Un Sistema de Verificación de Huellas Dactilares Basado en un Banco de Filtros de Gabor 2D

David Augusto Rojas Vigo

Facultad de Ingeniería Electrónica y Eléctrica, Universidad Nacional Mayor de San Marcos, Lima, Perú

RESUMEN: Este artículo describe la implementación algorítmica de un verificador de personas a través de huellas dactilares. Se emplea una plataforma de desarrollo DSP, que permite comparar las características extraídas de la impresión dactilar con la registrada previamente. Se emplea una técnica de alta velocidad en la etapa de coincidencia principalmente cuando no se requiera un valor alto de FMR.

ABSTRACT: This paper describes the algorithmic implementation of people's verifier through its fingerprints. A DSP platform of development is used, it allows to compare the characteristics extracted from the fingerprint with the previously registered. A high-speed technique in the stage of coincidence is used principally when a high value of FMR is not needed.

PALABRAS CLAVES: Verificación de Huellas Dactilares, Procesamiento Digital de Imágenes, Tiempo Real.

I. INTRODUCCIÓN

Con el gran avance en el desarrollo de los sistemas empotrados (*embedded systems*), la verificación por huellas dactilares viene migrando rápidamente hacia el mundo de los equipos móviles, convirtiéndose en una parte crítica para aplicaciones tales como los sistemas de acceso, PDA's, notebooks e incluso teléfonos celulares. Por lo que la verificación de la identidad juega un rol básico debido a que permite el acceso a lugares estratégicos y recursos a ser controlados.

El problema de la verificación de personas por huellas dactilares en tiempo real se puede tratar bastante bien utilizando como plataforma de hardware un computador de escritorio. Sin embargo, cuando se implementa esta aplicación en un dispositivo embebido encontramos ciertas dificultades causadas por la menor velocidad de la CPU, por la ausencia de una memoria caché e incluso en ciertas situaciones por la ausencia de una unidad de punto flotante. La optimización del proceso de verificación no implica sacrificar la confiabilidad, por lo tanto se requiere una técnica rápida, de bajo costo y precisa para desarrollar aplicaciones embebidas.

Debido a que los algoritmos de extracción de características y coincidencia involucran una gran cantidad de cálculo, no es una tarea trivial realizar éstos en dispositivos embebidos. Existen varias técnicas que pueden ser utilizadas para la verificación de personas por huellas dactilares en sistemas embebidos. Éstas incluyen los circuitos integrados de aplicación específica (*Application Specific Integrated Circuit* – ASIC), los procesadores digitales de señal (*Digital Signal Processor* – DSP), los sistemas de sólo un chip (*System-on-Chip* - SoC), entre otros.

Este artículo describe la implementación algorítmica de un verificador de personas por huellas dactilares en una plataforma de desarrollo DSP basada en el procesador digital de señales TMS320C6711 de *Texas Instruments* que permite autenticar la identidad declarada por un individuo, comparando las características extraídas de su impresión dactilar con la registrada previamente.

II. MARCO TEÓRICO

El diagrama general del proceso de verificación por huellas dactilares se muestra en la Figura 1., su funcionamiento puede resumirse en los siguientes pasos:

1) El usuario ingresa su identidad (mediante un teclado, tarjeta magnética, entre otros), así como su característica biométrica requerida (en este caso, su huella dactilar). Obsérvese que en las aplicaciones

de identificación el usuario no suministra su identidad.

2) El sistema extrae la representación de la característica biométrica de entrada, que consiste en la obtención de la respectiva plantilla (usualmente unos centenares de bytes).
3) Se compara la plantilla de entrada con la plantilla almacenada vinculada a la identidad suministrada previamente. Esta representación (la plantilla almacenada) se obtiene durante la inscripción de dicha persona.
4) El resultado de la comparación es una distancia o probabilidad de semejanza entre ambas plantillas. Se compara esta distancia con un valor umbral, si es menor que el valor umbral se acepta al usuario; en caso contrario se niega el acceso. Uno de los aspectos claves es la determinación de dicho valor umbral, puesto que condiciona los valores de FMR y FNMR [1], en algunas aplicaciones interesa que una de las dos tasas sea baja a costa de elevar la otra.

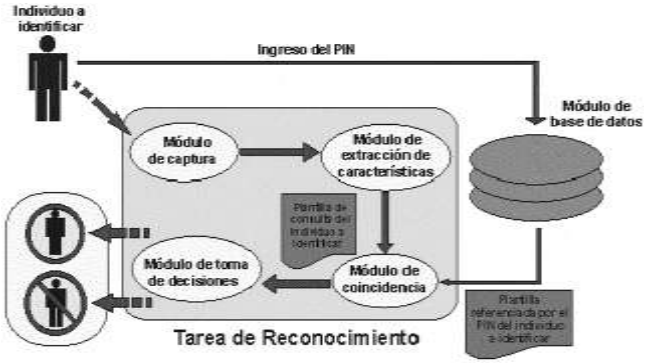

Fig. 1. Componentes de un Sistema Biométrico en modo verificación

III. HERRAMIENTAS DE DESARROLLO UTILIZADAS

El diseño de un sistema de verificación requiere las consideraciones de costo, tiempo de desarrollo y desempeño en tiempo real. Debido a la gran cantidad de datos que deben ser procesados y además a la complejidad computacional de los algoritmos empleados, todo sistema de verificación requiere una plataforma de desarrollo que provea herramientas de soporte tanto de hardware como de software.

De acuerdo con el Internacional Biometric Group, actualmente Texas Instruments ofrece una de las plataformas de desarrollo más flexibles de la industria biométrica, pues ha añadido a sus Plataformas de Desarrollo DSP (DSP Starter Kit - DSK) la posibilidad de conectarse a un dispositivo externo que realiza la captura de las huellas dactilares denominado Herramienta de Desarrollo para Verificación por Huellas Dactilares (Fingerprint Authentication Development Tool - FADT) a través de un bus de expansión, suministrando así una efectiva herramienta de bajo costo para construir sistemas de verificación por huellas dactilares con los productos basados en los procesadores de Texas Instruments, tal como se muestra en la Figura 2.

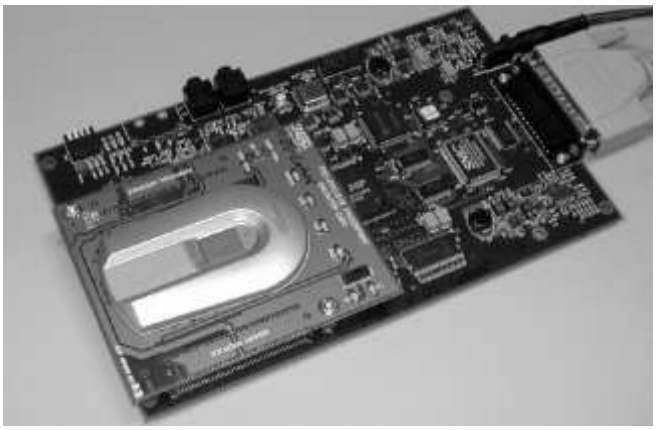

Fig. 2. Sistema FADT conectada al DSK TMS320C6711 de Texas Instruments

Utilizando las herramientas de hardware y software incluidas en el DSK es posible la construcción de un sistema autónomo de verificación por huellas dactilares, representado a través del diagrama de bloques de la figura 3. El hardware involucrado consiste básicamente en:

a) Un bloque de captura de la huella dactilar a través del sensor capacitivo FPC1010 que es monolítico versátil y compacto.
b) Un bloque de extracción de características y coincidencia realizado por el procesador DSP TMS320C6711 junto con las memorias externas incluidas en el DSK.
c) Una interfaz serial se incluye también en el DSK, que permite al DSP comunicarse con cualquier dispositivo de control y para conectarse al computador durante la inscripción de un nuevo usuario.

Se empleó el DSK TMS320C6711 del laboratorio de la Facultad de Ingeniería Electrónica de UNMSM con el objetivo de probar la funcionalidad de la técnica de verificación seleccionada, se utilizaron las bases de datos estándares para la evaluación del sistema de verificación. En esta sección se describe la implementación de los algoritmos en el procesador TMS320C6711 para la tarea de verificación de

personas por huellas dactilares, utilizando imágenes en escala de grises de ocho bits de profundidad, las que pueden ser obtenidas de cualquier tipo de escáner electrónico de huellas dactilares así como también de imágenes almacenadas en disco.

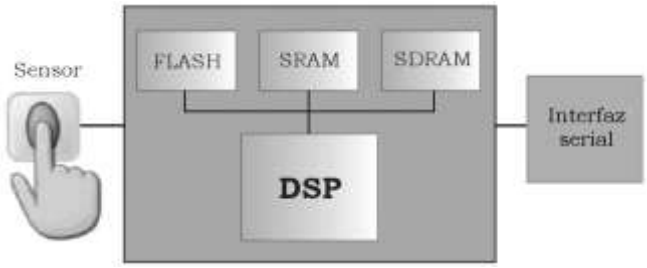

Fig. 3. Diagrama de bloques del sistema implementado

III. METODOLOGÍA DE IMPLEMENTACIÓN

Se ha elegido la técnica de verificación basada en el análisis de texturas orientadas debido a la menor complejidad y al costo computacional que requiere su implementación frente a las técnicas de correlación y minucias, en donde se muestra el buen compromiso que existe entre la complejidad algorítmica y el tiempo de procesamiento de las etapas de extracción de características y coincidencia. Por otra parte, tal como es remarcado en [2] esta técnica de verificación resulta ser muy adecuada para implementarse en un sistema embebido tal como un procesador DSP de alto rendimiento.

Existen dos formas de implementar el código para el verificador de personas utilizando una plataforma DSP:

1) Codificar manualmente todos los algoritmos en lenguaje assembler. Esta técnica provee un alto desempeño, pero requiere un elevado tiempo de desarrollo.
2) Codificar inicialmente todos los algoritmos en el lenguaje estándar ANSI C para el DSP, convertirlo a lenguaje máquina usando el compilador del CCS y luego incluir progresivamente técnicas de optimización para reducir el tiempo de procesamiento [3].

Se utilizó la segunda forma pues permite reducir el tiempo de desarrollo, luego empleamos el lenguaje assembler sólo en las partes más críticas del código.

La figura 4 muestra el diagrama de flujo de la rutina principal ejecutada desde el procesador DSP. El flujo se divide en dos partes, dependiendo del estado de la variable "fase" que indica al procesador si debe almacenar la plantilla en memoria o si debe realizar la verificación de la imagen de entrada. Los bloques que contienen (RTDX) realizan la comunicación en tiempo real con el computador utilizando uno de los canales de transmisión o recepción inicializados en el principio de la rutina a través de la librería de funciones que provee el CCS ubicada en el archivo de cabecera "rtdx.h", que permite establecer el enlace entre el DSP y el computador.

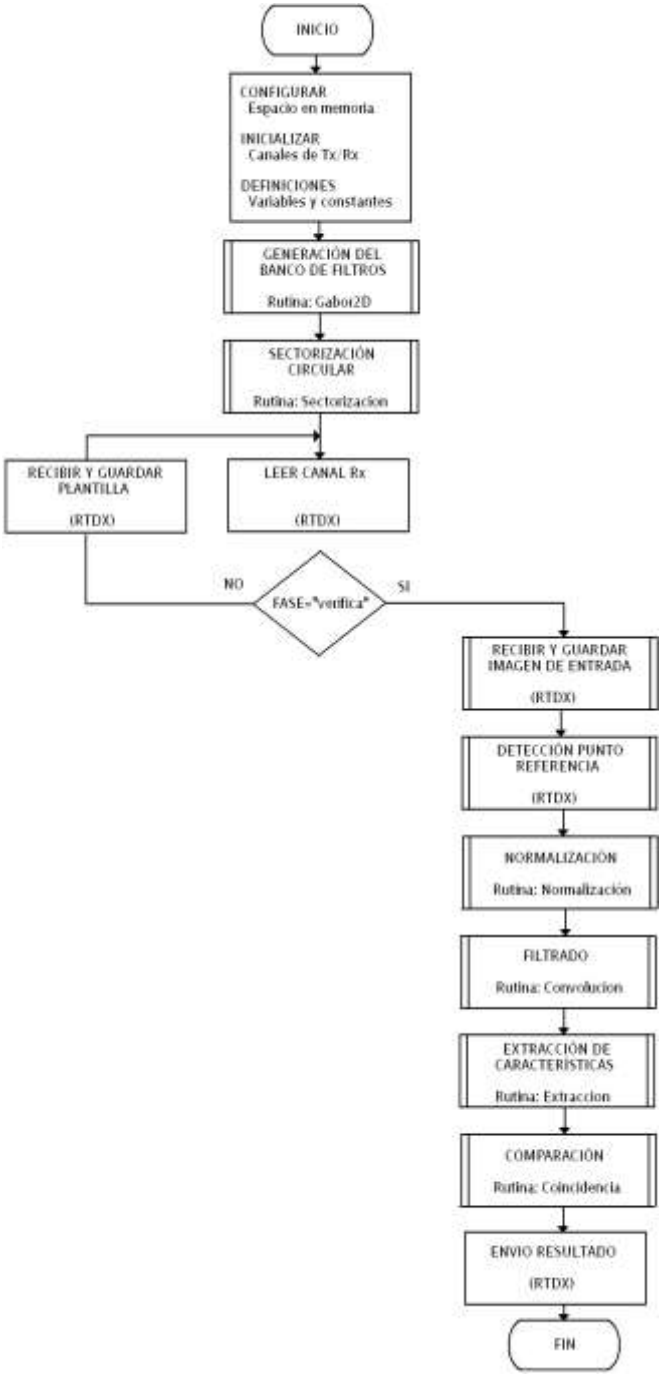

Fig. 4. Diagrama de flujo de la rutina principal implementada

A. *Rutina Gabor2D*

Esta rutina realiza la generación del banco de filtros de Gabor bidimensional (ver figura 5), es decir crea las máscaras de Gabor en las ocho orientaciones predefinidas, las cuales son utilizadas posteriormente para realizar el filtrado en el dominio espacial. Debido a las propiedades de la función de Gabor bidimensional, la creación de las máscaras requiere de las funciones matemáticas provistas por las librerías del estándar ANSI C, localizadas en el archivo de cabecera "math.h". Además, puesto que estas máscaras no cambian en el tiempo, no es necesario volver a calcularlas cada vez que se presenta una nueva imagen de entrada, por lo tanto el tiempo de procesamiento que requiere esta rutina no es considerado como parte del tiempo de extracción de características, sino más bien como parte del tiempo de inicialización del procesador.

Fig.5. Implementación de los ocho filtros de Gabor bidimensionales de diferentes orientaciones empleando el CCS

B. *Rutina Sectorización*

Esta rutina realiza la sectorización circular de la región de interés proveniente de la rutina Referencia (ver figura 6), la cual consiste básicamente en crear una matriz del mismo tamaño que la imagen recortada, que contenga las etiquetas del sector al que pertenece el píxel correspondiente de la región de interés. Debido a las propiedades geométricas de la sectorización, se requiere utilizar las funciones matemáticas del estándar ANSI C, localizadas en el archivo de cabecera "math.h". Puesto que la sectorización circular es la misma para cualquier imagen de entrada, no es necesario calcular nuevamente las etiquetas para cada nueva imagen, por lo tanto el tiempo de procesamiento que requiere puede ser considerado como parte del tiempo de inicialización.

C. *Rutina Referencia*

Esta rutina realiza la extracción del punto de referencia para determinar la región de interés (ver figura 6). Ésta tiene como entrada a la imagen a ser verificada y genera una imagen recortada que tiene como centro el punto de referencia, donde se encuentra ubicada la región de interés que será procesada y comparada por el sistema. Esta rutina requiere realizar el cálculo de los Vectores Gradiente, promediándose luego a nivel de bloques y obteniendo el punto de referencia [2]. El tiempo que requiere este procesamiento si es considerado como parte del tiempo de extracción de características.

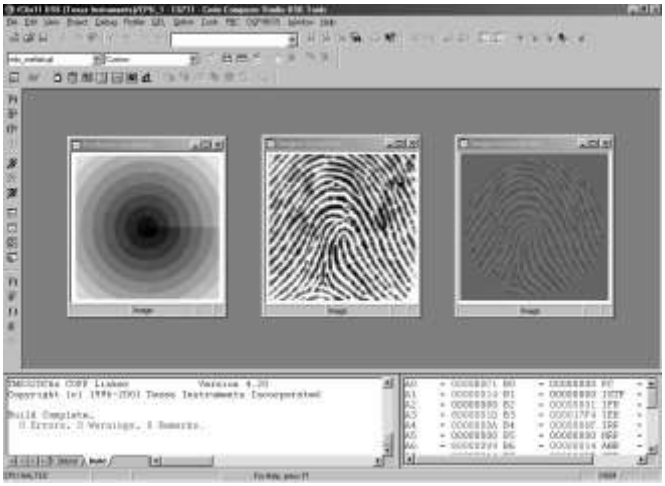

Fig.6. Implementación de los sectores circulares, detección del punto de referencia y normalización de la región de interés empleando el CCS.

D. *Rutina Normalización*

Esta rutina realiza la normalización de cada uno de los sectores circulares (ver figura 6). Ésta tiene como entradas a la imagen recortada y la matriz de etiquetas obtenidas de la rutina Sectorización y generado una nueva imagen normalizada sólo dentro de la región circular de interés. El tiempo que requiere esta rutina es parte del tiempo de extracción de características.

E. *Rutina Convolución*

Esta rutina realiza el filtrado en el dominio espacial de la región de interés con las máscaras de Gabor obtenidas de la rutina Gabor2D (ver figura 7), que tiene como entradas las máscaras de Gabor y la imagen

normalizada, generando las imágenes filtradas que a su vez son escaladas para que varíen en el rango de 0 a 255 debido a que se obtienen valores de píxeles grandes por el tamaño de las máscaras de convolución.

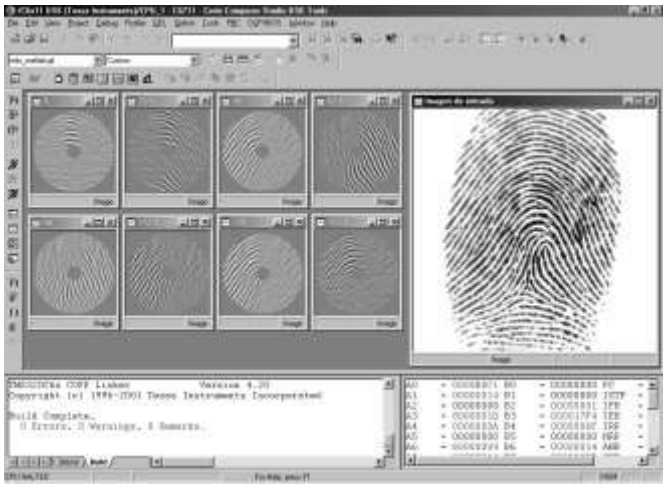

Fig. 7. Implementación del filtrado a la región de interés empleando el CCS.

Esta rutina representa cerca del 99% del tiempo de extracción de características, puesto que no sólo se requiere de un enorme número de operaciones de multiplicaciones y adiciones debido al tamaño de la imagen y de la máscara, sino también estas operaciones a su vez requieren del acceso de una gran cantidad de datos de la memoria (SDRAM externa), en donde se hayan ubicados los datos de entrada y de salida. El escalamiento es necesario para mantener los valores dentro del rango de trabajo de las variables en el procesamiento posterior.

Fig.8. Implementación del extractor del vector de características basado en la varianza empleando el CCS.

F. *Rutina Extracción*

Esta rutina calcula el vector característico denominado FingerCode (ver figura 8) a través de la varianza de la respuesta de los filtros en la imagen. Ésta utiliza como entrada las imágenes filtradas y escaladas, generando un vector de características que contiene 640 valores (bytes) para cada imagen capturada.

G. *Rutina Coincidencia*

Esta rutina realiza la comparación entre el vector de características extraído de la imagen de entrada con las versiones rotadas de las plantilla(s) previamente almacenada(s) en la memoria del DSP. El resultado de esta rutina es un valor binario basado en un valor umbral que determina si la característica biométrica del usuario es auténtica o no.

IV. INTERFAZ GRÁFICA DE USUARIO

La aplicación de comunicación entre el computador y el DSP fue realizada utilizando el software MATLAB [4], en donde se diseñó la GUI mostrada en la figura 9 y además se programaron las rutinas de comunicación con el procesador DSP a través del RTDX. Esta GUI posee dos funcionalidades: registrar y verificar.

La opción Registrar permite inscribir el nombre o PIN del usuario junto con su impresión dactilar previamente adquirida y almacenada en disco. La opción Verificar permite autenticar la identidad del usuario comparando la impresión dactilar de entrada con la previamente registrada. El resultado es un mensaje de aceptación o rechazo de acceso.

Fig. 9. Interfaz gráfica de usuario desarrollada para las opciones de registrado y verificación

IV. RESULTADOS EXPERIMENTALES

Un análisis detallado del desempeño de los algoritmos de verificación empleados fue descrito en el capítulo V. En esta sección se describen brevemente los tiempos de procesamiento de las principales rutinas que realiza el procesador DSP para la verificación. Para acelerar el proceso de desarrollo de los algoritmos, se realizó la codificación de todas las rutinas empleando lenguaje estándar ANSI C. Sin embargo, la extracción de características posee un elevado tiempo de procesamiento, principalmente la rutina de convolución espacial, debido al gran número de operaciones que realiza [5], por lo tanto se utilizaron técnicas de optimización a éstas, asignando racionalmente los tipos de datos adecuados para aprovechar el multiplicador de 16 bits, añadiendo rutinas que operan en lenguaje assembler en las partes más criticas del código, además de otras optimizaciones realizadas a nivel del compilador y descritas en [6]. Como resultado se obtuvo una considerable reducción en el tiempo de procesamiento.

Adicionalmente a estas técnicas de optimización, es posible acelerar aún más el proceso de convolución si se realiza en el dominio de la frecuencia, lo que requiere transformar tanto las máscaras de Gabor como la imagen de entrada recortada. Sin embargo, los requerimientos de memoria se hacen mucho mayores, además resulta poco útil para una aplicación real implementar un algoritmo FFT utilizando el lenguaje ANSI C, éste debe estar optimizado tanto en memoria como en tiempo de procesamiento a nivel de assembler.

La figura 10 muestra una comparación entre los tiempos de extracción utilizando un computador Pentium MMX de 166 MHz con 128 MB de RAM frente al DSP de 150 MHz con 16 MB de RAM (antes y después de optimizarlo). Es posible realizar la implementación utilizando únicamente rutinas en lenguaje assembler y assembler lineal, que permiten aprovechar el pipeline, reducir las dependencias secuenciales del código para que puedan ejecutarse en paralelo y utilizar el DMA para acelerar los procesos de carga/almacenamiento en la memoria SDRAM, sin embargo este nivel de optimización requiere un tiempo de desarrollo exponencial respecto al presentado en este capítulo y está fuera de los objetivos planteados para este trabajo, pero debe ser considerado cuando se requiera la implementación de un producto comercial.

Es importante también resaltar una situación que debe ser considerada, al codificar un algoritmo en un procesador DSP de punto flotante se generan muchas más instrucciones en lenguaje máquina que en un procesador de punto fijo, tal como se demostró en [7], en donde se realizó una comparación entre el tiempo de procesamiento para un mismo algoritmo utilizando un procesador de punto flotante y uno de punto fijo, determinándose una proporción de uno a trece.

Estos resultados parecen convertir a los procesadores de punto fijo como un candidato bastante fuerte para realizar el proceso de verificación, pues podría alcanzar tiempos de procesamiento inferiores a los tres segundos, lo que lo haría adecuado sobretodo en aplicaciones de tiempo real.

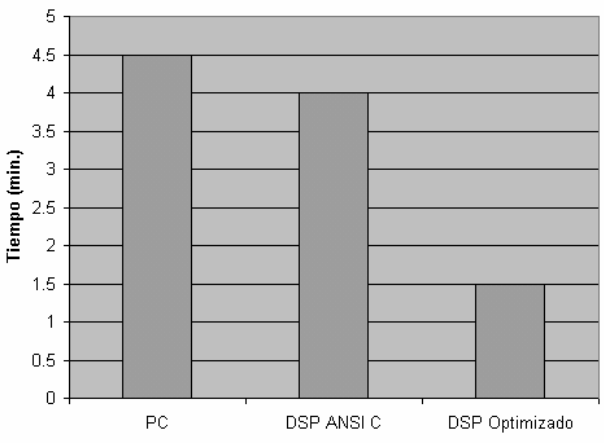

Fig. 10. Comparación en el tiempo de extracción de características en un computador personal, en DSP optimizado

V. CONCLUSIONES Y TRABAJO FUTURO

Se ha implementado una técnica de verificación de personas en un DSP utilizando la técnica de representación basada en un banco de filtros, permitiendo aprovechar tanto las características locales como globales en las huellas dactilares para mejorar la toma de decisiones en la verificación de la identidad.

Se ha comprobado de manera sencilla el correcto funcionamiento en cada etapa de los algoritmos codificados en el DSP, además han sido comparados con los resultados obtenidos en el computador gracias a las herramientas de visualización gráfica provistas por el IDE del CCS. Es importante recalcar las ventajas que presenta la optimización del código y la inclusión de rutinas en lenguaje assembler en las partes más críticas de éste, que reducen considerablemente el número de los ciclos de reloj que requiere su ejecución.

Se ha implementado una técnica de verificación de personas sobre un DSP utilizando la representación basada en un banco de filtros de Gabor bidimensional, puesto que la extracción de características y la coincidencia son más factibles de implementarse en hardware que los métodos tradicionales basados en minucias, caracterizados porque su desempeño depende fuertemente de la calidad de la impresión dactilar.

Se ha comprobado de manera simple el funcionamiento correcto en cada etapa acerca de los algoritmos codificados en el DSP, además se han comparado con los resultados obtenidos en el computador gracias a las herramientas de visualización gráfica provistas por el IDE del CCS. Es importante recalcar las ventajas que presenta la optimización del código y la inclusión de rutinas en lenguaje assembler en las partes más críticas de éste, que reducen considerablemente el número de ciclos de reloj que requiere su ejecución. Un componente clave en esta implementación ha sido el RTDX, pues nos ha permitido establecer la comunicación e intercambiar datos, incluso imágenes en tiempo real desde el computador al DSP y por lo tanto evaluar adecuadamente el funcionamiento de la verificación. Sin embargo, esta valiosa herramienta posee como principal limitación un "ancho de banda" angosto, pues ha sido diseñado principalmente para aplicaciones de control.

El siguiente paso en el desarrollo de un verificador de personas por huellas dactilares presentado en este trabajo consiste en el análisis de los algoritmos empleados en este trabajo utilizando un procesador DSP de punto fijo, lo que nos asegura que servirá para los requerimientos de tiempo real que son necesarios para todo dispositivo autónomo.

Una considerable mejora en el tiempo de extracción de características del FingerCode puede ser realizada a través de la convolución en el dominio de la frecuencia, una excelente alternativa puede ser utilizar la biblioteca de procesamiento digital de señales para la familia C6000 escrita en lenguaje assembler denominada DSPLIB para realizar la transformada rápida de Fourier bidimensional a través de la unidimensional, y aprovechar tanto la parte real como la compleja, procesando dos filas o columnas de la imagen en una sola transformación.

Varias mejoras pueden realizarse tanto para el procesamiento como para el reconocimiento por huellas dactilares a través de las herramientas de desarrollo DSP en tiempo real. Por ejemplo, puede emplearse una Plataforma de desarrollo EVM basada en el procesador DSP TMS320C6201 (de punto fijo), que posee una interfase PCI para establecer una comunicación mucho más rápida con el computador, de tal forma que permita realizar la tarea de reconocimiento fusionando la técnica basada en minucias (en el microprocesador) y la basada en texturas (en el procesador DSP) ejecutándose ambas en paralelo y tomando una decisión más acertada basada en ambos índices de similitud, obteniéndose mejores resultados en un tiempo de procesamiento mucho menor.

REFERENCIAS

[1] J. Wayman, et al., *Biometric Systems: Technology, Design and Performance Evaluation*, Springer Verlag London, 2005

[2] S. Prabhakar, *Fingerprint Classification and Matching Using a Filterbank*. PhD Thesis, Michigan State University, 2001

[3] Texas Instruments, *TMS320C6000 Programmer's Guide*, (SPRU198F) USA, 2001.

[4] C. Pérez *Matlab y sus Aplicaciones en las Ciencias y la Ingeniería*. Prentice Hall, 2002.

[5] P. Embree, B. KimballL , *C Language Algorithms for Digital Signal Processing*. Prentice Hall, 1991.

[6] Texas Instruments, *TMS320C6000 Optimizing Compiler User's Guide*, (SPRU187I) USA, 2001

[7] P. Chen et al., *Enabling Fingerprint Authentication in Embedded Systems for Wireless Applications*, The Chinese University of Hong Kong, 2002.