Collecting information from sensors using drone and LoRaWAN Protocol Top Level Design

Document Owners	*Max Kolchinsky, Ilya Freydkin*
Instructor / Guide	Itzik Ashkenazi, Aviel Glam
Table of contents	
Date	08.04.2019
Document Version	1

1. Introduction

Background

LoRaWAN is a point-to-multipoint networking protocol that uses Semtech's LoRa modulation scheme which is designed for long range, low power wide area networks with low cost, mobility, security and bidirectional communication. LoRaWAN uses ADR to transmit. The selection of the data rate is a trade-off between communication range and message duration. LoRaWAN supports coverage range of 2-5km (urban environment), 15km (suburban environment).

We would like to create sensor-gateway communication between endpoints which are sensors located in a large polygon area and a gateway located on a flying drone. For this purpose, LoRaWAN protocol fits our purpose best as a MAC layer protocol with LoRa at the physical layer.

Overview

The main idea of the project is to create a software which allows based on an input of sensor locations, obstacles in polygon area and a start point to create an optimal path for a drone to fly and gather data from the sensors.

The idea is to create a static path based on available parameters of the polygon area and to make both post-fly path adaptations (phase 1) based on the inputs we received from the sensor and dynamic adaptations on the fly (phase 2).

In this project we will write an algorithm for PP for phase 1 and potentially also an algorithm for phase 2. We will establish a sensor-gateway communication using LoRaWAN and establish gateway-server communication using the following method:

- Loading and unloading communication using ethernet (get the data from the drone after it returned successfully)

References

Jun M., D'Andrea R. (2003) Path Planning for Unmanned Aerial Vehicles in Uncertain and Adversarial Environments. In: Butenko S., Murphey R., Pardalos P.M. (eds) Cooperative Control: Models, Applications and Algorithms. Cooperative Systems, vol 1. Springer, Boston, MA

Radmanesh, Mohammadreza & Kumar, Manish & H. Guentert, Paul & Sarim, Mohammad. (2018). Overview of Path Planning and Obstacle Avoidance Algorithms for UAVs: A Comparative Study. Unmanned Systems. 6. 1-24. 10.1142/S2301385018400022.

Terminology and Acronyms

Term	Meaning
ADR	Adaptive Data Rate
UAV	Unmanned aerial vehicle

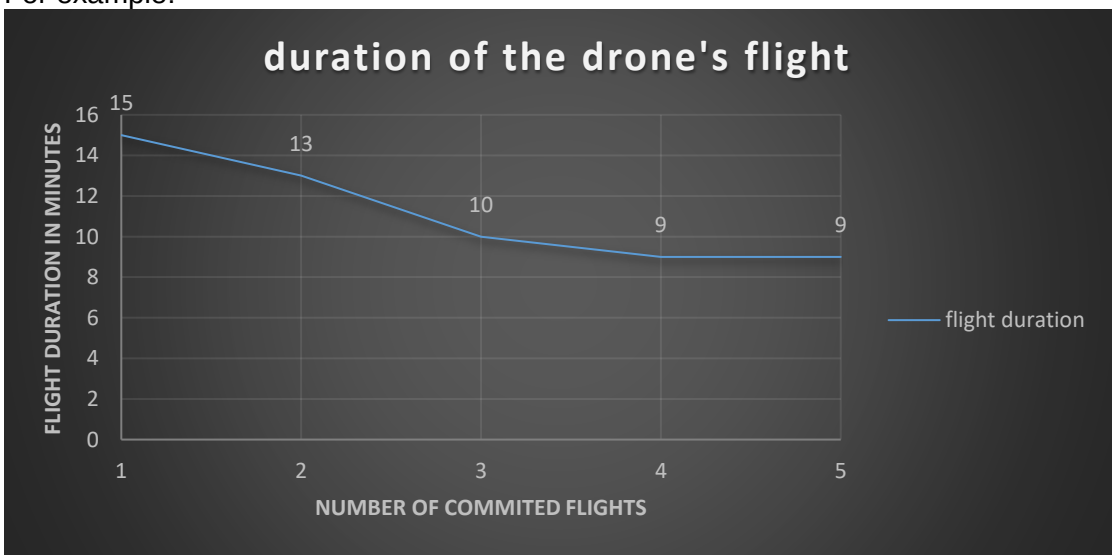
2. Project General Architecture

System description

As we said before, using LoRaWAN protocol we will create sensor-gateway communication between endpoints, which are sensors located in a large polygon area, and a gateway located on a flying drone.

We will load all input data, which is starting position of the drone, obstacles, sensors, location and their minimal range to our Path Planning algorithm. The output will be the optimal path for drone using this information.
During the flight, the drone (gateway) will collect data from sensors. Besides receiving data from them, we will also be able to figure out more accurate information about sensors range. We will use this information for optimization of the path and in this way, we will be able to reduce duration of the drone's flight. The more flights the drone commits, the more accurate the information is received and thus the more optimal path we will be generated for the drone.

For example:



3. Management overview

CLI overview

The main part of the project, as described previously, is the path planning algorithm which will be used to send the drone and collect the data from sensors. The algorithm input will consist of an input containing:

1) Start point of the drone (x_d, y_d)
2) Positions and radii of n sensors as 3d points $\{(x_{s_1}, y_{s_1}, r_{s_1}), (x_{s_2}, y_{s_2}, r_{s_2}), \dots, (x_{s_n}, y_{s_n}, r_{s_n})\}$
3) Positions of m obstacles as bboxes $\{((llx_1, lly_1), (urx_1, ury_1)), ((llx_2, lly_2), (urx_2, ury_2)), \dots, ((llx_m, lly_m), (urx_m, ury_m))\}$

The input file shall be a csv file s.t the 3 points mentioned above will be 3 lines in the csv file. The groups mentioned in points 2 and 3 shall be displayed as flat csv data.

Configuration examples

```
# First input data for 2 sensors with 0 radius for first traversal
100,100
120,260,0,130,260,0
120,140,150,200
```

4. Project Planning

- Time Table

Week	Plan
1	-
2	Overview and understanding of the project. Learning about Lora and LoRaWAN
3	Setting up the environment. Creating working connection between sensor, gateway and server. First proposal of the Path Planning Algorithm.
4	Completing environment setup. Writing HLD presentation and finishing Path Planning algorithm design.
5	Implementation of Path Planning static algorithm and dry tests. Understanding how to control the drone and be able to use algorithm output to send him flying.
6	Continuing implementation of Path Planning algorithm to make it dynamic. Parsing SNR results to add adaptive capabilities to the algorithm and improve after a flight.
7	First week of experiments. Field testing of the algorithm and fixing arising issues. Testing on small area.
8	Second week of experiments. Testing over large area with more sensors.
9	Extra week space in case issues arose in previous weeks that might've caused a delay
10	Extra week space in case issues arose in previous weeks that might've caused a delay
11	Making final presentation of the project
12	Final presentation of the project

- Mile Stones

1. Setting up the environment, understanding how the code works and being able to do modifications.
2. HLD presentation
3. Working dry implementation of Path Planning algorithm.
4. Wet simulation of drone flying by using our Path Planning algorithm
5. Getting improved results after every flight
6. Final presentation

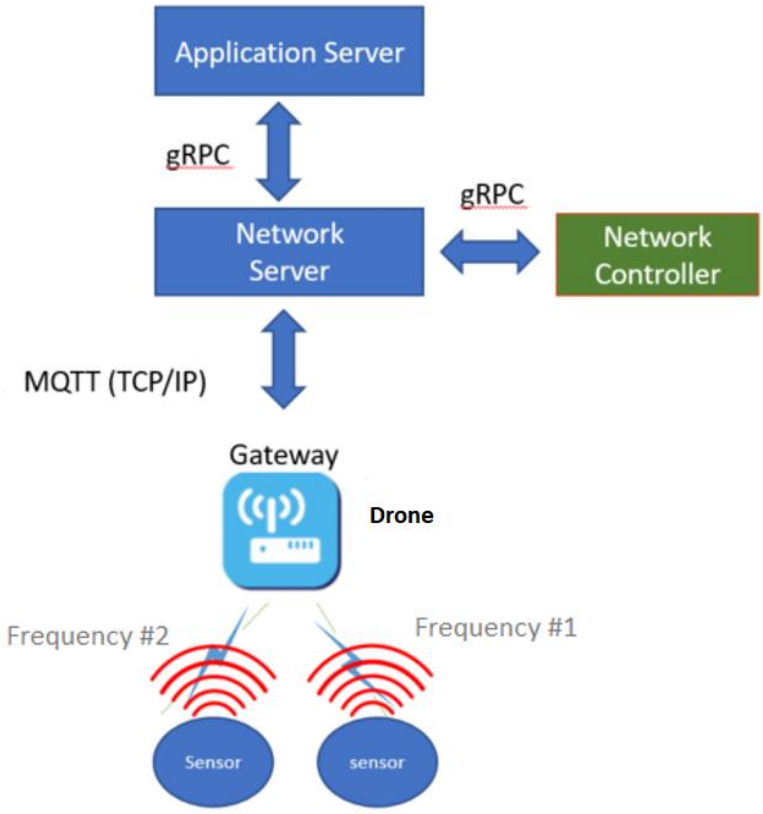
- Risks

1. Sensors wake up by duty cycle. Need to make sure all sensors wake up at the right time, so the drone can collect all the data.
2. Issues may arise with setting up the environment due to not having enough background in this specific environment, working with hardware and networking in general.
3. Some issues may arise only during wet tests and might leave us little time to work on fixing them.

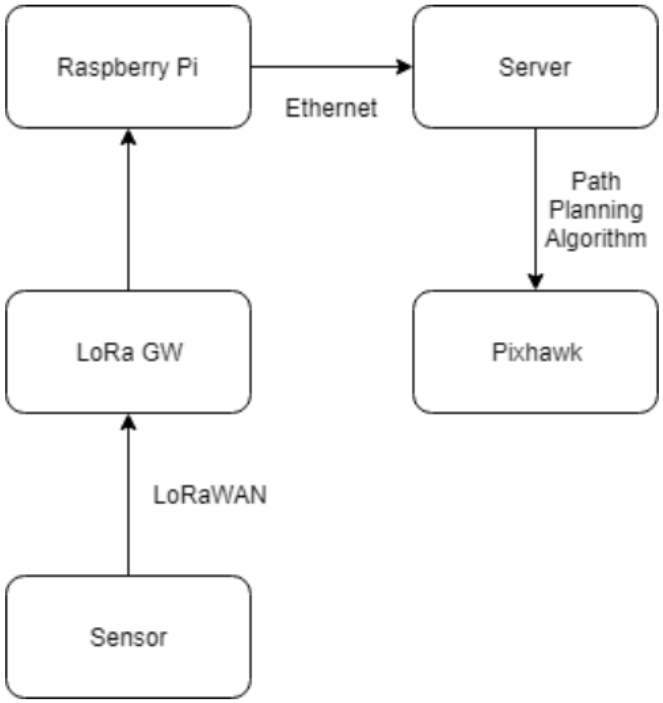
4. We might not reach the expected results.
5. Equipment might be damaged during simulation. Always need to have backup drone, sensors and gateways.

Appendix

Architecture



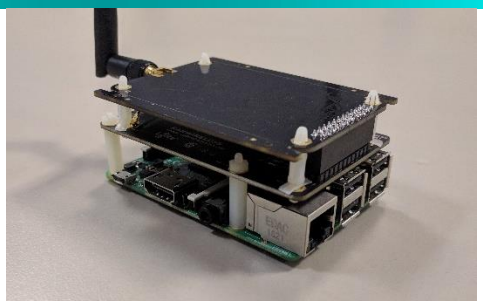
Architecture Block Diagram



Sensor (end node):

- We will use STM Nucleo evaluation board with Semtech sx1272 LoRa transceiver for the sensor boards (same as last project).
- We will use STM iks01a2; temperature, pressure and humidity sensor.
- The development of the sensor's firmware is largely based on an OpenSource project called st-lrwan developed by ST.
- The IDE is System Workbench for STM32, which is a proprietary development environment for STM32 architecture (based on Eclipse).

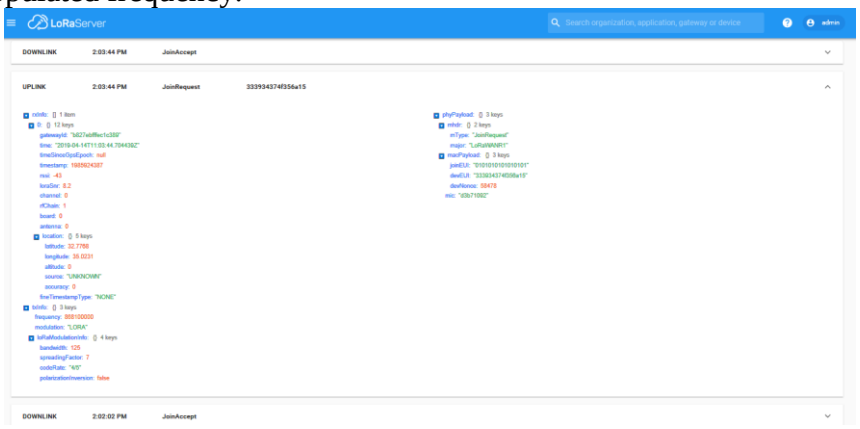
RAK831 – Multi-channel Gateway:



- We will put a Raspberry Pi with RAK831 LoRaWAN multichannel transceiver on drone.
- The software for this gateway is a gateway software developed by Semtech specifically for Raspberry Pi.
- Semtech is the main manufacturer of LoRa transceivers and the main maintainer of the LoRaWAN standard.
- This gateway implementation was specifically designed to provide testing capabilities for all of the features of the LoRaWAN standard.

LoRaServer – Network Server:

- For the network server we will use LoRaServer OpenSource LoRaWAN server implementation.
- We will connect the server to gateway over the network (Ethernet).
- Configure the server to communicate with the sensor using OTAA.
- The LoRaServer supports external network-controller.
- The controller act as follow
- Every uplink transmission is registered with its address and TX frequency.
- All of the devices are stored in a local DB.
- The controller look for “unbalanced frequencies”, i.e. two devices uses the same frequency while another frequency is empty.
- If “unbalanced frequencies” is detected the controller sends a special mac command to one of the sensors in the densely populated frequency.
- The sensor receive the message reconfigures to transmit using the under populated frequency.



Path Planning Algorithm

Definitions:

Sensor intersection: for n sensors $s_1, \dots, s_n$ located in points $\{(x_1, y_1), \dots, (x_n, y_n)\}$ for which there is an intersection between the signal radii, if $n > 1$ let p_{si} be the sensor intersection polygon between them which we will call in short sensor intersection. If $n = 1$ let p_{si} be the point of sensor s_1 bloated by the signal radius of s_1.

Closest point: Let (x_d, y_d) be the location of the drone and p_{si} be the sensor intersection of the next node in traveling salesman path. Then closest point (x_c, y_c) will be the point in polygon p_{si} s.t

$$(x_c, y_c) = \min_{x_i, y_i}(\sqrt{(x_d - x_i)^2 + (y_d - y_i)^2} \qquad \forall (x_i, y_i) \in p_{si}$$

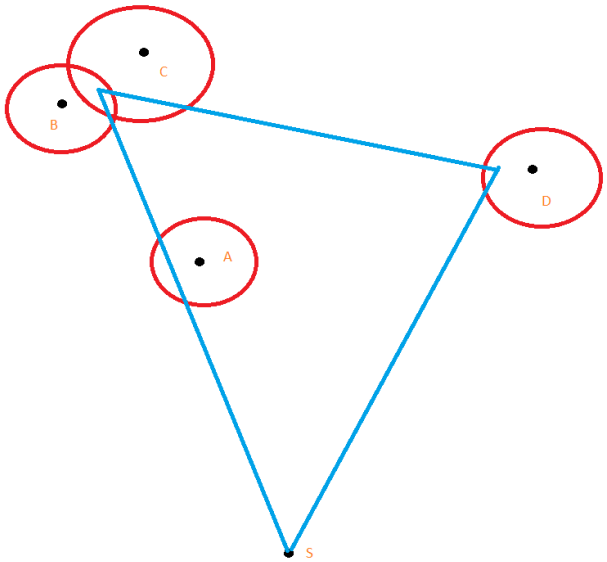
Closest distance points: Let p_{si_1} be the polygon of one sensor intersection and p_{si_2} be the polygon of a second sensor intersection. We will define:

$$\{(x_{cd_1}, y_{cd_1}), (x_{cd_2}, y_{cd_2})\} = \min_{x_i, y_i, x_j, y_j}(\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \qquad \forall (x_i, y_i) \in p_{si_1}, \forall (x_j, y_j) \in p_{si_2}$$

Problem Graph: A full graph for which the nodes are sensor intersections and the edges are closest distance points between each 2 sensor intersections. The distance of closest distance points is calculated by the distance on the polygon grid generated by "Path Planning for unmanned aerial vehicles in uncertain and adversarial environments"[1].

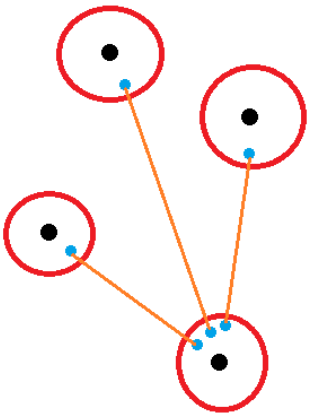
The solution to the path planning algorithm will be divided into steps.

1) Build the Problem Graph – For first traversal graph, each sensor has 0 radius, meaning that a node in the graph is the sensor with sensor intersection containing a 0 sized polygon which is the point itself.
2) First traversal - Calculate travelling salesman solution for the previous problem graph using brute force.
3) Parse sensor output and calculate signal radii using SNR for each of the sensors.
4) Build problem graph using previous step input and adapted edge weight calculation with closest points. Go back to stage 2.

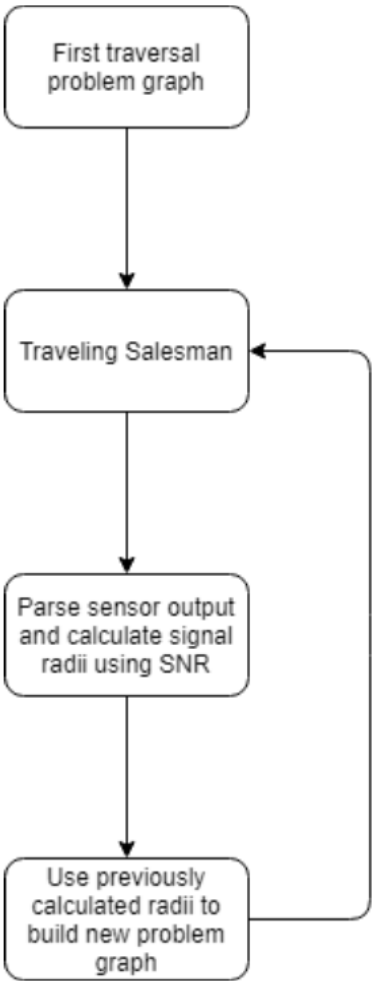


Adapted edge weight calculation

This is an algorithm used to calculate problem graph for points that have a signal radius to calculate an optimal path between those points. The input to the algorithm is a list of polygons (sensor intersections) and a start point.
For each two polygons, calculate the distance between them as the distance between closest distance points of the polygons.



Algorithm block diagram



- References

[1] Jun M., D'Andrea R. (2003) Path Planning for Unmanned Aerial Vehicles in Uncertain and Adversarial Environments. In: Butenko S., Murphey R., Pardalos P.M. (eds) Cooperative Control: Models, Applications and Algorithms. Cooperative Systems, vol 1. Springer, Boston, MA