JÓVENES EN LA CIENCIA
REVISTA DE DIVULGACIÓN CIENTÍFICA

OPTIMIZACIÓN DE UNA ARQUITECTURA EN FPGA PARA FILTROS GAUSSIANOS EN IMÁGENES

Loza Pérez, Narvid Jaziel (1), Rodriguez Doñate, Carlos (2)

1 Licenciatura en Ingeniería en Mecatrónica, Universidad de Guanajuato | nj.lozaperez@ugto.mx

2 Departamento de estudios multidisciplinarios, División de ingenierías, Campus Irapuato-Salamanca, Universidad de Guanajuato | c.rodriguezdonate@ugto.mx

Resumen

En este documento, se propone una arquitectura en FPGA optimizada para filtrado espacial gaussiano en imágenes, donde se aprovechan las propiedades distributivas de las funciones gaussianas para optimizar el consumo de recursos del FPGA. En base a los resultados obtenidos, se logró obtener un tiempo de procesamiento de 197 fps para imágenes de 640 X480 pixeles y el consumo de hardware del FPGA utilizado fueron menores al 6%.

Abstract

In this paper, we proposed optimized architecture in FPGA for Gaussian spatial filtering in images, where the distributional properties of the Gaussian functions are used to optimize resource consumption of FPGA, Based on the results obtained, it is possible to time processing of 197 fps for images of 640 X480 pixels and consumption of FPGA hardware used were lower than 6%.

Palabras Clave Filtrado Espacial, mascara o kernel, procesamiento en tiempo real, VHDL

INTRODUCCIÓN

Hoy día, la visión artificial tiene la tarea de procesar y entender una imagen o secuencia de imágenes para asemejar la visión humana. Para este objetivo, la visión artificial tiene como elemento fundamental las cámaras digital , la cual tiene la función captar la luminiscencia de cada punto en una escena como un sensor formado por una matriz puntual de sensores más pequeños, tal como lo haría el ojo humano, la cámara a su vez transformar los niveles de voltaje que recibe en una señal digital, todo este proceso incluirá errores tanto en la percepción de los sensores, como en la transformación de analógico-digital, estos errores se acumulan para formar ruido en la imagen, el cual es visible una vez que se despliegan sobre una pantalla. De aquí la necesidad de dar un tratamiento a esta organización de datos dentro de la matriz de la imagen, para interpolar el valor de cada pixel con ruido a uno más próximo al real de forma matemática, esto se denominado procesamiento digital de imágenes, este procesamiento se lleva a cabo mediante diferentes técnicas, como lo son el filtrado espacial y filtrado puntual.

La reducción de ruido en procesamiento de imágenes ha tomado gran relevancia en áreas como medicina [1-2], ya que este ruido en las imágenes puede ocasionar un diagnóstico erróneo, este procesamiento puede también ayudar a las maquinas a entender posiciones, velocidad y aceleración, en aplicaciones como robótica [3] y procesos industriales [4], la detección de bordes y entre otras muchas aplicaciones [1-5]. Este tipo de aplicaciones requieren un procesamiento en tiempo real, en la literatura se han desarrollado varios estudios para evaluar las plataformas para implementar los algoritmos de procesamiento digital de imágenes como es el caso en [6] donde los FPGA (Arreglo de compuestas programables en campo, *field programmable gate array*), han destacado ya que estos dispositivos ofrecen alto grado de procesamiento en paralelo y pipeline, además de trabajar a una alta frecuencia de operación.

En este trabajo se presenta una arquitectura basada en una FPGA, para el filtrado espacial en imágenes en tiempo real utilizando funciones gaussianas. Donde a partir de las propiedades distributivas de este tipo de filtros se propone la optimización del diseño para reducir el consumo de hardware del FPGA. En base a los resultados se obtuvo un tiempo de procesamiento de 187 fps (cuadros por segundo) para una imagen de 640 X 480 pixeles, y el hardware utilizado del FPGA fue menos del 2% y 6% de las unidades lógicas y multiplicadores, respectivamente.

Fundamento matemático

El filtrado espacial utiliza una máscara, también llamada kernel, la cual es una matriz $n \times n$, donde n se define matemáticamente por (1), esto para tener siempre un pixel central rodeado por la misma cantidad de vecinos.

$$n = 2*m+1; \quad m = 1,2,3,4,\ldots \qquad (1)$$

Los coeficientes de la máscara varían según el objetivo que se pretenda alcanzar, ya sea para eliminar ruidos específicos, resaltar caracterizas u obtener determinada información. En (2) se presenta una máscara de 3x3, donde el coeficiente $C_{(0,0)}$, multiplica al pixel central, y aquellas que lo rodean serán la vecindad, este pixel se ve influenciado por sus vecinos según el valor de los demás coeficientes dentro de la máscara.

$$k(\mathrm{i},\mathrm{j}) = \begin{vmatrix} C_{(-1,1)} & C_{(0,1)} & C_{(1,1)} \\ C_{(-1,0)} & C_{(0,0)} & C_{(1,0)} \\ C_{(-1,-1)} & C_{(0,-1)} & C_{(1,-1)} \end{vmatrix} \qquad (2)$$

El proceso de filtrado en imágenes se realiza por medio de la convolución en 2 dimensiones, donde la máscara hace un barrido sobre toda la imagen para procesar cada pixel. Esta operación se define por (3), donde $k(\mathrm{i},\mathrm{j})$ es la máscara, $\mathrm{f}(\mathrm{x},\mathrm{y})$ la imagen original y $g(\mathrm{x},\mathrm{y})$ la imagen procesada,

$$g(\mathrm{x},\mathrm{y}) = \sum_{i=-a}^{a} \sum_{j=-b}^{b} k(\mathrm{i},\mathrm{j}) * \mathrm{f}(\mathrm{x}+\mathrm{i},\mathrm{y}+j) \qquad (3)$$

En este proyecto se utilizará una distribución gaussiana bidimensional definida por (4), en la que el pixel central y aquellos que estén más cercanos a este tendrán una mayor influencia en la salida del

pixel procesado, esta influencia está definida por σ que representa la desviación estándar de la distribución, una máscara de $m = 2$ y $\sigma = 1.3$ se muestra en la imagen1.

$$D(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad (4)$$

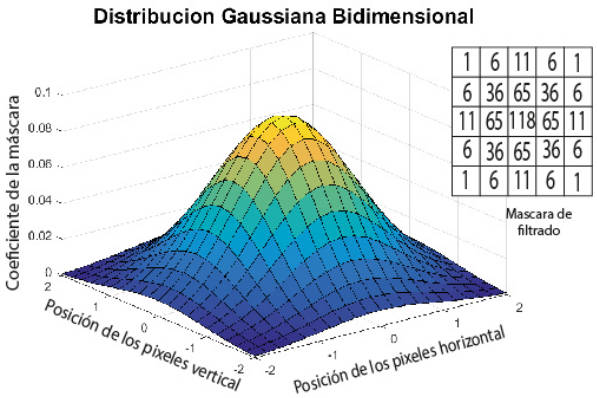

IMAGEN 1: Distribución gaussiana bidimensional diseñada para una máscara de filtrado espacial.

MATERIALES Y MÉTODOS

Para el desarrollo del proyecto se utilizó la arquitectura del sistema embebido presentado en [7] el cual contiene: un controlador de la cámara para capturar la imagen, un decodificador de Bayer quien reconstruye la captura a una matriz de colores (R,G,B), un controlador de puertos a la SDRAM, un controlador de la LCD/VGA y un módulo de control que se comunica mediante el protocolo RS-232 quien recibe y modifica los parámetros del sistema, dentro de este se integró el bloque de procesamiento digital, la arquitectura diseñada. El diagrama de la figura 2 muestra la interconexión entre cada uno de los bloques del sistema embebido.

La metodología consiste en el diseño del bloque de procesamiento espacial que se muestra en la imagen 3, el cual contiene dos módulos: Ordenador de Pixeles y el Filtro tipo gaussiano, lo que representa las dos etapas principales del procesamiento.

El Ordenador de Pixeles, se conforma de un arreglo de siete buffers que reciben y almacenan siete reglones de la imagen, esto para presentarlos a la salida de manera paralela uno a la vez para la siguiente fase del proceso, la bandera de Pix_VA, se activa para indicar que se han organizado y ordenado los pixeles (Pix_1A, Pix_2A, Pix_3A, Pix_4A, Pix_5A, Pix_6A, Pix_7A) y están lista para ser procesados.

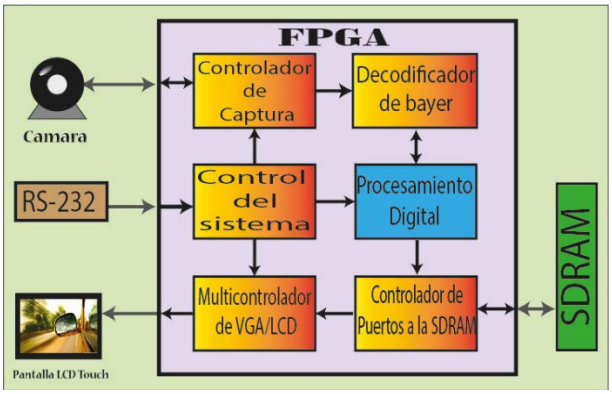

IMAGEN 2: Esquema general del sistema embebido en la FPGA.

IMAGEN 3: Diagrama general del bloque de procesamiento espacial, con sus módulos y señales internas.

El bloque de filtro tipo gaussiana como ya se mencionó anteriormente se basa en realizar la convolución de la imagen y una máscara con una distribución tipo gaussiana, si observamos la máscara que se presenta en la imagen1, es posible notar que los coeficientes se repiten en posiciones específicas, sin importar como varié la desviación estándar o el tamaño de la matriz esta secuencia seguí el mismo patrón.

El número de multiplicadores necesarios para realizar esta operación en paralelo, sería equivalente al tamaño de la matriz, es decir, $n \times n$, mientras tanto al utilizar la propiedad distributiva de la suma, es posible reducir estos multiplicadores en una cantidad mucho más pequeña que se rige (5).

$$N = \sum_{s=1}^{m+1} s = \frac{(m+1)(m+2)}{2} \qquad (5)$$

El filtro tipo gaussiano se encarga de realizar esta convolución entre la máscara y la imagen, su diseño esquemático se muestra en la imagen 3, en la primera etapa del proceso se realiza la suma de cuatro sumandos de todos los pixeles que tienen el mismo coeficiente, la segunda etapa realiza una segunda suma sobre aquellos grupos de 8 pixeles que comparten el mismo coeficiente, además se realizan las multiplicaciones. Cabe destacar que cada multiplicador de la arquitectura diseñada es utilizado para realizar dos operaciones ya que trabaja cada medio ciclo de reloj da un resultado, con lo que se logra reducir de forma considerable los recursos del FPGA, la última etapa realiza la suma total de los resultados en 3 ciclos de reloj, la programación se basa en pipeline ya que es necesario esperar 16 disparos positivos del reloj, para comenzar a obtener una respuesta de forma consecutiva.

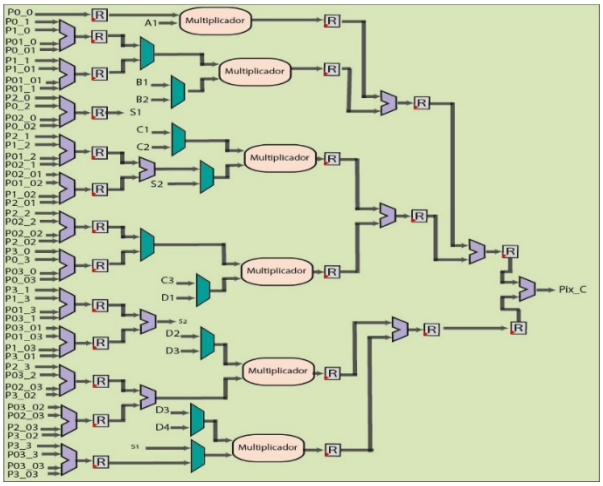

IMAGEN 4: Diagrama esquemático del bloque de filtro tipo gaussiano

RESULTADOS Y DISCUSIÓN

La puesta del experimento para validar esta propuesta se presenta en la imagen 5, donde se utilizó la cámara CMOS TRDB-D5M [8], es un sensor de 5 megapíxeles (2752 columnas, por 2004 renglones) con una frecuencia de 96 MHZ, y 12 bits por pixel y el kit de desarrollo DE1-Soc [10], que tiene el FPGA Cyclone V 5CSEMA5F31C6 y una SDRAM de 64 MB. En base a la síntesis del diseño, el consumo de hardware de la propuesta es de 1924 elementos lógicos, 258048 bits de memoria embebida y 18 multiplicadores embebidos, lo que implica que se realizó un diseño de muy bajo consumo de hardware. Para el FPGA utilizado representan el 6%, 6% y 21% de los elementos lógicos, memoria embebida y multiplicadores embebidos respectivamente.

IMAGEN 5: Dispositivos requeridos por el sistema embebido

Otro resultado que cabe destacar es el tiempo de procesamiento del diseño, ya que supera el estándar de procesamiento en tiempo real (30 fps). En la tabla 1 se presentan el tamaño de imágenes y su respectivo tiempo de procesamiento en fps.

Tabla 1: Desempeño del núcleo IP para diferentes tamaños de Imágenes

Tamaño de imagen	Fps (Frame per Seconds)
1. 256 X 256	924
2. 640 X 480	197
3. 800 X 480	159
4. 800 X 600	127
5. 1024 X 1024	58

Finalmente, la imagen 6 muestra el resultado de la arquitectura procesando en tiempo real la imagen que es enviada por la cámara, donde se puede apreciar cómo cambia la nitidez de las imágenes de manera inversamente proporcional a la variación de la desviación estándar del filtro gaussiano.

IMAGEN 6: La imagen superior izquierda es la escena original sin el filtro, la superior derecha es con una desviación de 1.3, la inferior izquierda con una de 2 y la inferior derecha con una de 3.3

CONCLUSIONES

Con este proyecto se logró obtener un núcleo IP portable, basado en un FPGA, que realiza un procesamiento de filtrado espacial gaussiano sobre imágenes, usando una máscara de filtrado de tamaño 7×7, cuyos coeficientes pueden ser modificados y en base a las pruebas realizadas sobre el mismo, se alcanzó una velocidad de 197 fps en imágenes VGA, por ello esta unidad de filtrado gaussiano puede ser utilizado en un procesamiento en tiempo real. Referente al consumo del hardware se logró diseñar un sistema optimizado con bajo consumo de recursos en hardware, ya que se utilizó el 6% de los elementos lógicos, 6% memoria embebida y 21% de multiplicadores embebidos, todo al considerar el tamaño de la máscara requerido para la convolución.

AGRADECIMIENTOS

Dirección de Apoyo a la Investigación y Posgrado (DAIP) de la Universidad de Guanajuato, y a Altera University Program.

REFERENCIAS

[1] Rabeh Amira, B., Faouzi, B., Hamid, A. & Bouaziz, M. (2014). Computer-assisted diagnosis of Alzheimer's disease. Imagen Processing, Application and Systems Conference (IPAS), 1-5.

[2] Olfa, M & Nawres, K.(2014). Ultrasound image denoising using a combination of bilateral filtering and stationary wavelet transform. Imagen Processing, Application and Systems Conference (IPAS), 1-5.

[3] Shaikh, A., Khaladkar, G., Jage, J. & Pathak, T. (2013). Robotic arm movements wirelessly synchronized with human arm movements using real time image processing. India Educators' Conference(TIIEC), 277 – 284.

[4] Maeder, A., Bistry, H.& Zhang J. (2007). Towards intelligent autonomous vision system -smart image processing for robotic application -. Robotics and Biomimetics, 1081 – 1086.

[5] Hanumantharaju, M. C., Ravishankar, M. & Rameshbabu, D. R. (2013). Design and FPGA implementation of an 2D Gaussian surround function with reduced on-chip memory utilization. Advances in Computing, Communications and Informatics (ICACCI), 604 – 609.

[6] Pauwels, K., Tomasi, M., Alonso, J. D., Ros, E., & Van Hulle, M. M. (2012). A comparison of FPGA and GPU for real-time phase-based optical flow, stereo, and local image features. IEEE Transactions on Computers, 61(7), 999-1012.

[7] Loza Perez, N. J. & Rodríguez Doñate, C. (2015). Diseño e implementación del control para una cámara digital CMOS mediante una pantalla táctil usando FPGA. Concyteg.

[8] Altera University Program, (2008). Terasic TRDB-D5M Hardware specification, Terasic Technologies Inc.

[9] Altera University Program, (2003). Multi-touch LCD Module User Manual, Terasic Technologies Inc.

[10] Altera University Program, (2003). DE1-Soc User Manual, Terasic Technologies Inc.