Course notes module 6

UAS positioning & navigation

Kjeld Jensen kjen@mmmi.sdu.dk

1 Agenda

1. Mid-term evaluation
2. Module theory and introduction to the exercises.
3. Exercises.

2 Course mid-term evaluation

It is time to perform a midterm evaluation of the IDT course. Here is the procedure that we will follow:

1. You will temporarily be divided into new teams.
2. Within the team you will discuss the topics listed below. The objective is to reach a conclusion within the team and list suggestions for improvements.
   - learning goals
   - level of difficulty
   - teaching methods
   - teaching materials
   - own performance

   One team member should be appointed as referent posting conclusions and suggestions to: https://kjen.dk/go/idt/eval
3. At the beginning of next module we will briefly discuss key points provided in your evaluation.

3 Module theory and introduction to the exercises

The theory required for this module has essentially been covered in previous modules. We will elaborate on the GNSS theory and look into the questions that may arise while walking through the exercises.

4 Exercises

The objective of these exercises are to acquire some experience in working with geographic and UTM coordinates and at the same time become familiar with UAS routes and tracks. The exercises will take you trough recording a GNSS based track and then convert this into a route plan by following these steps:

1. Record GNSS track.

2. Convert coordinates to UTM.
3. Remove outliers.
4. Simplify the track.
5. Convert to geographic coordinates.
6. Export as a route plan to QGroundControl software.
7. Fixed wing optimization.

In all exercises we will assume that the altitude is static, which essentially means that math will be reduced to 2D. In many applications this is the case anyway as the desired altitude is controlled by other parameters and the extrapolation to 3D is not that complicated when relevant.

This week's lab report concerns only step 3, 4, 6, (7) and should as usual not be longer than 4 pages of text, graphs and images (reasonably downscaled). The relevant material is your algorithms, the results and findings. Please discuss the results and please remember that report quality is more important than quantity.

4.1 Record GNSS track

To record the GNSS track you will need to connect your Pixhawk flight controller with GNSS module installed to your computer via USB. Please take good care of the Pixhawk and GNSS connectors, and keep them together by using electrical tape.

Using the ROS2, the ROS2 mavros package and example Python scripts create a ROS node that reads data from the Pixhawk in the MAVLINK format and saves the GNSS postions to a file. Please note that this part of the exercise builds upon module 5 which you will find at our confluence page:

https://kjen.dk/go/idt/confluence/

With the software running it is time to take the drone outside, obtain a GNSS fix and test that you are capable of saving positions to a file.

Then please spend some time recording various tracks by walking around e.g. building 44. Try to move close to building walls a few times to see if you can confuse the GNSS receiver.

4.2 Convert coordinates to UTM

The coordinates saved to the file should be converted to UTM using the same procedure as you did in module 2.

4.3 Remove outliers

Having the track represented as UTM coordinates it is now time to remove outliers if any. Outliers are single or a few off-track positions usually due to multipath problems that the GNSS was not able to cope with.

The first thing to do is to plot the track. This can easily be done using matplotlib, just remember to use `axis('equal')` in order to avoid skewing the plot.

Now create a Python class that contains a simple algorithm for removing outliers based on the maximum distance between two points relative to the time between recording and the maximum speed of the drone (in this case your walking speed). If your recordings do not contain any outliers you can manually add a few in order to test your algorithm.

4.4 Simplify the track

Create a Python class to generate a route plan consisting of waypoints (latitude, longitude, altitude) based on removing excessive track points, thereby simplifying the track.

The class should be able to perform simplification based on the different methods below:

1. maximum waypoints allowed.
2. minimize distance deviation from track.
3. minimize distance and bearing angle deviation from track.

If you by studying literature decide for a slightly different approach, this is ok as long as you compare the results from the implemented methods.

In this exercise it is acceptable to perform a visual comparison of the results and the original track using matplotlib. You may also wish to use the `exportkml.py` available under the module 2 course materials which enables you to view the result in e.g. the google earth app.

You may find inspiration here:

- Ramer–Douglas–Peucker algorithm[1]
- Visvalingam-Whyatt algorithm[2] [3]
- gpsbabel[4]
- gpsvisualizer[5]

4.5 Convert to geographic coordinates.

The coordinates should be converted to geographic coordinates using the same procedure as you did in module 2.

4.6 Export as a route plan to QGroundControl software.

Export the coordinates to a route plan format readable by QGC. At the course materials for this module you will find an example file `qgc_generate_plan_example.py` on how to generate a route plan readable by QGC. Please encapsulate the export functions in a class for future code reuse.

When you have exported the route plan, you can make a final test by uploading the plan to your flight controller using QGC then downloading it again and save it to see if there are any changes.

4.7 Fixed wing optimization

The purpose of this exercise is to take a route plan (simplified track) and then add interpolation waypoints where relevant to ensure that a fixed wing drone will follow the route smoothly instead of greatly overshooting each time a waypoint is reached.

The file `hermite.py` contains a simple example of a cubic hermite spline algorithm. By traversing the track and for each leg observing the angles that the fixed wing drone enters and exits the leg, it is possible by adding a few interpolation waypoints to obtain a smooth flight.

[1] https://en.wikipedia.org/wiki/Ramer-Douglas-Peucker_algorithm
[2] Paper in course materials: *1992 Line Generalisation by Repeated El imination of the Smallest Area.pdf*
[3] https://github.com/Permafacture/Py-Visvalingam-Whyatt
[4] https://www.gpsbabel.org/htmldoc-development/filter_simplify.html
[5] http://www.gpsvisualizer.com/tutorials/track_filters.html