Report on

Final Project

Submitted to

Rodolfo Coutinho

COEN 320

Section W

Submitted on April 8th, 2024

By

Patrick Riachi - 40104015 - riachipatrick1@gmail.com
Tatiana Blogu - 40172342 - t.blogu@gmail.com
Andre Hei Wang Law - 40175600 - heiwangandrelaw128@gmail.com

“We certify that this submission is the original work of members of the group and meets the Faculty's Expectations of Originality”

Objective

The main objective of this project is to design and implement a real-time system for the Air Traffic Monitoring and Control (ATC) system. The system will be able to monitor and help manage a specific area of airspace to ensure safe navigation and movement of the aircrafts.

Introduction

Air Traffic Monitoring and Control (ATC) plays an important role in managing the busy skies as the demand for air travel continues to increase. The primary aim of any ATC system is to ensure a smooth flow of air traffic and prevent collisions. These tasks involve detailed collaboration among various specialists and technological systems. The main operational zones of an ATC system are: the tower control area, the Terminal Radar Approach Control (TRACON) area, and the en-route area. Each of these zones is in charge of a designated sector of airspace which are called sectors.

Our project focuses on the en-route area, which begins once an aircraft is handed off from the TRACON to en-route air traffic control. This area requires high attention since it involves guiding the aircraft at high altitude until they approach their destination and are handed off again to the next sector's TRACON.

As we are developing a simplified ATC system, we will focus on a 3D rectangular airspace that is 15000 feet above the sea level, measuring 100000 by 100000 feet in the horizontal plane and 25000 feet in height. The system's task will be to alert the controller if there are two aircrafts within 1000 units in height and 3000 units in width/length. In that case, the speed or position of aircrafts will have to be adjusted as necessary.

Analysis

After collectively brainstorming and discussing with the TA, we have decided to make the project modular due to the involvement of multiple components. It is important for each component to work both independently and in conjunction with others. This modular approach will ensure easy and effective maintenance, testing, and potential future enhancements of the system. In addition, the main requirement is the continuous monitoring of airspace in order to detect potential collisions between aircraft and to ensure the minimum distance criteria between planes which are 1000 units in height and 3000 units in width/length. Therefore, multiple periodic tasks will be involved to simulate the key functions of our simplified ATC system.

Starting with the input file, which will contain aircraft data such as ID, position and velocity, the Radar Process will continuously send aircraft position updates. This will be set as a periodic task which will run every second to detect planes within 10000 x 10000 feet area. This process should work precisely since it serves as the initial data entry point for the Computer System Process.

The Computer System Process will handle an analytical aspect of the project. It will receive the aircraft position and speed updates from the process above. By using the Euclidean distance equation, it will calculate the distance between each aircraft. Following these calculations, the collision detection will be performed. An algorithm will be designed in order to predict potential collisions within the next three minutes, which will take into account the velocities in each spatial dimension (x, y and z).

Moreover, the Operator Console Process will serve like a user interface of our project, which will display alerts received from the Computer System. It allows the operator to take immediate action to resolve any potential conflict by adjusting the direction or/and speed of the involved aircrafts and to request more information about the planes.

The Data Display Console will display the radar's output by performing another periodic task. It will display aircraft positions on x and y planes every five seconds. This real time representation is crucial for the operator to maintain a visual understanding of potential issues.

Finally, the Communication System serves as the link between the operator's commands and the aircraft, sending the necessary instructions to adjust the aircraft's direction or velocity in response to the given circumstances.

Design

The main objective of this project is to have a radar system that detects the aircrafts that are passing within the en-route boundaries. Once detected, the information of the planes will be fetched by the radar, which will then be forwarded to the computer system to be printed and to be analyzed for any future collisions.

When the computer system receives the information of the planes that are within the detection range, using the current positions of each dimension with the different velocities, we were able to calculate the future positions of each aircraft in the next 3 minutes (180s). Comparing the calculated values of all the aircraft pairs, we can predict the collision positions of two planes. Those collisions are based on the calculation of the Distance between the aircraft using the Euclidean Equation for the XY-plane and the subtraction of the Z-indexes.

If the result of the Z distance is greater than 1000 units, the XY coordinates can be ignored since two planes with enough distance on the Z-axis can cross on top of each other with no collision. Otherwise, if the planes are too close to each other on the Z-axis, we will check the Euclidean distance between those aircrafts. If the XY-distance is over 3000 units at any time in the next 3 minutes, the planes will never collide. However, if the planes are too close to each other on the XY-axis, a VIOLATION message will be displayed by the operator console to fix the problem by changing one of the plane's position or velocity.

Through the communication system, the operator console will be able to send commands to specific aircraft.

Last but not least, we will have the display thread that runs every 5 seconds and outputs the radar visualization of the planes moving in the different directions.

Implementation

In the implementation process, we started by designing the aircraft files, which will be different threads that represent the planes. Using an input file, we are able to add/start different aircrafts with different positions and different velocities. Each thread holds the specific aircraft's information such as the ID, posX, posY, posZ, velX, velY, velZ. Using the velocity and the current position, we are able to make the aircraft thread update the positions, by adding the velocity to each position every second.

We have also developed a timer which allows us to make the threads periodic and allows us to manage our aircraft using a time constant.

Once we have the different aircraft threads running, using the inter-process message passing implemented within QNX, we are able to develop a periodic Radar task that runs every second, and receive the aircraft's updated information as a message. Once received, we will be able to take this data and check if the aircraft is within the en-route boundaries. If they are, the detected aircraft information will be sent/forwarded to the Computer System thread using the QNX inter-process message passing as well.

In the computer system, we will create a periodic task which will run every second, and this task will always forward the received information to the display thread which will print the different positions of the aircraft. This allows us to have a constant check on the updated positions of the aircraft.
In the computer system, the key functionality is the collision prediction. Using the current positions that are received by the radar, the computer system will compare every single pair of detected aircraft and check if there will be any collision in the next 180s by comparing each position of each second. Using the following equation: $|posZ(1) - posZ(2)|$ we will check if the result is greater than 1000:

- If **true**, this means we can skip the rest and move to the next pair, since these two aircraft can pass on top of each other (no collision) even when they have exactly the same XY coordinates.
- If **false**, this means the planes are too close to each other altitude wise:
    - At this point, using $\sqrt{|posX(1) - posX(2)|^2 + |posY(1) - posY(2)|^2}$, we are able to check the distance between the aircraft on the XY-plane. We will check if $\sqrt{|posX(1) - posX(2)|^2 + |posY(1) - posY(2)|^2} > 3000$:
        - If the result is greater than 3000, we will ignore and move to the next pair. This is because the planes are far enough from each other at any time within the next 180s.

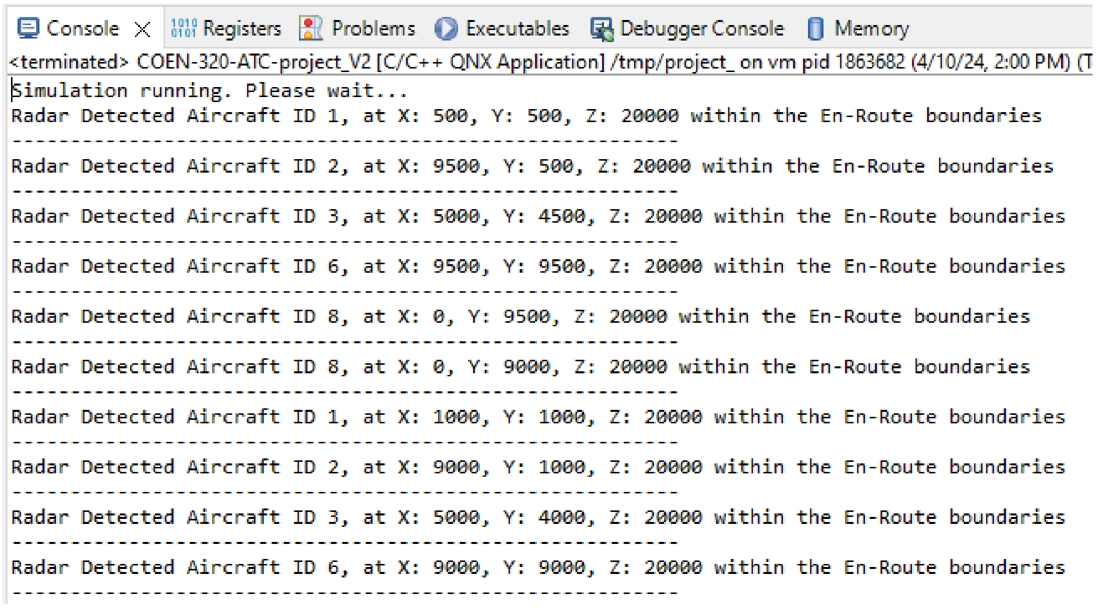
- If the result is less than 3000, this means the aircraft are in distance violation. The computer system will send the aircraft IDs of the pair that was just compared, with the violation message.

Now that we have the main logic of the implementation figured out, we will implement the display thread, which is a periodic thread running every 5 seconds and outputs the detected aircrafts on a 50x50 grid of the XY-plane. Each grid represents 2000 feet which in total will represent 100,000x100,000 feet of the XY-plane. If two aircraft collide, it will show the ID on the radar plane view with the altitudes and everything.

Moving forward, we have developed the Operator Console system which is utilized for the Violation Alerts function. These alerts are printed on the console with the predictions as well.

For the outputs, we used a mutex functionality in order to make the threads not output together at the same time.

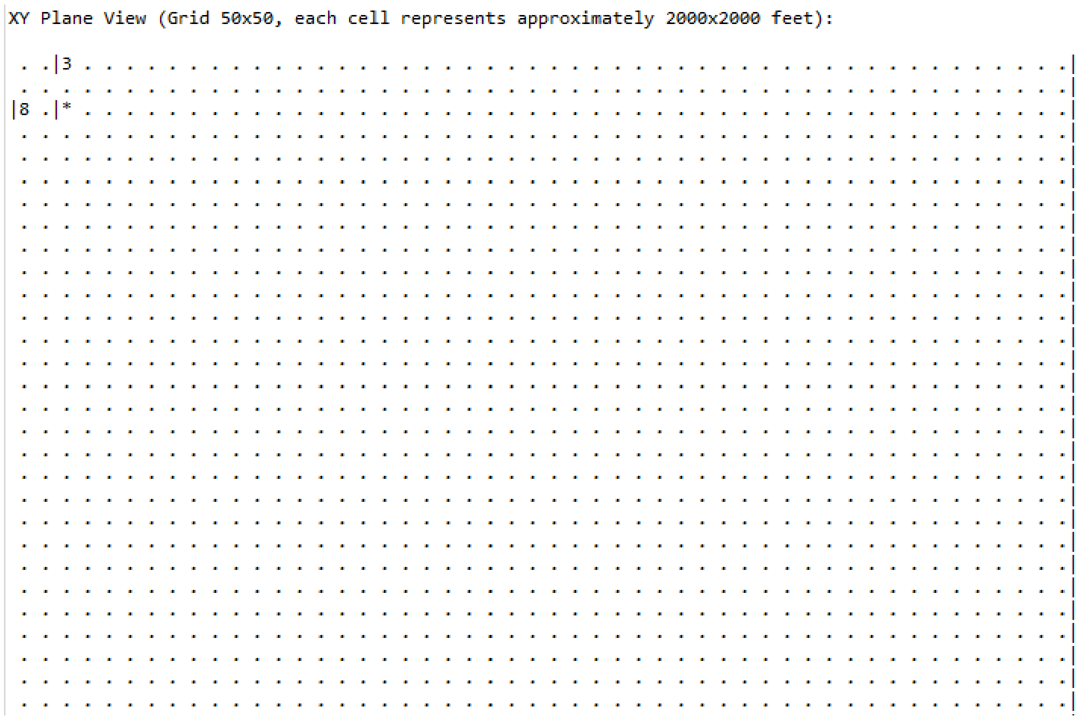
The following figure represents the radar system's log as it detects and updates the positions of aircraft within the boundaries.



```
Console  Registers  Problems  Executables  Debugger Console  Memory
<terminated> COEN-320-ATC-project_V2 [C/C++ QNX Application] /tmp/project_ on vm pid 1863682 (4/10/24, 2:00 PM) (T
Simulation running. Please wait...
Radar Detected Aircraft ID 1, at X: 500, Y: 500, Z: 20000 within the En-Route boundaries
---------------------------------------------------------
Radar Detected Aircraft ID 2, at X: 9500, Y: 500, Z: 20000 within the En-Route boundaries
---------------------------------------------------------
Radar Detected Aircraft ID 3, at X: 5000, Y: 4500, Z: 20000 within the En-Route boundaries
---------------------------------------------------------
Radar Detected Aircraft ID 6, at X: 9500, Y: 9500, Z: 20000 within the En-Route boundaries
---------------------------------------------------------
Radar Detected Aircraft ID 8, at X: 0, Y: 9500, Z: 20000 within the En-Route boundaries
---------------------------------------------------------
Radar Detected Aircraft ID 8, at X: 0, Y: 9000, Z: 20000 within the En-Route boundaries
---------------------------------------------------------
Radar Detected Aircraft ID 1, at X: 1000, Y: 1000, Z: 20000 within the En-Route boundaries
---------------------------------------------------------
Radar Detected Aircraft ID 2, at X: 9000, Y: 1000, Z: 20000 within the En-Route boundaries
---------------------------------------------------------
Radar Detected Aircraft ID 3, at X: 5000, Y: 4000, Z: 20000 within the En-Route boundaries
---------------------------------------------------------
Radar Detected Aircraft ID 6, at X: 9000, Y: 9000, Z: 20000 within the En-Route boundaries
---------------------------------------------------------
```

Figure 1: Radar detection log

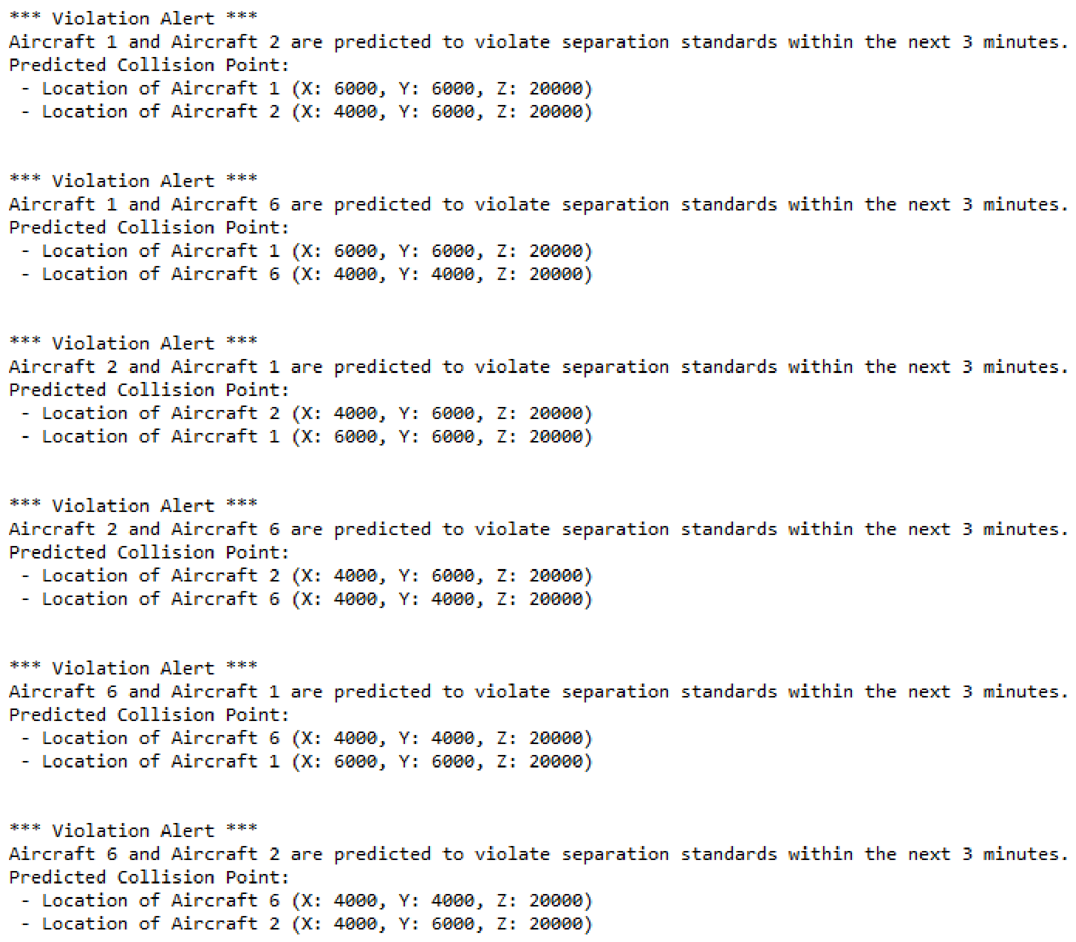
Figure 2 represents the graphical display of the Data Display Console, showing aircraft positions on a 50x50 grid where each cell represents an area of 2000x2000 feet.



```
XY Plane View (Grid 50x50, each cell represents approximately 2000x2000 feet):

 . .|3 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
|8 .|* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .|
```

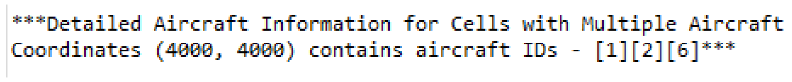
Figure 2: Aircraft Position Grid on XY Plane

Figure 3 displays the collision prediction alerts. The system forecasts possible violations of separation standards between aircrafts within the next three minutes.



```
*** Violation Alert ***
Aircraft 1 and Aircraft 2 are predicted to violate separation standards within the next 3 minutes.
Predicted Collision Point:
 - Location of Aircraft 1 (X: 6000, Y: 6000, Z: 20000)
 - Location of Aircraft 2 (X: 4000, Y: 6000, Z: 20000)


*** Violation Alert ***
Aircraft 1 and Aircraft 6 are predicted to violate separation standards within the next 3 minutes.
Predicted Collision Point:
 - Location of Aircraft 1 (X: 6000, Y: 6000, Z: 20000)
 - Location of Aircraft 6 (X: 4000, Y: 4000, Z: 20000)


*** Violation Alert ***
Aircraft 2 and Aircraft 1 are predicted to violate separation standards within the next 3 minutes.
Predicted Collision Point:
 - Location of Aircraft 2 (X: 4000, Y: 6000, Z: 20000)
 - Location of Aircraft 1 (X: 6000, Y: 6000, Z: 20000)


*** Violation Alert ***
Aircraft 2 and Aircraft 6 are predicted to violate separation standards within the next 3 minutes.
Predicted Collision Point:
 - Location of Aircraft 2 (X: 4000, Y: 6000, Z: 20000)
 - Location of Aircraft 6 (X: 4000, Y: 4000, Z: 20000)


*** Violation Alert ***
Aircraft 6 and Aircraft 1 are predicted to violate separation standards within the next 3 minutes.
Predicted Collision Point:
 - Location of Aircraft 6 (X: 4000, Y: 4000, Z: 20000)
 - Location of Aircraft 1 (X: 6000, Y: 6000, Z: 20000)


*** Violation Alert ***
Aircraft 6 and Aircraft 2 are predicted to violate separation standards within the next 3 minutes.
Predicted Collision Point:
 - Location of Aircraft 6 (X: 4000, Y: 4000, Z: 20000)
 - Location of Aircraft 2 (X: 4000, Y: 6000, Z: 20000)
```

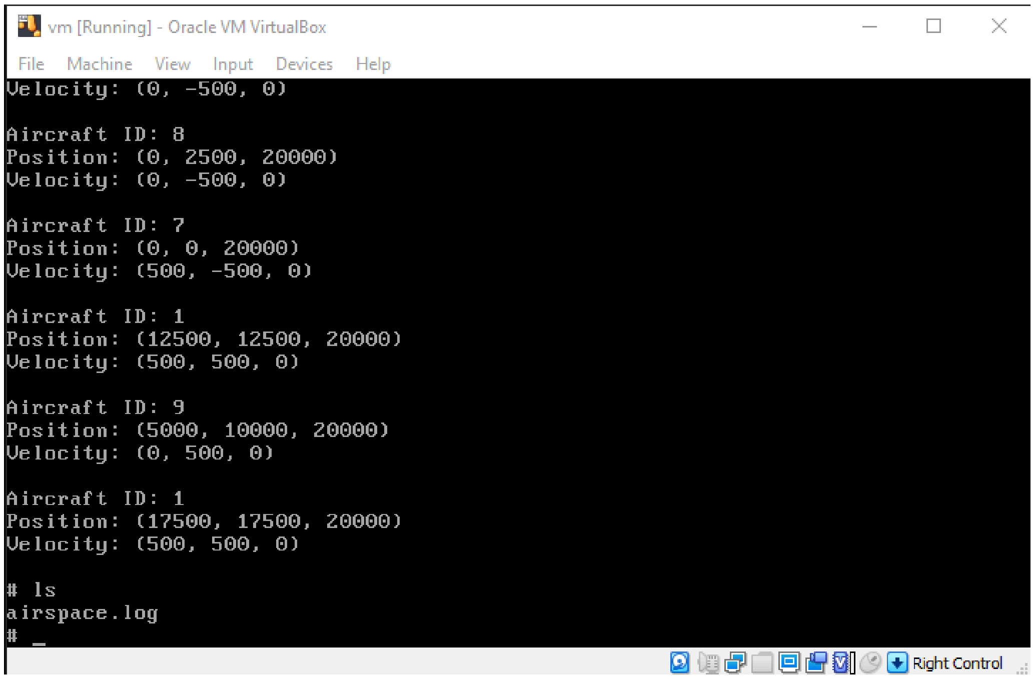
Figure 3: Collision prediction alerts



```
***Detailed Aircraft Information for Cells with Multiple Aircraft
Coordinates (4000, 4000) contains aircraft IDs - [1][2][6]***
```

Figure 4: Aircraft proximity information

The last figure shows the operator (user) querying individual aircraft information (position and velocity data) and it is recorded in the system's airspace log within the VM.


vm [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Right Control

```
Velocity: (0, -500, 0)

Aircraft ID: 8
Position: (0, 2500, 20000)
Velocity: (0, -500, 0)

Aircraft ID: 7
Position: (0, 0, 20000)
Velocity: (500, -500, 0)

Aircraft ID: 1
Position: (12500, 12500, 20000)
Velocity: (500, 500, 0)

Aircraft ID: 9
Position: (5000, 10000, 20000)
Velocity: (0, 500, 0)

Aircraft ID: 1
Position: (17500, 17500, 20000)
Velocity: (500, 500, 0)

# ls
airspace.log
# _
```

Figure 5: Airspace log and aircraft information retrieval

Lessons Learned

Working on this project, we learned the importance of the use of the modular design since it allows for more manageable maintenance of the system and the code. Also, it is helpful for the future enhancement of the project. Creating a modular framework allowed us to implement each component individually and then to ensure that they work seamlessly together without any problem.

Moreover, our team gained some interesting and valuable experience while working with QNX and C++. It was challenging to figure out how it works, but with the TA's help, we were able to understand the operating system and proceed with the project. We also encountered challenges with inter-process communication and synchronization in a real-time system. However, after extensive research, we were able to successfully meet all the project requirements.

If we had more time, we would place more emphasis on earlier testing, and we would explore additional algorithms that could be helpful in completing and even enhancing this project. This experience has significantly helped us to understand how real-time systems work and their applications.

Conclusion

In conclusion, we successfully met the project's primary goal, which was designing and developing a functional real-time ATC system. We were able to simulate the monitoring of airspace and manage control tasks for the en-route sector. The use of a modular design provided an effective development process which helps with the maintenance and potential future enhancements. Working with the QNX operating system provided us hands-on experience and we were able to use and understand better the real-time operating concepts learned in class. Despite having challenges with inter-process communication and system synchronization, we resolved these issues effectively. For future work, we plan to enhance our collision prediction algorithm in order to improve the system reliability. Also, we would like to explore more advanced features for airspace management such as machine learning based prediction models for even better system autonomy.

Contribution

<u>Patrick:</u>

-Worked on the Radar System, the Computer System, the Data Display and the Operator Console

<u>Tatiana:</u>

-Worked on the Radar System, Aircraft, the Input Strings, the Timer and the Final Report

<u>Andre:</u>

-Worked on the Communication System, the Computer System, storing the log files and on Main