Reconocimiento de cultivos, procesamiento y parametrización de riego

Amato Luciano, Borda Jonathan, Lancuba Emiliano, Gomez Nicolas, Olivera Florencia.

[1]Universidad Nacional de La Matanza,
Departamento de Ingeniería e Investigaciones Tecnológicas,
Florencio Varela 1903 - San Justo, Argentina
luchiiamato@gmail.com, jonathanmatiasborda@gmail.com, lancuba.emiliano@gmail.com,
nneegomez@gmail.com, o.florencia123@gmail.com

Resumen. En la actualidad, existen diversas metodologías y algoritmos conocidos de detección de objetos y reconocimiento de imágenes. A pesar del avance y evolución de estos con la llegada de las nuevas tecnologías, aún no se consigue un algoritmo que se adapte completamente a todos los puntos deseables de un sistema o producto (Costo, tiempo y alcance). En este documento, particularmente, se buscará demostrar teóricamente la implementación del algoritmo SIFT de reconocimiento de imágenes con el objetivo de identificar cultivos particulares, procesarlos y mejorar la precisión en el riego sobre el cultivo identificado. Adicionalmente, se buscará embeber el proyecto en un ecosistema IOT el cual beneficiará y brindará soporte al módulo de reconocimiento, aumentando así, las prestaciones que SmartGarden puede ofrecer al usuario. Motivado a la implementación de esta estructura nos encontraremos con la necesidad de cubrir diversas aristas y realizar modificaciones a gran escala en la estructura actual del proyecto, las cuales serán abordadas en posteriores secciones.

Palabras claves: SIFT, reconocimiento de imágenes, IOT, MIMD, MPI.

1 Introducción

Actualmente SmartGarden se compone de un sistema embebido y una aplicación móvil. El sistema embebido consiste en un dispositivo de riego autónomo, encargado de obtener valores de su entorno, realizar cálculos internos y establecer la intensidad de agua involucrada en un riego. La función de auto aprendizaje es la principal ventaja del producto: el sistema aprende de su entorno, y mejora con el transcurso del tiempo, la precisión de la intensidad de agua destinada al riego, y, por consiguiente, la calidad del mismo. Además, posee la tecnología apropiada para disponer de una comunicación por medio del protocolo *Bluetooth*, que, utilizando una aplicación como medio, permite establecer conexiones con *smartphones* que posean sistema operativo Android (API mayor o igual a 19 -Android 4.4-), permitiendo el intercambio de información entre estos. Esta aplicación posee funcionalidades que

abarcan tanto consulta de datos/estado como implantación de tareas manuales al sistema embebido, es decir, es posible consultar datos registrados, y -por medio del uso de sensores del dispositivo móvil- establecer directrices al dispositivo, tales como, iniciar registro de entorno, iniciar riego, detener riego, entre otros.
El objetivo del desarrollo e implementación del módulo de reconocimiento de cultivos es otorgar una nueva funcionalidad orientada a profundizar y mejorar la inteligencia del modelo. Mediante este, se lograran identificar objetos particulares, los cuales variarán i) el nivel de intensidad del agua destinada al riego ii) el tiempo de riego sobre los objetos. Estas modificaciones en los parámetros de riego serán procesadas en los momentos previos al pautado según la función de aprendizaje automático de SmartGarden.
Un ejemplo general de uso ya existente: Como se indica en el articulo de La Voz, en 2019, la inteligencia artificial de Facebook, interpreta las fotografías de los usuarios para adecuar la publicidad dirigida a cada cuenta, generando así, un mayor aprovechamiento de los recursos de la red social (imágenes de usuario) para generar un beneficio extra (publicidad dirigida). En este punto, el objetivo de SmartGarden dista del mecanismo implementado por Facebook, aunque se pueden encontrar un punto en común: el aprovechamiento de los recursos disponibles, de modo que, SmartGarden aprovecharía los recursos que tiene a su alcance, es decir, su entorno, para así generar una funcionalidad extra que impacte en un beneficio tanto para el usuario como para el entorno del sistema.

2 Desarrollo

A la hora de implementar este módulo, se debe comenzar con las modificaciones lógicas e infraestructurales a realizar en el sistema embebido. El módulo añade una cámara al dispositivo (*endpoint*) la cual será la encargada de obtener los datos del entorno, previo al riego, capturando imágenes del mismo. Luego, mediante la conexión con la aplicación móvil el dispositivo SmartGarden se encargará de enviar estos datos de entorno a la misma para su tratamiento.
La aplicación recibirá las visuales requeridas, y realizará una serie de validaciones, tales como, nitidez de imágenes o cantidad de capturas tomadas, entre otras. Finalmente, enviará el paquete de información a un *datacenter* de servidores externos, pertenecientes a SmartGarden, a través de ethernet. Estos servidores externos son los que, implementando SIFT (Definido por Hima Bindu, en 2019, como un poderoso algoritmo de descripción de características en el campo de la visión por computadora y el reconocimiento de patrones) junto a otras tecnologías, obtendrán los datos necesarios de intensidad y tiempo de riego sobre los cultivos, interactuando con una base de datos Firebase, la cual se retroalimentará de cada nuevo reconocimiento realizado por los SmartGarden. Como ya se ha mencionado en el documento, SIFT será la técnica utilizada para identificar un cultivo como tal. Entre sus ventajas principales -y la principal desde el punto de vista de SmartGarden- se encuentra la gran performance que sostiene ante efectos lumínicos altos, tales como, alta y baja luminosidad, sombras, etc. Esta es una característica importante, debido a que los *endpoints* se encuentra expuestos a condiciones luminosas cambiantes. Por otra parte,

entre las desventajas de SIFT, encontramos que el tiempo de procesamiento suele dispararse a niveles muy elevados (Ebrahim Karami, 2017). Según Karami y Cordero, este algoritmo se puede explicar en cuatro pasos:

- **Detección de extremos en el espacio-escala**: Se busca dentro de la imagen puntos los cuales no dependan de la rotación, escalado y traslación de la imagen.
- **Localización exacta de puntos clave:** Se deben eliminar aquellos que no sean estables a cambios de iluminación y ruido.
- **Asignación de orientación:** Mediante la asignación de una orientación a cada punto de la imagen basada en características locales de la misma, los puntos clave pueden ser descritos relativos a estas orientaciones y de esta manera lograr características invariantes a las rotaciones.
- **Descriptor de puntos clave:** Determinar para cada punto clave un descriptor (ubicado en la Firebase) relativamente invariante a cambios de iluminación y afinidades.

Motivados en agilizar este proceso de detección SIFT, se implementará MPI, una interfaz estándar para bibliotecas de paso de mensajes (Snir, 2005), con el objetivo de realizar procesamiento distribuido entre todos los servidores del *datacenter* previamente mencionado. Desde SmartGarden creemos que si bien MPI otorga un gran salto en cuanto a performance, equiparando así las pérdidas sufridas por la ejecución de SIFT, no será suficiente para que se logre responder a las solicitudes en el tiempo esperado. Es por esto que, además, se optará por implementar el paradigma MIMD (*Multiple Instruction Multiple Data,* definido por Joel Saltz como una técnica de reorganización de patrones de dependencia de datos que mejoran la utilización del procesador) complementando a MPI, y logrando así, una mayor eficiencia en las detecciones de puntos clave y descriptores. Cabe destacar que al construir un ecosistema de IOT las solicitudes por tasa de tiempo tendrán un valor elevado y deberán ser respondidas con la en un tiempo establecido. En resumen, MPI nos permitirá distribuir la carga de los datos entre los servidores y MIMD, a agilizar el procesamiento en cada servidor. Si bien sabemos que SIFT requiere tiempo de procesamiento alto, utilizando la combinación de MPI + MIMD obtendremos un equilibrio necesario para obtener los tiempos de respuesta esperados.
Nota: se debe implementar la arquitectura requerida en los servidores del datacenter para poder utilizar MIMD.

3 Explicación del algoritmo

El siguiente pseudocódigo buscará mostrar la implementación del algoritmo SIFT, junto con el uso de MPI + MIMD para la distribución de carga, basándose en lo demostrado por Marc-Andre Hermanns y MEKHIEL. No se abordarán las implementaciones del sistema embebido. No se abordarán las implementaciones de la aplicación móvil a excepción del desarrollo del módulo de recepción de información desde los servidores externos.

Servidores Externos – SIFT, MPI+MIMD.

Recibir información de Mobile y envió

```
#include "/usr/local/include/mpi.h"; #include "/usr/local/include/mimd.h"

images[] record;
string [] hosts, [] descrip, V, usr=root, pass=root;//Usuario y pass para todos los servidores.
int i=0;
Riego Result;
image_read(img);
hosts = get_hosts_network();//Obtengo los hosts de la red de servidores.
record = receive_images(emisor); //Recibo capturas de entorno de mobile por internet.
MIMD_define();//Definición de parámetros de MIMD.
MIMD_send_directives (hosts);//Envió de parámetros de MIMD a servidores. involucrados
   MPI_init(hosts, usr, pass);//Comienzo sector de paralización.
      MPI_Comm_images(MPI_COMM_WORLD, &record);//Se cargan las imágenes.
      //Se realiza comunicación entre servidores con primitivas MPI_send() y MPI_rcv();
     foreach img in record{ //Por cada imagen, obtengo los puntos, este proceso es
paralelizado al estar dentro del contexto MPI.
      Get_orientation(x);
      Create_dog(y);
      …//más información para construir descriptor
      V[i] = Get_points(x, y, …); //Obtengo los puntos clave
      descrip[i] = match_database(DB, FIREBASE_USER, FIREBASE_KEY, V[i]); //Busco
el descriptor en la base de datos y lo retorno. Nota: La base de datos se alimenta de los
puntos clave no encontrados, de modo que, si no se encontrara un descriptor para un punto
clave, ese punto clave pasara a ser un descriptor de sí mismo, alimentando para futuras
detecciones.
      i++}
    MPI_Finalize();//Termino paralización. La idea es paralelizar el proceso de
reconocimiento de las imágenes, sumado al de obtener los puntos clave, y la obtención del
descriptor en la base de dato, haciendo más rápido este proceso de tres partes.
Result= Procesar_descriptores(descrip)//Formo el objeto en cuestión y calculo intensidad de
riego y tiempo.
EnvioSolicitudDeInfoLista(emisor); //Se le avisa al emisor de la solicitud que la info esta lista
If(emisor_acepta)
   EnvioDatosAAplicacionMobile(emisor, Result);
}
```

Aplicación Mobile – Recepción de información.

Recibir información de embebido y envío

```
RecibirValidarEntorno{
  Video = RecibirDatosEmbebido();
  A = VerificarPesoDeVideo(video);
  B = VerificarCalidadDeVideo(video);
  C = VerificarTiempoDVideo(video);
  If (A==1 Y B==1 Y C==1)
    enviarImagenesValidadasASVsExterno(video);
    crear_hilo_pendiente_de_rta(respuesta) //cuando la señal de respuesta esté lista para ser
transmitida desde alguno de los servidores, el hilo encargado devolverá un booleano.
  else
    solicitarNuevaTomaDeVideoAEmbebido();
  evento_rta_1(respuesta); //Cuando el hilo devuelva un "true", un evento se disparará e
iniciara la rutina de recepción y envió de información al embebido.}
```

4 Pruebas que pueden realizarse

Para probar y verificar la funcionalidad del módulo basta con colocar el dispositivo regador en la ubicación donde desee que riegue, y activar el módulo mediante la aplicación. Una vez hecho esto observaremos, cuando comience el riego, la moderación del flujo y tiempo de riego en determinado sector. Particularmente esto será observado claramente donde existan cultivos que requieran mayor flujo de agua (un árbol) o requieran menor flujo de agua (Un cactus).

5 Conclusiones

Hemos recorrido las modificaciones a realizar en el proyecto SmartGarden si se busca añadir una funcionalidad compatible con el reconocimiento de objetos. Se ha observado el alto costo de desarrollo de la misma además de los esfuerzos por obtener eficiencia y respuestas en tiempos estipulados.
Las lecciones que se dejan asentadas luego de este análisis se observan en el entendimiento de la complejidad que un sistema de inteligencia avanzado atrae, junto con las metodologías complejas utilizadas para ello. Se deja a entrever también, que existen campos para continuar avanzando sobre una base: Los algoritmos de reconocimiento no son del todo precisos y muchos de ellos ofrecen ventajas que otros no. Luego de recorrer este camino, muchas ideas en torno a este tipo de inteligencia comienzan a surgir, por ejemplo, llevar este módulo de reconocimiento de objetos más allá y utilizar un reconocimiento facial para, por medio de gestos, la aplicación detecte los mismos y se realicen acciones que impacten en el *endpoint*.

6 Referencias

Ebrahim Karami, S. P. (2017). Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images. Memorial University, Canada: Faculty of Engineering and Applied Sciences.
Hima Bindu Ch, D. K. (2019). SIFT and It's Variants: An Overview. Amity University Rajasthan, Jaipur, India: SUSCOM-2019.
Marc-Andre Hermanns, K. M. (2018). Introduction to the MPI_T Events Interface. Tools Working Group.
MEKHIEL, N. (2015). MPI Instructions. Ryerson EE Network.
Cordero Andrea (2012). Estudio y selección de las técnicas SIFT, SURF y ASIFT de reconocimiento de imágenes para el diseño de un prototipo en dispositivos móviles.
Snir, A. G.-L. (2005). MPI-2: Extending the message-passing interface.
Joel Saltz, V. N. Towards developing robust algorithms for solving partial differential equations on MIMD machines.
La Voz (2019). La caída de Facebook dejó a la vista sus mecanismos automáticos de reconocimiento de imágenes.