Development of Autonomous Drones for Adaptive Obstacle Avoidance in Real World Environments

Arne Devos
Faculty of Engineering Technology,
KU Leuven,
Belgium
arne.devos@student.kuleuven.be

Emad Ebeid
SDU UAS Center, MMMI,
University of Southern Denmark,
Denmark
esme@mmmi.sdu.dk

Poramate Manoonpong
Embodied AI and Neurorobotics Lab, MMMI,
University of Southern Denmark,
Denmark
poma@mmmi.sdu.dk

***Abstract*—Recently, drones have been involved in several critical tasks such as infrastructure inspection, crisis response, and search and rescue operations. Such drones mostly use sophisticated computer vision techniques to effectively avoid obstacles and, thereby, require high computational power. Therefore, this work tuned and tested a computationally inexpensive algorithm, previously developed by the authors, for adaptive obstacle avoidance control of a drone. The algorithm aims at protecting the drone from entering in complex situations such as deadlocks and corners. The algorithm has been validated through simulation and implemented on a newly developed drone platform for infrastructure inspection. The design of the drone platform and the experimental results are presented in this study.**

***Index Terms*—Autonomous drone system, adaptive obstacle avoidance, simulation, implementation.**

I. Introduction

Unmanned Aerial Vehicles (UAV) or drones have been a subject of great interest over the last decades. Applications range from the first response in crisis situations [1] to agriculture [2],[3] and safety inspections [4],[5]. The environment in which they operate is mostly known and away from obstacles. When moving to more complex surroundings, like indoors, there is often a qualified pilot required to operate these drones. However, it is not always possible or wanted to have a pilot controlling the drone. With the market for drones growing rapidly [6], there is a need for drones to be operated autonomously. An important challenge for the drone is to autonomously and adaptively avoid obstacles it may encounter in complex environments.

To address the challenge, there has been a lot of research done to explore different methods. Some rely on computer vision and algorithms to identify objects [7]. This provides the distance to the object such that the drone can move to ensure a safe flight trajectory. Despite, this approach is working well in simple environments, it does not perform well in surroundings with complex structures or cramped indoor spaces. A way to resolve this is to make a map of the surroundings with the SLAM (Simultaneous Localization and Mapping) technique [8],[9]. Once the map has been obtained, a path can be planned for the drone. Nevertheless, this control strategy uses cameras, laser scanner or complex sensor arrays to create a map and perform navigation with obstacle avoidance. These sensor equipment are heavy and usually need complex algorithms, which require high computational and processing power, for signal processing.

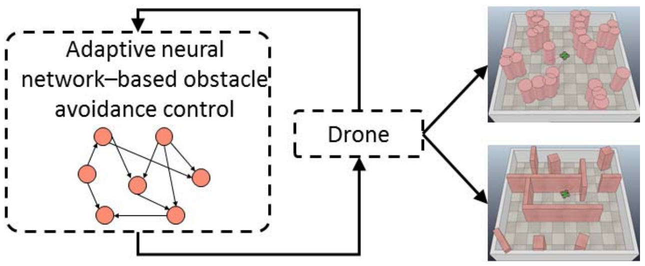


Fig. 1: The adaptive obstacle avoidance control acts as a closed-loop control system which can be directly applied to simulated and real drones

To achieve an autonomous drone with less complex sensors and low computational and processing power for adaptive obstacle avoidance in real-world environments, a drone development and its adaptive obstacle avoidance control is presented here. This study continues previous work [10]. It is based on a simple two-neuron recurrent network with synaptic plasticity [11], uses only two small and light-weight LiDAR (Light Detection and Ranging) sensors to detect obstacles and enable the drone to autonomously avoid them. Due to the neural dynamics and synaptic plasticity of the network, the drone can also effectively adapt its obstacle avoidance behavior to successfully navigate in complex environments with obstacles, corners, and deadlocks.

The paper is structured as follows; Section II describes the adaptive obstacle avoidance control. Section III presents the complete design of a LiDAR-based obstacle avoidance drone system. Section IV presents the simulation results of adaptive obstacle avoidance and navigation behaviors of a simulated drone in complex environments.

II. Adaptive obstacle avoidance control

The adaptive obstacle avoidance control for a drone was developed and presented in [10] (see Fig. 1). It is based on a simple two-neuron recurrent network with synaptic plasticity [11]. Here we shortly introduce the key components and

978-1-5386-7377-5/18/$31.00 ©2018 IEEE
DOI 10.1109/DSD.2018.00009

features of the control network where its details are referred to [10]. The network consists of three main discrete-time non-spiking neurons. It receives obstacle detection signals from two LiDAR sensors, installed at the front part of the drone. The range of each sensor is set to 50 cm. The raw sensory signals are mapped to a range of [-1, 1] where -1 means no obstacle in the range and 1 means that an obstacle is near (about 20 cm distance). The output of the control network is converted to a yaw command for steering the drone. According to this setup, a positive yaw value (meaning that the drone detects an obstacle on the left) will steer the drone to turn right and a negative yaw value (meaning that the drone detects an obstacle on the right) will steer it to turn left.

By exploiting short-term memory in the neural dynamics of this recurrent control network, the drone can continuously turn to successfully avoid an obstacle although the sensors do not detect an obstacle anymore. In other words, this turning behavior is guided by first LiDAR sensory feedback and then later by the short-term memory. Such a memory-driven turning behavior is important to deal with corner and deadlock situations. By applying synaptic plasticity [11] to the control network, the short-term memory will be regulated online during the interaction between the drone and the environment. This temporal memory regulation leads to optimal turning behavior to avoid obstacles, corners, and deadlocks in different environments. As a result, the drone can successfully navigate in complex environments with obstacles without getting stuck.

III. LiDAR-based obstacle avoidance drone design

The most important requirement for a drone, especially when flying beyond visual line of sight, is safety including an ability to autonomously and adaptively avoid obstacles [12]. The drone should be reliable so the mission can be completed without endangering people or damaging property. The battery has to be large enough to ensure a sufficiently long flight time, the control strategy should be easy to implement in the controller, and the drone should be able to lift the necessary equipment. Since the end-goal is to implement the control strategy in small drones, weight and dimensions of the parts should be considered as well. The first section presents the basic mechanical components that make up the drone so it fits the preconditions. The second section presents the electronic components, like the controller and sensors for autonomous obstacle avoidance, as well as how to set up these sensors correctly and calculate the minimum width of detectable obstacles. the third section presents the software design of the drone system.

A. Mechanical components

The first decision to be made is how many rotors the drone should have. In this work, a quadcopter has been chosen as a study platform since this meets the design requirements in regard to weight and power usage. The next step is to find a suitable frame. It has to be strong enough to carry sensors and other equipment (e.g., inspection camera). Carbon fiber is commonly used since it is rigid enough to withstand the forces of the motors along its arms without adding too much weight. In this work, Tarot 650 Sport frame has met these requirements (Fig. 2).

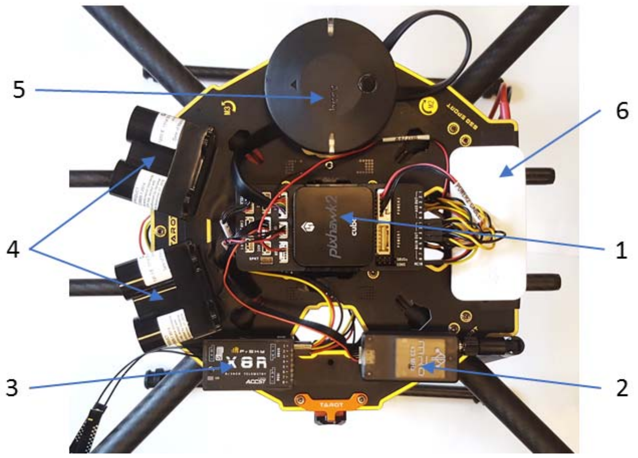


Fig. 2: The drone with 1) Pixhawk, 2) telemetry, 3) receiver, 4) LIDAR sensor, 5) GPS module, 6) Raspberry Pi

When dimensioning the motors, the type and length of the propellers have to be chosen based on the desired amount of thrust that the drone needs to lift its payload. For example, if the propeller is too small, the drone will not have enough thrust to lift its equipment. To have a first idea of how the drone will perform, a software tool like eCalc [13] can be used. eCalc offers a large database of common parts and it is easy to view which parameters can be changed.

To meet the requirements of lifting power, the drone will be equipped with 370 KV motors together with carbon fiber propellers of 14" long. The carbon fiber gives extra stiffness to the propeller so it does not bend when it is rotating at high speed. This helps to dampen unwanted vibrations.

Every motor needs to be driven by an Electronic Speed Control (ESC) to couple with the flight controller. It tells the motors how much power should go to each motor to give the correct steering. For our configuration, the motor should be equipped with an ESC of maximum 40A. This will be enough to cover the demanded power by the motors.

To decide on the size of the battery, it is important to find a balance between weight and flight time. A battery of 22.2 V and a capacity of 8000 mAh was chosen for the drone. This allows for a flight time of 25 minutes without extra payload.

B. Hardware components

To be able to add other components and implement developed control methods on the drone, the controller should be flexible to use [14]. According to this, an open-source flight controller is used, named PIXHAWK 2 [15], [16] (see Fig. 2-1). This controller equips with a 3D gyroscope, an accelerometer and, a barometer. Other installed equipment includes telemetry for data exchange with a ground station (Fig. 2-2), a receiver for manual control (Fig. 2-3), a GPS module with an internal compass for outdoor navigation (Fig. 2-5) and a

Raspberry Pi Zero to run the algorithm (Fig. 2-6). The RPi is connected with the flight controller through the serial link. Details about the software components is presented in III-C.

Classical obstacle avoidance strategies to control the drone are based on computer vision [7], [8] which needs a great deal of processing power. The proposed neural-based control method (see Section II) uses two simple sensors which can be ultrasonic or LIDAR. This reduces the amount of high dimensional signal processing in comparison to image processing for computer vision. Here two LIDAR-Lite v3 sensors from Garmin are used (Fig. 3 and 2-4). The sensor is low-cost, lightweight, and easy to interface with the PIXHAWK module of the drone via I2C, with a viewing angle of 2° (Fig. 3). It is assumed that the sensors are mounted at the center of the frame with an orientation angle of Θ. This angle depends on the sensor range L_s and the safety margin L_m as shown in Fig. 4. These two parameters are considered in order to make sure that the drone does not hit an obstacle with its propellers.

The minimum width of the detected obstacle (W_{obj}) can be determined by the following equation:

$$W_{obj} \geq 2\ tan(\frac{\Theta}{2})(L_x + \frac{L_d}{2}), \quad (1)$$

where L_x is the distance between the drone and the obstacle. This distance depends on the speed of the drone as well as the response time in which the drone can safely turn. The drone's response time varies based on the drone's mechanical and electronic selected parts. However, the worst-case response time of a drone can be calculated from the drone's behavior on air and its empirical data.

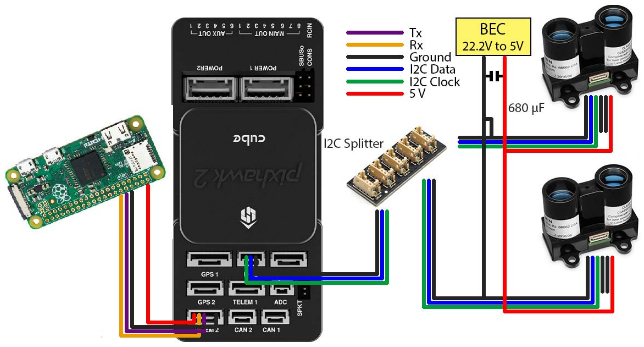


Fig. 3: Schematics to connect the sensors and Raspberry PI with the controller

C. Software components

In order to control the drone through the RPi, first, the RPi has to communicate with the drone's flight controller through MAVLink [17] (Micro Air Vehicle Link) protocol. In order to do so, MAVProxy [18], a MAVLink ground station written in Python, is installed in the RPi OS (Raspbian) to translate readable commands into MAVLink command messages.

To streamline and make automation easier, a Python-based module named DroneKit [19], a collection of drone APIs with underlying functionality to control a drone, is used on top of

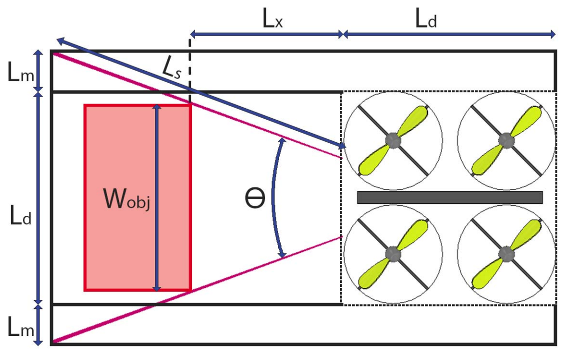


Fig. 4: Dimensions of the drone and the sensors in regard to the width of the obstacle

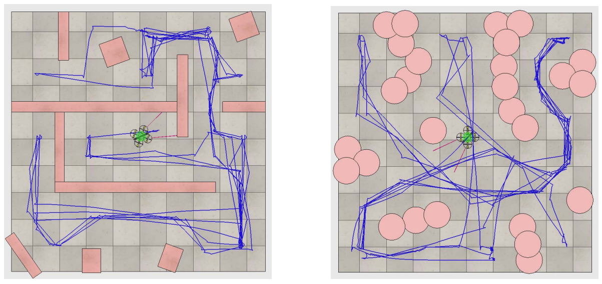
MAVProxy to gain easier access to the command library of the drone. Thus, the controlling algorithm can run onboard and send commands to the drone through the DroneKit framework.

IV. Experimental results

In this study, preliminary results of the performance of the adaptive obstacle avoidance control in the simulation are shown. A V-REP simulation [20] was used to simulate a drone and the controller was implemented in C++. The communication between the simulated drone and the neural control network is based on the Robot Operating System (ROS). With this implementation, the controller can easily be transferred to the real drone hardware.

The width of the drone in the simulation is 34 cm (L_d, shown in Fig. 4). Together with a safety margin of 6 cm (L_m, shown in Fig. 4) and a sensor range of 50 cm (L_s, shown in Fig. 4), this makes for an orientation angle Θ (Fig. 4) of 40° between the sensors. In this test, the speed of the drone was set to 0.1 m/s.

Two complex environments with different obstacles, corners, and deadlocks (see Fig. 1) were created. The first one was made to look like a room that has to be explored. The second environment has more sharp turns and deadlocks. The minimum width of the obstacles was calculated with equation 1. If the drone can safely turn when the object is 15 cm away, the minimum width of the obstacles has to be 24 cm. All objects in the maps are larger than this value. Figure 5 shows the results of the experiments. Each environment was run for about 30 minutes with the drone having no knowledge of the map. The drone successfully navigated both environments without getting stuck and colliding with the obstacles. It can also explore small corridors. The results have been recorded and can be watched from [21]. Other quantitative tests in different environments with a low density of obstacles, a high density of obstacles, and a high density of obstacles corners and deadlocks can be also seen at [10]. There, we observed that the drone had 100 % success to navigate without getting stuck or crashing in the environment with a low density of obstacles, 90 % success in the environment with a high density of obstacles, and 90 % success in the environment with a high density of obstacles as well as corners and deadlocks.



(a) Environment 1 (b) Environment 2

Fig. 5: Results of adaptive autonomous obstacle avoidance

V. Conclusion

In this paper, an adaptive neural control with synaptic plasticity for autonomous obstacle avoidance and exploration was explained. The V-REP simulator was used to simulate a drone and different environments as well as to evaluate the performance of the developed controller and demonstrate the obstacle avoidance behavior. Complete design of a drone platform was presented. The drone is developed to fit the requirements of the adaptive obstetrical avoidance algorithm. Therefore, two light-weight LiDAR sensors were selected and mathematical equations were formulated to find the exact locations of the LiDARs to satisfy the predefined design requirements such as the minimum width of detectable obstacles. Future work aims to test the drone in similar simulation environments and compare its results.

Acknowledgment

The research leading to these results has received funding from SDU Strategic Focus *Drones for Energy* project.

References

[1] L. Apvrille, T. Tanzi, and J.-L. Dugelay, "Autonomous drones for assisting rescue services within the context of natural disasters," in *General Assembly and Scientific Symposium (URSI GASS), 2014 XXXIth URSI.* IEEE, 2014, pp. 1–4.

[2] T. Pobkrut, T. Eamsa-ard, and T. Kerdcharoen, "Sensor drone for aerial odor mapping for agriculture and security services," in *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2016 13th International Conference on.* IEEE, 2016, pp. 1–5.

[3] V. Duggal, M. Sukhwani, K. Bipin, G. S. Reddy, and K. M. Krishna, "Plantation monitoring and yield estimation using autonomous quadcopter for precision agriculture," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on.* IEEE, 2016, pp. 5121–5127.

[4] R. Ashour, T. Taha, F. Mohamed, E. Hableel, Y. A. Kheil, M. Elsalamouny, M. Kadadha, K. Rangan, J. Dias, L. Seneviratne *et al.*, "Site inspection drone: A solution for inspecting and regulating construction sites," in *Circuits and Systems (MWSCAS), 2016 IEEE 59th International Midwest Symposium on.* IEEE, 2016, pp. 1–4.

[5] J. Irizarry, M. Gheisari, and B. N. Walker, "Usability assessment of drone technology as safety inspection tools," *Journal of Information Technology in Construction (ITcon)*, vol. 17, no. 12, pp. 194–212, 2012.

[6] D. Floreano and R. J. Wood, "Science, technology and the future of small autonomous drones," *Nature*, vol. 521, no. 7553, p. 460, 2015.

[7] H. Sedaghat-Pisheh, A. R. Rivera, S. Biaz, and R. Chapman, "Collision avoidance algorithms for unmanned aerial vehicles using computer vision," *Journal of Computing Sciences in Colleges*, vol. 33, no. 2, pp. 191–197, 2017.

[8] R. Mur-Artal and J. D. Tards, "Visual-inertial monocular slam with map reuse," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 796–803, April 2017.

[9] S. Moon, W. Eom, and H. Gong, "Development of large-scale 3d map generation system for indoor autonomous navigation flight work in progress," *Procedia Engineering*, vol. 99, pp. 1132 – 1136, 2015.

[10] C. K. Pedersen1 and P. Manoonpong, "Neural Control and Synaptic Plasticity for Adaptive Obstacle Avoidance of Autonomous Drones," *Lecture Notes in Artificial Intelligence*, 2018.

[11] E. Grinke, C. Tetzlaff, F. Wörgötter, and P. Manoonpong, "Synaptic plasticity in a recurrent neural network for versatile and adaptive behaviors of a walking robot," *Frontiers in neurorobotics*, vol. 9, p. 11, 2015.

[12] FreeD, "Free the Drones," https://www.sdu.dk/freed.

[13] Solution for All Markus Mueller, "eCalc," https://www.ecalc.ch/xcoptercalc.php.

[14] E. Ebeid, M. Skriver, K. H. Terkildsen, K. Jensen, and U. P. Schultz, "A Survey of Open-Source UAV Flight Controllers and Flight Simulators," *Microprocessors and Microsystems*, vol. 61, pp. 11 – 20, 2018.

[15] px4, "Pixhawk," https://www.pixhawk.org.

[16] Dronecode Project, "px4: the professional autopilot," https://www.px4.io.

[17] mavlink, "MAVLink," https://github.com/mavlink.

[18] ArduPilot, "MAVProxy," https://github.com/ArduPilot/MAVProxy.

[19] 3DR, "DroneKit," https:/dronekit.io/.

[20] E. Rohmer, S. P. Singh, and M. Freese, "V-rep: A versatile and scalable robot simulation framework," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on.* IEEE, 2013, pp. 1321–1326.

[21] "Recording of testings in different environments," https://tinyurl.com/ya7thupd.