

# Partición de Visual SLAM para su uso con celulares

Alejandro Silvestri, Jorge Eterovic, Alesio Sinopoli

Departamento de Ingeniería e Investigaciones Tecnológicas Universidad Nacional de La Matanza

Florencio Varela 1903 (B1754JEC), San Justo, (5411) 4480-8900

{jasilvestri; eterovic}@unlam.edu.ar; asinopoli@alumno.unlam.edu.ar

## RESUMEN

Un sistema de *Visual SLAM* obtiene en tiempo real la localización espacial de un dispositivo con cámara a partir del video que capta, en un recinto previamente mapeado. Esta capacidad se denomina **autolocalización**, imprescindible para la conducción autónoma.

*Visual SLAM* es una tecnología experimental que promete masificar la capacidad de autolocalización para **AMR** (*Autonomous Mobile Robot*), pequeños vehículos autónomos de carga para espacios controlados.

Es de sumo interés ejecutar *Visual SLAM* en celulares, pero la alta demanda computacional de estos sistemas lo hace inviable.

Este trabajo desarrolla un sistema partido de autolocalización visual basado en *Visual SLAM*, que ejecuta una parte en el celular y otra en una PC de escritorio.

**Palabras clave:** *VSLAM, STELLA\_VSLAM, VISUAL SLAM, BIORRUEDA, AMR, ORB-SLAM2*

## CONTEXTO

Un AMR es un pequeño vehículo autónomo de carga, que circula muchas veces a la par de las personas, en recintos controlados - no en la vía pública - como depósitos, plantas industriales, campus y predios en general.

Los AMR comerciales usan un sistema de autolocalización basado en LIDAR, un sensor efectivo pero muy costoso.

El proyecto PROINCE - Programa de Incentivos a Docentes Investigadores

Secretaría de Políticas Universitarias (2022-2023) de la Universidad Nacional de La Matanza consiste en el desarrollo de un AMR autolocalizado por *Visual SLAM*. El proyecto es multidisciplinario, con partes mecánicas, electrónicas, de control e informáticas.

El presente trabajo se concentra en el componente informático de autolocalización del AMR que se está desarrollando en el proyecto PROINCE de la Universidad Nacional de La Matanza.

## 1. INTRODUCCIÓN

### 1.1. Sobre Visual SLAM

Sensores como Lidar 2D y 3D hacen un relevamiento espacial del entorno. Montados sobre un vehículo, a medida que se desplaza sucesivas lecturas de estos sensores permiten relevar grandes áreas, llevando a cabo lo que se denomina **mapeo** del área. La composición en tiempo real de estas lecturas no es trivial, y se denomina *SLAM* (*Simultaneous Localization And Mapping*).

Un sistema SLAM mapea en tiempo real un área mientras el vehículo se desplaza, y a la vez localiza la posición del vehículo en ese mapa. Esta capacidad de autolocalización sobre un mapa relevado es la clave para conducir de manera autónoma al vehículo a través de una trayectoria hacia un destino.

Los Lidares son sensores costosos. En 2007 dos ensayos fundacionales lograron los

primeros sistemas de *Visual SLAM*, que reemplazaron el Lidar por una cámara de video. Estos sistemas son PTAM [1] y monoSLAM [2]. En pocos años el costo de una cámara para PC se redujo enormemente haciendo posible su masificación, poniendo a *Visual SLAM* como una tecnología candidata a sustituir los sistemas de SLAM con Lidar.

A estos trabajos pioneros de *Visual SLAM* le siguieron una sucesión de papers que marcaron una evolución acelerada durante 10 años. En 2014 y 2015 los ensayos LSD-SLAM [3] y ORB-SLAM [4] establecieron el hito máximo en cuanto a performance y precisión, sin alcanzar el “santo grial”: la posibilidad de poder ejecutarse en un celular.

En 2017 *Visual SLAM* se consideró un trabajo terminado, con ORB-SLAM2 [5] representando el estado del arte hasta la fecha. Varios ensayos posteriores han contribuido en mejorar diversos aspectos, entre los que se destacan el uso de cámara estéreo, omnidireccional, RGBD y combinación con acelerómetros, pero con apenas mejoras menores en la performance.

Cabe mencionar el surgimiento de una rama de investigación en 2018, denominada *Spatial AI* [6], que procura desarrollar sistemas de *Visual SLAM* con inteligencia artificial. Esta rama, con muchos méritos propios, no ha aportado a la mejora de la performance (sino todo lo contrario), divergiendo significativamente del objetivo de este trabajo.

En 2019 el ensayo OpenVSLAM [7] puso a disposición de la comunidad un sistema de Visual SLAM basado en ORB-SLAM2 pero con código abierto con licencia BSD (uso libre e irrestricto), con un código totalmente reescrito aplicando buenas prácticas de programación, y corrigiendo una infinidad de bugs que impiden el uso práctico de los otros sistemas publicados.

En 2021 OpenVSLAM se retiró por presuntas violaciones de licencias, y fue sustituido por **stella\_vslam** [8], un proyecto abierto dedicado a corregir todas las posibles violaciones de licencia de OpenVSLAM, convirtiéndose en la

única solución de código abierto para proyectos de ingeniería y productos comerciales.

## 1.2 Visual SLAM con celulares

Hay dos antecedentes de uso de *Visual SLAM* en celulares, ninguno de los cuales satisface la necesidad de un AMR.

En 2015 LSD-SLAM demostró que puede ser usado en celulares para autocalización sobre mapas relevados, desactivando el módulo de mapeo que es el de mayor demanda computacional. A priori esto satisface la necesidad de autocalización para AMR, pero falla en otros aspectos, principalmente en el enorme tamaño de los mapas en memoria, y en la incapacidad de guardarlos en un archivo para ser reutilizados en otro momento.

Entre 2018 y 2023 se han repetido con éxito varios intentos de compilación de ORB-SLAM2 (entre otros) en celulares de alta gama, pero obteniendo siempre una performance muy pobre.

La idea de partir el procesamiento entre un celular con cámara montado en el AMR y una PC fija conectada por WIFI claramente no elimina la necesidad de la PC, pero flexibiliza enormemente el despliegue del sensor (la cámara) en el vehículo experimental, habilitando incluso la experimentación con plataformas muy baratas como los pequeños vehículos robot Arduino.

El planteo simplista consiste en usar el celular como cámara IP y transmitir en tiempo real el video por WIFI a la PC con *Visual SLAM*. Lamentablemente la compresión del video imprescindible para acomodarse en el ancho de banda disponible introduce un retardo superior a 1 s que hace inviable su uso en navegación autónoma.

En Diciembre de 2022 alumnos de grado de ingeniería informática de la Universidad Nacional de La Matanza, participando en el proyecto PROINCE, presentaron con éxito su trabajo de grado UNSLAM [9], la primera aplicación con Visual SLAM partido, con

performance adecuada para la navegación autónoma de un AMR.

Este trabajo demuestra experimentalmente la viabilidad del concepto, logrando un retardo de 200 ms en un celular antiguo de bajo poder computacional, suficiente para la navegación autónoma, y muy mejorable con el uso de celulares de más potentes.

Más notablemente, el sistema no se implementó en una App, sino en una página web, lo que le confiere amplia compatibilidad y fácil despliegue.

UNLaM es la prueba de concepto del módulo de navegación del proyecto PROINCE. Adopta *stella\_vslam* como sistema de *Visual SLAM* ejecutando en una PC.

El *pipeline* de *stella\_vslam* es complejo, pero a los efectos de la exposición este trabajo lo dividió en dos partes: el preprocesamiento en celular de la imagen obtenida por la cámara, y el posprocesamiento en PC de la información extraída al combinarse con el mapa.

El preprocesamiento es el conjunto de todas las operaciones que se realizan sobre la imagen. Al finalizar, la imagen se descarta, y el sistema sólo utiliza la información extraída de ella, que ocupa una fracción menor que la imagen completa.

Ésta es la parte que se ejecuta en el celular:

1. captación de la imagen color y conversión a monocromática
2. generación de pirámide
3. detección de *keypoints* [10] en todos los niveles de la pirámide
4. selección de *keypoints* representativos por medio de *quadtrees*
5. extracción de descriptores [11, 12]

Este preprocesamiento está desarrollado en C++ utilizando la biblioteca OpenCV [13]. Su implementación en una página web requirió:

aislar los módulos de OpenCV utilizados  
compilarlos como bibliotecas estáticas

adoptar un soporte de paralelismo OpenMP [14] con una biblioteca para Web Assembly

compilar el código C++ del pipeline para Web Assembly con *emscripten* [15] y *llvm* [16], vinculando las bibliotecas estáticas  
elaborar la página web que capte la imagen con javascript y la preprocese con el módulo anterior en Web Assembly

enviar los resultados al servidor: la opción directa, *fetch* y JSON, se descartó para favorecer una más performante que evita el parseo: *web sockets* para enviar los datos de manera binaria

De lado de la PC se desarrolló una variante de *stella\_vslam* que admite como entrada los datos preprocesados por el celular en lugar de una imagen.

Además se implementaron *bindings* de *stella\_vslam* para Python, con dos propósitos principales:

permitir al servidor web que recibe los datos del preprocesamiento, enviarlos al *stella\_vslam*  
obtener el resultado buscado: la pose de localización

### 1.3. Sistema de referencia real

En general los sistemas de Visual SLAM presentan la información de pose en términos de una matriz de 4x4 en coordenadas homogéneas que representan una transformación euclidiana 3D.

Durante el mapeo los sistemas de *Visual SLAM* establecen un sistema de referencia espacial virtual, con un centro y una orientación de ejes arbitraria y desconocida por el usuario. En particular, los sistemas monoculares - aquellos que usan una sola cámara, en contraste con los estereoscópicos y los de profundidad - adoptan también una escala arbitraria, ya que carecen de referencia de escala real.

La segunda parte del presente trabajo fue diseñar e implementar una estrategia de vinculación entre el sistema de referencia virtual arbitrario del *Visual SLAM*, y un

sistema de referencia real definido por el usuario. Los pasos para la vinculación de estos sistemas de referencia son:

1. el usuario posiciona el celular en el punto que se elige como origen (0;0;0) del sistema de referencia, que suele ser en el suelo; el sistema registra las coordenadas virtuales asociadas obtenidas por Visual SLAM, y el acelerómetro del celular para relevar la dirección vertical
2. a continuación el usuario posiciona el celular desplazándolo 1 m sobre el eje X, a las coordenadas (1;0;0); el sistema registra esas coordenadas virtuales y la dirección vertical

Se asume que el suelo es horizontal, el eje Y es vertical y el eje Z es perpendicular a los otros dos.

Con las dos mediciones el algoritmo calcula la matriz de transformación que convierte del sistema virtual arbitrario de *Visual SLAM* al sistema métrico real elegido por el usuario.

Durante la operación, para cada imagen procesada el sistema de *Visual SLAM* informa sus coordenadas virtuales, y el código propio usa la matriz anterior para convertirlas a coordenadas métricas.

## 2. LÍNEAS DE INVESTIGACION Y DESARROLLO

Este trabajo es parte del proyecto PRINCE de la Universidad Nacional de La Matanza, dedicado al diseño de un AMR versátil de bajo costo como prototipo para uso académico y para uso comercial.

## 3. RESULTADOS OBTENIDOS/ESPERADOS

Los resultados obtenidos se describieron en la Introducción, y se resumen aquí:

prueba de concepto de Visual SLAM partido, con una medición de performance adecuada para AMR

método de vinculación entre sistemas de referencias virtual y real  
el sistema UNLaM completo y funcionando publicado en GitHub

Los resultados esperados son:

Aislación del autolocalizador partido como un módulo para su disponibilidad en otros proyectos, incluyendo PROINCE  
Modificación y documentación del módulo  
Tabla comparativa con pruebas de performance con varios celulares  
Sistema de navegación que indique la trayectoria a seguir a partir de la autolocalización.

## 4. FORMACIÓN DE RECURSOS HUMANOS

Cinco alumnos de ingeniería informática se recibieron de ingenieros con su trabajo de grado UNLaM.

El proyecto promueve la formación en proyectos de I+D de 4 docentes ingenieros.

## 5. BIBLIOGRAFIA

- [1] PTAM, Parallel Tracking and Mapping for Small AR Workspaces, Klein -Murray, ISMAR . 2007.
- [2] MonoSLAM: Real -Time Single Camera SLAM, Davidson et al, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, V OL. 29, NO. 6, JUNE 2007.
- [3] LSD -SLAM: Large -Scale Direct Monocular SLAM, Engels et al, Technical University Munich. 2014.
- [4] ORB -SLAM: a Versatile and Accurate Monocular SLAM System, Mur Artal et al, IEEE TRANSACTIONS ON ROBOTICS . 2015.
- [5] ORB -SLAM2: a n Open-Source SLAM System for Monocular, Stereo and RGB-D

Cameras, Mur Artal et al, IEEE TRANSACTIONS ON ROBOTICS . 2017.

[6] FutureMapping: The Computational Structure of Spatial AI Systems, Davidson et al. 2018.

[7] OpenVSLAM : A Versatile Visual SLAM Framework, Sumikura et al, Arxiv . 2019.

[8] [https://github.com/stellacv/stella\\_vslam](https://github.com/stellacv/stella_vslam)

[9] <https://github.com/UNLaM>

[10] Faster and better: a machine learning approach to corner detection, Rosten et al, 2008.

[11] BRIEF: Binary Robust Independent Elementary Features, Calonder et al, 2011.

[12] ORB: an efficient alternative to SIFT or SURF, Rublee et al, 2012.

[13] <https://docs.opencv.org/4.7.0/>

[14] <https://www.openmp.org/>

[15] <https://emscripten.org/>

[16] <https://llvm.org/>