Neural Control and Synaptic Plasticity for Adaptive Obstacle Avoidance of Autonomous Drones

Christian Koed Pedersen[1] and Poramate Manoonpong[1,2,*]

[1]Embodied AI and Neurorobotics Lab, Centre for BioRobotics, The Mærsk Mc-Kinney Møller Institute, University of Southern Denmark, Odense M, Denmark
[2]Bio-inspired Robotics & Neural Engineering Lab, School of Information Science & Technology, Vidyasirimedhi Institute of Science & Technology, Rayong, Thailand
(chped13@student.sdu.dk, *poma@mmmi.sdu.dk)
http://ens-lab.sdu.dk/

Abstract. Drones are used in an increasing number of applications including inspection, environment mapping, and search and rescue operations. During these missions, they might face complex environments with many obstacles, sharp corners, and deadlocks. Thus, an obstacle avoidance strategy that allows them to successfully navigate in such environments is needed. Different obstacle avoidance techniques have been developed. Most of them require complex sensors (like vision or a sensor array) and high computational power. In this study, we propose an alternative approach that uses two simple ultrasonic-based distance sensors and neural control with synaptic plasticity for adaptive obstacle avoidance. The neural control is based on a two-neuron recurrent network. Synaptic plasticity of the network is done by an online correlation-based learning rule with synaptic scaling. By doing so, we can effectively exploit changing neural dynamics in the network to generate different turning angles with short-term memory for a drone. As a result, the drone can fly around and adapt its turning angle for avoiding obstacles in different environments with a varying density of obstacles, narrow corners, and deadlocks. Consequently, it can successfully explore and navigate in the environments without collision. The neural controller was developed and evaluated using a physical simulation environment.

1 Introduction

The use of drones in various applications (including inspection, environment mapping, and search and rescue operations) has expanded in recent years [1,2,3]. The applications often take place outside in open areas away from obstacles and people, but as the technology advances the opportunity for flight in more complex areas arises. These are environments such as indoor or urban areas with many obstacles, sharp corners, and deadlocks. However, it does require the drone to feature a collision avoidance strategy to enable safe flight. Often this strategy is achieved by using cameras and computer vision with algorithms (like, SURF or classifiers) to detect frontal objects and estimate a distance to them during flying [4, 5]. The knowledge is then used to avoid obstacles before colliding them by moving to either side of the object. These vision-based algorithms sometimes cannot handle corner cases or scenarios where it is not a single enclosed entity in the way. To overcome this problem, other methods try to map the environment using algorithms (such as SLAM) and afterwards use the information for path planning to avoid obstacles [6]. This may require high computational power which can be difficult to implement on drones, especially when they are small in size. In this study, we propose an alternative approach inspired by [8] where they use simple two sensors and neural control with synaptic plasticity for adaptive obstacle avoidance of walking robots. Here we apply it to controlling autonomous drones. The neural control is based on a two-neuron recurrent network. Synaptic plasticity of the network is done by an online correlation-based learning rule with synaptic scaling. This neural-based control technique is a simple but effective way to enable a drone to navigate in complex environments with obstacles, narrow corners, and deadlocks. Due to low computational resource requirements for the neural algorithm implementation, this can open up for small indoor drone applications to fly in the environment with varying obstacle densities.

2 Materials and Methods

In this study, adaptive obstacle avoidance of an autonomous drone is achieved via a sensorimotor loop which involves neural dynamics, synaptic plasticity, sensory feedback, and a physical drone body (Fig. 1). Neural dynamics and plasticity are embedded in a neural control network (Fig. 2). Sensory feedback from the left and right obstacle detection sensors of the drone is processed through the network. The network outputs are used for pitch and yaw control of the drone. This results in an autonomous drone system that can navigate in a complex environment without collision. The drone is simulated in a V-REP and the communication between the simulated drone and the neural control network is based on the Robot Operating System (ROS) .

2.1 System Overview

The drone system is based on two main entities (see Fig. 1). It consists of a V-REP [7] simulation and a ROS-based neural control system.

Here we use a yaw rate and attitude roll and pitch parameters as control inputs to the drone. A positive value of the yaw rate will drive the drone to turn clockwise around its own axis. A negative value will make it turn counter clockwise. Two ultrasonic-based distance sensors are implemented at the front part of the drone for obstacle detection. Each sensor has a beam angle of 10 deg and a range of 0.5 m. The sensors are rotated 35 deg and positioned 5 cm with no pitch from the forward pointing axis of the drone. Height control is not the main focus here; therefore, the height setpoint is kept fixed at a certain height in all experiments. The neural controller is based on ROS and written in C++.

The ROS interface from the simulation enables the controller to get sensory information and send motor commands to the drone simulation. The controller implements the neural control network and synaptic plasticity described in the following sections.

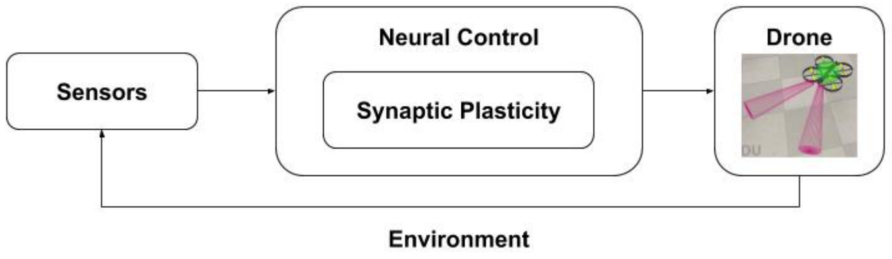

Fig. 1. System overview of adaptive embodied neural closed-loop control for autonomous obstacle avoidance and navigation behaviours of a drone.

2.2 Neural Control for Obstacle Avoidance of Autonomous Drones

The neural control for obstacle avoidance of an autonomous drone is derived from a neural network shown in Fig. 2. The basic principle of this neural control approach is inspired by Braitenberg vehicle [9]. Here it is adapted to gain additional dynamics for a drone platform rather than a two wheeled robot. All neurons of the network are modeled as discrete-time non-spiking neurons. The activity a_i of each neuron develops according to:

$$a_i(t) = \sum_{j=1}^{n} W_{ij} o_j(t-1) + B_i, i = 1, ..., n \tag{1}$$

where n denotes the number of units, B_i an internal bias term or a sensory input to neuron i, W_{ij} the synaptic strength of the connection from neuron j to neuron i. The output o_i of all neurons of the network is calculated by using a hyperbolic tangent (tanh) transfer function, i.e., $o_i = tanh(a_i), \in [-1, 1]$, except for the two neurons (N6 and N7, see Fig. 2) using a sigmoid transfer function.

The core of the network consists of the two recurrent input neurons (N1 and N2, see Fig. 2). They each receive obstacle detection signals from two ultrasonic-based distance sensors, installed at the front part of the drone. The range of each sensor is adjusted to 50 cm. Before feeding the raw sensory signals to the network, we map them to the 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 signals ($O_{1,2} \in [-1, 1]$) of the two neurons are transmitted to the output neuron (N3). The output of N3 is then mapped to a yaw command which is finally sent to the drone.

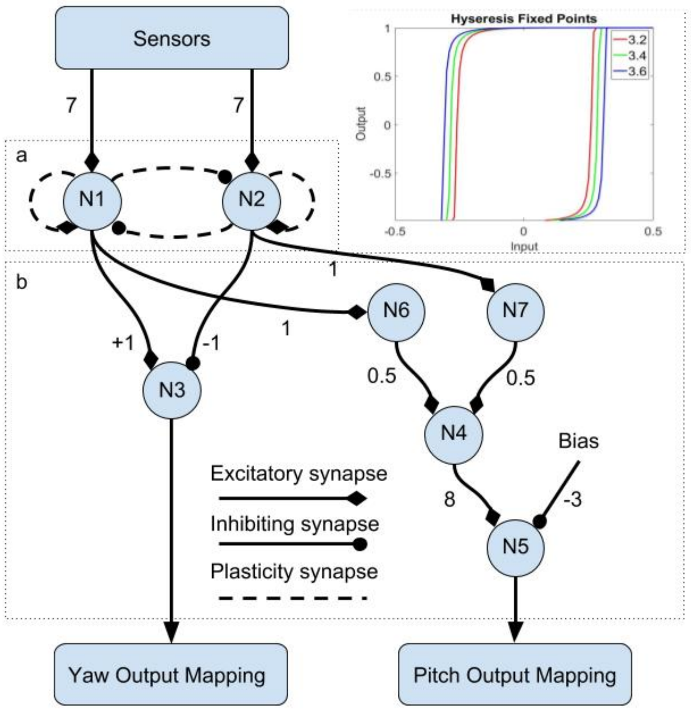

Fig. 2. The neural control network for obstacle avoidance of drones. (a) The two recurrent input neurons with plasticity synapses. (b) The drone specific network used for yaw and pitch control. (c) An example of input-output relation of a recurrent neuron (i.e., N1) of the network. The neuron shows different hysteresis loops at different self-connection weights.

According to this setup, the yaw command will be a positive value if the drone detects an obstacle on the left and a negative value if it detects an obstacle on the right.

The self-connections (W_{11}, W_{22}) can lead to a hysteresis effect in their neural activations. The hysteresis assures that the drone can perform a continuous turning behavior although the sensors do not detect an obstacle anymore. Note, the strength of the self-connections (> 1.0) defines the hysteresis width which determines the turning angle in front of the obstacles for avoiding them, i.e., the larger the hysteresis interval, the larger the turning angle. Additionally this hysteresis effect can reduce the amount of a wall following behavior that a network without the self-connections or Braitenberg vehicle-based control will generate.

Besides the self-connections, the neurons are mutually connected by two inhibitory synapses (W_{12}, W_{21}). This forms a so-called even loop [10]. In an even loop, in general, only one neuron at a time is able to produce a positive output, while the other one has a negative output, and vice versa. This guarantees the optimal functionality for avoiding obstacles and escaping from getting stuck in corner cases. In other words, in a corner the neural system can remember the first obstacle for a certain duration and executes the corresponding behavior (turn) and, in addition, it does not trigger the behavior induced by the second obstacle which would be a turning in the opposite direction.

However, since this network only affects yaw, the drone will always continue to move forward depending on the pitch setpoint given. In case there is not enough space to perform a forward-moving turn for avoiding an obstacle, the drone might collide the obstacle. To overcome the problem, the second control network (N4-N7) was developed (Fig. 2). It is applied for pitch control and works by first changing the activation range of the two input neurons from [-1, 1] to [0, 1] using simple neurons with a sigmoid transfer function. Next, it finds the average by having synapses with weights of 0.5 from each sigmoid neuron (N6, N7) to a hidden neuron (N4). The output of N4 represents the distance and presence of nearby objects with a range of [0, 1]. A bias term with inhibitory synapse is introduced to the final output neuron (N5), with the purpose of driving the output towards negative values when no objects are detected. Finally the resulting output from neuron N5 is used as a scaling of the pitch setpoint. This leads to reduction of forward motion when objects become closer and at a certain point also backward motion when objects are very close to the drone.

The ranges for yaw and pitch setpoints are arbitrary in the simulation and do not represent ordinary units such as m/s or similar, and thus the final yaw and pitch outputs need to be mapped appropriately. The weights of the recurrent and inhibitory synapses between the two input neurons, are responsible for the adaptive obstacle avoidance behavior of the drone. In this study, the self-connections are initially set to a positive value (e.g., 0.5) and the mutual connections between N1 and N2 are initially set to a negative value (e.g., -0.5). To obtain proper connection weights for avoiding obstacles, corners, and deadlocks in different environments, we apply synaptic plasticity mechanisms described in the following section. The other weights of the network are statically set as depicted in Fig. 2.

2.3 Synaptic Plasticity for Adaptive Obstacle Avoidance

To achieve adaptive obstacle avoidance of an autonomous drone (i.e., online adaption of its turning angle for avoiding obstacles in different environments with a varying density of objects, narrow corners, and deadlocks), we use here correlation-based learning [11] to modify the four synaptic weights ($W_{11,22}$ and $W_{12,21}$) of the two-neuron recurrent network, part of the neural control network (see Fig. 2). This learning rule is based on three factors: The output activity $O_i(t)$ ($i \in 1,2$) of the neurons ($N_{1,2}$) at the time step t, the output activity $O_i(t-1)$ of the neurons ($N_{1,2}$) at the previous time step (t-1), and a reflex signal $R_i(t)$ computed from the sensory input I_i as described in equation 2.

$$R_i(t) = \begin{cases} 1, & \text{if } I_i(t) > -0.5, \\ 0, & \text{if } I_i(t) \leq -0.5. \end{cases} \tag{2}$$

The reflex signal is used to control the learning process which will be activated as soon as the drone detects an obstacle at close range (about 0.3 m). Additionally, we also employ synaptic scaling [12] to avoid instability of the synaptic weights. To assure that the synaptic weights do not change their sign in order to maintain the hysteresis effects, for the learning rule, we map the outputs $O_i \in [-1, 1]$ to the positive interval $v_i \in [0, 1]$ as shown in equation 3.

$$v_i = \frac{O_i + 1}{2}. \tag{3}$$

The self-connections ($W_{11,22}$) are governed by Equations (2,3), respectively.

$$W_{11}(t+1) = W_{11}(t) + \mu_r \cdot v_1(t-1) \cdot v_1(t) \cdot R_1(t) + \gamma(k - v_1(t)) \cdot W_{11}(t)^2, \tag{4}$$

$$W_{22}(t+1) = W_{22}(t) + \mu_r \cdot v_2(t-1) \cdot v_2(t) \cdot R_2(t) + \gamma(k - v_2(t)) \cdot W_{22}(t)^2. \tag{5}$$

μ_r is the learning rate that changes the time scale of the learning process. It is here set to 0.01. γ is the forgetting rate that similarly changes the time scale of forgetting/synaptic scaling. It is set 0.0003. k is the offset that enables a constant reduction of the weights when no obstacle is detected. It is set to -0.01.

The two inhibitory connections ($W_{12,21}$) are modified in a similar fashion, but are changed equally to maintain symmetry; thereby forming the even loop [10]. This is done by introducing temporary variables ($q_{1,2}$) that calculate from them the average inhibitory connections. These parameters are updated as follows:

$$q_1(t+1) = \mu_q \cdot v_1(t-1) \cdot v_1(t) \cdot R_1(t) + \gamma(k - v_1(t)) \cdot q_1(t)^2, \tag{6}$$

$$q_2(t+1) = \mu_q \cdot v_2(t-1) \cdot v_2(t) \cdot R_2(t) + \gamma(k - v_2(t)) \cdot q_2(t)^2, \tag{7}$$

$$W_{12,21}(t+1) = W_{12,21}(t) + 1/2 \cdot (q_1(t+1) + q_2(t+1)). \tag{8}$$

μ_q is the learning rate and set to 0.015. γ and k are the forgetting rate and the offset and they are set to the values described above. Note that all meta parameters (i.e., the learning and forgetting rates and the offset) are empirically selected.

In principle, this learning mechanism changes the synaptic weights of the two-neuron recurrent network in a way that the sensory inputs and the weights will drive the neural output to reach and stay at the upper fixed point (≈+1) of the hysteresis loop (see Fig. 2c) while the drone is trying to escape from a corner or a deadlock. As a consequence, the drone will escape from the situation by performing a very large turning angle. Once it has escaped or does not detect an obstacle any more, the second part of the mechanism (synaptic scaling) will reduce the synaptic weights such that the neural output returns to the lower fixed point (≈-1) of the hysteresis loop (see Fig. 2c); thereby the drone will stop turning and continue to fly forward. In other words, the interaction of correlation-based learning and scaling moves the neural system between two fixed point states (i.e., hysteresis effects). Note that the used learning mechanism is independent of the initial weight values [12].

3 Experiments and Results

The neural controller described in this study was tested in different environments with and without the use of synaptic plasticity. Three environments with a varying density of objects (see Fig. 3) were created. The first and second environments (Figs. 3a and 3b) have sparse and dense random obstacles, respectively, while the third one has obstacles that were placed to form corners and deadlocks (Fig. 3c).

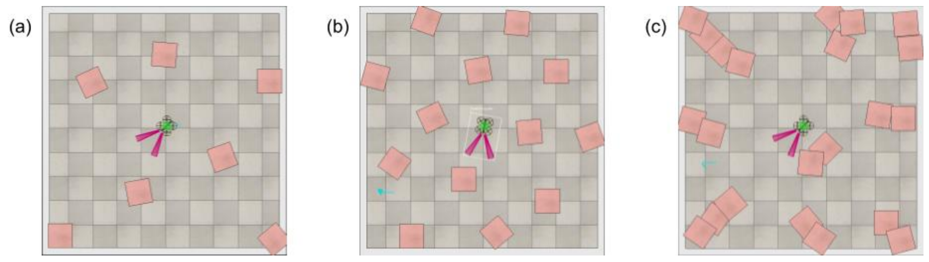

Fig. 3. Simulated environments. (a) Low density of obstacles. (b) High density obstacles. (c) High density of obstacles with corners and deadlocks.

The tests were conducted by setting a constant forward motion with a small random noise to the yaw control to enable an exploration behavior. Each test was run for approximately 30 minutes. Otherwise it was manually stopped if the drone got stuck. Figure 4 shows the simulation results of the drone using synaptic plasticity in the three environments. We initially set the synaptic weights of the recurrent network to $W_{11,22}$ = 0.5 and $W_{12,21}$ = -0.5 and changed them through the online learning rule described above. It can be seen that the drone could fly in all environments without getting stuck.

Figure 5 shows synaptic weight changes during the experiments. It can be seen that, in the environment with low dense obstacles (Fig. 4a), the weights changed slowly. This is because the drone did not frequently detect obstacles during the flight (Fig. 5a). In contrast, the weights increased quite fast (Fig. 5b) in the environment with high dense obstacles (Fig. 4b) and very fast (Fig. 5c) in the environment with high dense obstacles, corners, and deadlocks (Fig. 4c). In all cases, if no input was present, the synaptic weights started to decay slowly due to synaptic scaling which prevents the divergence of the weights. Table 1 shows three sets of average weights obtained from Fig. 5. These weights were used as the fixed weights ($W_{11,22,12,21}$) of the recurrent network to test the drone without online adaption.

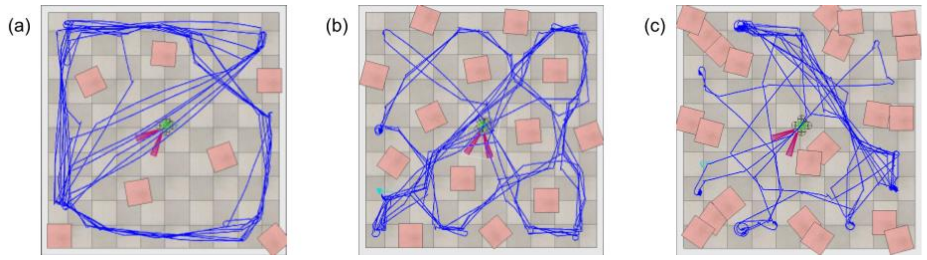

Fig. 4. (a),(b),(c) The trajectories of the drone in the three different environments or maps. Notice that no matter the density of objects, the drone is capable of moving around in all areas of the maps.

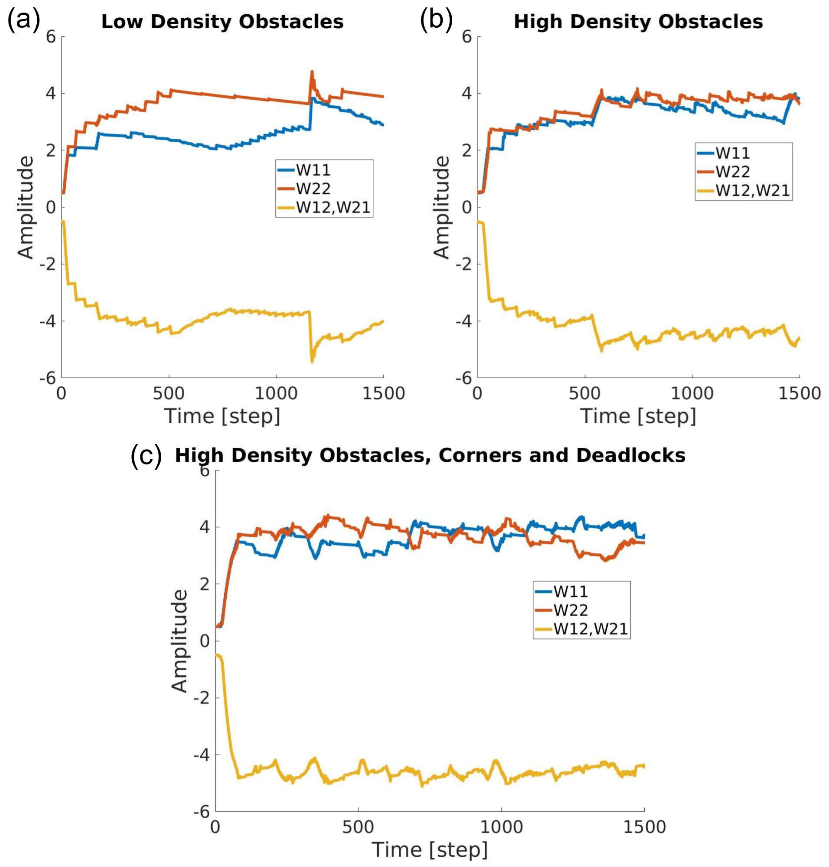

Fig. 5. The progress of weights during tests using synaptic plasticity. The weights change differently depending on the environment. (a) In the sparse environment, the weights changes slowly, up until a point (around 1200 time steps) where the drone enters one of the few narrow corner cases. (b),(c) When the density is increased, the weights start to change more rapidly.

Table 1. Weight averages

Environment	$W_{11,22}$	$W_{21,12}$
Low density of obstacles	3.1707	-3.9627
High density of obstacles	3.3689	-4.2286
High density of obstacles corners and deadlocks	3.6105	-4.5302

Figure 6 shows the simulation results of the drone with the three sets of the fixed average weights (Table 1) without online adaption in the corresponding environments. In the environments with high dense obstacles, corners, or/and deadlocks, the drone could not fully explore the areas. It basically got stuck in a corner (Fig. 6b) or a deadlock (Fig. 6c). Only in the environment with low dense obstacles, the drone can fly without getting stuck (Fig. 6a).

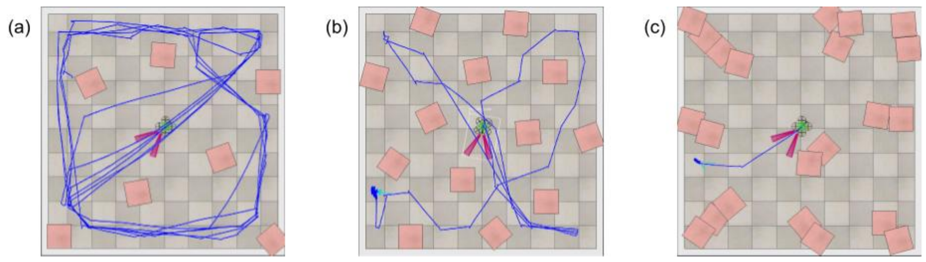

Fig. 6. (a),(b),(c) The trajectories of the drone without online adaptation in the three different environments. Notice that in the environment with low density of obstacles (a), the drone is capable of getting to most areas and avoids most obstacles. In the environment with high density obstacles (b), it ends up at a corner and cannot get away. In the environment with high density of obstacles, corners, and deadlocks (c), it almost immediately hits a corner where it gets stuck.

A quantitative test was made to further verify the improvements of the synaptic plasticity. The test was conducted by simulating each environment, with and without the synaptic plasticity. A run was deemed successful when the drone was able to fly for 5 minutes without either getting stuck or crashing. A total of 5 runs was done for each test and the results are showed in table 2.

The results of the experiments show that synaptic plasticity adds adaptation to a simple recurrent network, and thus enables the drone to fully explore in the environments with a varying density of obstacles, narrow corners, and deadlocks. It also allows the drone to autonomously avoid obstacles and escape from a corner and a deadlock. This effect can be compared when plasticity (see Figs. 4b, 4c) and non-plasticity (see Figs. 6b, 6c) are used. When observing the weights during flying in the sparse obstacle environment (Fig. 5a), there are small incremental changes for a long period. This is expected to happen when the drone reaches corners or obstacles where it only needs a larger turning angle to avoid getting

Table 2. Results of quantitative tests run on each environment. The runs are deemed successful if the drone is able to fly for 5 minutes without crashing or getting stuck.

Environment	Low density	High density	High density, sharp corners and deadlocks
Ratio of succesfull runs with synaptic plasticity	5/5	4/5	4/5
Ratio of succesfull runs without synaptic plasticity	5/5	3/5	0/5

stuck. However, at around the 1200 time steps (Fig. 5a) there is a large spike in the weight values indicating that it has reached a case where a large turning angle is required to escape from the situation. Without this adaptation, a large turning angle would be required at all times even though it is unnecessary and excessive in most cases. The results indicates that this method could very well be used for collision avoidance.

4 Discussion

Obstacle avoidance is an important basic function for autonomous navigation of drones. According to this, different obstacle avoidance techniques have been developed [4,5,6,16,17]. Many of them use visual cameras to extract information regarding obstacles. The information is used to find a suitable path to fly around the obstacles. Another way is to use Braitenberg's approach [9] which reactively controls an agent based on the activations of sensory inputs. For this approach, the agent will turn as long as it detects an obstacle. At a corner or deadlock, it might switch between turning left and right several times or it sometimes gets stuck. To deal with such a problem, Toutounji and Pasemann [18] proposed self-regulating neurons with short-term plasticity [19]. This allows an agent to avoid sharp corners. However, in this approach, different to our control proposed here, it requires multiple distance sensors for successful obstacle avoidance and the activity states have to be predefined.

In contrast to the previous approaches, we introduce neural control with synaptic plasticity which results in an adaptive obstacle avoidance strategy by using only two distance sensors. It is based on a two-neuron recurrent network with online correlation-based learning which adapts the synaptic weights of the network and thus changes the neural dynamics [13].

Recurrent networks have been proven useful in robot control [14, 15] and in this case enables the drone to successfully navigate in complex environments by avoiding collision and escaping from deadlocks or sharp corners. The simple recurrent network does not rely on global or memorized information, but, instead, learns and adapts to each situation as its arise.

In the future, we plan to transfer the neural control approach to a real drone. Accidental collisions can lead to damage to the drone. Thus, to ensure the safety in case the proposed control fails, propeller protection or prop guards will be installed on the drone, like the Lumenier Danaus drone[1]. This will allow for minor collisions where the control will still be able to learn and react before serious damage will occur.

5 Conclusion

In this study, we introduced neural control and synaptic plasticity for adaptive obstacle avoidance of an autonomous drone. 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. The core control mechanism is based on a two-neuron recurrent network. The network receives sensory inputs and translates them into proper motor commands through post-processing neural units for pitch and yaw control. Online correlation-based learning with synaptic scaling was employed for synaptic plasticity of the recurrent network. In principle, this online learning mechanism can increase or decrease the weights with respect to the interaction of the drone with its environment; thereby changing neural dynamics in the network (e.g., various hysteresis effects). This changing neural dynamics can be utilized for generating different turning angles with short-term memory when facing to different obstacles, corners, and deadlocks. This results in adaptively avoiding obstacles and escaping from corners/deadlocks. Furthermore, this also enables the drone to successfully explore and navigate in cluttered unknown environments. In the future, we will transfer the developed neural control approach to a real drone in order to test the adaptive obstacle avoidance behavior in a real environment.

Acknowledgments

This research was supported partly by Center for BioRobotics (CBR) at the University of Southern Denmark (SDU) and Startup Grant-IST Flagship research of Vidyasirimedhi Institute of Science & Technology (VISTEC).

References

1. Ashour, R., Taha, T., Mohamed, F.: Site inspection drone: A solution for inspecting and regulating construction sites. In: Proceedings of the IEEE 59th International Midwest Symposium on Circuits and Systems. (2016) 1-4
2. Sanfourche, M., Le Saux, B., Plyer, A., Le Besnerais, G.: Environment mapping & interpretation by drone. In: Joint Urban Remote Sensing Event. (2015) 1-4
3. Pobkrut, T., Eamsa-ard, T., Kerdcharoen, T.: Sensor drone for aerial odor mapping for agriculture and security services. In: Proceedings of the 13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology. (2016) 1-5

[1] http://www.lumenier.com/products/multirotors/danaus

4. Mori, T., Scherer, S.: First results in detecting and avoiding frontal obstacles from a monocular camera for micro unmanned aerial vehicles. In: Proceedings of the IEEE International Conference on Robotics and Automation. (2013) 1750-1757
5. Sedaghat-Pisheh, H., Amaury, R., Rivera, Biaz, S., Chapman, R.: Collision Avoidance Algorithms For Unmanned Aerial Vehicles Using Computer Vision. Journal of Computing Sciences in Colleges **33** (2017) 191-197
6. Magree, D., Mooney, J. G., Johnson, E. N.: Monocular visual mapping for obstacle avoidance on UAVs. In: Proceedings of the International Conference on Unmanned Aircraft Systems. (2013) 471-479
7. Rohmer, E., Singh, S. P. N., Freese, M.: V-REP: A versatile and scalable robot simulation framework. In: Proceedings of the IEEE International Conference on Intelligent Robots and Systems. (2013) 1321-1326
8. Grinke, E., Tetzlaff, C., Wörgtter, F., Manoonpong, P.:Synaptic plasticity in a recurrent neural network for versatile and adaptive behaviors of a walking robot. Frontiers in Neurorobotics **9** (2015) 11
9. Braitenberg, V.:Vehicles, Experiments in Synthetic Psychology. (1986)
10. Pasemann, F.: Discrete dynamics of two neuron networks. Open Systems and Information Dynamics **2** (1993) 49-66
11. Kolodziejski, C., Porr, B., and Wörgötter, F.: Mathematical properties of neuronal TD-rules and differential hebbian learning: a comparison. Biol. Cybern. **98** (2008) 259-272
12. Tetzlaff, C., Kolodziejski, C., Timme, M., and Wörgötter, F.: Analysis of synaptic scaling in combination with hebbian plasticity in several simple networks. Front. Comput. Neurosci. **6** (2012) 36
13. Neves, G., Cooke, S. F., & Bliss, T. V. P. I.: Synaptic plasticity, memory and the hippocampus: A neural network approach to causality. Nat. Rev. Neurosci. **9** (2008) 65-75
14. Huelse, M., and Pasemann, F.: Dynamical neural schmitt trigger for robot control. In: Proceedings of the International Conference on Artificial Neural Networks. (2002) 783788.
15. Pasemann, F., Huelse, M., and Zahedi, K.: Evolved neurodynamics for robot control. In: European Symposium on Artificial Neural Networks. (2003) 439444.
16. Zufferey, J.-C. & Floreano, D.: Fly-inspired visual steering of an ultralight indoor aircraft. In: Proceedings of the Transactions on Robotics. (2006) 137-146.
17. Franceschini, N., Ruffier, F., Serres, J. & Viollet, S.: Optic flow based visual guidance: from flying insects to miniature aerial vehicles. INTECH Open Access Publisher. (2009)
18. Toutounji, H., and Pasemann, F.: Behavior control in the sensorimotor loop with short-term synaptic dynamics induced by self-regulating neurons. Front. Neurorobot. **8** (2014) 19
19. Zahedi, K., and Pasemann, F.: Adaptive behavior control with self-regulating neurons, in 50 Years of Artificial Intelligence, eds Lungarella, M., Iida, F., Bongard, J., Pfeifer, R. (Berlin; Heidelberg: Springer), (2007) 196205.