Simulation of obstacle avoidance of an UAV

P. Udvardy* B. Beszédes*B. Tóth*, Á. Földi*, A. Botos*
* Óbuda University, Alba Regia Technical Faculty, Székesfehérvár, Hungary
udvardy.peter@amk.uni-obuda.hu, beszedes.bertalan@amk.uni-obuda.hu

2020 New Trends in Aviation Development (NTAD) | 978-1-7281-7325-2/20/$31.00 ©2020 IEEE | DOI: 10.1109/NTAD51447.2020.9379113

Abstract

The aim of the study was to create a model environment for an UAV and to run simulations for dynamic obstacle avoidance in this environment in order to make an optimum method for real flight later. This paper presents the steps for building a simulation environment for UAV flight, the program codes and map creation. The ultimate goal of this project is to develop a working method for automated UAV flight in any real world environment.

I. INTRODUCTION

Using unmanned aerial vehicles (UAVs) in aerial photography, aerial measurements or in any other aerial applications needs a good dynamic obstacle avoidance method to be used in order to keep safety.

Using real flights for the first occasion can be dangerous and not too cost effective thus modelling and simulation can be an acceptable solution before in situ flight. There are many types of modelling and simulation software and platforms available as open source assets or shareware. After creating a good model and the simulation leads to satisfaction the UAV flight will be successful.

In situ testing of an UAV during field work can lead to loss of data and the UAV itself due to a bug during programming and designing paths. Not only the loss of the UAV and the gathered data can happen but serious damage can be caused by the flight [1].

Navigation problems of the UAVs were investigated by many researchers in the recent years. Collision avoidance of a group of UAV flying in formation requires fast image processing algorithm which can be done by many ways such as rapidly exploring random tree (RRT) [2].

Obstacle avoidance can be reached based on the application of differential geometry. This method can be feasible in collision avoidance with aircrafts which fly in a straight line. The differential geometry approach has been useful in determining a guidance algorithm to produce a constant curvature evasion manoeuvre to avoid collision with an Aircraft [3].

Some avoidance method uses the detection of the change of the obstacle size during flight as it simulates the human eye's method where the field of view is bigger when the object comes closer [4].

Small UAVs generally faces with the problem of the lack of differens sensors as battery constraint and small or zero payload capacity limit their application. These UAVs use mainly CMOS cameras instead of complex systems such as lidar system, laser or radar equipment. For 3D model of the obstacles two cameras needed on board but with one camera obstacle avoidance also can be solved if there are tracked waypoint for reference [5].

Figure 1 shows the vision based navigation method of the UAVs.

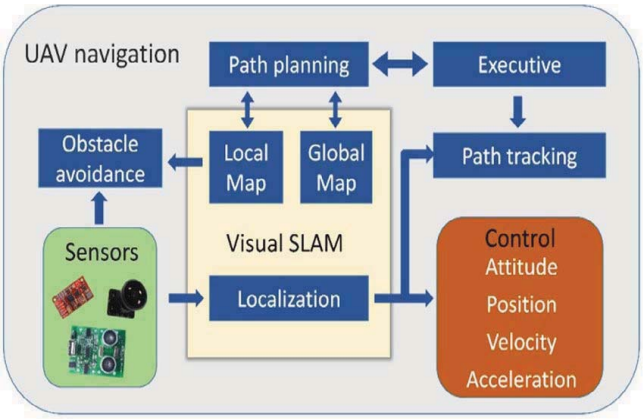


Figure 1. Vision based UAV navigation [4]

Micro aerial vehicles can be equipped with a resource efficient real time 3D terrain reconstruction system using only one down looking camera and an inertial measurement unit (IMU) which helps them fully autonomous landing and land spot detection [6].

The UAVs in this project comes from the DJI factory as our experience showed that DJI drones are reliable at a moderate cost. The DJI Phantom 3 Advanced and the Mavic Air 2 UAV's are available for research purposes at our University.

II. COMPLETED WORK

A. The UAVs

UAV model flights in simulation environment are based on real DJI UAV's. After completing the modelling in the robot simulation environment these two UAV's below are used in real simulation.

The Chinese DJI company can be considered as one of the leading UAV manufacturers in the world. The DJI Phantom 3 Advanced UAV is available since 2015 and beside its good features cannot be considered as the latest techniques [7].

This UAV weighs 1280 grams and has a 15.2V 4480mAh battery which allows maximum 23 minutes' flight time. The onboard Sony EXMOR 1/2.3" camera has 12 million pixels and has a FOV of 94° with 20 millimeters focal length. The ISO value changes between 100 and 3200 for video recording and 100 and 1600 for photographs. The shutter speed varies between 8 and 1/8000 seconds. The resolution of the video caption is 2.7K (30fps).

978-1-7281-7325-2/20/$31.00 ©2020 IEEE

The camera is hanged on-board with the help of a 3-axis gimbal which helps to stabilize the camera during flight.

For navigation DJI Phantom 3 UAV uses GPS/GLONASS system outdoor and has a vision positioning system for indoor positioning. The hover's horizontal accuracy is between 0.3 and 1.5 meters and the vertical accuracy is between 0.1 and 0.5 meters depending on the positioning method.

This UAV can reach approximately 3 kilometers distance outdoor from the home point which is marked before the flight starts. The 'Go home' function helps to find the hover in case of being lost or invisible.

The other UAV is the DJI Mavic Air 2 has similar features but is a more portable UAV available with enhanced features such as 48 million pixels for photographs and 4K video caption at 60 fps. The camera has a FOV of 84° with 24milimeters focal length. This UAV weights 570 grams and can be folded. The maximum flight time is 33 minutes and the maximum distance is 18 kilometers [7].

Figure 2 shows the DJI Mavic Air 2 UAV.

Figure 2. DJI Mavic Air 2 UAV [7]

B. Robot simulation environments

Different robot simulation environment software types are widely available in order to build a virtual environment for real robot modelling without depending on the real robot or UAV. This means lower cost, faster development possibilities and versatile modelling scenes.

These simulators are available for Linux, MacOS, Windows platforms and some of them for Android, too. The main software types are Gazebo, RoboDK, Sim Spark, Webots, OpenRAVE and CoppeliaSIM (formerly known as V-Rep). The main programming languages are C++, Python or Ruby and they support many formats such as OBJ, STL, Collada. Matlab software is also suitable for controlling.

These platforms support generic kinematic chains, force controlled motions as actuators and IMU, collision sensor, GPS, cameras, laser scanners as sensors. Not all but some of these simulation environments support dynamic obstacle avoidance and aerial robots (UAVs).

The CoppeliaSIM (V-Rep) simulation environment was chosen as general environment for modelling. V-Rep environment is a versatile platform for building simulation environment and for modelling and has a modular and decentralized architecture for robot simulation. This provides a more diverse and simplified connectivity between the independent calculation modules [8].

CoppeliaSIM has three types of licensing, the player and the educational versions are free after registration and a pro version is also available.

C. Modelling and building the simulation environment

In CoppeliaSIM environment the models are made of primitive shapes, joints, sensors and path. The mesh can be built by primitive shapes, mainly triangles. The higher the number of the tringles the better the resolution of the model but the slower the model running. For model building meshes can be imported from other sources such as AutoCad applications (OBJ, STL, DFX or 3DS extensions) but the limited number of triangles must be considered to avoid model collapse [9].

To simplify the mesh there are many methods: automatic mesh division, extract the convex hull, decimate the mesh or remove the inside of the mesh.

The shapes that belong together can be merged or grouped depend on the model structure. Colours and textures also can be set. The model elements are in hierarchical structure and the elements which belong to the same group must be named similarly.

There are eight visibility layers in CoppeliaSIM software and these layers can be switched on or off during the modelling and simulation process. The first visibility layer is linked to the shapes and can be adjust in the object common properties menu.

The shapes can act as a simple 'draw' object or as a real physical object depends on the dynamic properties settings of the body.

The modelling environment contains a 5x25 meters resizable floor as a base and a pile of twelve obstacles made of primitive shapes which are non-static, respondable objects.

Figure 3 shows the built simulation environment. This figure contains the whole simulation environment in order to ensure additional information about the particles of the objects and their linkage to each other.

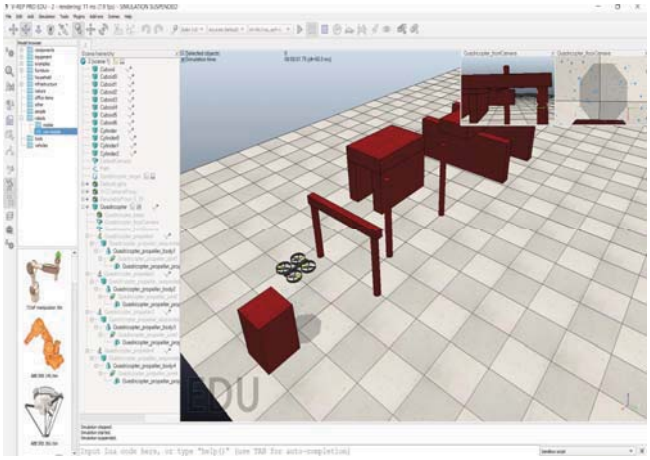


Figure 3. The modelling environment

978-1-7281-7325-2/20/$31.00 ©2020 IEEE

The UAV flies through a pre-defined path and avoid collisions. The path can be determined as a circle type path or a segment type path. The segment type path was selected as it has starting and ending points and the UAV can take off at the beginning, go through it and land at the path's end. If the path must be modified new waypoints can be inserted in the path edit mode.

The UAV model in the simulation is a quadricopter which was chosen from the model browser library and its properties were set manually. There are five programme codes in the UAV model of which four are responsible for the movement of the four rotors and the fifth one is for the realistic movement.

Two cameras were set onboard, one front and one floor camera to monitor the flight process in time and air turbulence can also be seen during UAV simulation flight.

Figure 4 shows the quadricopter's target in visible layer and the hierarchy levels of the quadricopter and the simulation environment.

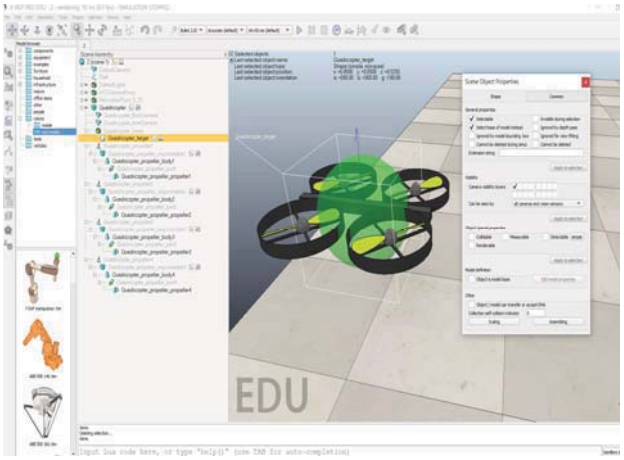


Figure 4. The UAV's target in visible layer

The quadricopter target is an invisible globe under the UAV model (its visible layer was switched off) and the UAV follows this target during the flight process. The target follows and moves through the UAV path and the quadricopter follows it through the pathway.

Figure 5 shows the UAV during simulation together with the path and target which were set to visible.

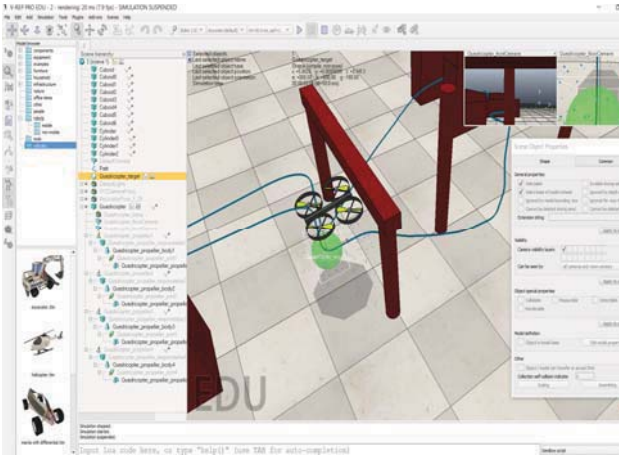


Figure 5. The UAV, the path and the target

Collision avoidance algorithms can adapt to various environments thus the UAV get the target easily [10].

New programme code was written for path following. As the target is a globe its orientation is not important during the movement but as the quadricopter follows it the orientation made a huge failure during flight; the UAV turned downwards and sideward and sometimes felt down. After switching off the target's orientation properties changes the simulation went good.

D. The program code

Except for the dynamics or physics modules that directly operate on all dy-namically enabled scene objects, other calculation modules require the definitionof a calculation task or calculation object, that specifies on which scene objects the module should operate and how [11].

Figure 6 shows the general simulation loop of the simulation environment.

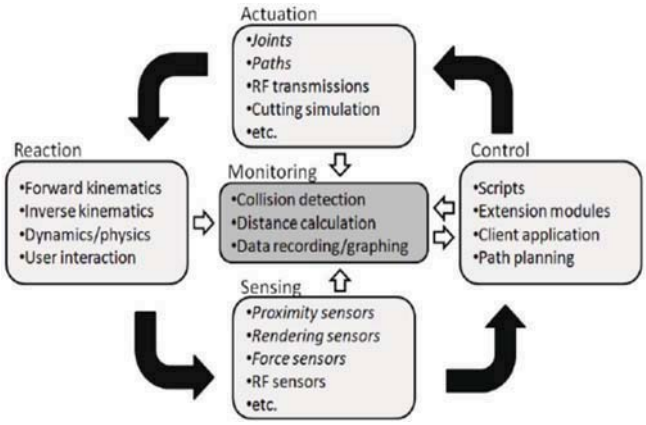
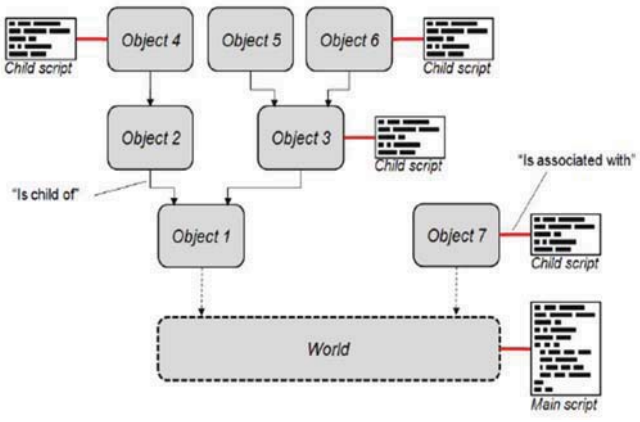


Figure 6. Simulation loop in V-REP environment [11]

A simulation is handled when the client application calls a main script, which in turn can call child scripts. Each simulation scene has exactly one main script that handles all default behaviour of a simulation, allowing simple simulations to run without even writing a single line of code [11]. Figure 7 shows the general script of the V-REP.



978-1-7281-7325-2/20/$31.00 ©2020 IEEE

Figure 7. Main script and child script of the environment [11]

During the simulation process program codes must be changed in order to get the UAV precise movement. The following codes were made:

the "path" value was set to the simulation requirements:
path=sim.getObjectHandle('Path')

the object value was set to target value of the simulation:
object=sim.getObjectHandle('Quadricopter_target')

the path length was defined:
pathLength=sim.getPathLength(path)

the UAV's point of origin was also defined:
posOnPath=0

the UAV's velocity was defined:
v=0.5

the path length was defined:
while true do
l=posOnPath/pathLength
if (l>pathLength) then
l=pathLength
end

values were added to position and orientation:
position=sim.getPositionOnPath(path,l)
--orientation=sim.getOrientationOnPath(path,l)

finally the calculation of target movement was determined:
sim.setObjectPosition(object,-1,position)
--sim.setObjectOrientation(object,-1,orientation)
*posOnPath=posOnPath+v*sim.getSimulationTimeStep()*
sim.switchThread()
end

Figure 6 shows the path from a different point of view, from below during the simulation. Obstacles were set to invisible to show the flight route.

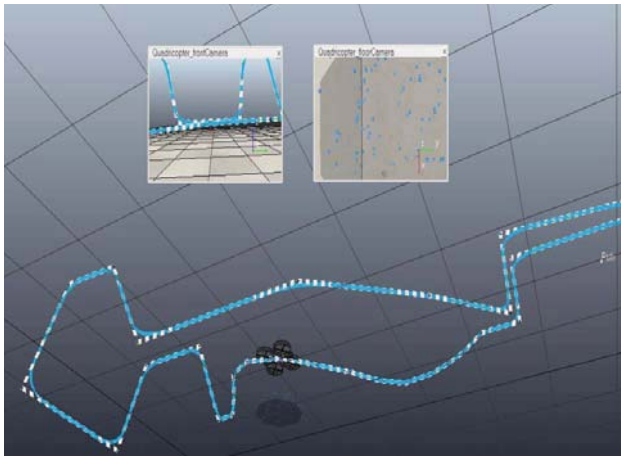


Figure 8. The path seen from below in simulation

III. SUMMARY AND CONCLUSIONS

Nowadays using UAVs in aerial photography, aerial monitoring and in aerial measurements become widespread and is often used in engineering. UAV's can complete their missions in full autonomy using their on-board sensors during their flight. The so-called 'drones' are widely available on market at low price at a good quality but for example DJI products can be used in scientific work with their high performance sensors and high capacity batteries and reliability.

Although these UAVs are suitable for full autonomous flight WiFi signals and microwaves can disturb their functions in some areas and this leads to failure. The other failure factor is the software itself, as a programme bug can destroy the mission. If the UAV breaks down not only the flying machine but the collected data are lost.

That is the reason why modelling, testing and simulation are so important before the in situ flight.

There are many modelling and simulation platform available for testing. One of these platforms is CoppeliaSIM (formerly as known as V-REP) that was used for modelling and testing in this project.

First the simulation environment was built using primitive shapes and giving them physical body features to be realistic during the test phase. The UAV was chosen from the model library browser and its properties was set in order to be similar to the DJI UAVs. The programme codes were set to ensure precise flight during the simulation. A path was created which avoided the physical obstacles and the UAV followed this path.

In the next step the simulation showed the UAV to turn and fall down many times. This irregular behavior was caused by the so-called target which was followed by the UAV and which went through the path. The target is a globe which is set to invisible during flight and its orientation disturbed the UAV. As the orientation was set and the globe did not spin this caused normal movement of the UAV.

Next step of this project is to make in situ flight with DJI Phantom 3 Advanced and Mavic Air UAV's on a field with pre-defined path. The path will be edited and uploaded to the UAV by DJI PC Ground Station and by UgCS ground station software [12], too. The different ground station softwares will be compared in the future studies.

ACKNOWLEDGMENT

This research was supported by Obuda University. We thank our students for their assistance during the field work and modelling.

REFERENCES

[1] Benedetti, Massimiliano De et al. "3D Simulation of Unmanned Aerial Vehicles." WOA (2017).

[2] X. Wang, V. Yadav and S. N. Balakrishnan, "Cooperative UAV Formation Flying With Obstacle/Collision Avoidance," in IEEE Transactions on Control Systems Technology, vol. 15, no. 4, pp. 672-679, July 2007, doi: 10.1109/TCST.2007.899191.

978-1-7281-7325-2/20/$31.00 ©2020 IEEE

[3] Brian A. White, Hyo-Sang Shin, Antonios Tsourdos, UAV Obstacle Avoidance using Differential Geometry Concepts, IFAC Proceedings Volumes, Volume 44, Issue 1, 2011, Pages 6325-6330, ISSN 1474-6670

[4] Yuncheng Lu, Zhucun Xue, Gui-Song Xia & Liangpei Zhang (2018) A survey on vision-based UAV navigation, Geo-spatial Information Science, 21:1, 21-32

[5] A. Al-Kaff, Qinggang Meng, D. Martín, A. de la Escalera and J. M. Armingol, "Monocular vision-based obstacle detection/avoidance for unmanned aerial vehicles," 2016 IEEE Intelligent Vehicles Symposium (IV), Gothenburg, 2016, pp. 92-97

[6] C. Forster, M. Faessler, F. Fontana, M. Werlberger and D. Scaramuzza, "Continuous on-board monocular-vision-based elevation mapping applied to autonomous landing of micro aerial vehicles," 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, 2015, pp. 111-118

[7] https://www.dji.com/hu

[8] Freese M., Singh S., Ozaki F., Matsuhira N. (2010) Virtual Robot Experimentation Platform V-REP: A Versatile 3D Robot Simulator. In: Ando N., Balakirsky S., Hemker T., Reggiani M., von Stryk O. (eds) Simulation, Modeling, and Programming for Autonomous Robots. SIMPAR 2010. Lecture Notes in Computer Science, vol 6472. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-17319-6_8

[9] https://www.coppeliarobotics.com/

[10] Liu J., Wang Z., Zhang Z. (2020) The Algorithm for UAV Obstacle Avoidance and Route Planning Based on Reinforcement Learning. In: Wang R., Chen Z., Zhang W., Zhu Q. (eds) Proceedings of the 11th International Conference on Modelling, Identification and Control (ICMIC2019). Lecture Notes in Electrical Engineering, vol 582. Springer, Singapore

[11] Freese, Marc & Singh, Surya & Ozaki, Fumio & Matsuhira, Nobuto. (2010). Virtual Robot Experimentation Platform V-REP: A Versatile 3D Robot Simulator. 6472. 51-62. 10.1007/978-3-642-17319-6_8.

[12] https://www.ugcs.com/

 978-1-7281-7325-2/20/$31.00 ©2020 IEEE