

# Reconocimiento de Mate por imagenes

Maximiliano Lima, Matias Tomaino, Ivan Baca, Pablo Martin Cruz

<sup>1</sup>Universidad Nacional de La Matanza,  
Departamento de Ingeniería e Investigaciones Tecnológicas,  
Florencio Varela 1903 - San Justo, Argentina  
Maxi.lima94@gmail.com, MatiasTomaino@live.com.ar, Ivane.baca@gmail.com,  
pablocruz90@gmail.com

**Resumen.** La investigación presentada tiene como fin conocer el mundo del reconocimiento de imágenes a través de patrones. El fin se basa en incorporar al proyecto “Smarte” la inteligencia suficiente para detectar mates que requieran cambio de yerba. Para la investigación se va a hacer uso del algoritmo SURF (Speeded-Up Robust Features) para reconocer patrones en el mate, como puede ser palos flotando o el tono de la yerba mas oscuro, que indiquen la necesidad de realizar un cambio de Yerba.

**Palabras claves:** SURF, CUDA, GPU, OpenCv, Machine Learning.

## 1 Introducción

Debido a la voragine de los tiempo actuales, las presiones laborales y la urgencia constante en el ambito laboral, las personas deben optimizar cada minuto de la jornada laboral y reducir al minimo las preocupaciones extra laborales. Es por esto que con el estudio del estado del mate a través de imágenes y patrones se busca quitar al usuario la tarea de verificar en cada mate el estado del mismo y reducir el tiempo que esto conlleva.

Existen varios metodos y algoritmos hoy en dia para el reconocimiento de imágenes y patrones. Dentro de los algoritmos más utilizados se encuentra SIFT, SURF, FAST, ASIFT entre otros. En este estudio se utilizará SURF, una evolucion del metodo SIFT. Si bienambos algoritmos obtienen los puntos de interes invariantes de la imagen en escala, orientacion e iluminacion el algoritmo SURF permite una velocidad de aplicacion mayor lo que hace que el algoritmo pueda ser utilizado en aplicaciones de tiempo real.

El algoritmo SURF permite la deteccion de puntos caracteristicos basado en la aproximacion de la matriz Hessiana y los puntos caracteriticos se detectan utilizando el maximo valor de la determinante de la matriz.[1]

## 2 Desarrollo

Para llevar a cabo la implementación del algoritmo SURF se requiere un dispositivo que contenga GPU incorporado y compatible con el manejo de instrucciones en paralelo a través del lenguaje CUDA.

Para el desarrollo se utilizarían las siguientes herramientas:

Lenguaje C++: Facilita el procesamiento de imágenes a bajo nivel. Es muy utilizado en el procesamiento de imágenes y es portable a múltiples plataformas.

Librería OpenCv: Librería de C++ que permite conseguir imágenes a través de la webcam u otros dispositivos. Es soportado por Windows y Linux.

Android NDK: Conjunto de herramientas para poder utilizar código C/C++ en aplicaciones Android. Se suele utilizar en aplicaciones que requieran procesamiento de grandes cantidades de datos a la mayor velocidad posible.

Código CUDA: Fue diseñado para computación de Propósito General utilizando la GPU a través del lenguaje C. Existe un toolkit creado y distribuido por NVIDIA que contiene los drivers y el software necesario para implementarlo. CUDA considera al GPU como una plataforma independiente que puede proveer de un ambiente de desarrollo minimizando el entendimiento del pipeline gráfico. Además, la CPU junto con la GPU conforman un sistema heterogéneo.

Algoritmo SURF: Se utiliza para detectar puntos clave y extraer puntos de interés de imágenes. Se puede aplicar SURF utilizando los datos provistos por OpenCV que además de leer imágenes y es capaz de aplicar Machine Learning.[2]

SURF utiliza una detección de puntos de interés o keypoints. Dada una imagen, el detector de SURF se aplica para detectar keypoints basado en aproximaciones de la matriz Hessianas.

Luego se representa las cercanías de cada punto de interés en un vector. Las partes vecinas de cada keypoint, en forma de cuadrados, son almacenadas junto con los puntos de interés[4]. Dependiendo del proyecto, el tamaño del vector puede variar. En este caso, por el tamaño del proyecto, se podría utilizar un vector de 64 dimensiones[4].

Finalmente, para comparar dos imágenes, un algoritmo de matcheo deberá ser aplicado. La comparación de imágenes se realiza utilizando el signo de Laplace, distinguiendo los puntos brillantes en fondos oscuros. Finalmente, las imágenes matchean cuando contienen el mismo contraste (basado en el signo de Laplace)[4]

## 3 Explicación del algoritmo.

El algoritmo se basaría en la comparación de patrones entre una imagen del mate en el estado actual y una base de conocimientos previa. La base ya vendría precargada y se usaría para comparar patrones del mate con patrones de las imágenes que se encuentran en la base.

En una primera instancia se aplicaría el algoritmo SURF a las imágenes precargadas de mates lavados para encontrar regiones críticas o puntos de interés. Una vez que se obtengan los puntos de interés en común en todos los mates lavados se

tendrá una base de conocimiento apta para realizar la comparación con mates en tiempo real en nuestro producto final.[3]

Para realizar la comparación en tiempo real, se podría, en primera medida, definir un porcentaje de afinidad. Cuando más alto el porcentaje, mas acertado será el resultado pero con menos aciertos. Una vez definido el porcentaje adecuado y equilibrado, el algoritmo para identificar mates lavados sería el siguiente:

Aclaración: dado que la temperatura del agua influye en la rapidez en la que un mate se lava, la primera foto se tomaría en el rango de los primeros 5 (si el agua se encuentra hirviendo) a los 15 (si el agua se utilizó a una temperatura de 75 grados) mates cebados. Luego de obtener la imagen, el algoritmo se aplicaría cada 3 mates cebados:

Cada 3 mates cebados {

**getFotoMate:** Obtener la foto del mate que se encuentra en el producto Smarte. Para obtener la foto se utilizaría la librería OpenCv.

**dividirImagenEnRegiones** = Dividir la imagen del mate en regiones. Dado que el algoritmo SURF nos permite trabajar en regiones.

**cargarRegionesAGPU** = Reservar la memoria necesaria en la GPU para aplicar el algoritmo en la imagen obtenida.

**ejecutarSURF** = Ejecutar el algoritmo SURF utilizando la ventaja de la GPU. Es decir, diviendo en threads el procesamiento de cada sección de la imagen.

**obtenerResultadoDeGPU** = obtener el resultado de la comparación realizada.

**alertarUsuario** si el resultado de acierto se encuentra dentro del porcentaje de afinidad = Si el resultado obtenido se encuentra dentro de los umbrales seteados de acierto, se informará al usuario que debe cambiar la yerba.

**actualizarBaseDeConocimiento** si el mate se encuentra lavado = para aumentar la cantidad de aciertos, si el mate se encuentra lavado, se actualizará la base de datos con los puntos de interés detectados en la imagen analizada.

}

#### 4 Pruebas que pueden realizarse

Las pruebas podrían realizarse con la utilización del producto en si. Para poder hacer uso del algoritmo, además de la programación del mismo, solo ser requerirá una cámara capaz de fotografiar el mate, un celular con GPU (muchos en la actualidad poseen), una base de datos con al menos imágenes de 10 mates lavados y el prototipo del SMARTE funcionando. Aplicando el algoritmo desarrollado se podrá realizar las pruebas necesarias.

## **5 Conclusiones**

El procesamiento de imágenes en tiempo real es complejo y pesado de realizar. Gracias a algoritmos complejos como SURF o SIFT en conjunto con las ventajas que brinda el procesamiento en paralelo utilizando la GPU de los celulares hace que el procesamiento de imágenes en tiempo real sea algo que se pueda aplicar en dispositivos móviles.

## **6 Referencias**

1. Junchul Kim, Younghun Jung, Xuenan Cui, Hakil Kim: A Fast Feature Extraction in Object Recognition Using Parallel processing on CPU and GPU.(2009)
2. Suping Wu, Bing Feng.: Proceedings of the 2019 International Conference on Modeling, Simulation, Optimization and Numerical Techniques (SMONT 2019). (2019).
3. Karel Horak, Jan Klecka, Ondrej Bostik, Daniel Davidek: Classification of SURF Image Features by Selected Machine Learning Algorithms.(2017)
4. Ebrahim Karami, Siva Prasad, Mohamed Shehata. Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparision for Distorted Images. (2017).