Reconocimiento de objetos con GPU en Boya inteligente

Hector Rojas Stoll1, Juan Tocino, Ezequiel Laurenti, Damian Vinci

1Universidad Nacional de La Matanza,

Departamento de Ingeniería e Investigaciones Tecnológicas,

Florencio Varela 1903 - San Justo, Argentina

[Hector.rojass18@gmail.com](mailto:Hector.rojass18@gmail.com), [tocino.juan@gmail.conm](mailto:tocino.juan@gmail.conm), [ezee.laurenti@gmail.com](mailto:ezee.laurenti@gmail.com), [damianvinci@gmail.com](mailto:damianvinci@gmail.com)

**Resumen.** Este trabajo de investigación intenta comprobar si es posible utilizar reconocimiento de objetos en nuestro proyecto de Boya inteligente a través de imágenes captadas por una camara integrada a la boya y de esta manera poder dar avisos en su correspondiente aplicación de android “BoyApp” sobre la presencia de objetos en una pileta, utilizando procesamiento en paralelo con GPU.

**Palabras claves:** Boya inteligente, BoyApp, HPC, GPU, OpenCV, Object Recognition, Reconocimiento de objetos.

1. Introducción

Esta investigacion se basa en añadir a nuestro proyecto Boya inteligente, que consiste en realizar distintas mediciones tales como temperatura, opacidad o movimiento en una pileta que nos den una idea sobre el estado en tiempo real de la misma para poder tomar una decision y realizar una accion de forma remota desde una aplicación movil o que la haga el embebido de forma automática y poder mantener siempre la pileta en optimas condiciones, una nueva funcionalidad que haga uso de computacion de altas prestaciones (HPC) y base su solucion en alguno de los paradigmas vistos en clase (OpenCL, MPI o GPU).

En nuestro caso, haremos uso del modulo GPU integrado en nuestro dispositivo Android, que a través del procesamiento de imágenes e informacion de forma paralela nos permititra saber en tiempo real, si existen objetos en el fondo de la pileta brindandonos además informacion de los mismos, y a su vez, le da al usuario la posibilidad de seguir utilizando las funcionalidades convencionales de la app y la boya sin afectar la performance o tiempos de respuesta.

La motivacion del desarrollo de esta nueva funcionalidad es por un lado saber sobre la acumulacion de agentes externos que colaboran con la descomposicion del agua tales como algunos insectos, moluscos o incluso hojas, sobre todo en epoca otoñal, y por otro lado la deteccion de objetos personales que a las personas se les suelen caer y perder dentro del agua como por ejemplo joyas las cuales son importantes ubicar y recuperar rapido.

El tema del analisis de imágenes para la deteccion y obtencion de informacion de objetos es un tema que se viene desarrollando desde hace tiempo y ya existen aplicaciones que cumplen estan funciones, por ejemplo una de las formas mas basicas de reconocimiento de imágenes es el codigo de barras que se comenzo a utilizar para el reconocimiento de objetos en almacenes e industrias y mas tarde se haria popular su uso en supermercados. Luego surgio el codigo bidimensional (QR) que permite almacenar mucha mas informacion que un codigo de barra, hasta llegar a hoy en dia en donde una de las aplicaciones más populares de reconocimiento de imágenes en dispositivos móviles es la aplicación de Google: Google Goggles, la cual es capaz de, al hacer una fotografía, reconocer ciertas imágenes que aparecen en ella y hacer una consulta en su motor de búsqueda al respecto, además de otras más específicas como Naruto Card Scanner o Power Rangers Scanner, de Bandai América, que reconocen cartas de colección de dichas series.

1. Desarrollo

Para aplicar la funcionalidad propuesta en esta investigacion, como primer medida, necesitaremos integrar a la boya una camara como medio de obtencion de informacion, para que, de esta manera, paralelamente se pueda procesar la misma y llevar a cabo el reconocimiento del objeto.

Las imágenes captadas por la camara son enviadas a la placa arduino a la cual estara conectada, y luego seran enviadas al dispositivo movil donde las imágenes seran procesadas y analizadas mediante el modulo GPU con el fin de detectar la presencia de algun objeto en el fondo de la pileta, este modulo ademas mediante algoritmos de reconocimiento de patrones y puntos caracteristicos comparara esta informacion con otra almacenada en una base de datos que implementaremos y donde se encontraran imágenes de posibles obejtos que se puedan encontrar en una pileta generalmente, con su respectiva informacion.

Finalmente, una vez reconocido el objeto, en el caso de tratarse de hojas o bichos solo se enviara una notificacion al dispositivo movil avisando de la presencia del mismo, y en caso de tratarse de objetos personales o no identificados se procedera ademas a enviar la imagen del mismo acompañado de su informacion en caso de tenerla, para que el usuario sepa de antemano si solo necesita limpiar o el fondo o se trata de la deteccion de ese objeto perdido tiempo antes por el o alguien mas y realizar la accion correspondiente.

El diseño implementado permitirá que toda esta dinámica se haga en el dispositivo, sin tener que acceder a Internet para ello, y devolviendo resultados en un tiempo aceptable para el usuario, para lo cual utilizaremos algoritmos basados en el uso de técnicas para el reconocimiento de imágenes tales como puntos característicos o keypoints y descriptores.

Un punto característico es un lugar concreto de una imagen que contiene una propiedad que la define. Esto puede ser, por ejemplo, esquinas, bordes, formas u otros elementos que representen una particularidad de la imagen. Existen multitud de tipos de puntos característicos, entre ellos SIFT y SURF, que son los más extendidos, y otros como los BRIEF o KAZE que por sus características, son menos eficientes en su funcionamiento, aunque usados en ocasiones concretas. Una vez extraídos los puntos característicos, se genera un descriptor para cada uno de ellos. Los descriptores son estructuras que contienen varios datos sobre el punto y su entorno, para poder definir un contexto para el mismo. Idealmente, estos descriptores deben ser invariantes a la escala, traslación o rotación de la imagen, así como cambios de iluminación o ruido. Otro de los tipos de punto y descriptor más conocidos son los tipo ORB (Oriented FAST and Rotated BRIEF), la combinación de FAST y BRIEF, el cual podría plantarle cara a los puntos SIFT y SURF en cuestión de eficiencia y precisión.

En nuestro proyecto se utilizaran los puntos y descriptores ORB, ya que ofrecen ciertas ventajas:

1. Son más rápidos en su procesamiento, ya que sus descriptores son binarios.

2. En cuestión de precisión, son equivalentes a SIFT y SURF, que son los más extendidos.

3. ORB es de uso libre, mientras que SIFT y SURF operan bajo derechos de autor.

1. Explicación del algoritmo.

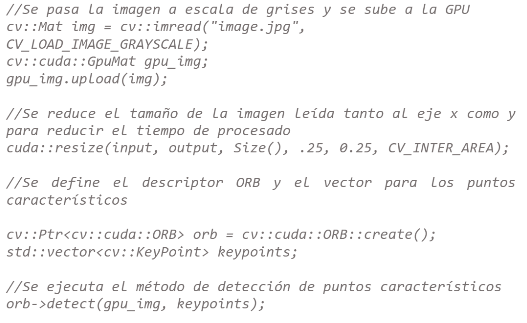
En primera instancia se realiza el emparejamiento de puntos, proceso mediante el cual los descriptores de dos imágenes son comparados y unidos en parejas. Existen varios métodos para este proceso, el más sencillo es el del vecino más próximo, dónde se calcula la distancia entre los descriptores y se emparejan con el más cercano.

Una vez se tienen los puntos emparejados, se busca una homografía que transforme los puntos de la primera imagen en la segunda. Una homografía es una transformación proyectiva entre dos figuras geométricas planas. En general se consideran que dos imágenes son equivalentes si existe dicha transformación. Para determinar los parámetros de la transformación, normalmente se utilizan sólo unas pocas parejas de puntos. La transformación resultante se aplica al resto de puntos, comparando el punto transformado con el punto emparejado. La homografía debe ser independiente del tamaño, localización y rotación del objeto de la escena. Este procedimiento es usado para comprobar qué cantidad de puntos de la imagen de entrada pertenecen a la imagen de la base de datos y si estos coinciden. Si se dispone de las suficientes parejas buenas, la imagen ha sido reconocida.

Para poder hacer programas utilizando los procedimientos descritos en este apartado, se utilizara la interface de Android de OpenCV, diseñado para la eficiencia computacional con fuerte enfoque en aplicaciones de tiempo real, del cual utilizaremos concretamente sus funciones relativas al tratamiento de imágenes (como pasar de una foto en color a otra en escala de grises) y funciones para la extracción de puntos característicos, así como sus descriptores, emparejamientos, homografía y vecinos más próximos.

Además debido a que necesitamos paralelizar el trabajo para agilizar los tiempos de procesamiento, será necesario aplicar la arquitectura CUDA, (Arquitectura Unificada de Dispositivos de Cómputo) que hace referencia a una plataforma de [computación en paralelo](https://es.wikipedia.org/wiki/Computaci%C3%B3n_paralela) incluyendo un [compilador](https://es.wikipedia.org/wiki/Compilador) y un conjunto de herramientas de desarrollo creadas por [nVidia](https://es.wikipedia.org/wiki/NVidia" \o "NVidia) que permiten a los programadores codificar algoritmos en [GPU](https://es.wikipedia.org/wiki/GPU).

El siguiente código ejemplifica el uso de con el algoritmo ORB.



En este ejemplo se aplica paralelismo al trabajar con la imagen, lo cual será necesario ya que al ser imágenes tomadas bajo el agua se necesitara de mucha resolución de la cámara, lo que se traduce en más pixeles a analizar y por lo tanto más carga computacional, y luego se utiliza el algoritmo ORB para detectar los keypoints necesarios para realizar la comparación con otra imagen.

1. Pruebas que pueden realizarse

Las pruebas que podrían realizarse para verificar el correcto funcionamiento de esta función implican arrojar diferentes objetos al fondo de la pileta, ya sean las que están cargadas en nuestra BD como otros que no lo estén, a partir de esto la boya debería detectar la presencia de los mismos enviando la correspondiente notificación al dispositivo móvil con un aviso, una imagen del objeto con o sin información adicional según corresponda.

1. Conclusiones

A través de esta investigación intentamos demostrar que se puede implementar el reconocimiento de objetos mediante la integración de una cámara en nuestro proyecto Boya inteligente, la utilización de las funciones de la librería OpenCV para Android y el modulo GPU de nuestro dispositivo móvil.

A partir de esta experiencia pudimos aprender sobre los grandes beneficios que conlleva la utilización de HPC a la hora de desarrollar un software que sea paralelizable, con alta demanda de poder de cómputo, con algoritmos muy complejos o neceidad de mucha memoria o ancho de banda, concluyendo que es una solución viable en cuanto a lo tecnológico y económico para tomar en cuenta en futuros proyectos.

Una mejora a futuro podría ser optimizar la función de comparación de la imagen recibida con las imágenes de la BD, ya que mientras la cantidad de imágenes sea baja no habrá problema, pero si esta empieza a crecer puede que se tarde un tiempo en realizar el proceso, para ello en vez de llamar a la función de comparación por cada imagen, arrancaría un hilo, que sería el encargado de realizar la tarea. Por tanto, se lanzarían tantos hilos como imágenes contenga la base de datos.

1. Referencias

1. Lamiaa A. Elrefaei, Mona Omar Al-musawa and Norah Abdullah Al-gohany.: Development of an android application for object detection based on color, shape, or local features (2017)

2. Gautier Berthou.: Implementation of an object-detection algorithm on a CPU+GPU target (2016)

3. Oskar Andersson, Steffany Reyna Marquez.: A comparison of object detection algorithms using unmanipulated testing images Comparing SIFT, KAZE, AKAZE and ORB (2016)

4. Ethan Rublee, Vincent Rabaud, Kurt Konolige, Gary Bradski.: ORB: an efficient alternative to SIFT or SURF (2015)