Contents

[**1.** **Pre-Calculation** 2](#_Toc8855093)

[**1.1.** **Field of view of the camera** 3](#_Toc8855094)

[2. Bibliography 3](#_Toc8855095)

[Figure 1.1 3](#_Toc8854942)

# **Pre-Calculation**

Initially, before dividing the total area, the maximum area that the command center can handle must be calculated. This is important to know how many command center is required to serve a particular amount of area. This is calculated by taking certain factors into consideration like the number of drones that command center can use at that instant of time, the distance of the disaster area as the drones have to make two-way travel from the command center and the destination, battery capacity of individual drones, properties of the camera present in the drones, maximum speed that the drone can travel and an average current consumption of the drone when it is traveling at its top speed. Currently, some of the nominal values are assumed for the variables. The calculation for the maximum area that the command center can handle is as below.

The capacity of the battery is usually given in Ah format. Hence it is required to convert it into mAH form. This is done by dividing the capacity by 1000. For any battery, it is not optimal for its health, if it is drained to its 100% of the capacity. Hence here the capacity is peaked out at its 90%. Thus actual capacity can be calculated from the below formula

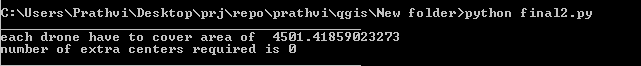
The maximum time that each drone can fly can be calculated using the formula as below.

Here the nominal current is approximated to be 8A. The resulting value will be in seconds. Using the above value, the distance that each drone can fly is calculated. Here, the round trip distance for each drone is also considered. The maximum time is calculated using the formula given below

This is the maximum distance that the drone can fly. The drone has to sweep the entire area without crossing its limitation of this maximum distance. The Maximum area that the drone can sweep also depends on the field of view of the camera (1.1) and the height at which the drone is flying. Here taking for the worst condition, say the field of view of the camera as and let the drone be hovering at the **height of . using t**he formula below, maximum effective distance can be calculated. This value is taken as 'n' in the code.

Thus maximum distance for entire command center can be calculated as below.

This value is helpful in determining if only this command center is sufficient to cover the entire area or not. If it is sufficient, it continues further by executing the required algorithms. If it is insufficient, then it calculates the number of extra command centers required and pings them as required. After dividing the area with the other command centers, it obtains a new area for itself. These pre-calculations are common for all the algorithms applied in this project.



* 1. **Field of view of the camera**

For many optical instruments, the field of view is termed as an angular field of view or linear field of view (Howard and Rogers 2014). Here angular field view is used for the calculation. It is a solid angle through which a detector is sensitive to electromagnetic radiation. For a normal lens, the diagonal field of view can be calculated as below.

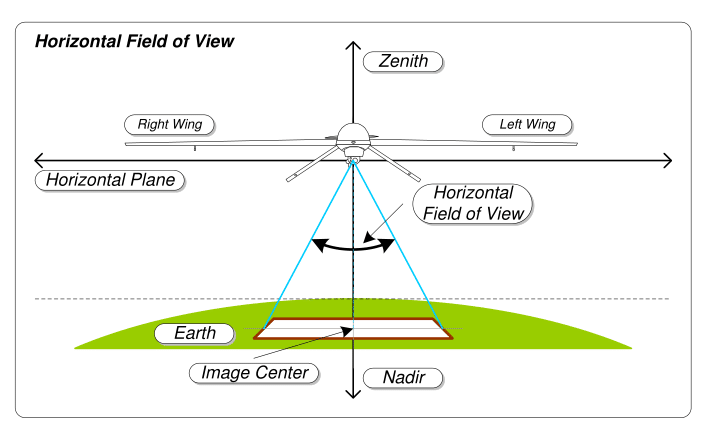


Figure 1.1

* 1. **Calculation of area**

The positioning format used in this project is in the form of latitude and longitude. To calculate the area of a polygon with given points is not so simple as in the Cartesian coordinate system. Hence the distance and srea formula cannot be used in this case due to the curvature of the earth. Because of this, the area or distance calculated from the Cartesian co-ordinates principle will be lesser than the actual value. Thus complicated calculations must be done in order to calculate them. However, for a very small distance, the curvature of the earth can be neglected as it can be considered approximately flat due to a very large radius of the earth. To decrease the hassle of calculation, as explained above, a predefined python module named 'geographiclib' is used.

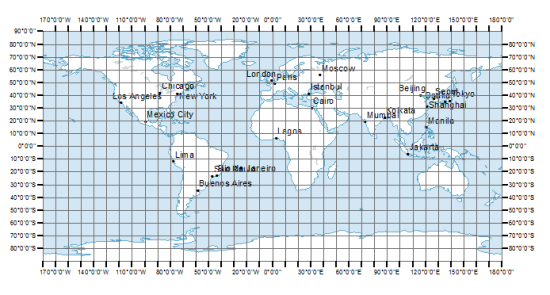
The area calculation if defined in ‘PolygonArea’ class. Which has following functions defined in it.

* [Clear()](https://geographiclib.sourceforge.io/html/python/code.html#geographiclib.polygonarea.PolygonArea.Clear) - reset the polygon
* [AddPoint()](https://geographiclib.sourceforge.io/html/python/code.html#geographiclib.polygonarea.PolygonArea.AddPoint) - add a vertex to the polygon
* [AddEdge()](https://geographiclib.sourceforge.io/html/python/code.html#geographiclib.polygonarea.PolygonArea.AddEdge) - add an edge to the polygon
* [Compute()](https://geographiclib.sourceforge.io/html/python/code.html#geographiclib.polygonarea.PolygonArea.Compute) - compute the properties of the polygon
* [TestPoint()](https://geographiclib.sourceforge.io/html/python/code.html#geographiclib.polygonarea.PolygonArea.TestPoint) - compute the properties of the polygon with a tentative additional vertex
* [TestEdge()](https://geographiclib.sourceforge.io/html/python/code.html#geographiclib.polygonarea.PolygonArea.TestEdge) - compute the properties of the polygon with a tentative additional edge

Compute() function calculates perimeter and area of the points defined by the polygon using AddEdge() or AddPoint() functions.

* 1. **Map Coordinate System**

Anything on Earth can be located in terms of latitude and longitude coordinates. The field of study that measures the shape and size of the Earth is geodesy. Many coordinate systems are proposed such as WGS84, NAD27, and NAD83. In each coordinate system, each position on the earth is given a unique value. A geographic coordinate system defines two-dimensional coordinates. The equator is the border where we measure north and south. Everything north of the equator has positive latitude values and everything south of the equator has negative latitude values and define a zero line at the equator. The Greenwich Meridian (or prime meridian) is a zero line of longitude from which we measure east and west



1. **Algorithms**

After completing the necessary calculation to determine the maximum area that an individual command center can handle, division algorithms are applied to get the total area divided into smaller chunks for each drone to navigate inside it to find any anomalies and report back to the command center. Several different algorithms can be used depending on the constraints. The constraints may be the time, resources available, speed *etc.* Three different algorithms are developed here in this project which can be useful and reliable for different cases. The names of the algorithms are as follows.

a. Area Division Algorithm

b. Line Division Algorithm and

c. Spiral Algorithm

* 1. **Area division Algorithm**

Here as the name depicts the total area is divided into smaller chunks of square-shaped area. In short, one can picture this algorithm as just drawing horizontal and vertical lines to the master grid (Given area), to get a smaller area for each drone to navigate.

The input to the code will be the latitude and longitude values of the corner polygon of the master grid in the KML format. There are different geo formats available. But, the reason for using this format is because of huge documentation, community help and also it is easier to understand the format.

Firstly, the input KML file must be parsed into workable integer format in the form of latitude and longitude values. To do this predefined python module named pykml is used. This will read the coordinates values and will be stored in the 'cords' variable in the code. In the KML, the coordinates will be stored in comma delimited format for latitude and longitude and 0.0, delimited format for two points. This value is copied and stored in another variable for the backup. This is done by using an inbuilt function called as deep copy. Using this the two variables are allocated with a different memory location and hence handled differently. The flow chart and the explanation of the working principle of the algorithm are as bellow.

a. The master grid is considered to be in the square shape for simplicity

b. The bottom left corner point is considered as an origin for the subsequent calculation. This won't affect any calculation. This is just for the reference to have a better pictorial understanding of the algorithm.

c. Another point 90 for the origin is also considered initially. This will help in easily propagate in further iterations.

d. As the maximum area is available, maximum side length is calculated by taking the square root of area

e. Next, shift the two points to a distance of length eastwards and note the final two positions. The previous two and the new two coordinates form the first square. Continue this till the east corner is reached

f. Move the bottom variable to the top and calculate new top variable

g. Continue this until the north corner is reached. This will divide the major grid into smaller sections.

The number of smaller grids to be made depends on the total number of drones that are present. The major disadvantage of this algorithm is the area loss. This algorithm divides the complete area into a number of smaller area if and only if the total number of smaller grids are a perfect square. According to the given problem statement, if there is an anomaly in that region, then it would not be possible for the drone to find it as none of the drones are allocated to that area. Another major disadvantage of this algorithm is that the divided area works better only if the major grid is in the square shape and it gives only square grids. This puts a lot of other constraints. The comparative results of each algorithm will be explained in the later chapters.

* 1. **Line Division Algorithm**
  2. **Spiral Algorithm**

# Bibliography

Howard, Ian P., and Brian J Rogers. 2014. *Binocular vision and stereopsis.* New York: Oxford University Press. p. 32.