Distributed Systems 72020
Final Project Report
Prof. Kaliappan Ravindran
Date of Submission: 5/28/2020

Pollution Detecting Mobile Drones in a Wireless Ring Configuration

Carlos Tavarez Martinez
City College, CUNY, NY
ctavare003@citymail.cuny.edu

Vivek Sharma
Graduate Center, CUNY, NY
vsharma@gradcenter.cuny.edu

Edis Bibuljica
Graduate Center, CUNY, NY
ebibuljica@gradcenter.cuny.edu

Goal: Designing a functional wireless network with ‘air pollution’-sensing drones.

1.Introduction:

A Content Delivery Network (CDN) is a network containing data centers and proxy servers [1]. The aim of the CDN is to provide high availability of the service. This task is fairly achievable with the engineering advancement in a wired network. But when it comes to wireless networks with mobile nodes, the challenge of reconfiguration requires a different perspective in achieving a similar goal. The aim here is to design a system with a wireless network and mobile nodes which provides continuous service and as well is fault tolerant. Since the nodes are mobile in a wireless environment, and different factors come into play, the challenge of reconfiguring becomes much more challenging. To top it all, the limitation of a mobile node like limited processing power, limited power, and limited storage also adds up to the problem. This system design is an attempt to solve the availability problem, addressing various factors that may affect the service.

One motivation to this particular problem is the use of drones to reach and explore locations to identify places where toxic gases might be generated. Normally, identifying the source of toxic gases is a matter of receiving a high signal from sensors then building a map of the spatial concentration of the gas using methods such as inverse distance, nearest neighbor or Kriging [2]. Building a spatial concentration of toxic gases in conflict zones having military disputes and lacking a network infrastructure is possible using drones carrying air quality reader sensors. Yajie et al proposed a low cost and fault-tolerant distributed system of sensors to monitor air pollution in London using stationary and mobile nodes [3]. The use of drones have become popular in artificial intelligence and computer vision [4]. Drones have also been used in networking applications to create a mesh network and provide high speed WiFi and transmit video in areas without a network infrastructure [5]. Our goal is to use drones to identify and track high mass concentrations of toxic gases without the need of a proper network infrastructure.

We design this wireless network in the pseudocode section to be used this air pollution'-sensing system. It involves interaction between Ground Control/Base Station(GCS), leader drone and the worker drone. The GCS contains three sub-modules. The Geo map maps sets the destination and sends the coordinates to the worker drone. The Pollution map performs computation on the raw data and with the help of the Kriging map, determines the update in the coordinates for the particular drone. The scenarios covered in this section involves, software failure recovery(roll-forward), hardware failure(replacing the faulty drone), assigning position to the drone, checking the status of the drone, sensing the air pollution level, forwarding the measured data and performing computation on it.

Our work will begin with exploring the data structures employed. We show the components of our mobile network with the use of distributed systems. We evaluate run-time traces with the use of efficient algorithm design. To take our network tree and develop a mathematical model to the reader to show communication channels throughout the distributed computer system.

2. Procedure/Simulation:

We start by defining our pollution searching wireless network with mobile drones using distributed computer systems. Our network will be devised using a CDN (Content Delivery Network). It will be a fully synchronous network so our upper layer-algorithm DMAX will be defined so that maximum delay in the channel will be known.The components to our wireless network will allow for maximum efficiency in communication so that optimal locations of pollution can be quickly found in this WIFI based communication transmission. This will allow for a distance of about 160 meters when it comes to video communication and about 200 meters for data communication.

Our distributed system will consist of 3 main components, routers, sensors, and drones. The router's purpose is to bridge the communication with drones. We will have sensors such as the stationary pollution sensors that will identify pollution targets of interest on the geographic map. Lastly the drones with pollution sensors and cameras for visualization will enable us to have mobility when it comes to accurately determining precise locations of pollution on the field. Drones will be able to move at a maximum travel speed of 10 meters per second. The mobile drones will have neighbor to neighbor communication in the wireless tree network of drones. We also have a system of backup drones in case of malicious attacks used on the network's distributed system. Main communication will be done using the ground base station. The master content pages will consist of a geographic map, pollution geolocation map, and a Kriging map. Let us define master content pages

- P-a- Geographic map
- P-b- Pollution based map

- P-c- Kriging map

All new information learned from analysing the new data from the wireless network of mobile drones will be stored as cached pages. We shall denote the memory as {x}. Any updates being sent along the network communication channels will be denoted as the function U({x}).

- {x} - cached pages
- U({x})- Function for sending updates of {x}

Other components in our distributed system include the stationary pollution server. These servers store all data locations on known mass concentrations of pollution in the world. The geographic map server stores all maps throughout the world, along with coordinates, ground height, and areas with large bodies of water. The ground base station server holds all relevant data of commanding drones and destination locations of high density pollution. We also incorporate interconnect servers so that all communications of drone information and updates are analysed, received, and sent.

In the development of the system we will have all messages collected by each drone in the cached pages {x}. Communication from drone to drone will be done through network channel communication denoted by function U({x}). Thus drone1 or leader drone will obtain information from its sensors and store it in the cache pages and we denote this by {x1}. When information is sent to drone2 along the network channel using function U({x1}) drone2 receives the data and adds its own readings from its sensor. It then sends the cached pollution concentration data {x2} to drone 3. This process continues until all drones are exhausted.

We then take our raw pollution mass concentration data and send it to the ground base station. There we begin by sending the data along the interconnect server to the pollution map where the stored master content page P-b lies. (See Fig. 1) The information is then appended so that the pollution map has an accurate time stamped data set based on location and pollution percentage. The data is then sent along the wired channel using function U({P-b}) to the interconnect server. After data is then passed to the master content server KrigMap P-c for analysis.

The KrigMap server then exports valuable areas of interest to the GeoMap master content server P-a. The GeoMap server takes the areas of interest and then commands the leader drone to survey those areas for possible pollution hotspots. This is done by sending U({P-a}) thru the wired interconnect server and out of the ground base station. After U({P-a}) is broadcasted wirelessly to the leader drone. After the leader drone assigns destinations for each drone to survey possible pollution locations.

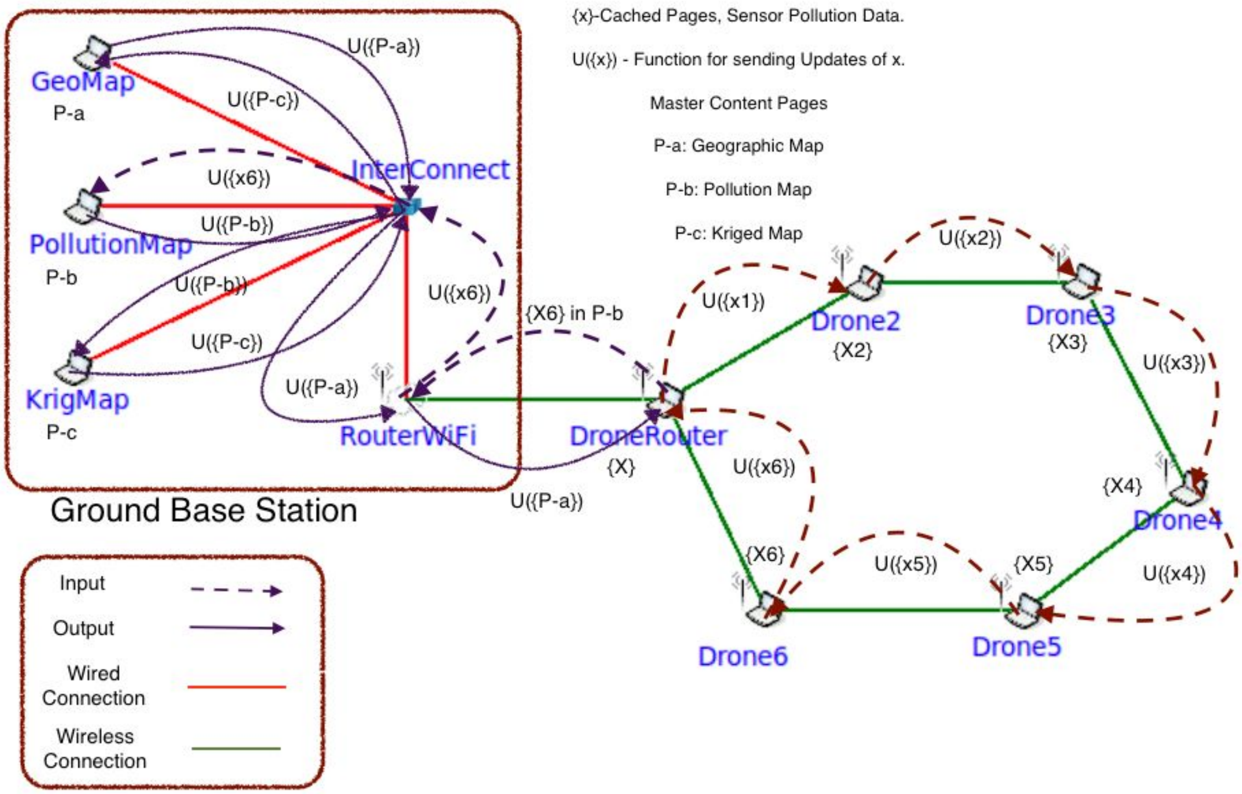

Fig. 1: Snapshot of the system with state of the environment.

Let each client in the distributed system be a drone. Each drone will be expressed as a process pi for i=1,2,3,...,N. The leader drone will be determined from an election algorithm. The leader will be determined by four factors.

a) Battery Strength of the drone
b) Signal Strength of the drone
c) Number of neighboring drones

a) **Battery Strength**: Assuming that the battery capacity for each of the drones remains the same. Each drone computes the remaining battery percentage. Suppose if 1100mAh is the remaining charge of a 2000mAh battery, the battery strength is 55%.

   For a mobile device, battery is one of the limited resources. A leader has to continuously coordinate between drones and Ground Control Station(GCS) and also perform many important tasks. It also needs to have a clear and strong signal strength among its peers.

b) **Signal Strength :** Signal strength is the measure of the power of the signal received. It is an indirect measure of the distance between a drone and the Ground Control

Station(GCS). The leader has the following two factors that can measure the signal strength of a data link.

- **Received Signal Strength Indicator (RSSI):** It measures the power of the received signal. The strength of the signal is generally measured in decibel milliwatts(dBm). Following table shows the measured power and the strength of the signal.

Power	Strength
>-76 dBm	Excellent
-89 to -77 dBm	Very Good
-97 to -90 dBm	Good
-103 to -98 dBm	Low
-112 to -104 dBm	Very Low
< -113 dBm	Unlikely Connection

Table 1: Signal Strength of a signal based on their power

- **Signal to Noise Ratio (SNR):** It is given by ratio of power of signal to power of the noise in that signal. High SNR value means high quality of signal.
  - $SNR = \frac{P signal}{P noise}$

  Using one among these two factors, we can find the score and normalize it.

c) **Node Connectivity:** It is the ratio of the total number of neighboring drones to the total number of deployed drones. A leader is required to be at the suitable position from where it can send and receive information from the neighbouring nodes.
Let,
N be the total number of deployed drones for the event and
n_i be the total number of neighbouring nodes of a drone p_i.

The Node Connectivity(conn) of the drone will then be given by

$$conn = \frac{n_i}{N}$$

Let δ be the time of network channel delay in the distributed system.

Based on these three factors, the rank of a drone is determined by a total score. The one with the highest score becomes the leader.

So based on the factors, formula for rank can be given by

Let ,

batt = Normalized Battery Strength
signl = Normalized Signal Strength
conn = Node Connectivity
b, s, n = their respective weights.

The accurate weights can be found by experiments, but for our research we assign the highest weight to the battery followed by signal strength and then node connectivity. So b=0.35, s=0.5 and n=0.15.

$$Rank(p_i) = b\ x\ batt + s\ x\ signl + n\ x\ conn$$

3. Drone to Drone Communication:

The job of the leader drone is to decide which drones are going to work and lead the drones. Also, get the data from other drones and send data to the base station. The base station communicates with the leader drone to start an analysis of testing pollution concentration in the field. The drones will communicate neighbor to neighbor. So drone 1 will communicate a list to drone 2 of pollution concentration. This will follow up with drone 2 appending the list with its own record of pollution concentration, and this will go on to the last drone. Let δ be the time of network channel delay in the distributed system.

In our analysis of runtime traces in finding mass concentrations of pollution our drones will need to communicate to the base station effectively so that we can accurately determine the locations on the map. We will need an identification ID for each drone that we define as pi for i=1,2,3,....,N. Since pollution concentrations can move from a variety of factors such as wind, rain, and temperature. We need a timestamp to accurately determine pollution mass travel.

Also to correctly define locations of pollution we need to define the latitude, longitude, and altitude. We define them as xi for latitude, yi for longitude, and ai for altitude for i = 1,2,3,...,N. Our variable for pollution mass concentration percentage will be denoted as ci for i = 1,2,3,...,N. (See Table 2.) The table will then be used to analyze optimal areas on the geographic map to search for the sources of the mass concentrations of pollution. These tables can be updated for specific constraints based on what is being analyzed such as adding additional columns to show wind speed, humidity, temperature, and etc. We will use this data along with the Kriging formula for visualization to predict possible locations of mass concentrations of pollution. Below in mathematical expressions is an in depth perspective on how we will apply Kriging to finding precise locations of pollution concentration.

DRONE_ID	TIME_STAMP	LATITUDE	LONGITUDE	ALTITUDE	CONCENTRATION_%
p1	t	x1	y1	a1	c1
p2	$t+\delta$	x2	y2	a2	c2
p3	$t+2\delta$	x3	y3	a3	c3
…...	…...	…...	…...	…...	…...
pi	$t+(i-1)\delta$	xi	yi	ai	ci
...	…...	…...	…...	…...	…...
pN	$t+(N-1)\delta$	xN	yN	aN	cN

Table 2. Drone pollution mass concentration data on map.

To show the cause and effect relationship of the distributed system we develop a concrete example. In the tree construction we have two main performance indices time-to-reconfigure and network-wide message overhead. When a tree node fails, time-to-reconfigure is a measure of the parallelism in the search process for the network to heal around a failed node. If a drone dies in the distributed system the neighbor drone will wait $x\delta$ time for a response to see if that drone received a message. If no response is received then the drone will communicate back to its previous neighbor informing the system that a drone is missing. This will then patch the system of the missing drone and re-link the distributed system. We will ping the drone to see if the neighbor is working properly.

The network-wide message overhead incurred for the reconfiguration of failed mobile drone nodes is dependent on how large the distributed system is made. If we have a large set of drones it will depend on particular distributed computer system hardware and since we have a fully synchronous system we can calculate network delay and have a max number of messages exchanged between nodes in the entire network as part of reconfiguration for failed drones in the system. Figure 2 shows the drones as they are designated in their assigned locations after a system check(ready()) by the GCS. The drones report back to the leader drone once they reach their destination and then start sensing the air pollution level once it receives the start command. In case of internal failure, the drone roll-forwards and starts sending the measured mass concentration to the GCS via neighboring drones and the leader drone.

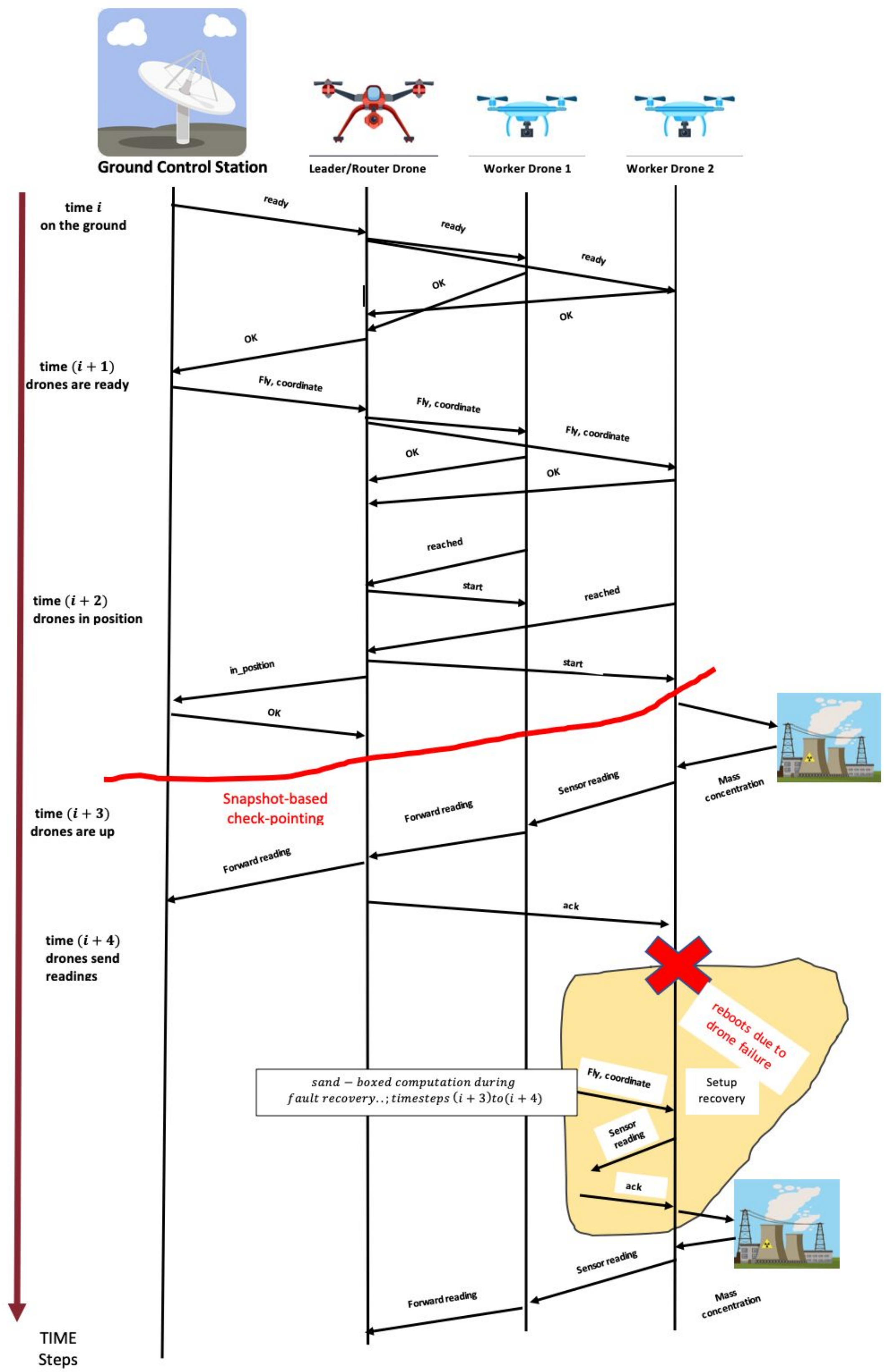

Fig. 2: Scenario roll forward execution in mobile drone system

Figure 3 shows that an unresponsive/dead drone is decommissioned from the service after a repeated start command. The GCS arranges for the backup drone, pulls up the last saved location from the geomap module and commissions a new drone to arrive at the location. Once reached, the newly commissioned drone sends an in-position message and starts sensing the air pollution level.

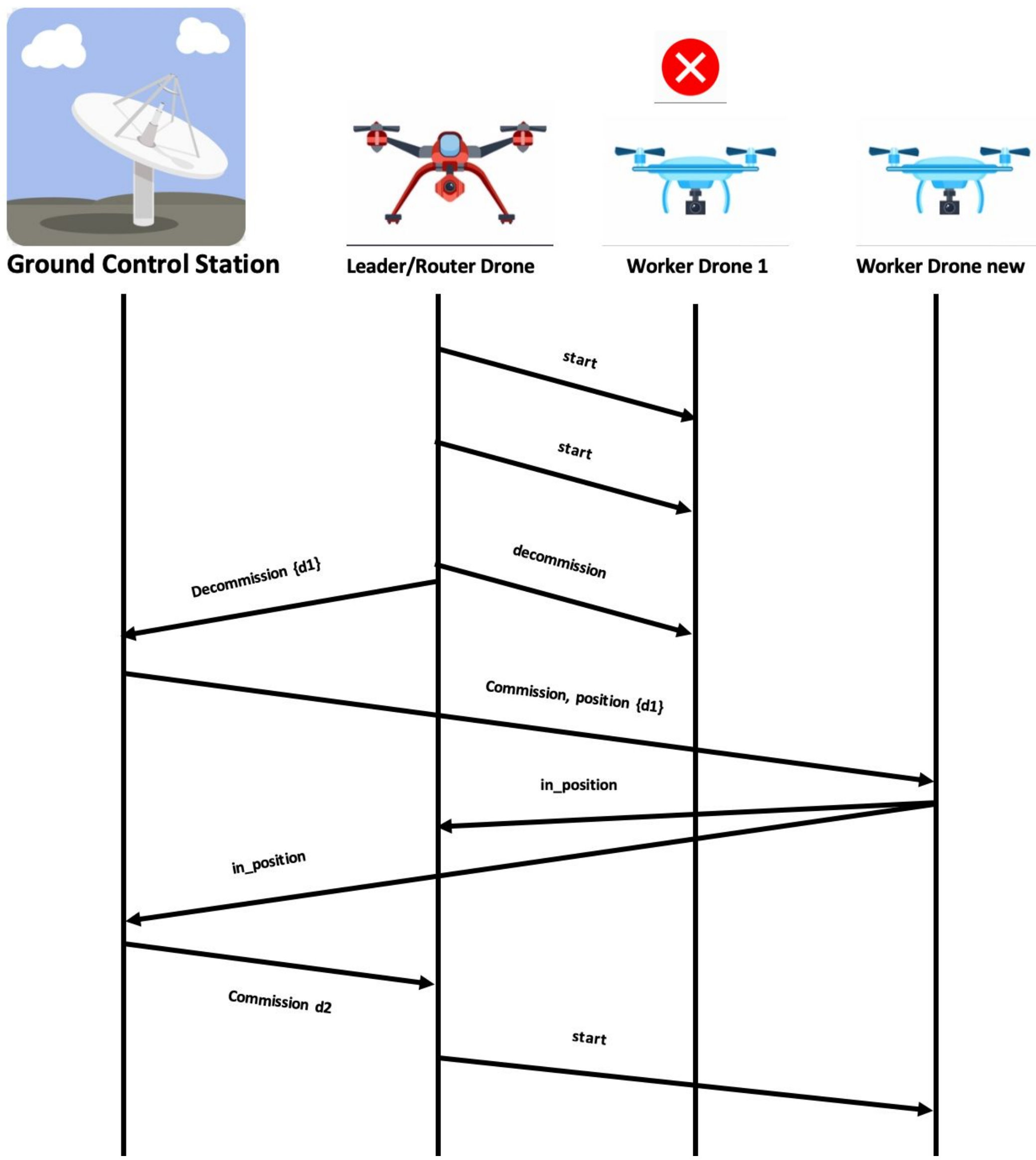

Fig. 3: Fault recovery of service by replacing the faulty drone

4. Pseudocode:

Following contains the code that would run in Ground Base/Control Station(GCS), leader drone and worker drone respectively. The GCS will have three sub-modules within it which will be identified as gcs.geo_map, gcs.pollution_map, gcs.krig_map in the code.

a) gcs_controller_main:

```
while(TRUE):
    Initialize the list_of_drones and leader_drone;
    if(__init__): //start of the message
        send(list_of_drones, loc_i) to leader;
        send(ready) to leader_drone;
    if(receive(ready) == OK):
        receive locations loc_i from GeoMap;
        send(loc_i) to leader; //loc_i is the destination location for drone d_i
    if(receive(in_position)):
        add_drone(d_i) to list_of_drones;
    if(receive(U{x_6})): //U{x_6) is the raw sensor data
        forward x_6 to module gcs.pollution_map;
    if(receive(decommission{d_i)):
        remove d_i from list_of_drones;
        d_j = available_drone();
        loc_i = gcs.geo_map.getLocation();
        commission_drone(d_j, loc_i);
```

Explanation:

Roughly, a GCS controller can receive three broad categories of signals,

1) **Initialization signal:** This signal is received at the start of the initialization. Right after the GCS sends a ready command to the leader drone, it waits for the OK message from the leader drone.

2) **Data Signal:** The worker drone measures the mass concentration from the field and sends the raw data to the leader drone. The leader drone then forwards the data to the GCS which gets forwarded to the pollution map module.

3) **Error Signal:** Once the drone faces a hardware failure, it is decommissioned by the leader drone. The location of the decommissioned drone is sent by the last position stored in the geomap module to the new drone.

```
b) leader_drone_main:
dead_drones = list_of_drones;
while(TRUE):
    if(receive(ready)):
        forward ready to all d_i;
    if(receive(OK)):
        forward OK to GCS;
    if(receive(fly,coordinates)):
        i:=0;
        do:
            send fly_to(loc_i) to d_i;
        until(i<=n);
    if(receive(reached)):
        flag:=0;
        dead_or_alive(); //This function is defined below
        if(dead_drones ≠ NULL):
            flag:=1;
            dead_or_alive();

    if(receive(x_6)):
        forward(x_6) to GCS.
    if(receive(commission{d_j})):
        Send start() to d_j

dead_or_alive():
    if(flag == 0):
        send start d_i;
        wait for time δ.N; //N is the total number of employed drones
        if(receive(sensor_reading) == TRUE ∀ d_i):
            remove d_i from dead_drones list;
            forward(sensor_reading) to GCS;
    elseif(flag == 1):
        send decommission(d_i) to dead_drones and GCS.
```

Explanation:

A leader drone's primary task is to coordinate between the worker drones and the GCS. It needs to forward the sensor data to the GCS while continuing to monitor the health of the worker drone. If the leader drone finds that the worker drone has turned dead, it needs to decommission the drone from service while informing the GCS about the same. So that the GCS can arrange a back-up drone to take the place of the dead drone.

```
c) worker_drone_main:
while(TRUE):
        Mass_concentration = monitor();
        if(receive(ready)):
                if(check_systems() == OK): //involves checking memory, CPU and peripherals
                        send(OK) to leader drone;
        if(receive(fly,loc)): //loc is the location coordinate
                move(loc);
                send(OK) to the leader drone;
        if(receive(START)):
                Send mass_concentration to neighbour drone;
        if(receive(commission, position{d_i})):
                if(check_systems() == OK): //involves checking memory, CPU and peripherals
                        move(loc_i);
                        send(in_position) to leader and GCS;
```

Explanation:

The task of a worker drone is to continuously monitor the air pollution level and send the raw data to the GCS via neighbouring drone and the leader drone. It also needs to move to the coordinates computed by the geomap module of the GCS.Pretty good man!!!!

5. Mathematical Expression:

Simple Kriging is a geostatistical technique used to make linear spatial interpolation. To estimate the mass concentration of a gas using Kriging, it is necessary to know the distance of the point being estimated to the sensors that measure the concentration of the gas. Compared to other techniques such as inverse distance or nearest neighbor, Kriging distributes the weight coefficient among close neighbors. To interpolate using Kriging we need at least the measure and location of two sensors. This data is used to build a two points covariance matrix. Generally the covariance of a given point decreases as a function of distance to its initial point. To understand how Kriging works let us look at the Kriging equation.

$$Z(longitude, latitude) = \begin{pmatrix} Z_1 \\ \vdots \\ Z_n \end{pmatrix}^{`} \begin{pmatrix} Cov(x_1, x_1) & \cdots & Cov(x_1, x_n) \\ \vdots & \ddots & \vdots \\ Cov(x_n, x_1) & \cdots & Cov(x_n, x_n) \end{pmatrix}^{-1} \begin{pmatrix} Cov(x_1, x_0) \\ \vdots \\ Cov(x_n, x_0) \end{pmatrix}$$

In this equation, Z_1 to Z_n is what is being read by the sensors from position 1 to n. The second component on the right is the two points covariance matrix, which is the covariance from sensor 1 to sensor n, and the last term is the estimated covariance of the point to be interpolated based on its distance to each sensor. Ideally Kriging might give an initial snapshot of the spatial concentration of a gas and those estimated values can be used as a guide to inform the drones of a possible target.

The job of the drones is to read and follow strong signals of mass concentration. As a drone starts sending data of the actual state of the environment, such data can be used to improve spatial interpolation and at the same time localize the source of the gas. As a distributed system this will require the exchange of messages between a main server, the leader drone and the worker drones.

6. Challenges:

The main challenge is to have the tools to bring this project to life. Ideally we can do simulations, but the current open source for network simulation does not come with wireless nodes reading data from a sensor. Neither do open source for robotics simulation come with an integrated network infrastructure to simulate the communication between drones. Having drones sense air pollution correctly with respect to error from sensors. To remedy this factor we increase the error in sensor readings of latitude, longitude, and altitude to have an accurate location of where the air pollution came from. This will enable the drones to have some slack when it comes to natural errors in determining precise GPS locations from the sensors.

This problem could be improved if the drone would have an autonomic intelligent controller that learns from experience. One possible choice would be a reinforcement learning (RL) model. A reinforcement learning model learns by interacting with its environment and being rewarded by its positive actions. In this problem the reward for a RL would be tracking the high concentration of pollutants.

7. Conclusion:

In this paper we algorithmically explained how a fleet of drones can be used to track air pollution and build spatial concentration of pollutants. In an effort to make all drones work jointly, we propose an election algorithm suitable for this particular problem. To ensure the system recovers from failure, we also proposed a roll forward execution that will allow the substitute drone take-on where the failed drone was working.

In our creation of a pollution detecting mobile network of drones we show some areas of importance for future work. If we add increased visualization for drone obstacle recognition by future developing ideas from LIDAR based object detection [1]. This will enable drones to identify obstacles on the map and avoid making contact with them. This could also help with identifying the sources of the pollution, such as an uprooted tree with methane gas seeping through the ground from where the roots lie or a frozen lake with cracks that have natural gas leaking from the inner pockets of the earth.

Another future work is making drones more autonomous by using RL to pilot the drone to where the high signal of mass concentration is coming from. Another feature we would like an autonomous drone to learn is to sense the wind direction, this may result in an increase in the precision of gas tracking. We ask the community to continue researching this important subject of wireless networks using distributed systems since it can be used for a variety of applications worldwide.

References

[1] K. Ravindran, "Reconfigurable Peer-to-Peer Connectivity Overlays for Information Assurance Applications," *2009 IEEE International Conference on Communications*, Dresden, 2009, pp. 1-7, doi: 10.1109/ICC.2009.5199397.

[2] Ligas, Marcin, and Marek Kulczycki. "Simple spatial prediction-least squares prediction, simple kriging, and conditional expectation of normal vector." *Geodesy and Cartography* (2010).

[3] K. Itakura and F. Hosoi, “Automated tree detection from 3d lidar images using image processing and machine learning,” *Applied Optics*, vol. 58, DOI 10.1364/ao.58.003807, no. 14, p. 3807, May. 2019. [Online]. Available: https://doi.org/10.1364/ao.58.003807

[4] Ma, Yajie, et al. "Air pollution monitoring and mining based on sensor grid in London." *Sensors* 8.6 (2008): 3601-3623.

[5] Chand, Gudi Siva Leela Krishna, Manhee Lee, and Soo Young Shin. "Drone based wireless mesh network for disaster/military environment." *Journal of Computer and Communications* 6.04 (2018): 44.