**Autonomic movement of crazyflie drones algorithm**

There algorithm composed from two modules:

1. Building and publishing a grid from the received raw data (Grid node).
2. Using the built grid to send each drone its cj's which are coordinates to which the drone will be moving (Cj\_injectore node). This will be done in a few steps:
   1. identify the interest points (array of coordinates to which the drones should be moving in order to explore all of the unknown area). Carried out in GridPOI (POI – points of intrest) module.
   2. Identify the corner points. Once an interest point is associated to a specific drone, if there is no line of sight to it, the path to it will be built using the corner points. Carried out in GridPOI module.
   3. Associate each drone with a specific interest point. Here all drones position and goals are taking in account. Carried out in get\_goal\_point function.
   4. Build an optimal path to each interest point using Astar algorithm. Carried out in PathBuilder module.
   5. Use agent module to decide what will be the next step (goal) of each drone. This will be done according to Astar path, current drone position, the distance from its previous goal and the revealed obstacles (updated grid). Carried out in Agent module
   6. Publish chosen cj's to each drone. Carried out in cj\_injector\_container module.

In addition there is a Display node which subscribes to drones position and grid. It displays all received data at each time stamp.

**Grid node**

Subscribes to point cloud (PC) messages and to tf (current position sensing of the drone). Publish the updated grid. At the beginning (before any messages are transferred) the grid is a matrix of zeros composed from tails of resolution res. Each PC coordinate reveals a tail in the grid. The value "1" indicates open space and the value "2" indicates a wall. In addition there is a timer which activates mode show\_real\_pc. This mode reveals the grids relaying on the current PC and not on the addition of all PC messages from the beginning of the flight. This is useful when the drone doesn't move for a period of time and creates dummy walls.

**Cj\_injector node**

Subscribes to occupancy grid messages and to tf. Publish cj message for each drone.

Blocks diagram of Cj\_injector node:

Drone injector:

* Subscribes to Grid and position
* Finds POI every n seconds
* Chooses a specific point for each drone

Drone Cj injector:

* Finds the next position according to Agent module.
* Finds the next rotation angle (yaw) according to timer.
* Constructs Cj message and publish it.

Drone Cj injector:

* Finds the next position according to Agent module.
* Finds the next rotation angle (yaw) according to timer.
* Constructs Cj message and publish it.

Drone Cj injector:

* Finds the next position according to Agent module.
* Finds the next rotation angle (yaw) according to timer.
* Constructs Cj message and publish it.