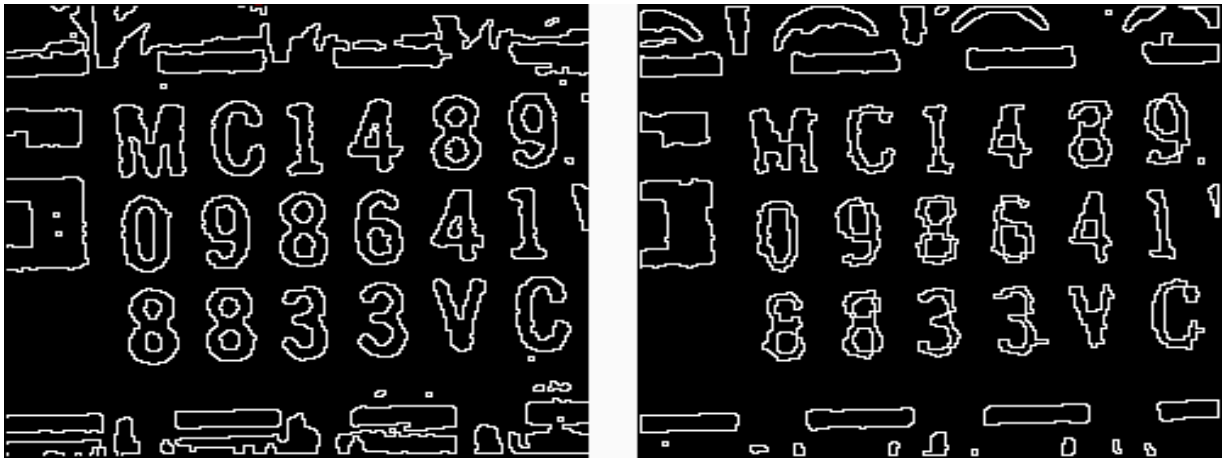

Figura 5: Segmentación de aristas en las imágenes de la Fig. 4.

Este ejemplo es uno de tantos, en los cuales para obtener el resultado deseado es necesario aplicar varios pasos de procesamiento, cada uno con parámetros que puede ser necesario modificar para mejorar los resultados. Con imágenes similares, es necesario seguir los mismos pasos, pero modificando algún parámetro de procesamiento. En nuestro ejemplo, si la imagen de entrada es un poco más oscura, es necesario modificar el valor umbral durante la binarización. Esta situación es característica en muchas ramas de la ciencia donde se utiliza el PDI (imágenes satelitales, medicina, microscopía electrónica, etc.) donde a un conjunto de imágenes similares es necesario aplicarle un mismo tipo de procesamiento. Con los sistemas comerciales esto involucra repetir manualmente cada uno de los pasos, lo cual es evidentemente tedioso e inconveniente.

3 El sistema PDICalc

Para solucionar estas dificultades, en este trabajo proponemos una manera diferente de abordar este problema, y de dotar de versatilidad a los sistemas de PDI. La idea principal es sencilla pero efectiva: concebir al PDI como una planilla de cálculo, cuyas celdas están ocupadas por imágenes. Esto da origen al sistema PDICalc, el cual es, efectivamente, una planilla de cálculo para PDI. 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 de PDICalc es semejante en su operación a una planilla común, de manera que cualquier usuario se sienta cómodo con su uso. El editor de expresiones se activa cada vez que se "pincha" una celda, y permite programar el contenido de la misma. Por ejemplo `celda3=celda1+celda2` expresa que la imagen en la celda 3 se obtiene computando pixel por pixel, en el espacio cromático RGB, la suma de los pixels de las celdas 1 y 2, ver Fig. 6.

Esta expresión también puede obtenerse "pinchando" la celda con el botón derecho, opción que despliega un menú en el cual se permite modificar el origen de una celda (archivo, ecuación computada a partir de otras celdas, valor constante, etc.), editar los

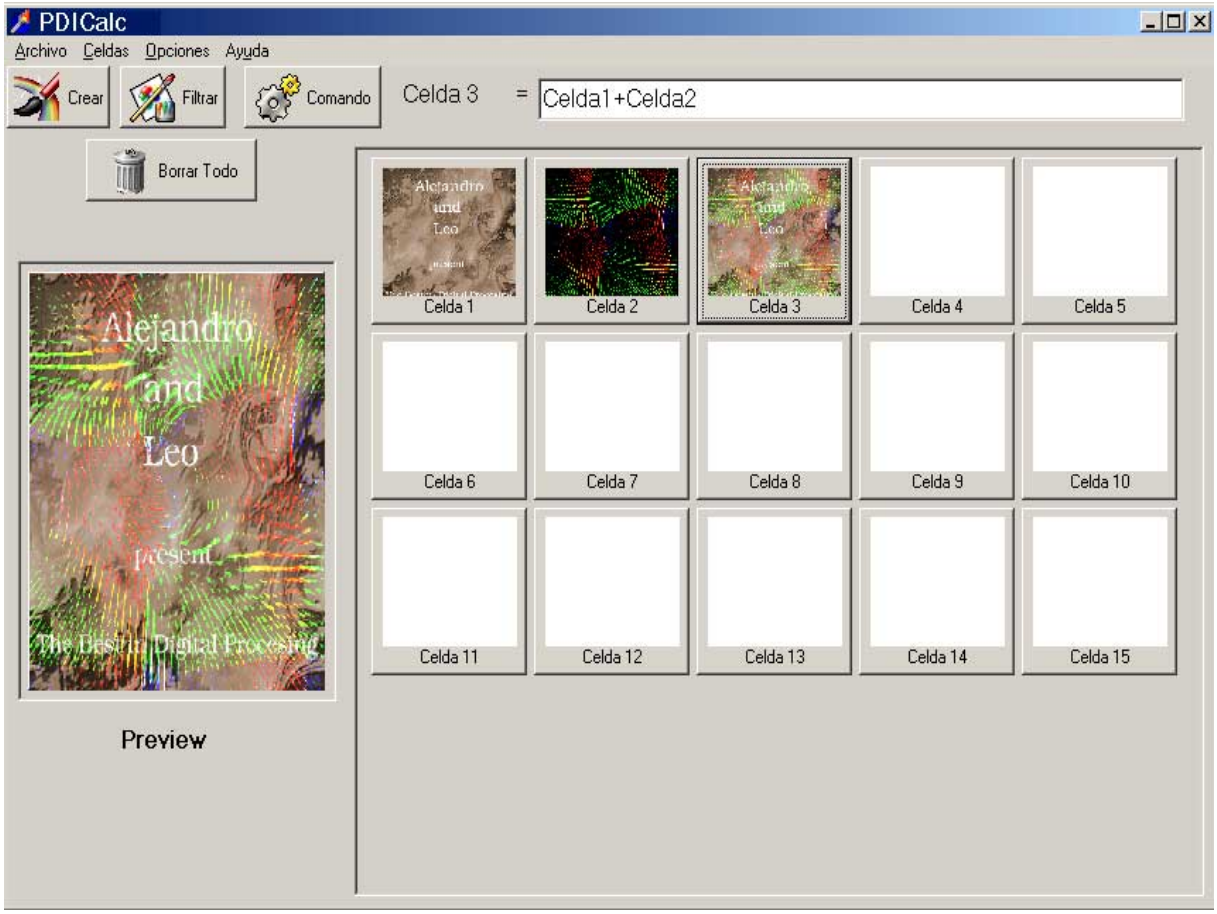

Figura 6: Cómputo de una imagen en una celda como suma de otras dos celdas.

parámetros de la imagen, los filtros, y varias otras que serán descriptas a continuación (ver Fig. 7). La gramática que permite formar expresiones para computar el contenido de una celda tiene como símbolos terminales `Celdai`, y algunas constantes de imagen (imagen blanca, con un valor dado de gris, con un valor dado de RGB, etc.). Las operaciones aritméticas se interpretan como operaciones pixel a pixel en el intervalo entero [0..255] (el usuario tiene opción de decidir en cada caso de qué manera se cierra el álgebra, por ejemplo cuando un pixel se hace negativo, mayor que 255, etc.).

La gramática incluye también la operación unaria `filtrar` que denota un filtro genérico (ecualización, convolución, morfológico, etc.), de manera de poder aplicar dicho procesamiento a una celda arbitraria. Muchos de estos filtros implican una conversión previa de la imagen al espacio cromático YIQ (por ejemplo para aplicar ecualización). La gramática permite combinar varias evaluaciones en una sola expresión, por ejemplo `celda3=filtrar(celda1+filtrar(celda2-g127))` denota la operación por la cual se le resta a la imagen en la celda 2 el valor de gris 127, al resultado se le aplica un filtrado (a elegir por medio del menú del botón derecho). A esta imagen se le suma la imagen en la celda 1, y al nuevo resultado se le aplica otro filtrado. Esta forma de proceder es posible, pero siempre es aconsejable descomponer estos procesamientos complejos en varias etapas simples (en nuestro caso con la secuencia `celda4=celda2-g127, celda5=filtrar(celda4),`

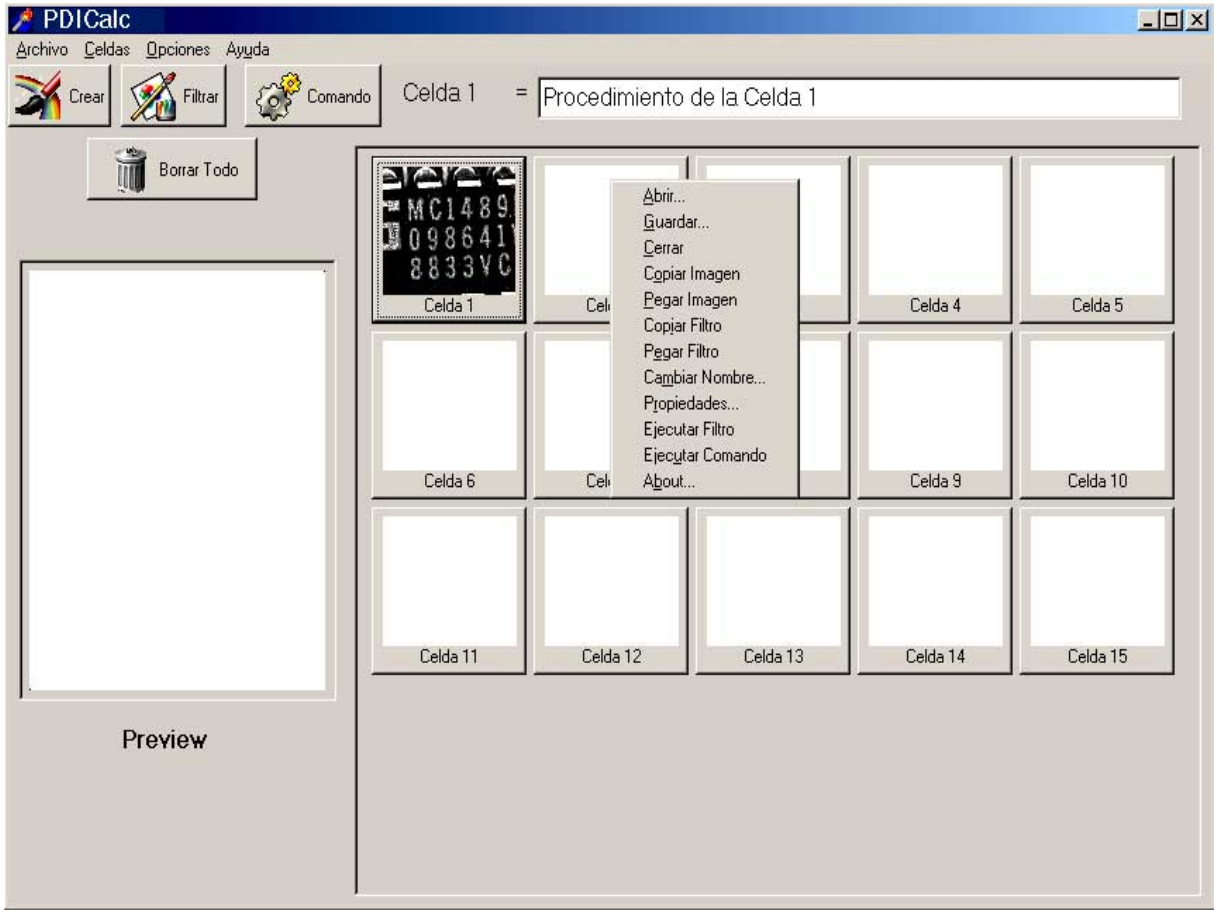

Figura 7: Opciones de edición con el botón derecho.

`celda6=celda1+celda5, celda7=filtrar(celda6)`. De esta forma, el procesamiento de imágenes como el mostrado en la Sección anterior se puede programar en muy pocos pasos, se puede acceder a modificar cada uno de los parámetros de cada filtro, se puede aplicar el mismo procesamiento a cualquier imagen en cualquier momento, y se puede guardar la programación del procesamiento en un archivo para continuar trabajando luego (ver Fig. 8).

La gramática permite expresiones circulares, como por ejemplo `celda1=filtrar(celda1)`. Este tipo de casos son útiles y necesarios cuando es necesario aplicar un procesamiento iterativo una cantidad indeterminada de veces (varias erosiones o varias dilataciones, por ejemplo). Por dicha razón hemos elegido que la evaluación de las expresiones sea *explícita* (por medio del botón `comando` en la GUI de la planilla) al contrario de lo que ocurre en las planillas de cálculo convencionales. De esa forma el usuario puede observar no solo el resultado de ir aplicando iterativamente estos filtros, sino el efecto que se produce cuando a la imagen obtenida se le aplica un procesamiento ulterior.

Al pinchar una celda con el botón derecho del mouse se descuelga un menú que permite, manipular el contenido de la celda de varias maneras (abrir una imagen desde un

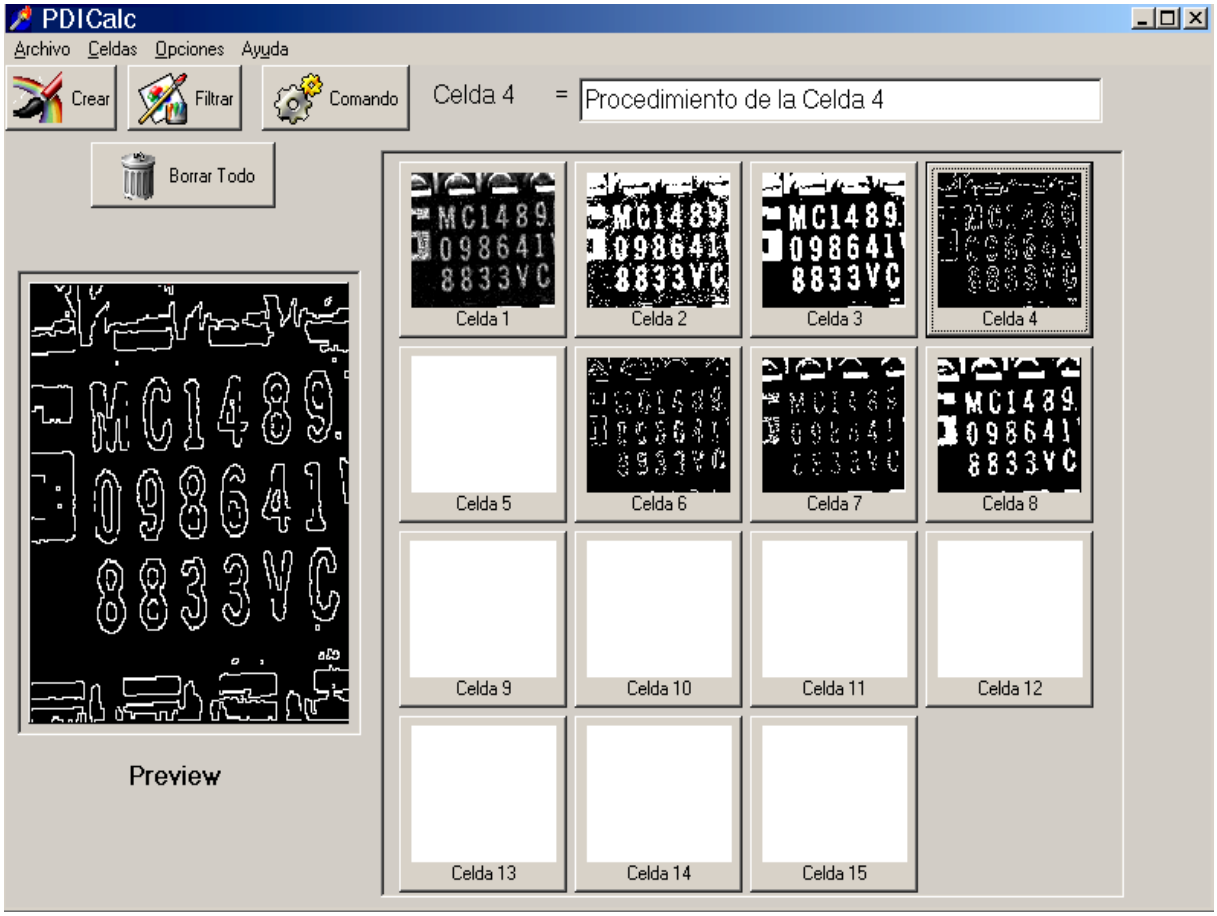

Figura 8: Programación del procesamiento ejemplificado en la Sección anterior.

archivo, o bien guardarla, copiar o pegar una imagen desde el clipboard, editar las propiedades del filtrado a aplicar, cambiar el nombre de la celda, eliminar la celda, etc.) Entre las propiedades del filtrado a aplicar es posible elegir la aplicación de filtrado por convolución, ecualización, filtrado morfológico, etc., donde cada uno de los parámetros específicos (kernel de los filtros, matrices morfológicas, etc.) se pueden elegir (ver Fig. 9). La programación de estos filtros también se puede copiar y pegar, lo cual es evidentemente de gran utilidad.

Al pinchar la celda con el botón izquierdo, el contenido de la misma se despliega en el `preview`. Pinchando sobre este lugar se despliega una ventana de `view` donde la imagen se aprecia en su tamaño real. La ventana de `view` tiene una cantidad programable de imágenes previas almacenadas, de manera de poder comparar en tamaño real varias imágenes al mismo tiempo. Por último, podemos mencionar que en el menú descolgable de archivo es posible guardar en disco como "proyecto" el estado completo de la planilla, incluyendo las imágenes contenidas en las celdas, la programación de los filtros, etc. La imagen de cada celda se guarda como archivo .bmp, y el estado de los filtros se guarda en un formato propietario de pequeño tamaño.

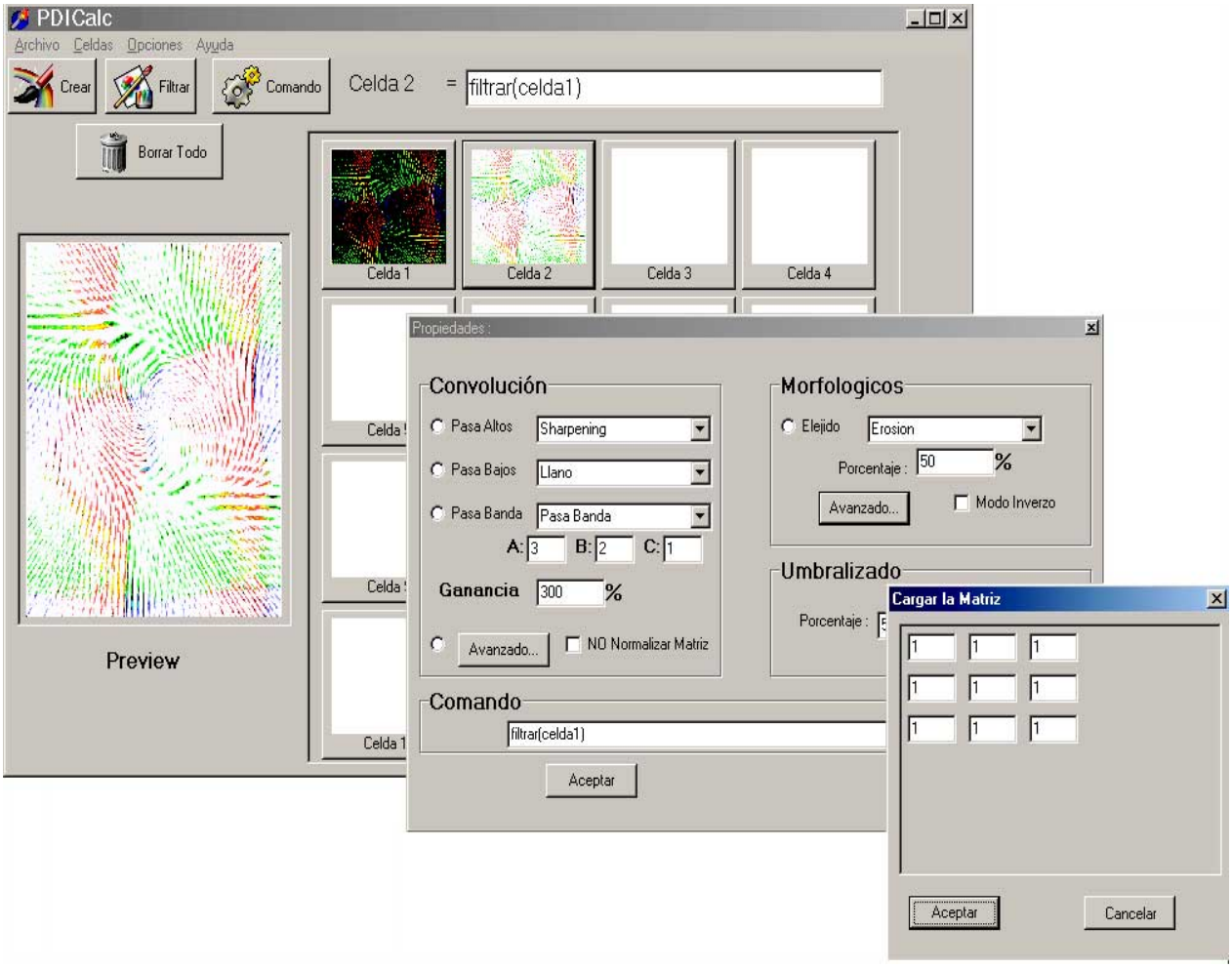

Figura 9: Menú de programación de los filtros.

4 Procesamiento avanzado

Muchos problemas de PDI requieren técnicas más avanzadas que la simple aplicación de filtros. Un caso particular muy importante es el procesamiento espectral de la imagen, normalmente manipulando la transformada rápida de Fourier (FFT) de la imagen. Por ejemplo, si la imagen está deteriorada por una señal no ruidosa (periódica) no limitada en banda, entonces es imposible recuperar la imagen original por filtrado. Sin embargo, en la FFT de la imagen normalmente es posible localizar la componente espectral de la señal indeseada. Esto permitiría reinstaurar la señal original por medio del "enmascaramiento" del especto de la señal indeseada, llevando a 0 los valores de energía correspondientes, y realizando la FFT inversa. Este procedimiento está ilustrado en la Fig. 10, donde se observa una imagen aérea con ruido periódico introducido por interferencia radial (celda 1). En la celda 2 se ha computado la FFT de dicha imagen, donde se observan claramente los dos puntos simétricos producidos por la señal de interferencia. En la celda 6 la zona aledaña a dichos puntos ha sido removida por medio de un enmascaramiento, lo cual produce, por medio de la transformada inversa, la imagen restaurada en la celda 7. Por

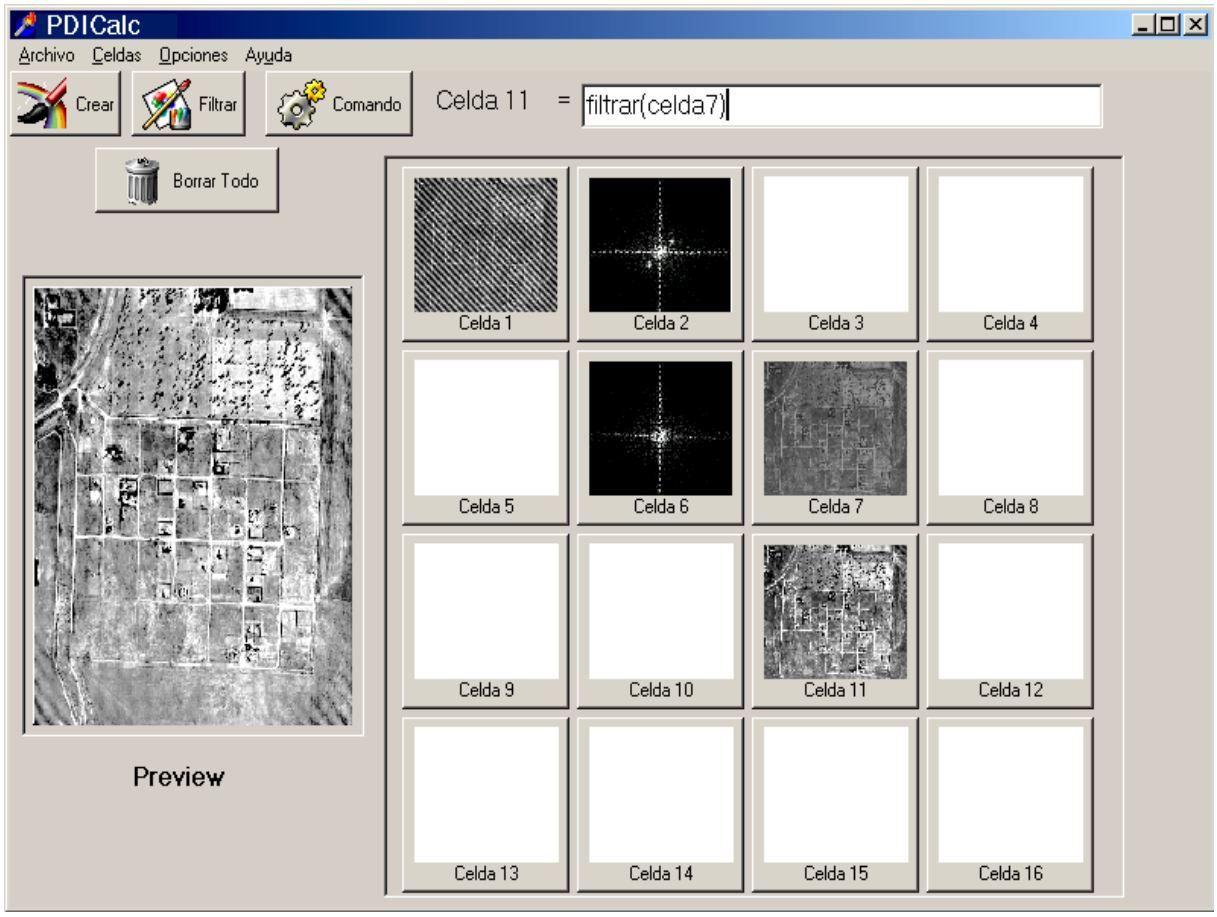

Figura 10: Filtrado por enmascaramiento espectral.

último, en la celda 11 y en el área de preview se observa la imagen final, obtenida por medio de una ecualización de la imagen de la celda 7.

Este tipo de procesamiento puede ser de gran utilidad en situaciones donde se desean hacer mediciones específicas. En la Fig. 11 podemos ver una secuencia similar de pasos con una fotografía de microscopio electrónico, donde se desea medir la cantidad de grafito (puntos negros) sobre una superficie metálica pulida, pulimiento que produce el patrón de rayas que dificulta la medición, tal como se observa en la imagen de la celda 1, proveniente directamente del microscopio. Los pasos dados en las celdas 2, 6, 7 y 11 son similares a la Fig. 10 (FFT, enmascaramiento, FFT inversa), con la diferencia de que el enmascaramiento es un poco más trabajoso, dado que el patrón de frecuencias de las rayas no es limitado en banda. Por último, se requiere un paso adicional de umbralizado (celda 12) en el cual las partículas de grafito se diferencian perfectamente del fondo, y luego su área se puede medir directamente contando la cantidad de pixels.

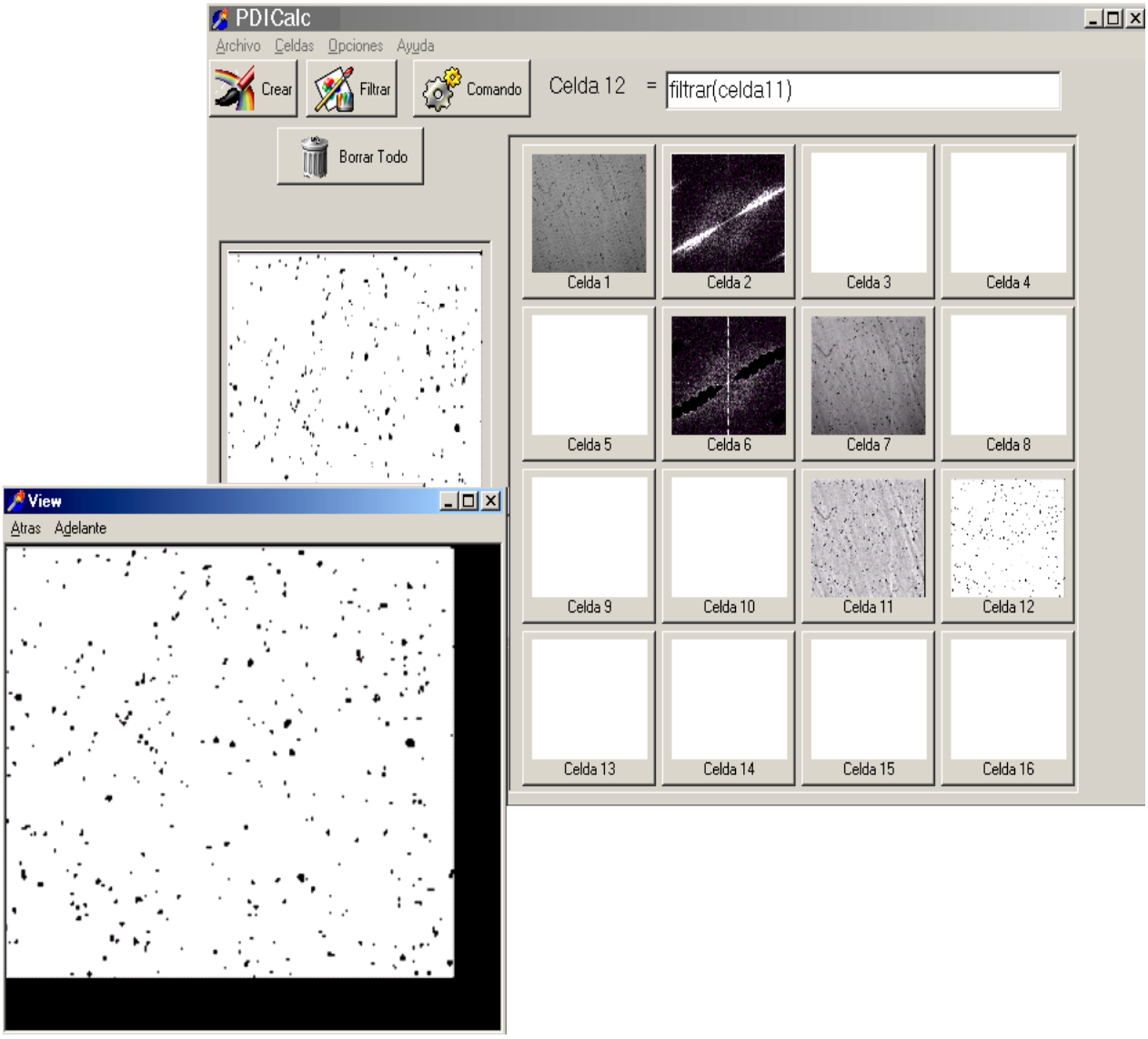

Figura 11: Filtrado por enmascaramiento espectral de una fotografía de microscopio electrónico (gentileza J. Insausti).

5 Conclusiones y trabajo futuro

Se presentó un sistema de procesamiento de imágenes cuyo funcionamiento es similar a una planilla de cálculo. 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. Si bien se han implementado la mayoría de las operaciones de filtrado que puedan necesitarse, el sistema se puede extender arbitrariamente por medio de la programación de la componente de software adecuada.

Entre los trabajos futuros, estamos considerando la posibilidad de manejar imágenes satelitales de tipo Tematic Mapper (6 bandas) y multiespectrales, a través de funciones de pseudocoloring adecuadas. También fue observado por usuarios que sería importante poder administrar varios proyectos en forma simultánea. Si bien este objetivo se puede lograr ejecutando varias instancias del PDICalc al mismo tiempo, una solución más práctica consistirá en un futuro en que el sistema cuente con varias "solapas", en cada una de las cuales administra un proyecto por separado, pudiendo de esa forma intercambiar información entre varios proyectos a partir de comandos especialmente introducidos.

Referencias

[1] Gregory A. Baxes. *Digital Image Processing.* John Wiley and Sons, New York, 1994.

[2] K. Castleman. *Digital Image Processing.* Prentice-Hall, New York, 1989.

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

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

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

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

[7] Anil Jain. *Fundamentals of Digital Image Processing.* Prentice-Hall, Cambridge, 1996.

[8] Jae Lin. *2D Signal and Image Processing.* Prentice-Hall, Cambridge, 1991.

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

[10] J. C. Russ. *The Image Processing Handbook.* CRC Press, Boca Raton, FL, 1989.