RIELAC, Vol. 41 3/2020 p. 53-65 Septiembre-Diciembre ISSN: 1815-5928

Implementación de un detector de movimiento para cámaras inteligentes sobre sistemas embebidos

Luis M. Garcés-Socarrás, Roberto Sánchez-Correa, Alejandro J. Cabrera Sarmiento, Santiago Sánchez-Solano, Piedad Brox Jiménez

RESUMEN / *ABSTRACT*

Este artículo describe la implementación sobre hardware reconfigurable de un detector de movimiento para cámaras inteligentes el cual puede ser empleado en varios campos de aplicación. El sistema propuesto detecta el movimiento en una secuencia de vídeo identificando la región de interés para reducir el tiempo de procesado de los algoritmos de análisis e identificación posteriores. Como parte de este trabajo se realizan tres módulos de detección de movimiento basados en el modelo planteado por Reichardt & Hassenstein para la detección bioinspirada de movimiento elemental, así como otros módulos auxiliares. Estos bloques han sido incorporados a la biblioteca de procesado de imágenes y vídeos *XIL XSGImgLib*, la cual permite simplificar y reducir el tiempo de diseño de las aplicaciones de procesado de imágenes y vídeos sobre los FPGA y SoC FPGA de Xilinx. Para la comprobación de los detectores se presenta una aplicación de detección de movimiento para un flujo espaciotemporal de vídeo, proveniente de un punto de control de tráfico vehicular, en un FPGA *Spartan-6 LX45*, arrojando mejoras en el tiempo de ejecución de la implementación del bloque de detección de movimiento elemental comparado con desarrollos similares reportados en la literatura consultada.

Palabras claves: Cámara inteligente, procesado de imágenes, detección de movimiento elemental, FPGA, *XIL XSGImgLib*

This paper describes the implementation of a motion detector system for traffic control's smart cameras on embedded devices that can be used in multiple application fields. The proposed system detects moving areas in video sequences, thus reducing the processing time of video analysis algorithms by selecting regions of interest. As part of this work, three motion detection modules, and auxiliary blocks, were developed based on the bio-inspired elementary motion detector presented by Reichardt & Hassenstein. These motion detection modules have been incorporated in the portfolio of modules in the XIL XSGImgLib video and image processing library, which aims to speed-up and ease visual image processing applications over Xilinx FPGAs and SoC FPGAs. To test the detectors, a spatio-temporal motion detection application has been implemented on a Spartan-6 LX45 FPGA using video sequences from a vehicular traffic control point, showing execution time improvements on the elementary motion detector block compared with similar developments reported in the consulted literature.

Key words: Smart camera, embedded image processing, elementary motion detection, FPGA-based system, XIL XSGImgLib

Implementation of a Motion Detector for smart cameras on embedded systems

1.- INTRODUCCIÓN

El procesado digital de imágenes es un campo con múltiples aplicaciones en sectores de la industria, la robótica, la salud, el transporte y la seguridad [1–5]. Muchas de estas aplicaciones emplean cámaras inteligentes, las cuales se encargan del procesado inicial de las imágenes para reducir la carga computacional y el flujo de comunicación con un centro de control.

Recibido: 25/2/2020 Aceptado: 13/7/2020

Una cámara o sensor inteligente consiste en la trasformación de una cámara convencional a la que se le añade un sistema de procesado y control el cual realiza, como principales tareas, la detección y estimación de movimiento, la localización y reconocimiento de patrones, y la extracción de características en las imágenes capturadas. A pesar de la complejidad de los algoritmos, muchas de estas tareas deben realizarse en tiempo real. Sin embargo, las estrictas restricciones en términos de consumo de potencia, dimensiones y peso, no hacen viable su implementación mediante sistemas de visión portables con arquitecturas basadas en computadoras personales convencionales [1, 2, 6–8].

Los sistemas embebidos cumplen con las restricciones anteriormente descritas, especialmente los basados en arreglos de compuertas programables (FPGA: *Field-Programmable Gates Array*) y los novedosos SoC FPGA (*System-on-Chip Field-Programmable Gates Array*) son particularmente interesantes por la capacidad para explotar el paralelismo del algoritmo sobre el hardware, la posibilidad de combinar componentes hardware y software para la implementación del mismo, la flexibilidad para interconectar diferentes dispositivos internos y externos, y la facilidad de reconfiguración para adaptar la arquitectura a una amplia variedad de aplicaciones [1, 4, 6–9].

La detección de objetos en movimiento en una secuencia de imágenes es, en muchos casos, el primer paso para la extracción de información relevante en aplicaciones de visión computacional [10–12]. Esta operación permite reducir el área de interés del fotograma, concentrando el flujo de procesado exhaustivo sobre los píxeles en movimiento.

Las diferentes técnicas para la detección de movimiento se distinguen por los métodos para identificar las diferencias entre dos fotogramas. Estos métodos abarcan, desde técnicas tan sencillas como la sustracción matemática entre un fotograma y un modelo, hasta otras más complejas, como la obtención de los gradientes del desplazamiento mediante métodos diferenciales [13]. Para seleccionar un método de detección de movimiento se deben analizar distintos factores, tales como la ubicación de los elementos de captura y las características del entorno de muestreo. También se deben considerar los requisitos de la aplicación, como la velocidad de procesado necesaria y el consumo de recursos computacionales.

Este artículo describe el diseño e implementación en FPGA de los módulos para detección de movimiento basados en el detector de movimiento elemental (EMD: *Elementary Motion Detection*) de Reichardt y Hassenstein (*R&H*) que han sido incorporados a la biblioteca de procesado de imágenes y vídeos *XIL XSGImgLib* para incrementar su funcionalidad. En la sección 2 se introducen las diferentes técnicas de detección de movimiento, así como sus principales ventajas y desventajas. La sección 3 describe la implementación del detector de movimiento elemental y sus principales variantes, mientras que en la sección 4 se ilustra la implementación de un sistema de procesado espaciotemporal de detección de movimiento para una cámara inteligente basada en un FPGA, a partir del empleo de varios bloques de la biblioteca *XIL XSGImgLib*. Por último, se presentan las conclusiones de este trabajo.

2.- DETECCIÓN DE MOVIMIENTO

Las técnicas de detección de movimiento se basan en encontrar las diferencias entre dos fotogramas de una escena, obtenidos en diferentes instantes de tiempo con un mismo sensor óptico, con el objetivo de clasificar las regiones de las imágenes en frente o fondo [10]. En el frente se asocian las regiones de píxeles que sufren cambios, indicando la ocurrencia de movimiento, mientras que el resto de los píxeles, los cuales no presentan modificaciones relevantes, se clasifican como fondo [13].

Existen tres propuestas convencionales para detectar movimiento, la sustracción de fondo, el análisis del flujo óptico y la diferenciación temporal [3, 13–17].

2.1.- TÉCNICAS DE SUSTRACCIÓN DE FONDO

Las técnicas basadas en sustracción de fondo construyen un modelo de la escena estática para luego compararlo con cada fotograma de una secuencia, discriminando las regiones de interés frente a aquellas que permanecen estáticas o presentan movimientos repetitivos (como el de las hojas de los árboles o las masas de agua) [13]. Estas técnicas son simples en general, encontrándose la mayor dificultad en la búsqueda de un modelo adecuado, estático o adaptativo, que permita darle robustez al algoritmo [11].

La implementación de algoritmos básicos de substracción de fondo sobre dispositivos de hardware reconfigurable ha sido la más difundida por su simplicidad en los métodos de detección, que permiten operación en tiempo real empleando una reducida cantidad de los recursos lógicos disponibles [1, 3, 4, 11, 14, 16, 17].

El principal inconveniente de estas técnicas es su sensibilidad a los cambios en la escena, ya sea por las variaciones de iluminación o por las condiciones ambientales. Además, no son factibles para su uso en aplicaciones con cámaras móviles ya que la actualización del modelo complica el sistema de procesado. Por último, la información que brindan estas técnicas está

orientada solamente a la presencia de diferencias entre el frente y el fondo, no permitiendo conocer ni la velocidad, ni la orientación del desplazamiento del objeto que atraviesa la escena muestreada [4].

2.2.- TÉCNICAS DE ANÁLISIS DEL FLUJO ÓPTICO

El movimiento bidimensional en una secuencia de fotogramas y su proyección en el plano de la imagen, respecto a un sensor visual, es la base de la estimación de flujo óptico. El análisis de las secuencias ordenadas en el tiempo posibilita la estimación del movimiento en la imagen obteniendo la velocidad instantánea o el desplazamiento discreto de los objetos. Este análisis permite conocer la dirección y magnitud del desplazamiento de los píxeles entre dos fotogramas [17–20].

Los métodos principales para el cálculo del flujo óptico utilizan técnicas diferenciales y técnicas de coincidencia. Ambos métodos requieren la realización de cálculos intensivos y amplios recursos computacionales, dificultando su implementación para operar en tiempo real sobre plataformas embebidas secuenciales.

2.3.- TÉCNICAS DE DIFERENCIACIÓN TEMPORAL

Las técnicas de diferenciación temporal emplean fotogramas de la escena en instantes anteriores como modelos de comparación, destacándose entre las más populares la diferencia entre fotogramas adyacentes y el análisis de la frecuencia espaciotemporal.

La primera técnica consiste en detectar variaciones entre la imagen de estudio y otras capturadas con anterioridad mediante la sustracción de las mismas. Teniendo en cuenta que los cambios entre fotogramas consecutivos en una secuencia de vídeo no presentan diferencias significativas, la selección del intervalo de tiempo entre ambas imágenes influye en la variación de los valores de la intensidad de los píxeles para detectar el movimiento en la escena [15]. Para eliminar este inconveniente se emplea un valor de umbral que ajusta la sensibilidad de la técnica, considerando el movimiento para valores del resultado de la sustracción que sobrepasen el umbral [1, 15]. Las ventajas de esta técnica con respecto a la sustracción de fondo son la posibilidad de ser empleada en cámaras en movimiento y su menor sensibilidad a los cambios de las condiciones ambientales de la escena. Sin embargo, la información que se obtiene a partir de su aplicación se limita a la ocurrencia de cambios en la escena [15].

Por otra parte, el método basado en el análisis de la frecuencia espaciotemporal utiliza un conjunto de filtros que son activados selectivamente en función del movimiento presente en la imagen [21]. El detector de movimiento elemental de Reichardt y Hassenstein, propuesto en 1956, es un ejemplo conocido de estos filtros, el cual emplea la correlación de las señales de dos fotorreceptores para obtener una respuesta sensible al desplazamiento. En aplicaciones de procesado de imágenes se utilizan conjuntos de filtros EMD alineados en la dirección del movimiento que se desea detectar, precisando de la sintonización y ajuste de los parámetros del modelo para la detección de distintos movimientos y velocidades. [21–23].

El detector de movimiento elemental de Reichardt y Hassenstein emplea un reducido número de operaciones matemáticas simples, facilitando su implementación. Este es el caso del desarrollo propuesto en [24] para la detección de movimiento elemental mediante un detector basado en el análisis de la frecuencia espaciotemporal. El sistema, implementado sobre un FPGA *Stratix EP1S4* de Altera (actualmente Intel Corp.), procesa fotogramas en tiempo real con resoluciones de 256×256 píxeles. Otra implementación de estas técnicas se presenta en [25] para estudiar modelos bioinspirados de detección de movimiento y desarrollar un sistema inteligente autónomo para diversas aplicaciones. El algoritmo de detección de movimiento se desarrolla sobre un FPGA *Stratix II S60* de Altera para una resolución de 127×100 píxeles, ocupando el 13% de los recursos lógicos y el 39% de la memoria interna del dispositivo.

Las técnicas de análisis de la frecuencia espaciotemporal permiten una estimación de la orientación y la velocidad de los objetos que atraviesan la escena a través de los filtros de movimientos, siendo factible su implementación en aplicaciones con movimientos de la cámara [24, 26].

3.- DETECCIÓN DE MOVIMIENTO ELEMENTAL

La biblioteca de procesado *XIL XSGImgLib* proporciona un conjunto de módulos hardware reconfigurables para facilitar la rápida creación de prototipos de sistemas de procesado de imágenes y vídeos sobre un FPGA desde un ambiente de desarrollo amigable e idóneo para la comprobación de los algoritmos. Esta biblioteca ha incrementado progresivamente sus funcionalidades mediante la incorporación de nuevos elementos de procesado, como módulos de cálculo de histogramas [27] o bloques de convolución genéricos que usan el procedimiento de reconfiguración automática descrito en [6] para facilitar la utilización de la biblioteca y optimizar el consumo de recursos del FPGA [28].

Los nuevos bloques para detección de movimiento que han sido incorporados en la biblioteca *XIL XSGImgLib* se basan en el detector de movimiento elemental mostrado en la Figura 1. Este dispositivo realiza el análisis de la frecuencia espaciotemporal de dos imágenes mediante el cálculo de la correlación entre los píxeles ($R(t)$) capturados por dos fotorreceptores ($P1$ y $P2$). Estos fotorreceptores se colocan a una cierta distancia Δx , activándose en instantes de tiempo diferentes (Δt) y obteniendo una medida de la velocidad del movimiento del objeto por la zona de análisis. El detector obtiene la correlación entre las señales recibidas desde los fotorreceptores ($P1$ y $P2$) empleando un filtro de primer orden (Δt en la Figura 1) para retardar las señales ($P1_d$ y $P2_d$), como muestra la expresión (1).

$$R(t) = P2 \times P1_d - P1 \times P2_d \tag{1}$$

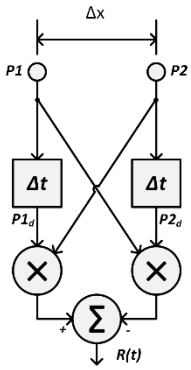


Figura 1

Detector de movimiento elemental de Reichardt y Hassenstein.

Esta correlación es máxima (Figura 2) si el tiempo que tarda una señal en moverse de un fotorreceptor a otro es igual al tiempo causado por la demora del filtro de primer orden. A velocidades de desplazamiento, frecuencias y dirección del movimiento diferentes a las parametrizadas en el modelo, la respuesta del detector se reduce [24].

Figura 2

Detección de movimiento. a) Fotograma anterior. b) Fotograma actual. c) Resultado de la correlación del detector de Reichardt y Hassenstein.

La implementación del algoritmo de detección de movimiento elemental para dispositivos programables (Figura 3) recibe los flujos de píxeles de los fotogramas por las entradas $P2$ (para el fotograma actual) y $P2_d$ (para el almacenado en memoria con una diferencia temporal Δt entre ellos). Estos flujos son demorados Δx instantes de tiempo (*Delay*$_x$) hasta la posición del otro receptor ($P1$ y $P1_d$), realizándose las operaciones matemáticas de multiplicación (*Mult*$_x$) y resta (*AddSub*) para obtener el valor de la correlación de los píxeles a la entrada ($R(t)$).

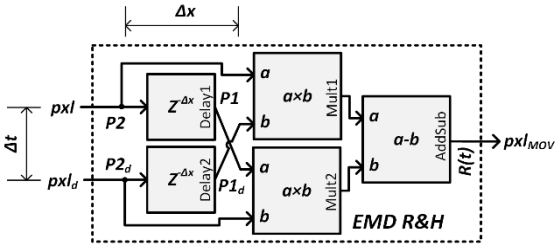


Figura 3

Arquitectura interna del bloque XIL EMD R&H.

3.1- MODIFICACIONES AL DETECTOR DE MOVIMIENTO ELEMENTAL

La salida del detector de movimiento elemental puede verse afectada por las condiciones del entorno, como la iluminación y el ruido, precisando de modificaciones para mejorar su respuesta. Estas modificaciones añaden elementos de filtrado pasa alto (*HP* en la Figura 4) en diferentes etapas del detector de movimiento básico de Reichardt y Hassenstein, con el objetivo de filtrar las señales de entrada, suprimir las diferencias de iluminación constantes y aumentar la ganancia a la salida [25, 29, 30].

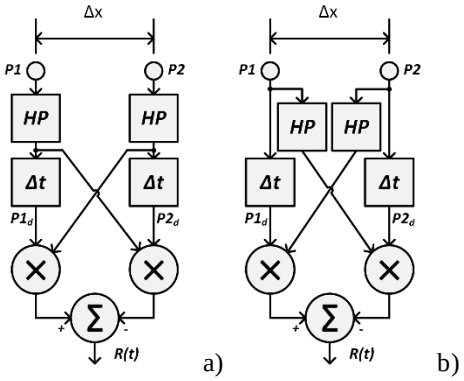


Figura 4

Alternativas al detector de movimiento elemental. a) Detector de Harris y O'Carroll. b) Detector de Kirschfeld.

La modificación propuesta por Harris y O'Carroll (Figura 4a) añade un filtro pasa alto (*HP*) entre los fotorreceptores y el filtro pasa bajo, con el objetivo de filtrar las señales de entrada, suprimiendo las diferencias de iluminación constantes presentes en algunos entornos, que causan falsas detecciones u omisiones (Figura 5a) [29, 30]. Otra variación del modelo original es la planteada por Kirschfeld (Figura 4b), que añade el filtro pasa alto en las ramas interiores del modelo, presentando una mejor ganancia a la salida ($R(t)$) con más sensibilidad a los ruidos provenientes de los fotorreceptores (Figura 5a) [25].

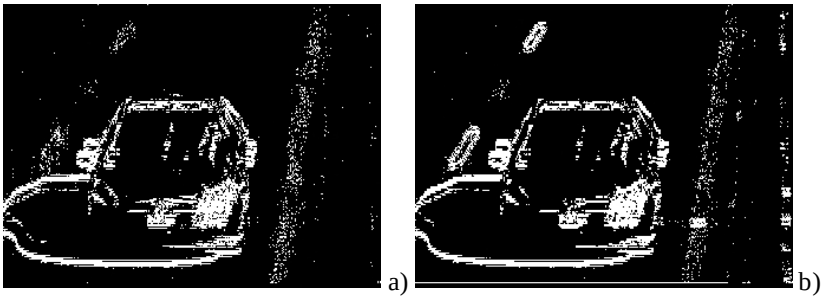


Figura 5

Detección de movimiento. a) Resultado de la correlación del detector de Harris y O'Carroll. b) Resultado de la correlación del detector de Kirschfeld.

La implementación de los módulos de detección modificados es similar a la del módulo básico, añadiendo los filtros pasa alto. La implementación de estos filtros se realiza mediante el uso de filtros específicos de realce (*Sharp*), en unión con bloques almacenadores de líneas (*Line Buffer*) para el procesado de una ventana de píxeles contiguos.

La variación de Harris y O'Carroll (Figura 6), también disponible en la biblioteca *XIL XSGImgLib*, aplica los filtros pasa alto (*Sharp*) a ambos fotogramas, para luego continuar con la arquitectura del EMD básico. En cambio, la modificación de Kirschfeld (Figura 7) solo emplea un filtro de realce (*Sharp*) en la rama del fotograma actual, no presentándolo para el fotograma anterior. En esta última arquitectura se emplea un bloque de retardo configurado para una latencia (τ_{HP}) igual a la suma de las latencias de operación del almacenador de línea y el filtro de realce.

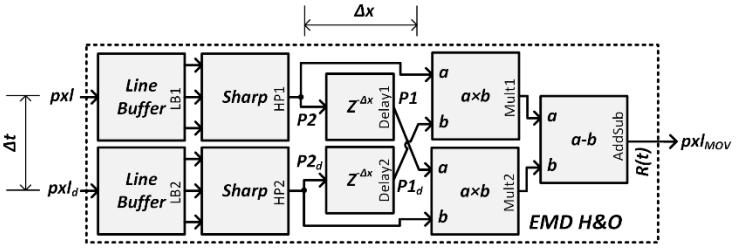


Figura 6

Arquitectura interna del módulo *XIL EMD H&O*.

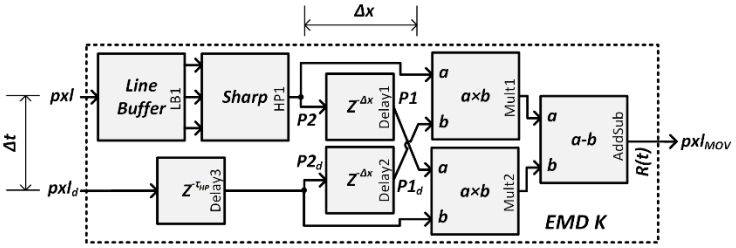


Figura 7

Arquitectura interna del módulo *XIL EMD K*.

4.- IMPLEMENTACIÓN DE UN SISTEMA DE DETECCIÓN DE MOVIMIENTO PARA CÁMARAS INTELIGENTES

La utilización de detectores de movimiento en cámaras inteligentes para aplicaciones de monitorización de tráfico, vigilancia o seguimiento facilita la identificación de regiones de píxeles que presentan un desplazamiento en la secuencia de imágenes para señalar y, si es necesario, segmentar la región en movimiento para entregársela a la siguiente etapa de procesado, la cual concentra el análisis sobre esta área en vez de emplear la imagen en su totalidad. Este mecanismo permite reducir el tiempo de procesado exhaustivo dedicado a la detección de características y la identificación de los objetos de interés en la secuencia.

El sistema de procesado espaciotemporal de la Figura 8 se emplea en la creación de un detector de movimiento elemental para una cámara inteligente en un punto de control de tráfico con límite de velocidad de circulación de los vehículos de 60 km/h. Las imágenes son capturadas por una cámara, almacenando varios fotogramas en memoria, de los cuales son extraídos tres para ser analizados por el sistema de procesado y señalar en el fotograma de salida la región en movimiento. El sistema ha sido implementado sobre una placa de desarrollo *Atlys*, con un FPGA *Spartan-6 XC6SLX45* de Xilinx, a la cual se conecta una cámara estereoscópica del fabricante Digilent Inc. (*VmodCAM - Stereo Camera Module*) mediante un módulo de control, desde la cual se toma la secuencia de fotogramas de resolución 640×480 y se almacenan en memoria externa al FPGA disponible en la placa de desarrollo.

El sistema de control de fotogramas está integrado por un bloque de escritura en memoria (*Módulo de Escritura*$_1$), el controlador de memoria externa del FPGA (*MPMC*) y tres módulos de lectura (*Módulo de Lectura*$_x$) para entregar los tres fotogramas necesarios a la unidad de procesado hardware (*Procesado HW*). El resultado de las operaciones es enviado directamente al controlador de vídeo (*Control de Vídeo*) para su visualización. Para esto es preciso que la unidad de procesado presente una frecuencia de trabajo mayor o igual a la empleada por el sistema de visualización, garantizando que la frecuencia de refrescamiento del monitor no sea afectada.

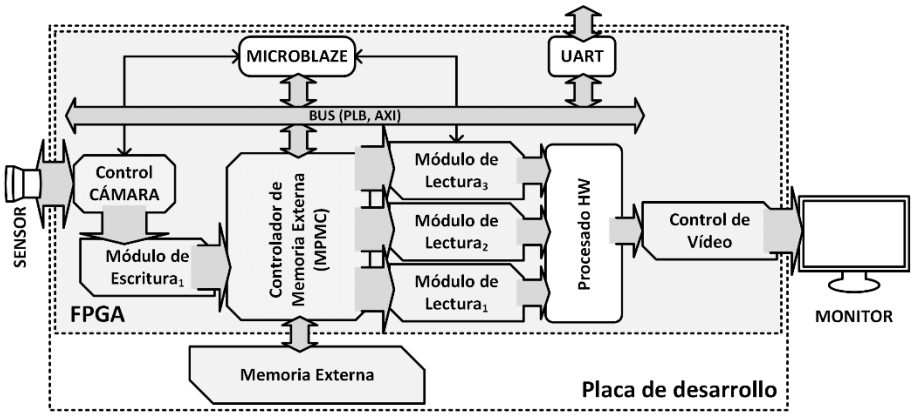


Figura 8

Sistema de procesado espaciotemporal para la detección de movimiento elemental para cámaras inteligentes.

La memoria externa almacena varios fotogramas de la secuencia de vídeo (Figura 9) en dependencia de la configuración del desplazamiento temporal del bloque de detección de movimiento (Δt). El módulo de escritura acumula los píxeles desde la interfaz de vídeo, sobrescribiendo el fotograma más antiguo en memoria (*Fotograma* t_1). A su vez, los tres módulos de lectura obtienen los fotogramas a analizar. Los fotogramas t_{-2} y t_{-1}, separados por Δt fotogramas, son empleados en la detección de movimiento, mientras se realiza la señalización de la región resultante de la operación anterior en el fotograma actual (*Fotograma* t_0). El sistema de manejo de memoria externa propuesto implementa una cola circular de fotogramas, realizando un desplazamiento en las direcciones de memoria de los módulos de lectura y escritura cuando se recibe un nuevo fotograma.

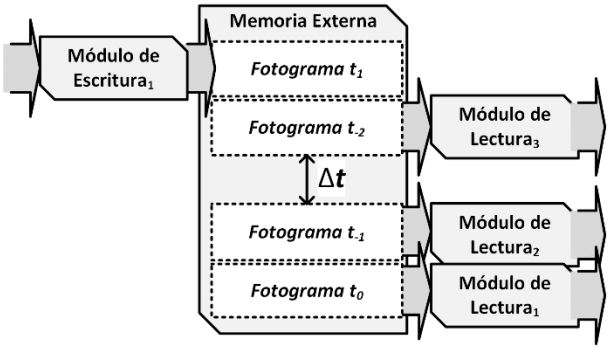


Figura 9

Distribución de los fotogramas en la memoria externa para el detector de movimiento elemental.

La unidad de procesado (Figura 10) analiza los fotogramas en memoria t_{-2} y t_{-1} para detectar el movimiento y luego elimina la información no relevante por medio de operaciones morfológicas. Este resultado es procesado para detectar las coordenadas donde se concentran los cambios en las imágenes, dibujando un rectángulo en el fotograma t_0 y entregando nuevas señales de activación. La unidad de procesado incluye un módulo de temporización de fotogramas (VTC: *Video Timing Controller*) para suministrar las señales de sincronismo y activación, necesarias para el funcionamiento de los bloques de procesado y visualización del resultado.

La unidad de procesado se compone de diez bloques disponibles en la biblioteca de procesado *XIL XSGImgLib*. Los módulos de conversión (*XIL RGB2Gray*) reciben, a través de los módulos de entrada, fotogramas en colores correspondientes a una trama de vídeo capturada por una cámara con resolución de 640×480 píxeles (Figura 11a y b), transformándolos a escala de grises (Figura 11c y d).

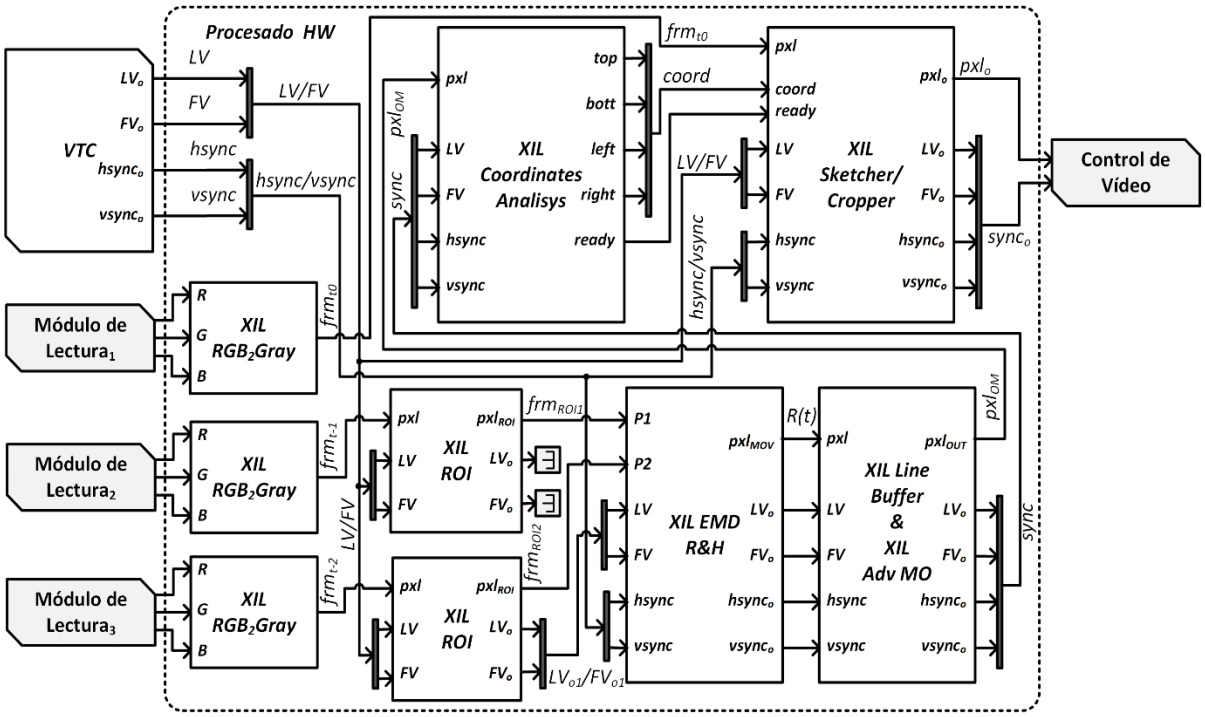


Figura 10

Unidad de procesado del detector de movimiento elemental.

Los fotogramas t_{-2} y t_{-1} son ajustados a la región de interés (ROI: *Region of Interest*), eliminando un 15% de ambos laterales de los fotogramas (Figura 11e y f), zona por donde es menos probable ocurra el desplazamiento de un automóvil al pasar por el punto de control de tráfico. Los bloques de región de interés (*XIL ROI*) generan las nuevas señales de activación de la imagen (LV_{o1} y FV_{o1}) que ajustan el procesado al 70% de la imagen.

El bloque de detección de movimiento (*XIL EMD R&H*) se basa en el diseño básico del detector de movimiento elemental planteado por Reichardt y Hassenstein, configurado para la dirección vertical, suprimiendo cualquier movimiento en la dirección horizontal. El bloque recibe los dos fotogramas a analizar (frm_{ROI1} y frm_{ROI2}) y las nuevas señales de activación (*LV* y *FV*) desde los bloques de región de interés. Además, emplea las señales de sincronismo (*hsync* y *vsync*) generadas por el VTC, las cuales se incluyen, a partir de este bloque, al flujo de procesado. La salida del bloque de detección de movimiento ($R(t)$) devuelve los cambios entre los fotogramas contiguos t_{-2} y t_{-1} (Figura 11g). El resultado muestra que el detector no es sensible a una distribución constante en los niveles de la imagen, donde no se detectan cambios.

La operación de cierre que se lleva a cabo en la siguiente etapa del sistema elimina los puntos aislados de la detección de movimiento (Figura 11h). En esta operación se emplea un bloque almacenador de líneas, para paralelizar los píxeles, y un operador morfológico avanzado (*XIL Line Buffer & XIL Adv. MO*).

El resultado de la operación de cierre es analizado por el bloque de análisis de coordenadas (*XIL Coordinates Analisys*) para obtener las coordenadas de los puntos superior izquierdo e inferior derecho que limitan la zona de la imagen en la que se detecta movimiento. Para incluir el dibujo de un rectángulo en la imagen, este bloque entrega las coordenadas de los puntos (*coord*), junto con el fotograma actual (frm_{t0}), al bloque de segmentado de regiones (*XIL Sketcher/Cropper*). El resultado del sistema de procesado (pxl_o) es, por último, enviado al controlador de vídeo para su visualización, en conjunto con las señales de sincronismo ($sync_o$). La imagen resultante (Figura 12) muestra un rectángulo que abarca el automóvil en movimiento.

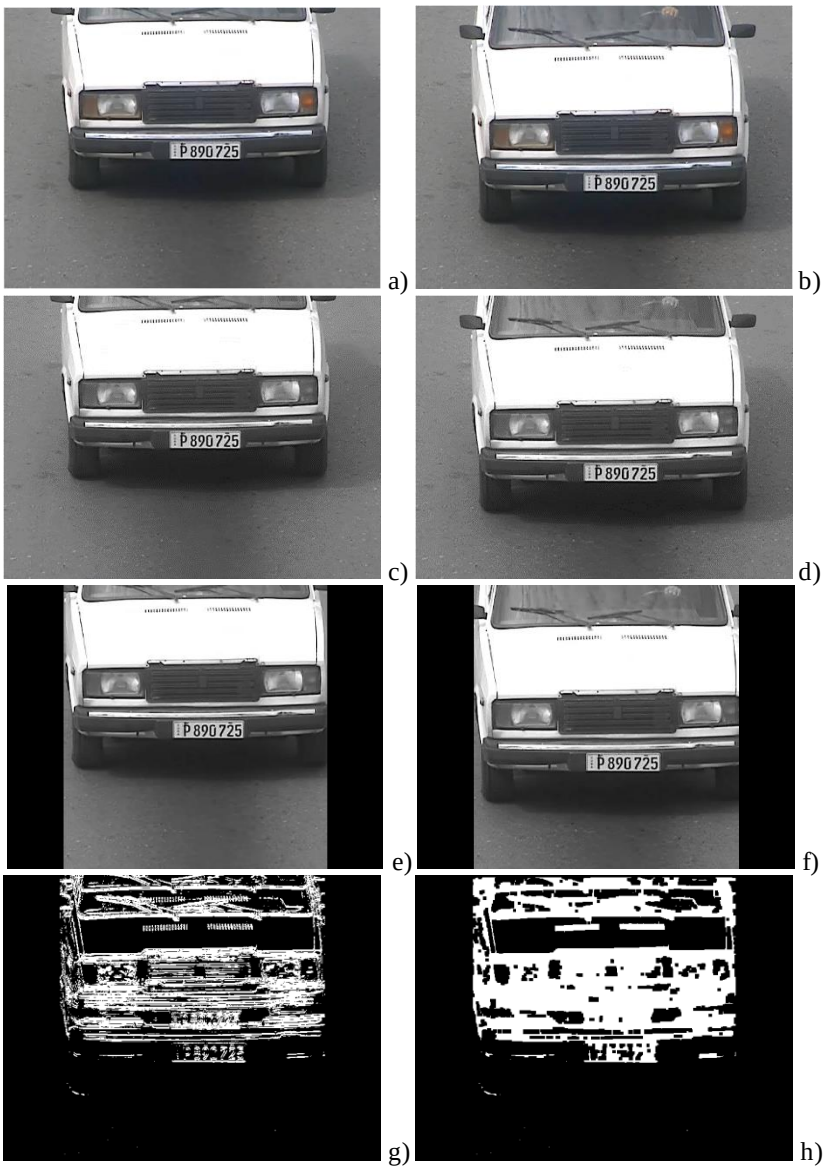
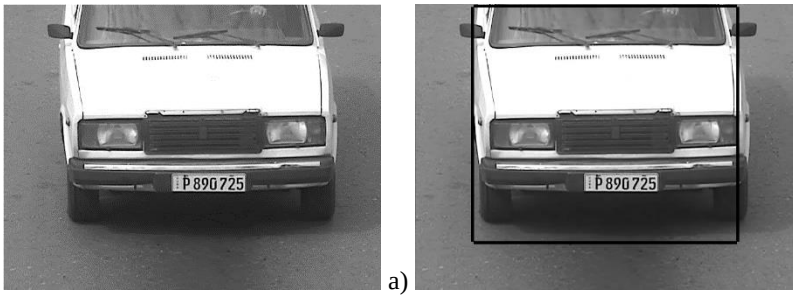


Figura 11

Operaciones del sistema de detección de movimiento para una aplicación de control de tráfico. a) Fotograma t_{-2} en formato color. b) Fotograma t_{-1} en formato color. c) Fotograma t_{-2} en escala de grises. d) Fotograma t_{-1} en escala de grises. e) Salida del bloque *XIL ROI* para el fotograma t_{-2}. f) Salida del bloque *XIL ROI* para el fotograma t_{-1}. g) Salida del bloque *XIL EMD R&H* para los fotogramas t_{-2} y t_{-1}. h) Salida del bloque *XIL Adv MO* para los fotogramas t_{-2} y t_{-1}.

El consumo de recursos de la unidad de procesado hardware, empleando un FPGA *Spartan-6 LX45*, se muestra en la Tabla 1. El consumo de recursos lógicos supone un 8,54% de los registros y un 21,64% de las tablas de búsquedas disponibles en el FPGA, siendo el módulo de detección de movimiento configurado para la orientación vertical el de mayor consumo, incluyendo dos multiplicadores dedicados (3,45 %). La unidad de procesado opera a una frecuencia de 57,60 MHz con un rendimiento de procesado normalizado de 17,35 ns/píxel, permitiendo el análisis de una imagen de alta definición (HD) con refrescamiento a 60 Hz. En este aspecto, la implementación propuesta es más eficiente que otras reportadas en la literatura empleando el mismo principio de detección de movimiento, como el desarrollado por Zhang *et al*. en [24] el cual alcanza un rendimiento de 43,59 ns/píxel (implementado sobre un FPGA *Stratix EP1S4* para resoluciones de 256×256 píxeles a 450 fps) o el presentado por Köhler *et al*. en [25] con un rendimiento de 393,70 ns/píxel (implementado sobre un FPGA *Stratix II S60* para resoluciones de 127×100 píxeles a 200 fps). En comparación con la implementación de otros algoritmos de detección de movimiento sobre FPGA reportados en la literatura consultada, la unidad de procesado hardware presenta un rendimiento

ligeramente superior al sistema de diferenciación temporal presentado por Kau *et al.* en [15] con rendimiento de 18,38 ns/píxel (implementado sobre un FPGA *Ciclone II 2C70* para resoluciones de 640×480 píxeles a 177,08 fps), mientras que es inferior al sistema de sustracción de fondo reportado por Kapude *et al.* en [3], con rendimientos de 4,91 ns/píxel y 9,52 ns/píxel (implementado sobre un FPGA *Virtex-6 LX75T* y un FPGA *Spartan-6 LX75T* para resoluciones de 256×256 píxeles a 3097 fps y 1602 fps respectivamente), y al sistema de estimación de flujo óptico obtenido por Allaoui *et al.* en [17], con rendimiento de 3,74 ns/píxel (desarrollado sobre un FPGA *Virtex-6 LX75T* para resoluciones de 640×480 píxeles a 870 fps).



a) b)

Figura 12

Identificación de la región en movimiento. a) Fotograma más reciente en memoria (t_0) en escala de grises. b) Segmentación del movimiento en el fotograma más reciente (t_0).

TABLA 1

Consumo de recursos de la unidad de procesado hardware para el sistema de detección de movimiento

Recursos (Spartan-6 LX45)		Detección de movimiento	
Tipo	**Disponibles**	**Ocupados**	**Por ciento**
Registros	**54 576**	4 569	8,54%
LUT	**27 288**	5 904	21,64%
DSP48A	**58**	2	3,45%

El sistema completo de procesado espaciotemporal para detección de movimiento emplea el 15,06% de los registros y el 44,12% de las tablas de búsqueda de un FPGA *Spartan-6 LX45* (Tabla 2). El 5,17% de los bloques de multiplicación (*DSP48A*) son utilizados en el cálculo de la dirección inicial donde se posicionan los módulos de escritura y lectura cuando los mismos son desplazados, mientras que los bloques de memoria de 16 bits implementan las estructuras de sincronismo necesarias para las interfaces con la memoria externa de la placa (MCB: *Memory Controller Block*).

Tabla 2

Consumo de recursos del sistema de procesado espaciotemporal para detección de movimiento en cámaras inteligentes.

Recursos (Spartan-6 LX45)		Sistema de procesado	
Tipo	**Disponibles**	**Ocupados**	**Por ciento**
Registros	**54 576**	8 218	15,06%
LUT	**27 288**	12 039	44,12%
RAMB16	**116**	33	28,45%
RAMB8	**232**	1	0,43%
DSP48A	**58**	5	8,62%
MCB	**2**	1	50,00%

5.- CONCLUSIONES

La detección de movimiento, como primera etapa de procesado en los sistemas de cámaras inteligentes, permite concentrar la ejecución del algoritmo en el área en movimiento, reduciendo el tiempo de procesado.

Los métodos de detección de movimiento analizados presentan ventajas y desventajas. Los métodos de substracción de fondo no son factibles para aplicaciones con cámaras en movimiento y no brindan información sobre la velocidad de cambio de los objetos y la dirección del movimiento. Los métodos de substracción de fotogramas adyacentes solucionan el primer inconveniente, manteniendo los restantes, mientras que el análisis del flujo óptico requiere sistemas de procesado muy complejos. Las técnicas de análisis frecuencial, como el detector de movimiento elemental descrito en este trabajo, presentan un adecuado compromiso entre la complejidad de la implementación sobre dispositivos embebidos y la información resultante del procesado.

El sistema de detección de movimiento desarrollado, que utiliza el detector de Reichardt y Hassenstein junto a otros bloques de procesado de imágenes disponibles en la biblioteca *XIL XSGImgLib*, permite el procesado de fotogramas en alta definición a 60 Hz. El consumo de recursos de la implementación del sistema ocupa 15,06% de los registros, un 44,12% de las tablas de búsquedas y un 8,62% de los multiplicadores dedicados, la mitad de estos empleados en el detector de movimiento, además del 29% de los bloques de memoria de un FPGA *Spartan-6 LX45*, lo cual permite la inclusión de otras funciones a la unidad de procesado.

REFERENCIAS

1. Yi, Q.: "FPGA Implementation of Video Capture and Moving Target Detection System," in 10th International Conference on Communications, Circuits and Systems (ICCCAS'18), 2018, pp. 373–377.
2. Yang, L. and Dong, W.: "Real-time Motion Detection with High-speed Vision and Local Parallel Processing," in IEEE International Conference on Mechatronics and Automation (ICMA'19), 2019, pp. 1079–1084.
3. Kupade, R. N.; Meenpal, T.; and Nath, P. K.: "FPGA Implementation of Moving Object Segmentation using ABPD and Background Model," in 2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT'19), 2019, no. 1, pp. 1–5.
4. Alshadoodee, H. A. A.; Dhahir, M. Q.; and Rasool, Z. H.: "Digital Camera in Movement Tracking on FPGA Board DE2," in 2019 International Russian Automation Conference (RusAutoCon'19), 2019, pp. 1–8.
5. Bilal, M.; Khan, A.; Umar, M.; Khan, K.; and Kyung, C.: "A Low-Complexity Pedestrian Detection Framework for Smart Video Surveillance Systems," IEEE Transactions on Circuits and Systems for Video Technology, vol. 27, no. 10, pp. 2260–2273, 2017.
6. Garcés-Socarrás, L. M.; Cabrera Sarmiento, A. J.; Sánchez-Solano, S.; Brox Jiménez, P.; Ieno, E.; and Pimenta, T. C.: "Modificación automática de arquitecturas de módulos hardware de procesado de imágenes," Revista de Ingeniería Electrónica, Automática y Comunicaciones, vol. XXXVII, no. 3/2016, pp. 21–33, 2016.
7. Yi, Q.: "Design of Moving Object Detection System Based on FPGA," in 10th International Conference on Communications, Circuits and Systems (ICCCAS'18), 2018, pp. 435–438.
8. Bourrasset, C.; Maggiani, L.; Sérot, J.; and Berry, F.: "Dataflow object detection system for FPGA- based smart camera," IET Circuits, Devices & Systems, vol. 10, no. 4, pp. 280–291, 2016.
9. Rohilla, R.; Raj, A.; Kejriwal, S.; and Kapoor, R.: "FPGA Accelerated Abandoned Object Detection," in International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT'16), 2016, pp. 302–306.
10. Singh, S.; Shekhar, C.; and Vohra, A.: "FPGA-Based Real-Time Motion Detection for Automated Video Surveillance Systems," FPGA and SoC Devices Applied to New Trends in Image/Video and Signal Processing Fields, vol. 5, no. 1, pp. 1–18, 2016.
11. Guo, G.; Kaye, M. E.; and Zhang, Y.: "Enhancement of Gaussian Background Modelling Algorithm for Moving Object Detection & Its Implementation on FPGA," in IEEE 28th Canadian Conference on Electrical and Computer Engineering (CCECE'15), 2015, pp. 118–122.
12. Jadhav, S.; Narvekar, R.; Mandawale, A.; and Elgandelwar, S.: "FPGA Based Object Tracking System," in Fifth International Conference on Communication Systems and Network Technologies, 2015, pp. 10–13.
13. Kryjak, T. and Gorgon, M.: "Real-time implementation of moving object in video surveillance systems using FPGA," Computer Science, vol. 12, pp. 149–162, 2011.
14. Vu Dang, N. T.; My, L. N. T.; Quoc, T. N.; and Hoang, L. T.: "Hardware Architecture Design For Vehicle Detection," in International Symposium on Electrical and Electronics Engineering (ISEE'19), 2019, pp. 33–36.

15. Kau, L.-J.; Jhao, G.-T.; Lai, W.-X.; and Liu, Y.-R.: "FPGA-based Moving object Detection with Interferences," in International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM'19), 2019, pp. 181–184.
16. Korakoppa, V. P.; Mohana; and Aradhya, H. V. R.: "An Area Efficient FPGA Implementation of Moving Object Detection and Face Detection using Adaptive Threshold Method," in 2nd IEEE International Conference On Recent Trends in Electronics Information & Communication Technology (RTEICT'17), 2017, pp. 1606–1611.
17. Allaoui, R.; Mouane, H. H.; Asrih, Z.; Mars, S.; El hajjouji, I.; and El mourabit, A.: "FPGA-based implementation of Optical flow Algorithm," in 3rd International Conference on Electrical and Information Technologies (ICEIT'17), 2017, pp. 1–5.
18. Uetsuhara, K.; Tahara, A.; Manabe, T.; and Shibata, Y.: "Implementation of a real-time image-based vibration detection and adaptive filtering on an FPGA," in Conference on Design and Architectures for Signal and Image Processing (DASIP'18), 2018, pp. 54–59.
19. Chang, C.-W.; Zhong, Z.-Q.; and Liou, J.-J.: "A FPGA Implementation of Farneback Optical Flow by High-Level Synthesis," in ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA'19), 2019, p. 309.
20. Seyid, K.; Richaud, A.; Capoccia, R.; and Leblebici, Y.: "FPGA-Based Hardware Implementation of Real-Time Optical Flow Calculation," IEEE Transactions on Circuits and Systems for Video Technology, vol. 28, no. 1, pp. 206–216, 2018.
21. Claveau, D. and Wang, C.: "Space-Variant Motion Detection for Active Visual Target Tracking," Robotics and Autonomous Systems, pp. 11–22, 2009.
22. Mota, S.; Ros, E.; Díaz, J.; Ortigosa, E. M.; Agis, R.; and Carrillo, R. R.: "Real-Time Visual Motion Detection of Overtaking Cars for Driving Assistance Using FPGAs," in 14th International Conference on Field Programmable Logic and Application (FPL 2004), Leuven, Belgium, 2004, vol. 3203, pp. 1158–1161.
23. Garland, M.; Le Grand, S.; Nickolls, J.; Anderson, J.; Hardwick, J.; Morton, S.; Phillips, E.; Zhang, Y.; and Volkov, V.: "Parallel computing experiences with CUDA," IEEE Micro, vol. 28, no. 4, pp. 13–27, 2008.
24. Zhang, T.; Wu, H.; Borst, A.; and Buss, M.: "An FPGA Implementation of Insect-Inspired Motion Detector for High-Speed Vision Systems," in IEEE International Conference on Robotics and Automation, 2008, pp. 19–23.
25. Köhler, T.; Röchter, F.; Lindemann, J. P.; and Möller, R.: "Bio-inspired motion detection in an FPGA-based smart camera module," Bioinspiration & Biomimetics, vol. 4, no. 1, pp. 1–15, 2009.
26. Mota, S.; Ros, E.; Ortigosa, E. M.; and Pelayo, F. J.: "Bio-inspired motion detection for a blind spot overtaking monitor," International Journal of Robotics and Automation, no. 1, pp. 190–196, 2004.
27. Garcés-Socarrás, L. M.; Romero Ares, D. A.; Cabrera Sarmiento, A. J.; Sánchez-Solano, S.; and Brox Jiménez, P.: "Model-based implementation of self-configurable intellectual property modules for image histogram calculation in FPGAs," Ingeniería e Investigación, vol. 37, no. 2, pp. 74–81, 2017.
28. Garcés-Socarrás, L. M.; Cabrera Sarmiento, A. J.; Sánchez-Solano, S.; Brox Jiménez, P.; Ieno, E.; and Pimenta, T. C.: "Self-Modifiable Image Processing Library for Model-Based Design on FPGAs," IEEE Latin America Transactions, vol. 17, no. 5, pp. 742–750, 2019.
29. Harris, R. A. and O'Carroll, D. C.: "Afterimages in fly motion vision," Vision Research, vol. 42, no. 14, pp. 1701–1714, 2002.
30. Shoemaker, P. A.; O'Carroll, D. C.; and Straw, A. D.: "Implementation of visual motion detection with contrast adaptation," in SPIE Electronics and Structures for MEMS, 2001, pp. 316–327.

AGRADECIMIENTOS

Esta investigación ha sido financiada parcialmente por el Consejo Superior de Investigaciones Científicas (CSIC) a través del Programa de Cooperación al Desarrollo iCOOP+ y por el Ministerio español de Ciencia, Innovación y Universidades a través del proyecto TEC2017-83557-R, así como por el programa CAPES/MES.

CONFLICTO DE INTERESES

Ninguno de los autores manifestó la existencia de posibles conflictos de intereses que debieran ser declarados en relación con este artículo.

CONTRIBUCIONES DE LOS AUTORES

Luis Manuel Garcés Socarrás: conceptualización, investigación, metodología, software, supervisión, visualización, redacción – borrador original.

Roberto Sánchez Correa: conceptualización, análisis formal, investigación, metodología, software, validación – verificación, redacción – revisión y edición.

Alejandro José Cabrera Sarmiento: conceptualización, curación de datos, adquisición de fondos, supervisión, redacción – revisión y edición.

Santiago Sánchez Solano: conceptualización, adquisición de fondos, supervisión, redacción – revisión y edición.

Piedad Brox Jimenez: conceptualización, adquisición de fondos, supervisión, redacción – revisión y edición.

AUTORES

Luis Manuel Garcés Socarrás, Ingeniero en Automática, Doctor en Ciencias Técnicas, Departamento de Automática y Computación de la Universidad Tecnológica de La Habana "José Antonio Echeverría" (CUJAE), La Habana, Cuba, lmgarcess@automatica.cujae.edu.cu, https://orcid.org/0000-0002-8158-6423. Actualmente desarrolla el tema de investigación de aceleración de algoritmos mediante hardware reconfigurable para el procesado de imágenes y vídeos.

Roberto Sánchez Correa, Ingeniero en Automática, Hardent Inc., Montreal, Canadá, roberto.sanchez-correa@polymtl.ca, https://orcid.org/0000-0001-6020-5305. Entre su campo de investigación está la aceleración de algoritmos de procesado de imágenes mediante hardware reconfigurable, así como la comunicación a alta velocidad y baja latencia en clústeres de FPGA.

Alejandro José Cabrera Sarmiento, Ingeniero Electricista, Doctor en Ciencias Técnicas Departamento de Automática y Computación de la Universidad Tecnológica de La Habana "José Antonio Echeverría" (CUJAE), La Habana, Cuba, alex@automatica.cujae.edu.cu, https://orcid.org/0000-0001-6710-163X. Sus líneas de investigación principales están relacionadas con el desarrollo de sistemas empotrados basados en FPGA y la aceleración de algoritmos sobre hardware reconfigurable, con aplicaciones en procesado de imágenes, control inteligente y criptografía.

Santiago Sánchez Solano, Licenciado en Física, Doctor en Ciencias Físicas, Instituto de Microelectrónica de Sevilla, IMSE-CNM, (CSIC/Universidad de Sevilla), Sevilla, España, santiago@imse-cnm.csic.es, https://orcid.org/0000-0002-0700-0447. Sus líneas de investigación se centran en el desarrollo de sistemas empotrados con componentes hardware/software sobre FPGA y la realización microelectrónica de sistemas neurodifusos, así como sus aplicaciones en robótica, procesamiento de imágenes, seguridad y redes de sensores inteligentes.

Piedad Brox Jiménez, Licenciada en Física, Doctora en Física, Instituto de Microelectrónica de Sevilla, IMSE-CNM, (CSIC/Universidad de Sevilla), Sevilla, España, brox@imse-cnm.csic.es, https://orcid.org/0000-0003-1059-5338Sus actividades de investigación incluyen el diseño e implementación de sistemas neurodifusos y su aplicación al procesado digital de imagen y vídeo, diseño de sistemas de visión empotrados, diseño de circuitos digitales de altas prestaciones incluyendo dispositivos programables (FPGA) y circuitos integrados (ASIC) para distintas aplicaciones (procesado de imagen y vídeo, criptografía, biometría (huella dactilar), controladores empotrados, sensado virtual).

Esta revista se publica bajo una Licencia Creative Commons Atribución-No Comercial-Sin Derivar 4.0 Internacional