Identificación de usuarios mediante el uso de reconocimiento de huellas dactilares.

Dal Verme Tomás, De Arias Axel, Figueredo Nicolás Leandro, Quinteiro Lucas, Rodríguez Maximiliano Pablo.

Universidad Nacional de La Matanza,
Departamento de Ingeniería e Investigaciones Tecnológicas,
Florencio Varela 1903 - San Justo, Argentina
tdalverme@gmail.com; maxirodriguez91@gmail.com; axeldearias@gmail.com;
nico_1507@live.com.ar; quinteiro.lucas95@gmail.com

Resumen. El objetivo de la siguiente investigación es el de ofrecer el reconocimiento de los distintos usarios de SmartBartman mediante sus huellas dactilares, de forma eficiente y rápida, utilizando técnicas de paralelismo para el procesamiento de la información de las msimas. Para ello se utilizará un algoritmo de extracción de puntos característicos de las huellas (minucias), para su posterior comparación con la base de datos de SmartBartman que contiene las huellas de todos los usuarios, asociadas a sus preferencias de bebidas.

Palabras claves: CUDA, thread, huellas dactilares, IOT, paralelización, HPC.

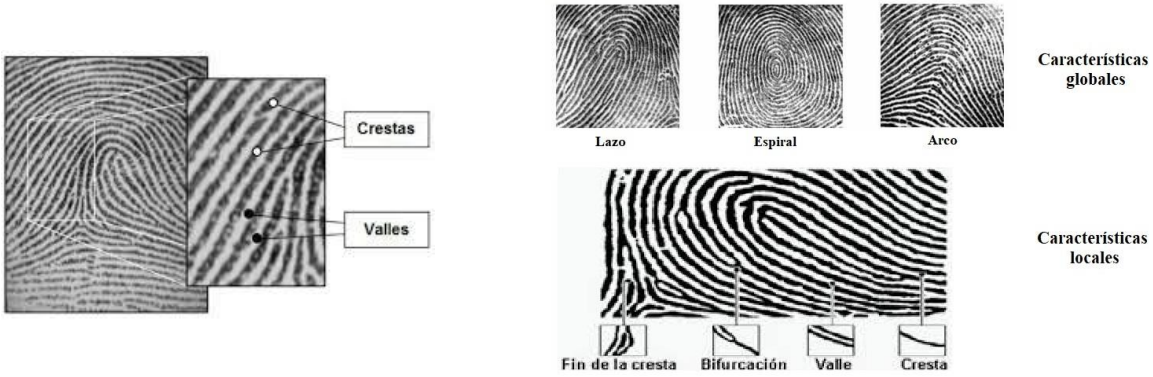
1 Introducción

El trago perfecto perfecto no existe, pero si existe el mejor trago para cada persona. El principal objetivo de SmartBartman es el de aprender las preferencias de sus usuarios y así lograr preparale su trago ideal. Para lograrlo, cuenta con un sistema embebido encargado de servir la bebida obteniendo información de sus distintos sensores, y una aplicación movil, que se conecta con el mismo a través del protocolo Bluetooth, en donde el usuario podrá consultar su nivel de alcohol en sangre y elegir el trago a preparar. Una vez finalizado, el usuario deberá responder una serie de preguntas que ayudará a SmartBartman a comprender sus preferencias y actualizarlas en la base de datos para que el próximo trago se aproxime mejor a su trago “perfecto”. Además, se utilizarán algunos sensores del smartphone (sensor de luz, giroscopio, etc.) que serán empleados para enviar instrucciones al sistema embebido, como por ejemplo, empezar a servir o prender LEDs.
Actualmente, la identificación de los usuarios se hace a través del smartphone, es decir, un usuario está asociado a un dispositivo movil, por lo que si cambiara de teléfono, perdería todos sus datos y preferencias asociadas. Este es el principal motivo que impulsa la incorporación del reconocimiento por medio de huellas dactilares, de forma que si un usuario desea utilizar la aplicación en otro smartphone (o en un futuro en otro dispositivo inteligente que soporte la aplicación), pueda contar con sus datos y poder preparar el trago que desee a su medida.

Para obtener la huella dactilar del usuario se utilizará el sensor de huella del smartphone. Estos sensores hoy en día permiten conseguir imágenes de alta reslución y por lo tanto se requiere de un gran procesamiento de información. Es por esto que proponemos utilizar la tecnología GPU con la que cuenta el smartphone para realizar las tareas de procesamiento de las huellas, paralelizando la carga de trabajo en distintos threads, y así lograr una respuesta rápida y eficiente.

2 Desarrollo

Existen principalmente dos métodos para el procesamiento e identificación de huellas dactilares: características locales y patrones globales. El primer método se basa en el análisis y comparación de lo que se conoce como minucias, que son bifurcaciones o terminaciones de las crestas de las huellas. En cambio, el segundo método realiza una aproximación macroscópica de la huella, observando el flujo de las crestas, clasificándolas en arcos, lazos y espirales.



En nuestro caso, utilizaremos el método de localización y comparación de minucias (características locales). El mismo consta de seis etapas:

1. **Adquisición.** En esta etapa se obtiene la huella del usuario a comparar mediante el sensor del smartphone.
2. **Aclaración.** En la práctica, las condiciones de la piel, el ruido de los sensores, la presión del dedo al apoyarlo en el sensor, entre otros aspectos pueden resultar en una mala calidad de la imagen y por lo tanto la detección de falsas minucias. Para ello se utilizan técnicas de mejora de la imagen (en nuestro caso mediante la transformada de Fourier) que permiten recuperar las zonas borrosas e interrumpidas usando la información contextual. Además, se realiza el proceso de binarización de la imagen, donde la imagen resultante solo queda formada por unos (pixel blanco) y ceros (pixel negro).
3. **Adelgazamiento.** Durante esta etapa se obtiene una imagen donde las crestas son de un solo pixel de grosor.

4. **Extracción de minucias.** Se divide la imagen en ventanas de 3x3, y se calcula el número de píxeles que cruzan el píxel central. Si resulta igual a 7, se dice que hay una minucia del tipo terminación, si es igual a 6 se dice que no hay minucia y si es menor o igual a 5 se dice que existe una bifurcación.
5. **Eliminación de falsas minucias.** Se analizan las minucias encontradas y se descartan aquellas que se cree que son minucias falsas.
6. **Reconocimiento.** En esta etapa se compara la matriz que contiene las minucias encontradas con cada una de las matrices de la base de datos. Si al compararla con alguna, resulta que las minucias coinciden, el usuario es identificado correctamente.

Huella tomada por el sensor | Huella mejorada | Huella binarizada | Huella adelgazada | Extracción de minucias

Realizar todos estos pasos es muy costoso computacionalmente y es aquí donde entra en escena la tecnología GPU que nos da la posibilidad de realizar multiprocesamiento a través de miles de threads ejecutando paralelamente.

3 Explicación del algoritmo

Debido a que la imagen se ve dividida en la etapa de aclaración (en bloques de 32x32) y luego en la etapa de extracción de minucias (en bloques de 3x3), podemos explotar las capacidades que nos brinda la tecnología GPU para paralelizar la carga de trabajo, y realizar el procesamiento más rápidamente.
Durante la etapa de aclaración, al realizar la transformada de Fourier a cada bloque de píxeles, podemos aprovechar y asignar cada píxel a un thread de forma que cada hilo sea procesado por un CUDA core de forma paralela. Si las imágenes obtenidas fueran de 128x128 píxeles, se obtendrían cuatro bloques de 32x32 píxeles y estos bloques podrían organizarse en una grilla de cuatro bloques, donde cada bloque está dispuesto en dos dimensiones de 32x32, resultando en 1024 threads por bloque. Por lo tanto, para el procesamiento de la imagen completa, se necesitarán 4096 hilos, cada uno asignado a un único CUDA core.
En cambio, durante la etapa de detección de minucias se divide la imagen en bloques de 3x3, obteniendo 43 bloques. Para esta etapa se podría organizar en una grilla de 1821 bloques y cada bloque dispuesto en dos dimensiones de 3x3, resultando en 9 threads por bloques. Además, podríamos hacer uso de la directiva ParallelReduction de CUDA para reducir cada matriz de 3x3 de forma paralela, y por lo tanto más rápidamente, y de acuerdo con el resultado obtenido clasificarlo en minucia o no.
Al aplicar todas estas mejoras lograremos reducir notablemente el tiempo de procesamiento, ya que estamos paralelizando complejas tareas y cálculos

matemáticos, logrando una rápida respuesta a un problema que parecía mucho más complejo debido al gran volumen de datos a procesar. A continuación se presenta el pseudocódigo de cómo se realizaría el algoritmo:

```
/** Obtiene la huella dactilar a comparar **/
Imagen huella = leerImagen(path);

/** Convierte la imagen en unos (pixeles blancos) y ceros (píxeles negros) **/
Imagen huellaBinaria;
huellaBinaria = binarizar(huella, 0, 255);

/** Divide la imagen en bloques de 32x32 píxeles **/
Imagen bloquesFourier[] = dividirImagen(huellaBinaria, 32, 32);

/** Se suben todos los bloques a la GPU **/
for(Imagen bloqueActual in bloquesFourier) {
	/** Se sube a la GPU un bloque de 32x32 threads **/
	subirBloqueGPU(bloqueActual, 32, 32);
}

/** Se ejecuta la transformada de Fourier sobre cada bloque de 32x32 en la GPU **/
Mat huellaMejorada = fourier();

/** Se ejecuta el algoritmo de adelgazamiento **/
Imagen huellaAdelgazada = adelgazarHuella(huellaMejorada);

/** Divide la imagen en bloques de 3x3 píxeles **/
Imagen bloquesMinucias[] = dividirImagen(huellaAdelgazada, 3, 3);

/** Se suben todos los bloques a la GPU en bloques de 3x3 threads **/
for(Imagen bloqueActual in bloquesMinucias) {
	subirBloqueGPU(bloqueActual, 3, 3);
}

/** Se obtienen las minucias de la huella. El método obtenerMinucias() debería hacer
uso de la primitiva ParallelReduction de CUDA **/
Huella huellaConMinucias = obtenerMinucias();

/** Se compara cada entrada de la base de datos con la huella obtenida, si coinciden
se permite el ingreso a la aplicación y se obtienen las preferencias **/
Preferencia preferencias;
for(Entrada entrada in BaseDeDatos) {
	if(entrada.comparar(huellaConMinucias) == true) {
		permitirIngreso();
		preferencias = entrada.getPreferencias();
	}
}
```

4 Pruebas que pueden realizarse

Para probar que todo funcione correctamente simplemente hay que registrarse en la aplicación ingresando la huella dactilar por primera vez, preparar varios tragos de manera que las preferencias por defecto se vean modificadas, y por último intentar utilizarla en otro smartphone y verificar que el sistema nos permite el ingreso correctamente, obteniendo las mismas preferencias que se tenían en el anterior dispositivo.

5 Conclusiones

Este estudio nos permitió demostrar que la idea de agregar el reconocimiento de huellas dactilares a SmartBarman es totalmente factible y una importante mejora, ya que actualmente si un usuario cambia de dispositivo, perderá todos sus datos y preferencias.
Mediante esta investigación pudimos dilucidar las complejidades que conlleva la computación de altas prestaciones, pero que son indispensables para el procesamiento de algoritmos costosos computacionalmente o cuando el volumen de datos es muy grande. Además, nos permitió conocer en profundidad el funcionamiento de los algoritmos de reconocimiento de huellas dactilares y lo complejos que son, por lo que resulta de vital importancia la necesidad de paralelizar su procesamiento y maximizar el uso de los recursos disponibles (en este caso, la tecnología GPU).
En un futuro se podría agregar al sistema un módulo que permita el pago de los tragos, de forma que el sistema pueda comercializarse a boliches donde cualquier persona pueda elegir el trago a preparar y pagar haciendo uso de su celular (ya sea mediante un código QR, NFC, tarjeta de crédito, etc.)

6 Referencias

1. Chambi Mamani, E.W.: Reconocimiento y detección biométrico basado en imágenes de huellas digitales. (2016)
2. Sarzuri Flores, V.: Algoritmo de clasificación de huellas dactilares basado en redes neuronales función base radial. (2014)
3. Enriquez Aguilera, F., Silva Aceves, J.M., Torres Argüelles, S.V., Martínez Gómez, E.A., Bravo Martínez, G.: Utilización de GPU-CUDA en el procesamiento digital de imágenes. (2018)
4. Aguilar, G., Sánchez, G., Toscano, K., Nakano, M., Pérez, H.: Reconocimiento de huellas dactilares usando características locales. (2008)
5. López García, J.: Algoritmo para la identificación de personas basado en huellas dactilares. (2018)
6. Ruiz, M., Morales, M., Hernández, Y.: Una estrategia de segmentación de imágenes digitales de huellas dactilares latentes. (2011)