Implementación FPGA del cálculo de profundidades en la recuperación de 3D usando luz estructurada

Díaz Hernández Carlos Alberto, López Gutiérrez Luis David, Arias Estrada Miguel O., Feregrino Uribe Claudia y Cumplido Parra Rene A.

Instituto Nacional de Astrofísica, Óptica y Electrónica – INAOE, Luis Enrique Erro 1
Santa María Tonantzintla, Puebla, 72840 México.
{cdiaz, luis_david}@ccc.inaoep.mx, {ariasm, rcumplido, cferegrino}@inaoep.mx.

Resumen: Se presenta la implementación en un FPGA Virtex-II del algoritmo para el cálculo de profundidades en el proceso de recuperación de información 3D a partir de imágenes utilizando luz estructurada. En la metodología, se explican las características del pipeline y la forma en que se establece la comunicación a través de una interfaz C/C++ entre una aplicación cliente y el proceso residente en el FPGA. Asimismo, se presentan resultados de pruebas funcionales del diseño y un análisis cuantitativo de los resultados de estas pruebas.

Palabras claves: FPGA, luz estructurada, procesamiento de imágenes, 3D.

1 Introducción

En la vida real y con mayor frecuencia en la industria, se presenta la necesidad de medir con alta precisión las formas de objetos tridimensionales y asegurar al mismo tiempo la calidad en la producción. Estas mediciones pueden realizarse por la inspección automática en línea y el reconocimiento de problemas a través de la medición de formas 3D de un objeto [3]. Uno de los métodos más ampliamente usados para recuperación 3D se basa en la proyección de luz estructurada. La información 3D se manifiesta por si misma en las deformaciones aparentes del modelo proyectado. Este procedimiento de análisis de las deformaciones, implica un consumo excesivo de recursos de cómputo, los cuales aminoran las expectativas del algoritmo, por lo tanto se debe buscar una manera de disminuir el consumo de tiempo de CPU en la computadora.

Generalmente el procesamiento de secuencias de imágenes se lleva a cabo por medio de procesadores de propósito general y de alta velocidad, sin embargo el tiempo de procesamiento resulta muy grande para aplicaciones con procesamiento de imágenes a velocidades convencionales de 25 y 30 cuadros por segundo.

Asimismo, existen en el mercado dispositivos específicos de cómputo que pueden realizar algunas de estas tareas con un mayor desempeño que un microprocesador de uso general, entre estos se encuentran los DSP's, coprocesadores, etc. Existen circuitos que proveen una funcionalidad mayor o menor a la requerida, o en caso extremo no proveer la funcionalidad.

No obstante actualmente, con el uso de nuevas tecnologías de circuitos integrados y la aplicación de novedosos algoritmos de procesamiento, es posible desarrollar incipientes formas de procesamiento de imágenes en tiempo real.

2 Estado del arte

Las aplicaciones que implican el análisis de imágenes, van acompañadas con algoritmos que involucran tiempos de ejecución largos a la vez que estos pueden ser ampliamente paralelizados. A continuación se menciona un trabajo importante en el área del procesamiento de imágenes que implica el uso de dispositivos reconfigurables para la ejecución de procesos críticos.

En [1], se presenta el diseño de un circuito para la modificación automática del histograma de una imagen de video. El circuito corrige automáticamente el histograma o nivel de grises de una imagen; recibe como entrada una imagen de 256 x 256 píxeles a una frecuencia de 50 o 60 imágenes por segundo.

Este circuito toma todos los bytes que representan el nivel de gris de cada píxel de una imagen y calcula el promedio. Para la siguiente imagen el circuito efectuará una corrección de cada píxel de manera que el promedio de grises de la imagen se encuentre más centrado. Básicamente este circuito es una unidad de transformación de imagen a imagen., sin embargo, como subproducto, dispone de una salida donde es posible obtener el nivel promedio de gris de la imagen, es decir, también se realiza una transformación de imagen a datos.

3 Metodología

Para llevar a cabo el análisis de las deformaciones en un tiempo menor, se realizaron las siguientes tareas:

- Simplificación del código
- Diseño del dispositivo
- Diseño e implementación de la interfaz de aplicaciones en C/C++

3.1 Simplificación del código

Se simplificó el código, con el fin de eliminar cálculos redundantes que implicarán consumo de tiempo de procesamiento sin beneficio. Al analizar el código se observó que se realizaban operaciones dentro de un ciclo anidado que eran redundantes, por lo tanto se procedió a reordenar el código con el objetivo de disminuir el tiempo en procesamiento y a la vez el espacio de almacenamiento necesario en la función.

Después del reordenamiento del código se hizo un análisis para determinar cuantas veces se ejecutaban cálculos redundante y se determinó que el valor está dado por $R \cdot P_R$, es decir, el número de regiones (R) por el número de puntos en cada región

(P_R), con el rediseño, 4 de 5 operaciones se realizan una vez, y una de estas se ejecuta el número de regiones que se tienen (R). Con el rediseño ya se obtuvo una gran mejora, ya que esas operaciones son de punto flotante y ocupan más tiempo en realizarse que las enteras.

3.2 Diseño en Handel-C y sintetizado

El diseño se realizó en Handel-C, se tomó el enfoque de un pipeline de 9 niveles como se muestra en la figura 1.

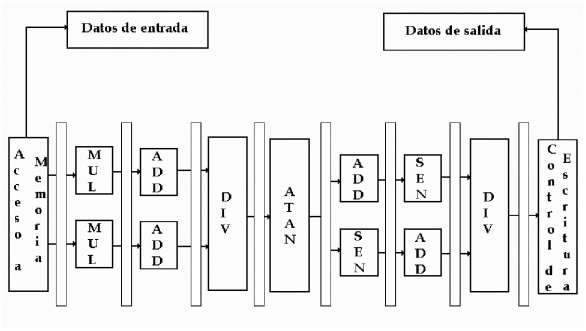

Fig. 1. Pipeline propuesto para el procesamiento de los datos

El acceso a memoria (nivel 1) se encarga de recuperar el valor de la columna en una posición dada por uno de los contadores. En el segundo nivel, MUL-MUL, se realizan dos operaciones de punto flotante, estos son los cálculos temporales necesarios para determinar los valores del numerador y del denominador. El ADD-ADD, tercer nivel, se encarga de sumar parámetros y resultados previos temporales para el cálculo de valores del numerador y denominador.

En el cuarto nivel (DIV), se divide el numerador entre el denominador, para que en el ATAN (nivel 5), se obtenga el arco-tangente del cociente. Es importante mencionar que el cálculo del arco-tangente se realizó a través de un polinomio de Taylor. En el sexto nivel, ADD-SEN, se suman el ángulo obtenido con otro parámetro, a la vez que se obtiene el seno del mismo ángulo obtenido. El SEN -ADD, séptimo nivel, calcula el seno de la suma de los ángulos y a la vez suma el seno obtenido del nivel previo con un parámetro. También en este caso, el seno del ángulo se determina a través de un polinomio de Taylor.

En el octavo nivel (DIV), se divide los valores calculados en el nivel 7, este viene siendo el resultado final, el cual es almacenado en el Control de Escritura, nivel 9, cuando se determina que es un valor válido.

Para aminorar el tamaño del diseño, se decidió tomar como aritmética de cálculo, operaciones con punto fijo con una precisión de 14 bits en la parte decimal (aproximadamente 5 decimales) y 20 bits en la parte entera (aproximadamente 6 decimales).

El sintetizado del diseño se realizó con la herramienta Xilinx ISE 5.1i.

3.3 Diseño e implementación de la interfaz de aplicaciones en C/C++

La figura 2, muestra un diagrama a bloques de la integración del dispositivo con las aplicaciones de usuario.

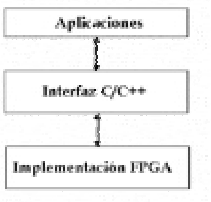

Fig. 2. Integración del dispositivo diseñado con el ambiente de aplicaciones

La primera capa está conformada por las aplicaciones que harán uso de la funcionalidad del dispositivo y su implementación depende del lenguaje en que se codifique, y es completamente aparte del diseño de proceso implementado en el dispositivo. La segunda capa, es la interfaz C/C++ que sirve como intermediaria entre las aplicaciones clientes y el dispositivo en el FPGA. En la capa de implementación FPGA, reside el proceso implementado en el FPGA, este proceso realiza la funcionalidad requerida en el análisis de las deformaciones.

Para acceder a la funcionalidad proveída por el FPGA, se realizó el diseño de una interfaz en C/C++ para el envío y recepción de datos hacia/de el FPGA. Esta interfaz a la vez fue diseñada cumpliendo con el flujo de datos propuesto por la arquitectura de la librería de MVTEC-Halcon [2], dicha arquitectura se muestra en la figura 3.

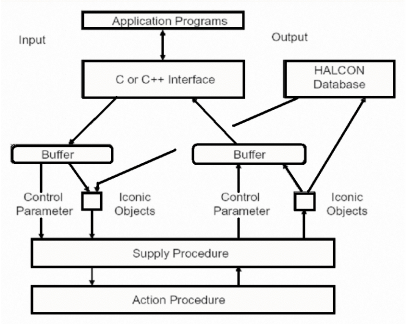

Fig.3. Arquitectura del flujo de datos en la librería de MVTEC Halcon.

El ajustar la interfaz con la librería de Halcon, se esta permitiendo su acceso desde la interfaz de usuario HDevelop. El procedimiento de abastecimiento (supply), recibe los datos desde la aplicación cliente que en puede ser el HDevelop, o cualquier otra aplicación desarrollada en Visual Basic, Visual C++, etc. Su función principal es extraer la información enviada y verificar que esta sea del tipo que esperado. Una vez obtenidos los parámetros, y los objetos internos de los parámetros, estos se pasan al procedimiento de acción que es el encargado de ejecutar finalmente el procesado de las deformaciones.

4 Resultados

A continuación se presentan los resultados de las pruebas. La diferencia en resultados no es distinguible para el ojo humano. Sin embargo el análisis del error medio generado durante el procesamiento debido al truncamiento, es aceptable. A continuación se muestran 3 imágenes: entrada, la obtenida por MVTec Halcon y finalmente la obtenida con el FPGA. En las pruebas realizadas se envía al operador el esqueleto de una tupla de regiones como la mostrada en la figura 5, el operador recibe todos los parámetros necesarios y como resultado final del procesamiento, el operador regresa una imagen como la mostrada en las figuras 6 y 7.

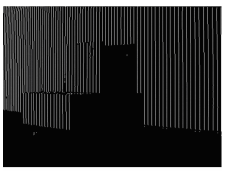

Fig. 5. Imagen de entrada.

Fig. 6. imagen de salida con el conjunto de operadores de Halcon

Fig. 7. Imagen de salida con el elemento de procesamiento diseñado para el FPGA.

En el análisis de tiempo de ejecución se obtuvieron los siguientes resultados:

Tiempo de procesamiento (en segundo)		Mejora
Con operadores de Halcon	Con el proceso en el FPGA	
113.293	28.871	3.924
116.507	18.114	6.432
147.098	26.694	5.511
120.974	21.863	5.533
102.757	20.588	4.991

Tabla 1

Las variaciones que se observan son debidas a que como se trabaja en Windows 2000, y no es un sistema en tiempo real, las medidas de tiempo no son precisas, además de que las asignaciones de tiempo de CPU para el caso del uso de los operadores de

Halcon, no se realizan siempre con la misma prioridad. De igual forma para el caso de transferencias de datos entre la computadora y la tarjeta, no siempre toma el mismo intervalo de tiempo.

La siguiente tabla lista los promedios de la tabla anterior, con lo cual se pretende tener una medida de comparación menos engañosa.

Promedios		
Con los operadores de Halcon	Con el proceso en el FPGA	Mejora
120.1258	23.226	5.278

Tabla 2

El resumen de los resultados obtenidos después de la síntesis en Handel-C es la siguiente:

Number of Slices: *15,360 out of 19,200 80%*
Total Number 4 input LUTs: *28,800 out of 38,400 75%*
Number of bonded IOBs: *243 out of 404 60%*
The Average Connection Delay for this design is: *3.078 ns*
The Maximum Pin Delay is: *19.871 ns*
The Average Connection Delay on the 10 Worst Nets is: *17.494 ns*

5 Conclusiones y perspectivas

En cada ciclo de reloj el pipeline propuesto, trabaja con nueve píxeles a la vez. Realizando en paralelo la recuperación de información 3D sobre nueve datos.

Es importante hacer mención, que la cuando el diseño ocupa demasiado espacio del FPGA, el proceso de compilación en Handel-C es demasiado lento, y en algunos casos saturas los requisitos de memoria del sistema.

A través de este proyecto, se ha conseguido mejorar hasta en 5 veces más un proceso crítico, logrando de esta manera el objetivo del mismo. Aunque, los resultados obtenidos con el presente diseño fueron satisfactorios. Sobre este diseño se pueden hacer mejoras, tales como:

- Aproximar las funciones trigonométricas de manera eficiente.
- Usar una aritmética de punto flotante que aporte mayor precisión en los cálculos.
- Habilitar un flujo continuo de datos entre la aplicación huésped y el FPGA.

6 Referencias

[1] Aitzol Zuloaga Izaguirre. Circuito para la modificación automática del histograma de una imagen de video. Universidad del País Vasco. 1996.

[2] HALCON Extension Package Interface. Programmer's Manual. MVTec Software GmbH. http://www.mvtec.com/halcon/

[3] Venustiano Soancatl Aguilar, Leopoldo Altamirano Robles. Recuperación de Formas 3D Usando Luz Estructurada. 2003.