International Conference on Military Technologies (ICMT) 2019
May 30 – 31, 2019, Brno, Czech Republic

Use of Virtual and Augmented Reality in Design of Software for Airspace

Tomas Malich*, Lenka Hanakova*, Vladimir Socha*, Sarah Van den Bergh*,
Michaela Serlova*, Lubos Socha†, Slobodan Stojic* and Jakub Kraus*
*Czech Technical University in Prague, Department of Air Transport, Prague, Czech Republic,
e-mail: *malictom@fd.cvut.cz, hanakle1@fd.cvut.cz, sochavla@fd.cvut.cz*
vandesar@fd.cvut.cz, serlomic@fd.cvut.cz, stojislo@fd.cvut.cz, krausjak@fd.cvut.cz
†Technical University of Kosice,Department of Air Transport Management, Kosice, Slovak Republic,
e-mail: *lubos.socha@tuke.sk*

***Abstract*— The aim of the article is to present a software design for the design and adaptation of flight routes and spaces, or respectively, of the Air Traffic Control sectors. Presented system uses the means of virtual and augmented reality. The functional and non-functional requirements that led to the software design of the introduced system were defined. Functionality has been verified by a partial implementation in C# language. With regard to real visualization in 3D, the future use of the system in air traffic control training is intended. There are also several visions of the future that could fairly change the air traffic control rules. Therefore, it is also intended to make the system flexible enough to be expanded for further applications.**

***Keywords*— air traffic control; airspace management; aviation; augmented reality; flight route; training; virtual reality**

I. Introduction

At the beginning of civil aviation, only visual aids were used for navigation. These were mainly landmarks, visible from the air or light beacons for night navigation. Navigation equipment of the aircraft was an aeronautical chart, a compass, a clock, and a pilots' sight. The possibility of flying was greatly affected by the weather and in adverse conditions, the safety of the flights was greatly compromised. Subsequently, the aviation industry focused on a technization, a great achievement and milestone were set with the invention of radar.

Together with the improvement of the navigation systems, the area of monitoring and communication was also developed. Space with radar coverage had been increasing rapidly, especially around the airports [1]. This has resulted in dividing the airspace both vertically and horizontally. The space was also categorized according to the intended traffic. Along with the categorization of spaces, a process of the specialization and division of the Air Traffic Control (ATC) services [2].

Given the continuing increase in air traffic density, it is still necessary to innovate the Air Traffic Management (ATM) technologies and processes. Nowadays, large civil aircraft arrive to its final destinations by the flight routes. These are determined by waypoints. During a controlled flight aircraft may occur in several different sectors, each of which is controlled by different ATC service providers.

Current ATC is a centralized-built system and uses flight routes which are firmly set in the airspace. The system is centralized because the flight trajectory decision depends on the Air Traffic Controller Officer (ATCo). ATCo is an active part of the system and the aircraft is a passive component of the system. Air transport under the radar coverage is arranged into flight routes. Despite the fact that nowadays aviation has the access to more or less advance technology that enables the flight to be flown dynamically according to the true trajectory between two airports, the usage of strictly determined flight routes remains. The main reason for still using the flight routes is that this way of organizing a large airspace is much more easily controlled by the ATCos instead of controlling more or less seemingly chaotic airspace if the aircraft flew straight from the departure to the final destination.

The location and division of these sectors along with the location of waypoints are a very important part of the efficiency of civil aviation. Followig this, there is a demand for software in which existing flight routes, waypoints and sectors can be modified and created. One of the main requirements for such software is that it is easy to use. First steps should be to select a destination on Earth where adjustments would be made and further to create changes using an intuitive graphic interface. The application should also allow the import of aeronautical charts so that it won't be necessary to enter routes, waypoints and airspace manually for each destination. Additional functions should be airspace and air traffic control sectors editing.

There are also several visions of the future that could fairly change the air traffic control rules. To do this, the software should be ready, and the developer should always keep in mind that changes to the program should be done simply and without major interference.

The aim of the article is to present a software design for the creation and adaptation of flight routes and spaces, or respectively, of the ATC sectors. One of the basic requirements is the intuitiveness of the system which allows the user to redefine (or optimize) the appearance of airspace with as little effort as possible. It is also intended to make the system multiplatform and flexible enough to be expanded for further applications.

978-1-7281-4593-8/19/$31.00

II. METHODS

A. Functional Requirements

Functional requirements of the system were formed as follows. The system allows a creation and adjustment of airspace, sector, waypoints and the system of airport runways and taxiways. In addition, the system allows real-time flight simulation, creation and adjustment of the air traffic scenario, the use of a virtual Air Traffic Control Officer (ATCo) to guide aircraft according to a predefined scenario, and user input into real-time simulation. It is then possible to store and load already created airspace, sectors, waypoints and air traffic scenarios. The system enables to import data from the aeronautical charts. These aeronautical charts are supported in *.xml format. Multiple users can cooperate within one simulation. The simulations can be displayed on a VR/AG device. It is possible to switch between simulation modes. The simulated situation can be displayed in a 2D or 3D view.

Based on the analysis of individual functional requirements, specified requirements are defined. The analysis further detects other requirements arising from the conditions for designing functional requirements [3].

Airspace is defined as a 3D object in space. Therefore, it has a unambiguously defined shape. The shape of the airspace is described by 3D coordinates. To determine the coordinates it was necessary to define the space with the Cartesian coordinate system and by the axes x, y and z. The airspace had to be placed according to its geographical coordinates. That means that a system contaning auronautical chart, where it is possible to place created airspace, was constructed.

The creation of space was done as follows. At first, a user picked the waypoints in the chart amid which connections were established. These connections among waypoints form 2dimensional shape of a space (when viewed from above). The shape of the connections can be adjusted so that they don't have a form of a line segment but a form of a curve. After that, by the selection of the first point, the shape creation of the airspace is finished. The user selects the airspace category. Subsequently, the upper and lower boundary lines of the airspace (the height above ground level) are set by the user. Finally, the user confirms the creation of the sector, thereby saving it. Once the space is created, it is possible to edit or delete it.

A sector is defined as a 3D spatial object. As with the airspace, the shape is described by the Cartesian coordinate system. The sector coordinates were created using the geographic coordinates. The lower and upper boundary lines are defined according to the heights above the ground level. The creation of the sector follows the same process as the creation of airspace up to a few minor differences. No category is defined in the sector space, but a responsible ATC service provider. It is possible to group sectors into larger sectors which are managed only by one ATCo. As swith the space, sectors can be too edited.

The waypoint is determined by a latitude and a longitude and it is valid for any height above ground. It is possible to select and further edit or delete the created waypoint in the editation panel. When creating and editing, it is possible to enter a name that consists of five characters of the Latin alphabet.

The airport runway is determined by two geographic coordinates and runway designation. The system does not require additional data, such as setting precision or non-precision approach procedures [4]. The airplane is able to land on the runway only if it approaches in the runway direction, or possibly with a minor deviation. The landing speed of the aircraft is resolved automagically. Appart from that, system can also change the landing speed so that the aircraft was able to land. However, the speed adjustment depends on whether it is possible with respect to the aircraft parameters and the necessary trajectory. If there is a conflict and the system does not find a solution to allow the aircraft to land on the runway, the user is warned and a Missed Approach Procedure (MAP) is commenced. The process of MAP is simplified within the simulation and follows the same process on all runways.

The software includes real-time simulation mode. After switching the real-time simulation mode on objects, that are defined as moving, start to move. Aircraft in the system change position according to defined scenarios, or possibly along the trajectory determined by the course and speed of the aircraft. In this mode, it is possible to control the aircraft by changing the course, flight speed or height. Furthermore, it is possible to navigate the aircraft to the waypoint, land at the selected airport, start the holding process according to the selected holding pattern and hand over the aircraft to another ATCo.

In the simulation mode on active aerodrome runways a new aircraft can be added to the system, which then takes off from the selected runway. In simulation mode, it is not possible to modify airspace, sectors, waypoints and airport runways.

The software allows to create a scenario that follows simulated air traffic. Running the simulation point defines the time T0. Scheduled events in scenarios are triggered either by reaching the time T0 + event start time or sequence continuity of the events. Each entity in the system contains a scenario manager and it is possible to define events with respect to the entity's capabilities. During one active scenario, it is possible to create and activate additional scenarios. Each scenario has defined links to the entities in the system that will be used during its run.

The software allows users' interventions into the simulation mode. These includes interventions in traffic scenarios, commands to the active aircraft and airports. In this mode, it is not possible to change the airspace, the sectors, the location of the airport runways, etc. These inputs are entered using the editing menus for each entity.

The software allows to preserve created spaces, sectors, waypoints, and scenarios by pointing to external files or databases. Spaces, sectors, and waypoints are saved into projects. Projects are data sets that, after loading into the system, create spaces, sectors, and waypoints as they were stored by the user. The project is defined by a unique

designation, and the information about the location and location of the site for editing and simulation are also a part of the saved project data. This location appears after the project is loaded, but it is possible to change the location later.

The software allows loading and subsequently displaying of the airspace and waypoints by importing the data from the aeronautical charts. The chart format is an AIXM *.xml format. These data are transformed so that they can be used in the system to create flight spaces and waypoints. The imported data are rewritten into the project structure so that they could be used in the system to create flight spaces and waypoints, and then to store them, modify, and load along with other project data.

The software allows to turn on a server mode. In this mode, it is possible to connect to an instance running on a single user machine from another machine (this instance is called a client), which sees the status of the simulation and editing on the server after connecting. Normally, the client is in observer mode and is not be able to interfere with the simulation. Subsequently, the server could allow to choose a different role for the client. Possible roles are: the sector manager, editor, pilot and observer. The ATCo is assigned a thus is able to control the aircraft that fall into its sector. ATCo is also allowed to pass on the aircraft to another ATCo when the aircraft is near the leaving point of the sector. Editor is able to create, edit, and delete airspace, sectors, waypoints, air traffic scenarios, and aerodrome runways in the editing mode. In simulation mode, the controller is allowed to control all entities in the simulation as well as server-side user. Pilot is able to control the assigned aircraft according to instructions that are not passed through the system but verbally (or by other means such as radio, telephone, etc.). The observer is only allowed to monitor the simulation.

The software allows the activation in VR, AG and VR + AG mode. In this mode, it is possible to connect as client to a running instance in server mode and is assigned the observer role. In VR mode, the user monitors the simulation in a completely virtual environment using a virtual reality device. In AG mode, the user is able to monitor the simulation provided by the augmented reality device. This is a mobile device. When the AG application is switched on, the user first detects a flat surface suitable for the simulated world (such as a blank table) and then simulates the selected area. In the VR + AG mode, the two above-mentioned modes are combined. The procedure is the same as for AG, except that the user uses a virtual reality device simultaneously to monitor the virtual world.

The system allows switching between simulation and editing mode. When switching from simulation mode to editing mode, it stops “time” and it is possible to create, edit and delete airspace, sectors, track points, airport runways, and default aero scenarios. When switching from edit mode to simulation mode, time will be triggered, and air traffic scenarios, aircraft commands and aircraft transfer between controllers will be possible.

The system grants viewing in a 2D perspective, i.e. from above, when the aeronautical chart is visible, and all active entities are rendered using 2D patterns. The system enables to switch to a 3D view where the chart is rendered as a 3D object by a chart depicting heights and the active entities are located in the space even according to their height above the ground.

The system allows for each sector to activate virtual ATCo. This makes it possible for aircraft in the sector to be automatically guided by the system without users’ interference according to their scenarios. In the future, it is intended to use the artificial intelligence (probably neural networks) to teach virtual ATCos using data from simulations that will be controlled by the user.

B. Non-Functional Requirements

The non-functional requirements are compiled using the practices described in the literature, [5], [6], [7]. Motivation to create dysfunctional requirements is to develop a quality and set-up system and setting the criteria for measuring quality. [8]. Presented software defines the following requirements: Performance, Sustainability, Reliability, Availability, Extensibility, Safety.

The performance requirement of the system is specified in such a way that it is necessary for the system to respond to user input with such delay so that the user does not get the impression that the application does not execute its commands immediately. In development, the system is expected to work on a mid-range machine available at the time of development. In order to determine the performance of the application, it will be necessary to test real users and then evaluate their feedback.

Sustainability is dependent on the ability to correct system failures so that the repair does not affect another part of the system. Sustainability contributes to the appropriate division of functional units into components that are minimally (ideally not at all) dependent on each other. Throughout the design, sustainability had to be taken as an important parameter for system architecture decisions.

The system must guarantee the integrity of the created data. Data created by the system are stored either in files or in a database. In both cases, it is necessary to maintain the data duplicitously until the moment when even if they are not duplicate, they won’t be lost in an unexpected event.

Accessibility has a lower priority in the proposed system. It is not a complex system that would require continuous traffic, but an application that will only run temporary. Therefore, system components will not be replicated to ensure consistent availability. In the event of an unexpected event, such as hardware, software, network failure, etc., the application becomes inaccessible.

Extensibility is one of the key non-functional requirements. Adding new functionality to the system or modifying existing functionality will be considered as a normal intervention. To achieve good extensibility, the smallest interconnection of system components, frequent use of interfaces, a high level of encapsulation of individual components, and a high-quality object model will be required. The system should have separate sections for the

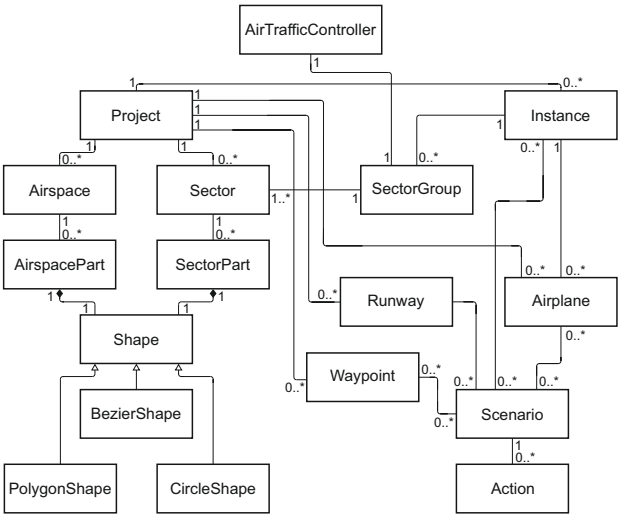

Fig. 1. Classes diagram

computational part of the simulation and imaging parts to replace the current 3D engine with the lowest number of system interferences.

The system is not designed to operate in a network that is freely available from the Internet. Nevertheless, security standards will need to be maintained to avoid data leakage during network communications, or to fake false data. The system will only use secure SSL for all network communications. Files and databases will not be encrypted. The system will require a machine that is secured by a user password to keep the stored data safe. In other words, the security of the stored data will be delegated to the operating system and user.

C. Classes Design

Based on the functional requirements, the UML diagram of the classes shown in Fig. 1 was created. These classes form a system data model. Because of the clarity of the figure, the attributes and functions of the classes are omitted in the diagram, and are described in detail below.

The Node class is used to preserve attributes that determine the position in the system. If the height is not defined, then it is ignored and it is assumed that the node is valid at all heights above the ground. Project encapsulates entity information and deployment at time T0. The instance encapsulates the entity deployment information at a time other than T0. Airspace class keeps information on airspace and contains links to its parts. The AirspacePart class contains information about one part of the airspace.

Shape is an abstract class from which the PolygonShape, BezierShape and CircleShape classes are separated. With respect to the PolygonShape class, the system will work only with polygons whose line segments do not cross each other. BezierShape is used to preserve the shape information of the object, which is described by Bézier's curve. The system supports linear, quadratic and cubic Bézier curves. The BezierCurveType class enumerator serves to determine the type of curve. Each type of curve requires a different point sequence in the nodeList attribute. These points serve to plot the curve in 2D or 3D space. CircleShape is used to store information about the shape of an object that is described by a circle.

The Sector class keeps information on the ACC sector and contains links to the individual sections by which the sector is constituted. Subsequently, it refers to a group of sectors to which it belongs and which is managed by one ATCo. SectorPart contains information about one part of the sectors by which the part is comprised. The AtcType class serves as an enumerator for determining the type of ATC service. The AirTrafficController contains information on ATCo. The ControllerType class serves as an enumerator for determining the type of the ATCo. Waypoint contains waypoint information. Runway contains information about the airport runway and the Airplane contains aircraft information in the system.

D. Component System Architecture

Based on the non-functional and functional requirements, a component diagram is created and the system architecture is designed. Fig. 2 depicts a design of a module and system components. The system is split into server and client partition. Communication with noncritical response requirements takes place through the REST interface. Communication with critical response requirements, e.g. change of the aircraft position) functions via the TCP protocol. Messages between the server and clients are in the JSON format. The database for data storage is MongoDB, which is an object database.

The Arda Server application was created on the Spring frame which is a JavaEE extension. Therefore, the Java virtual machine is required to run it. The reason for choosing Java was mostly because of it's easy to achieve multiplatformity. The Arda server provides the clients with an interface to obtain the state of the currently running instance, list of saved projects and list of saved instances for the selected project. Among mentioned attributes, the Arda server provides also data of the selected projects and instances.

Aircraft calculations are made on the server side. Inputs that can affect the flight trajectory will only come from flight scenarios. For example, if a client user sends a command to change the course of a selected aircraft to the server, this command is added to the aircraft scenario, and the server starts to change the aircraft's course according to the rules that are defined on the server side.

E. Maps SDK for Unity

Despite the fact that within the system the globe will be projected on the 2D surface, it is necessary to perform calculations for the movement of the aircraft on a 3D model of the globe. Available data on the current position of the aircraft in geographic coordinates (latitude and longitude), aircraft height above ground, aircraft speed and flight direction are used as inputs. The system will be required to find the best route possible for the aircraft in position A, which is defined by latitude and longitude,

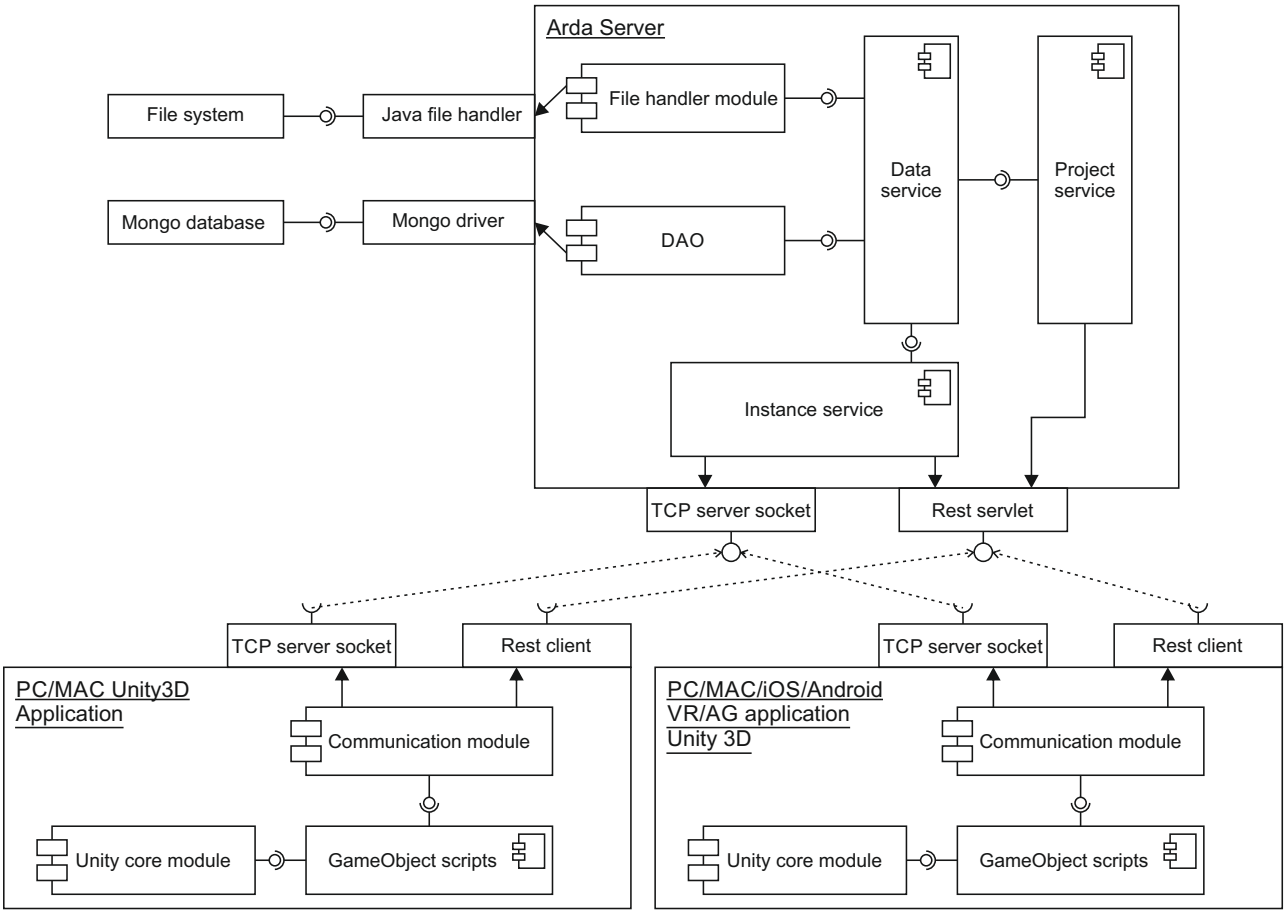

Fig. 2. Architecture of components and modules

to reach the position B in the shortest possible time. Maps SDK for Unity from Mapbox will be used for map representation. This framework allows many useful features. For instance, it is able to create an object on a generated map, based on a defined latitude and longitude. It is important to bear in mind that the generated Mapbox Map in Unity is embedded in a 3D scene and its centre is placed in the coordinates x, y and z.

F. Aircraft Navigation in the System

The Earth is not a perfect sphere, so this fact should be taken into account during calculations. The proposed software will use the WGS84 (World Geodetic System 1984) model. This model is a world-renowned standard for defining a coordinate system and a reference ellipsoid for navigation. The coordinates on the WGS84 model are based on the geographical coordinates. The location on the model is determined by latitude and longitude.

In order to move the aircraft in the WGS84 model, it was necessary to create an algorithm that would, within the short intervals, change the position of the aircraft in the geographical coordinates in relation to the course, velocity, altitude and current position of the aircraft. The result of this algorithm will always be an latitude and longitude. These are then used to position the aircraft on the map by using the function defined in the Mapbox SDK API named GeoToWorldPosition, which takes Vector2 as a parameter (a class defined in Unity3D) to store two parameters x and y into one instance of a class. It will also be necessary to properly rotate the aircraft in the scene coordinates so that the aircraft model is heading to the course

G. Aircraft Movement

Within the system, it was necessary to solve the change in position relative to the flight distance in time that passed between the last iteration and the new iteration of the algorithm. One of the parameters of the algorithm was therefore the time difference between two iterations in milliseconds. It was also necessary to know the velocity of the aircraft moving in the coordinate system. Since the aircraft does not move along a line but on a curve, and because the resultant angles (latitude and longitude) are required as a result of the algorithm, it was necessary to calculate the angle change in the given time interval and the ratio in which the angle change is to be applied between latitude and longitude. To calculate the angle change, it is necessary to know the radius of the circle that has the centre at the same point as the centre of the WGS84 system (World Geodetic System 1984) and the aircraft altitude. Thus, in each iteration, the value of r will be calculated using the equation:

$$r = r_e + h_a \tag{1}$$

where r_e is the distance from the centre to the Earth's surface in the WGS84 model and h_a is the aircraft altitude. The distance between the centre of the Earth and the Earth's surface at a given latitude and longitude is obtained through equation:

$$r_e = \sqrt{\frac{(a^2 \cdot cosx)^2 + (b^2 \cdot sinx)^2}{(a \cdot cosx)^2 + (b \cdot sinx)^2}} \quad (2)$$

where a is the length of the major half axis, b is the length of the minor half axis, and x is the latitude. A change of the angle $Delta\ alpha$ can be presented through equation:

$$\Delta\alpha = \frac{s}{r} \quad (3)$$

The resulting latitude is calculated, as a result of the relative distribution of the overall change in angle with the flight direction, by the equation:

$$x = x_0 + sin(h) \cdot \Delta\alpha \quad (4)$$

where h is the flight direction in radians and x_0 is the initial latitude. To calculate the resulting longitude, the following equation should be used:

$$y = y_0 + cos(h) \cdot \Delta\alpha \quad (5)$$

where y is the resulting longitude and y_0 is the initial longitude.

H. Aircraft rotation

In the system, the North is always located at the coordinates $[x, y, z] = [0, 0, \infty]$. Aircraft as Unity 3D objects are placed in the coordinate system so that their course is pointing directly in the z direction. Quaternions are used to rotate a 3D object. For numerical calculations, using the quaternions bears more advantages than calculating rotations through rotary matrices. The basic principle of calculating quaternion rotation is based on following formula:

$$Q = a + b \cdot i + c \cdot j + d \cdot k;\ i^2 = j^2 = k^2 = -1 \quad (6)$$

where a is the scalar component of Q and $\{b, c, d\}$ form the vector part, i, j, k present unit vectors, components of orthonormal basis [9].

The quaternions can be stored as 4 digits, rather than 9 for a rotation matrix. When combining several subsequent rotations, quaternions save 17 floating point operations compared to multiplying rotation matrices together. Moreover, in a practical computation, one has to ensure that the quaternion is normalized (or account for its norm in some other way). Renormalizing the quaternion insted of renormalizing a rotation matrix is far more quicker and effective.

Trying to calculate a rotated vector from a quaternion takes 26 more floating point operations. This is a important fact to realise that the convenience charges something back. Therefore, one has to consider whether the amount of a computation is chaining rotations together or actually computing rotated vectors. Tracking the orientation of a rigid body involves chaining rotations. In the end quaternions would cost a reduction in memory and arithmetic operations needed.

The Euler angles can also be used for the rotation. However, these have several disadvantages such as that they are coordinate-dependent, the coordinate system is local to the object, the complex rotation is dependent on the order of the individual rotations, they have a complex inverse problem and the existence of a gimbal lock, where, despite the changes in the values at one of the angles does not change the rotation of the whole body [10].

In Unity3D, the rotation of an object is described in Vector3 class instance which contains three parameters x, y, and z. Each object in the scene includes a vector that describes its rotation.

For proper rotation of the aircraft due to its course, the standard Unity3D Quaternion.Euler function was used and which requires three parameters. Rotating the 3D object of an aircraft according to its course is only required in the y-axis (Yaw axis). Therefore, the fuction is called in each Update cycle on the GameObject of the aircraft, and that as follows:

Quaternion.Euler (airplane.transform.rotation.x, airplane.flyDirection, airplane.transform.rotation.z), where the airplane is a reference to the GameObjekt of the aircraft, transform is a reference to the object that Unity3D uses to describe the location of the object in the scene [11], rotation is a reference to the Vector3 instance, which includes data about the current rotation of the GameObject, and flyDirection is the float value in which the current course of the aircraft is stored.

Note: GameObject is in Unity3D every entity that is added to the scene. Each such object contains the Start () and Update () functions. The Start () function is called when the object is created. The Update () function is called in each engine status update cycle, which can be several milliseconds in a row.

III. RESULTS

Verification of the design functionality was done through a simplified form of proof of concept implementation. The implementation of the system is neglected as a client-server design. The implementation is mainly focused on the functionality of proposed algorithms, data display in 3D engine and concept validation using VR and AG technologies. Source codes are written in C#, which is one of two supported languages in Unity3D.

A. 3D Sectors Rendering

The implementation contains a complete solution for creating and rendering a polygon-shaped sector. The user is prompted to create such a sector to choose from three to any number of points that define the sector. When creating a new point, the two previous points are connected together and form the line segment. After reselecting the first point in the chain, the chain closes and stops the shape of the sector. Subsequently, the lower and

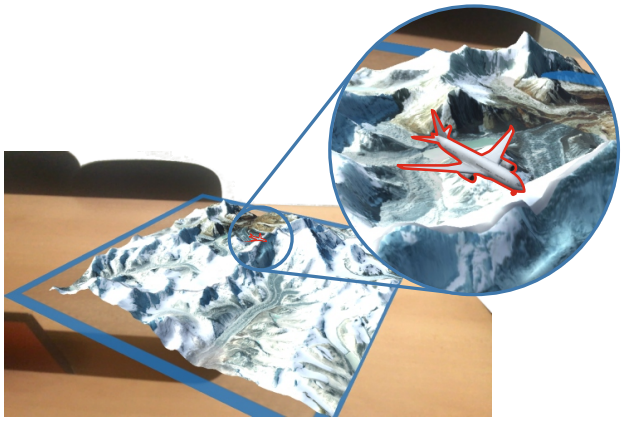

Fig. 3. Simulation in augmented reality

upper heights above the ground level are set and the sector is rendered.

For the rendering itself a triangulation algortihm is used dividing the polygon content into the finite number of triangles. The points of these triangles are then used to create the Mesh bottom base of the sector. The same Mesh as the base is also used for the top base of the sector. To create the points of the rectangulars forming the sides of the polygon another algorithm was used. These points are also used to create a few Mesh objects (as many as the number of the sides of the polygon). All created Mesh objects are then merged into one that is passed on for rendering. For the sector in 3D space, a transparent material is also created using a shader, and a line is applied to all the edges to distinguish the shape of the sector.

B. Augmented Reality

To implement augmented reality Unity AR Kit Plugin and Unity AR Interface were used. With these plug-ins, it was possible to build a 3D scene so that a map (Mapbox) was rendered in it and in which it is possible to place planes, waypoints and runways. The detection of space for the scene location is solved using the algorithm in the AR Kit plugin.

After selecting space, the search for such spaces is deactivated and a map and all other entities are displayed on the desktop. A sample of the scene using the augmented reality is shown in the figure 3.

IV. Conclusion

The goal of the paper was to design and partially implement the software for a creation and modification of the flight routes, air spaces and sectors. Both functional and non-functional requirements were defined for the presented system. Based on these requirements, the proposal of the presented system was prepared.

A functionality of the proposal was verified through a simplified implementation form, known as ”proof of concept”, which verified functionality of the proposed algorithms, data representation in 3D engine and the whole concept using VR and AG technology.

In the following development a client application is planned to be created. The technology of virtual and augmented reality has a high potential and it is very likely that it is only a matter of time before it enters into daily practice in air traffic control. The main advantage will be a real visualization in 3D, which can be very beneficial during an implementation of a process such as Free flight. Using an augmented reality and imaging devices for virtual reality, it will be possible for several air traffic controllers to work at one desk and to cooperate in the environment that they will see, however such environment will be only a projection. In this context, the proposed software could be used as a testing tool for implementation of these technologies into practice. In this context, the proposed software as a testing tool for the implementation of these technologies could be used to set in practice. There are other researches dealing with simulations in the ATC domain, but these simulations are used for different purposes (e.g. [12], [13]). However, the use of VR and AR in trajectory and airspace design hasn't yet been researched.

Due to a support of network features, the system could also be used for air traffic controllers' training programs. Existence of the air traffic scenarios will enable illustrative demonstration of the procedures used in practice. With the support of multiple platforms, including mobile devices, it is also possible to modify the software by creating a computer or mobile air traffic control simulation game. During the creation of the proposal, these visions were taken into account. The proposal should therefore be flexible enough in order to make these visions easy to implement.

Additionally, it will be possible to use systems and methods that are already used in practice, together with the new procedures that are still under preparation. The software can then be used as a tool for introduction of the new procedures and methods with the highest degree of optimization.

Acknowledgment

This work was supported by CTU in Prague research program no. SGS19/133/OHK2/2T/16 “Augmented Reality as a form of Pilots' Support during the Flight”.

References

[1] D. Smith, *Air traffic control handbook : the complete guide for all aviation and air band enthusiasts*. Sparkford, Yeovil, Somerset, UK Newbury Park, Calif: Haynes Pub. Haynes North America, 2010.

[2] M. Nolan, *Fundamentals of air traffic control*. Clifton Park, N.Y: Delmar Cengage Learning, 2011.

[3] H. Cervantes and R. Kazman, *Designing software architectures: a practical approach*. Addison-Wesley Professional, 2016.

[4] J. Kraus, “Determining acceptable level of safety of approach to landing,” in *Proc. of 20th International Scientific on Conference Transport Means 2016*. Kaunas University of Technology, oct 2016, pp. 230–235.

[5] L. Gorton, *Essential Software Architecture*. Springer, 2011.

[6] Microsoft, “Application architecture guide,” 2009.

[7] M. Cade, “Sun certified enterprise architect for java ee study guide,” 2010.

[8] P. Eeles and P. Cripps, *The Process of Software Architecting*. Addison-Wesley, 2010.

[9] R. Mukundan, "Quaternions: From classical mechanics to computer graphics, and beyond," in *Proceedings of the 7th Asian Technology conference in Mathematics*, 2002, pp. 97–105.

[10] D. Brezov, "Optimization and gimbal lock control via shifted decomposition of rotations," *Journal of Applied & Computational Mathematics*, vol. 07, no. 03, 2018.

[11] I. Buyuksalih, S. Bayburt, G. Buyuksalih, A. P. Baskaraca, H. Karim, and A. A. Rahman, "3D modelling and visualization based on the unity game engine: advantages and challenges," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. IV-4/W4, pp. 161–166, nov 2017.

[12] M. Bauer and D. Langr, "Workload—new possibilities for the atc simulation environment," in *2017 International Conference on Military Technologies (ICMT)*. IEEE, 2017, pp. 447–451.

[13] M. Bauer, "Air traffic controller workload environment - atc communication aspect," in *Transport Means - Proceedings of the International Conference*, 2018, pp. 903–908.