Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Escuela de Ciencias Exactas y Naturales
Departamento de Ciencias de la Computación
R-521 — Robótica Móvil

Trabajo Práctico

Graph SLAM

1. Introducción

El objetivo de este trabajo práctico es implementar un solucionador de Pose Graph SLAM utilizando la librería GTSAM. En particular, se le pedirá que implemente tanto una solución batch como una solución incremental para los problemas SLAM 2D y 3D utilizando conjuntos de datos estándar.[1]

Entrega

- Se debe proveer un repositorio git que contenga el código desarrollado y un archivo `README.md` con las instrucciones de compilación y ejecución. Se recomienda hacer una imagen Docker para facilitar la reproducción de los resultados.
- Se debe entregar un informe en Lyx o LaTeX explicando el trabajo realizado y analizando los resultados obtenidos.

Evaluación

- Haciendo únicamente los ejercicios obligarorios (no opcionales), el trabajo tiene una nota máxima de 8. Los ejercicios opcionales permiten llegar a la nota máxima de 10.
- Entregas Fuera de Término: Si la entrega se realiza durante la primer semana luego del plazo, se decuentan 2 puntos. Si la entrega es más tarde que esto último la nota máxima es de 6.

Pose Graph SLAM usando la librería GTSAM

En este trabajo práctico, resolverá el problema SLAM con Pose-Graph utilizando la librería GTSAM. Si no está familiarizado con GTSAM, encontrará un tutorial detallado en su sitio web: `https://gtsam.org/tutorials/intro.html`. Para instalar GTSAM en c++, deberá clonar el código del repositorio: `https://github.com/borglab/gtsam`, descargar (usando git) la última versión y compilar la librería siguiendo las instrucciones.

Después de instalar GTSAM con éxito, escriba una función para leer archivos G2O [2] y resuelva el problema de optimización de grafos para casos 2D y 3D usando GTSAM. En esta tarea, utilizamos los datos proporcionados en `https://lucacarlone.mit.edu/datasets/`.

Si bien GTSAM está desarrollado en C++, también proporciona un wrapper tanto para MATLAB como para Python. En esta tarea, eres libre de utilizar cualquiera de esos lenguajes.

Guía de instalación de GTSAM

A continuación proporcionamos una guía de instalación para las bibliotecas GTSAM, luego presentamos sus versiones C++, MATLAB y Python respectivamente.

[1] Esta tarea se basa en Homework 7 – SLAM of the NA 568 Mobile Robotics: Methods & Algorithms – Winter 2022 created by Maani Ghaffari – University of Michigan

[2] `https://github.com/RainerKuemmerle/g2o/wiki/File-Format`

Librería GTSAM

El primer paso es clonar e instalar la librería GTSAM. Las instrucciones detalladas se pueden encontrar en el repositorio.

Remark 1. *Si planea utilizar el wrapper de MATLAB o de Python, puede omitir esta parte.*

Remark 2. *GTSAM requiere la librería Eigen. Se puede descargar e instalar desde aquí:* `http://eigen.tuxfamily.org/index.php?title=Main_Page`

Remark 3. *Por favor, tenga en cuenta los requisitos previos: Boost >= 1.58 (Ubuntu: sudo apt-get install libboost-all-dev); CMake >= 3.0 (Ubuntu: sudo apt-get install cmake); Un compilador moderno, es decir, al menos gcc 4.7.3 en Linux.*

- Clonar el repositorio de gtsam `https://github.com/borglab/gtsam`

  `cd <path_to_your_repository>`

- Crear un nuevo directorio llamado "build":

  `mkdir build`

- Navegar al directorio de compilación:

  `cd build`

- Ejecutar cmake para crear enlaces esenciales para los archivos de compilación (**Nota**: si desea utilizar el wrapper de python, deberá hacer algo diferente aquí. Por favor, salte a la sección del wrapper de python)

  `cmake ..`

Remark 4. *Si recibe un error que indica que no se encuentra Boost, puede intentar instalarlo manualmente siguiendo este tutorial y asegúrese de que está apuntando al directorio correcto agregando los siguientes comandos en CMakeList.txt:*

```
SET (BOOST_ROOT "<your_boost_path>")
SET (BOOST_INCLUDEDIR "<your_boost_path>/boost")
SET (BOOST_LIBRARYDIR "<your_boost_path>/libs")
SET (BOOST_MIN_VERSION "1.58.0")
set (Boost_NO_BOOST_CHAKE ON)
```

- Crea el archivo en el directorio de compilación (El comando -j10 indica el número de hilos que se usarán durante la compilación. Esto acelerará el proceso. Se recomienda usar el número máximo de núcleos menos 2, de lo contrario, tu máquina podría bloquearse. Si tiene 12 hilos en su máquina, use -j10). `make -j10`
- Instalar la librería GTSAM en tu máquina. `sudo make install -j10`
- En este paso, podrás ver la ruta donde se están instalando todos los paquetes. Si más tarde no puedes vincular la ruta de instalación a tu código, puedes volver a este paso y ver dónde se instaló.

C++

Si desea utilizar C++, no necesita instalar nada más. Todo lo que necesita hacer es buscar y vincular la librería GTSAM agregando el siguiente contexto a su archivo CMakeList.txt.

```
find_package(GTSAM_REQUIRED)
include_directories(${GTSAM_INCLUDE_DIR})
target_link_libraries(<your_project> <your_project_lib> gtsam)
```

Puede consultar algunos ejemplos de GTSAM en C++ `https://github.com/borglab/gtsam/tree/develop/examples`.

Wrapper de MATLAB

Si tiene un sistema Ubuntu más reciente (posterior a la versión 10.04), deberá realizar una pequeña modificación en su instalación de MATLAB, ya que MATLAB se distribuye con una versión antigua de la biblioteca estándar de C++. Elimine o cambie el nombre de todos los archivos que comiencen con libstdc++ en el directorio de instalación de MATLAB, en las siguientes rutas:

- `/usr/local/MATLAB/{version}/sys/os/{system}`
- `/usr/local/MATLAB/{version}/bin/{system}`

Para el wrapper de MATLAB, se puede hacer de dos maneras:

a) download the precompiled wrapper from `http://www.borg.cc.gatech.edu/download.html`,

b) or install from sources following the instructions in `https://github.com/borglab/gtsam/tree/develop/matlab`.

Recomendamos usar el wrapper precompilado si no está familiarizado con los sistemas Linux. A continuación encontrará instrucciones sobre cómo agregar el wrapper de MATLAB precompilado a su conjunto de herramientas.

- Descargar el wrapper precompilado. Descargar el toolbox de MATLAB precompilado (compatible con MATLAB R2011a y versiones posteriores) según su sistema operativo (Mac OS de 64 bits / Linux de 64 bits / Windows de 64 bits).
- Extraer la carpeta. (**Nota: para sistemas linux**, después de la extracción, deberá hacer clic derecho en el archivo extraído → Open With → Archive Mounter. Luego verá un `gtsam-toolbox-3.2.0-lin64` en el lado izquierdo junto con Computadora, OS... La carpeta dentro llamada `gtsam_toolbox` es la carpeta que queremos usar.)
- Colocar la carpeta `gtsam_toolbox` en `<YOUR_MATLAB_INSTALL_PATH>/toolbox/`
- Cada vez que abra su MATLAB, en el lado izquierdo de su GUI, navegue hasta su `gtsam_toolbox` → clic derecho → add to path → selected folder. Luego debería poder ejecutar los códigos de ejemplo en la carpeta de ejemplos. O puede agregar los siguientes comandos al inicio del código.

  ```
  addpath('<your_gtsam_toolbox_path>')
  import gtsam.*
  ```

Puede consultar algunos ejemplos de GTSAM en MATLAB en el directorio `gtsam_toolbox/gtsam_examples` .

Wrapper de Python

Descargar e instalar Anaconda desde `https://www.anaconda.com/download`

```
conda create -n gtsam_env python=<your_python_version>
conda activate gtsam_env
conda install -c conda-forge cmake eigen pybind11 boost numpy matplotlib python-
   graphviz conda-forge::plotly conda-forge::pandas conda-forge::nbformat
```

```
git clone https://github.com/borglab/gtsam.git
cd gtsam
mkdir build && cd build
conda install -r <gtsam_folder>/python/requirements.txt
cmake .. -DGTSAM_BUILD_PYTHON=1 -DGTSAM_PYTHON_VERSION=<your_python_version>
make -j2 (2 is the number of threads you want to use)
make python-install
```

Puede consultar algunos ejemplos de GTSAM en PYTHON en el directorio `https://github.com/borglab/gtsam/tree/develop/python/gtsam/examples`.

2. Graph-SLAM 2D

2.1. A.

Escriba una función para leer el conjunto de datos Intel 2D[3] desde el formato G2O y generar poses y aristas. Estas poses y aristas se utilizan en problemas posteriores. Puede ser cualquier forma que desee siempre que pueda usarla para generar el resultado correcto.

Para datos 2D, la pose en formato G2O es `[VERTEX_SE2 i x y theta]` y el borde en formato G2O es `[EDGE_SE2 i j x y theta info(x, y, theta)]`, donde `info(x, y, theta)` es un vector 1×6 `[q11 q12 q13 q22 q23 q33]` donde los elementos son la matriz del triángulo superior de la matriz de información 3×3:

$$\Omega = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{12} & q_{22} & q_{23} \\ q_{13} & q_{23} & q_{33} \end{bmatrix}.$$

Al invertir esta matriz de información, se puede obtener la matriz de covarianza para el modelo de ruido.

Puede verlo en detalle en el repositorio de g2o[4].

Observación 5. *Si usas las funciones `readG2o()` y `load2D()` proporcionadas por GTSAM, entonces no podrás resolver el grafo incrementalmente (tarea 2.3 en este trabajo práctico).*

Sugerencia: Puede usar funciones `fscanf()`, `textscan()` o `readcell()` con formatSpec adecuado de MATLAB y verificar si el primer elemento en la entrada es `VERTEX_SE2` o `EDGE_SE2`.

2.2. B. Batch Solution

Una solución por lotes significa que primero construimos el grafo completo y luego lo resolvemos por completo. Cargar `data/input_INTEL_g2o.g2o` y construir un grafo de factores no lineales 2D usando GTSAM. Utilice el solver de Gauss-Newton. Visualice y compare la trayectoria optimizada con la trayectoria inicial. Incluye la gráfico en el informe. Describir el proceso de construcción del grafo y sus parámetros.

[3] `https://www.dropbox.com/s/vcz8cag7bo0zlaj/input_INTEL_g2o.g2o?dl=0`
[4] `https://github.com/RainerKuemmerle/g2o/wiki/File-Format-SLAM-2D`

Remark 6. *Para este problema, el solver de Gauss Newton caerá en un mínimo local si no le agregamos una perturbación. Está bien enviar un grafico que muestre que no funciona como se espera, pero incluya una discusión sobre por qué sucede esto.*

Hint: Puede usar `NonLinearFactorGraph` como grafo, usar `GaussNewtonOptimizer` como optimizador, usar `Values` para su estimación inicial, `noiseModel.Gaussian.Covariance()` para su modelo de ruido, `graph.add()` y `initial.insert()` funcionan como lo vea conveniente. Sin embargo, los nombres de las funciones pueden ser diferentes para las diferentes versiones de GTSAM.

2.3. C. Incremental Solution

Utilizar el solver ISAM2 para optimizar la trayectoria de forma incremental (a medida que construye el grafo gradualmente). Un algoritmo detallado se describe en Algoritmo 1. Visualice y compare la trayectoria optimizada con la trayectoria inicial. Incluir el gráfico en el pdf. Describir el proceso de construcción del grafo y sus parámetros.

Algorithm 1 `incremental_solution_2d(poses, edges)`

Require: poses: a N x 4 array that each row is $pose = (id_p, x, y, \theta)$; edges: a M x 11 array that each row is $edge = (id_{e1}, id_{e2}, dx, dy, d\theta, info)$

$isam \leftarrow$ gtsam.ISAM2() ▷ Initialize isam solver
for every pose in poses **do**
  $graph \leftarrow$ NonlinearFactorGraph ▷ Initialize the factor graph
  $initialEstimate \leftarrow$ Values ▷ Initialize the initial estimation
  $(id_p, x, y, \theta) \leftarrow pose$ ▷ Extract information from the current pose
  if $id_p == 0$ **then**
    $priorNoise \leftarrow$ some noiseModel ▷ Use a predefined noise model
    $graph.add(PriorFactorPose2(0, Pose2(x, y, \theta), priorNoise))$
    $initialEstimate.insert(id_p, \text{Pose2}(x, y, \theta))$
  else ▷ Not the first pose
    $prevPose \leftarrow result.at(id_p - 1)$ ▷ Use last optimized pose
    $initialEstimate.insert(id_p, prevPose)$
  end if
  for every edge in edges **do**
    $(id_{e1}, id_{e2}, dx, dy, d\theta, info) \leftarrow edge$ ▷ Extract information from the current edge
    if $id_{e2} == id_p$ **then**
      $cov = construct\_covariance(info)$ ▷ Construct a covariance matrix from the information vector.
      $Model \leftarrow$ noiseModel.Gaussian.Covariance(cov)
      $graph.add(BetweenFactorPose2(id_{e1}, id_{e2}, \text{Pose2}(dx, dy, d\theta), Model)$
    end if
  end for
  $isam.update(graph, initialEstimate)$
  $result = isam.calculateEstimate$
end for

Sugerencia: Puede usar `NonLinearFactorGraph` como grafo, usar `gtsam.ISAM2()` como algoritmo de actualización, usar `Values` para su estimación inicial y usar las funciones `graph.add()`, `initial.insert()`, `isam.update()` y `isam.calculateEstimate()` como vea conveniente. Sin embargo, los nombres de las funciones pueden ser diferentes para las diferentes versiones de GTSAM.

3. Graph-SLAM 3D

3.1. A.

Escriba una función para leer el archivo 3D Garage G2O[5] desde el formato G2O y genere poses y aristas.

Para datos 3D, la pose en formato G2O es `[VERTEX_SE3:QUAT i x y z qx qy qz qw]` donde `(x,y,z)` representa la traslación y `(qx,qy,qz,qw)` la rotación como un cuaternión. La ventaja en formato G2O es `[EDGE_SE3:QUAT i j x y z qx qy qz qw info(x, y, z, qx, qy, qz)]`, donde `info(x, y, z, qx, qy, qz)` es un vector 1×21 de la matriz de información 6×6. Después de un proceso similar en la tarea 1 A, puede obtener la matriz de covarianza. Puede consultar los detalles en el repositorio de g2o[6].

Remark 7. *Tenga en cuenta que el cuaternión en MATLAB está en el orden de `[qw qx qy qz]` y es diferente del orden en los archivos g2o que es `[qx qy qz qw]`. Puede usar la función `quat2rotm()` en matlab para construir una matriz de rotación a partir de cuaternión o usar la función `quat2tform()` en matlab para construir una matriz de transformación.*

3.2. B. Batch Solution

Cargue `data/parking-garage.g2o` y construya un factor-graph 3D no lineal usando GTSAM. Utilice el solver de Gauss-Newton. Visualice y compare la trayectoria optimizada con la trayectoria inicial. Incluya un gráfico 3D o dos gráficos 2D en su pdf. Describir el proceso de construcción del grafo y sus parámetros.

3.3. C. Incremental Solution

Utilice el solver ISAM2 para optimizar la trayectoria de forma incremental. Visualice y compare la trayectoria optimizada con la trayectoria inicial. Incluya un gráfico 3D o dos gráficos 2D en su pdf. Describir el proceso de construcción del grafo y sus parámetros.

[5] `https://www.dropbox.com/s/zu23p8d522qccor/parking-garage.g2o?dl=0`
[6] `https://github.com/RainerKuenmerle/g2o/wiki/File-Format-SLAM-3D`

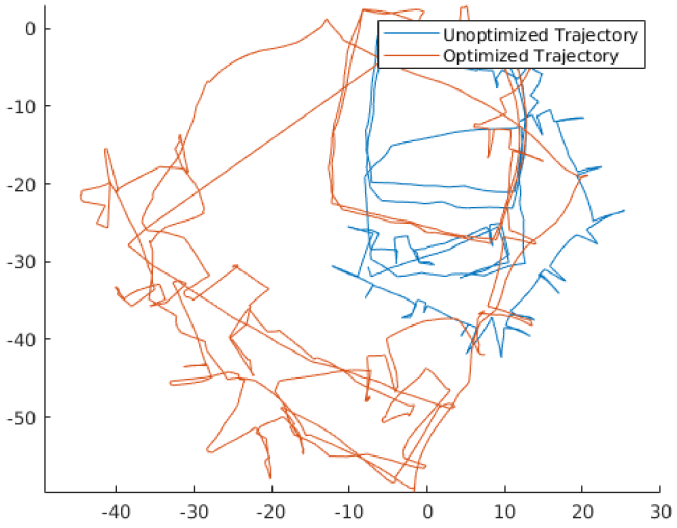

Figura 1: Expected result for task 1 B.

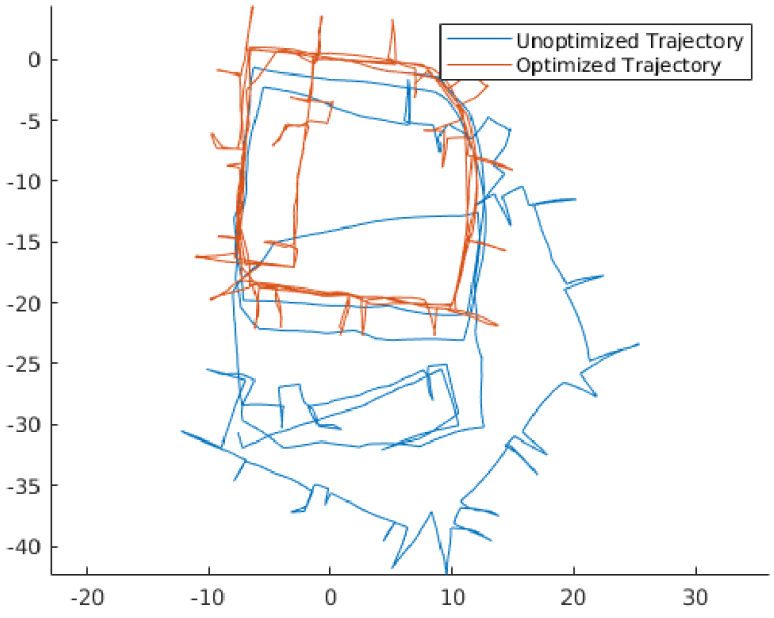

Figura 2: Expected result for task 1 C.

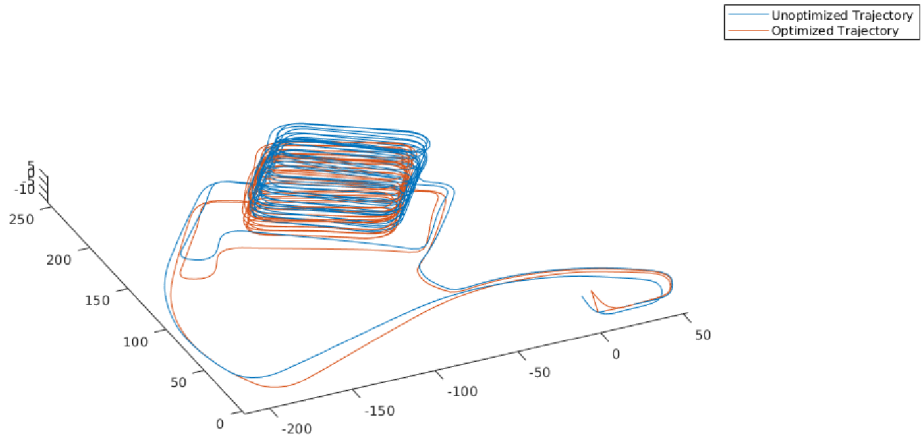

Figura 3: Expected result for task 2 B.

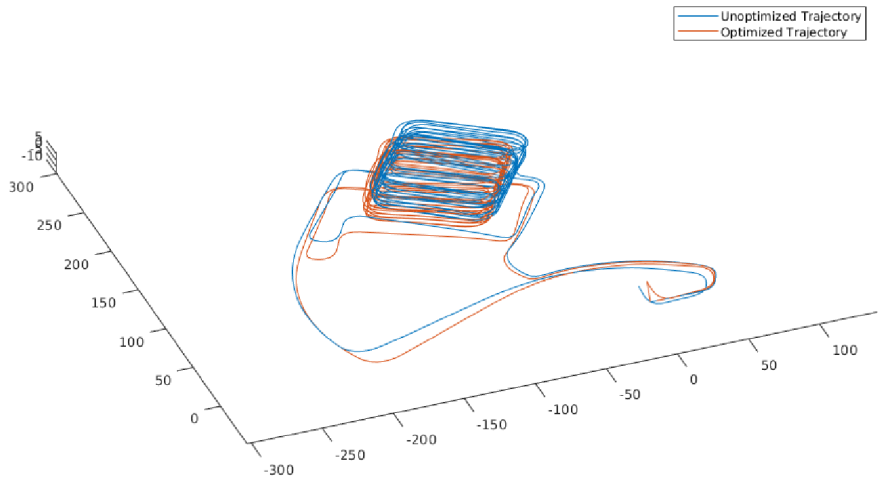

Figura 4: Expected result for task 2 C.