CSE 483: Mobile Robotics - Assignment 04

Due: **14th November 2016**, 2300 hrs

November 2, 2016

General Instructions

1. **Form teams of 2 people for this assignment.** In this assignment, you would immensely benefit from collaborations. Working alone is *discouraged*.
2. The assignments are to be done in C++, on the linux version of GTSAM-3.2.1. The code provided along with the assignments may run across other platforms, and on other versions of GTSAM as well. However, there are no guarantees on this front. Evaluation will be carried out in a machine running Ubuntu 14.04.3, and with GTSAM-3.2.1 installed.
3. Plagiarism is strictly prohibited.
4. Ensure that the plots included in the report are reproducible from the code that you submit.

Landmark SLAM using the GTSAM Toolbox (10 points)

In this assignment, we will use the GTSAM toolbox to test variants of Smoothing and Mapping (SAM) on a *toy* problem.

The example we consider for this assignment is present in the `example.graph` file in the `Data` folder in the GTSAM `examples` directory (in GTSAM-3.2.1). The .graph file has observations gathered from a robot over 100 poses (2D) and about 30 landmarks.

A sample C++ code to get you started on the assignment is present at the following URL:
`https://github.com/krrish94/gtsam-experiments`

In the `landmarkSLAM` folder of the code downloaded from the above URL, the file `landmarkSLAM.cpp` is of interest. It loads the .graph file (be sure to change the paths appropriately inside the cpp file) and constructs a pose graph out of it.

1. `Nonlinear Batch SAM`:
   Write code (about 5-6 lines is what it takes) to perform batch-mode SAM with a Levenberg-Marquardt solver. You can use the `simplePlanarSLAM` code (present in the same codebase) as a template. Repeat the experiment using a Gauss-Newton solver (again, it's only 2-3 lines of code).

2. `Linear Batch SAM`:
   Instead of using the nonlinear version of batch SAM (i.e., the Levenberg-Marquardt optimizer), linearize the constructed nonlinear factor graph and solve it. (Search around for possible ways of linearizaing a factor graph in GTSAM.)

3. `iSAM`:
   In this question, you're required to complete the code in the cpp file `landmarkISAM.cpp`. This file runs an version of iSAM2 (iSAM with the Bayes Tree graphical model), and performs linearization after every few time steps. Most of the code has been written for you. Make sure you understand the code. You will only have to tweak around with different parameters and report performance statistics as mentioned in the file.

Required analysis

For each of the above techniques presented, GTSAM outputs a factor graph. That factor graph can be written out to a pdf file, using graphviz (look at the GTSAM examples folder for a file `Pose2SLAMExample_graphviz.cpp` that has the 2-3 lines of code which perform this task).

Plot the initial factor graph, and the optimized factor graph (after all nodes have been added and the system has been solved) for the above cases. For each case, also give the running time of the algorithm. Use the functions `gttic_` and `gttoc_` for better time estimates. If you face issues with this, other standard C++ timers can be used.

Deliverables

1. A zipped (`.zip`) folder (try not to use any other format such as `.bz2`, `.tar.gz`, etc.) whose name is composed of the ID numbers of the two team members, separated by an underscore, eg. `201507666_201507555.zip`. The folder should have a `report.pdf` file which contains the other deliverables for the assignment.

2. Additionally the folder `landmarkSLAM`, which contains an edited version of CMakeLists.txt and the completed code files.

3. Please keep the input/output signatures for each function the same. Also do not change the directory structure, and do not add additional files, unless otherwise specified. The code will be evaluated on the same test case presented, i.e., on `example.graph`.