A Comparative Study of Collision Avoidance Algorithms for Unmanned Aerial Vehicles: Performance and Robustness to Noise

Steven Roelofsen, Denis Gillet, and Alcherio Martinoli

Abstract Over the past years, the field of small unmanned aerial vehicles has grown significantly and several applications have appeared, requiring always more autonomous flight. An important remaining challenge for fully autonomous unmanned aerial vehicles is collision avoidance between aircraft. In this work, we will compare two collision avoidance algorithms in terms of performance and robustness to sensor noise. We will leverage both experiments with real vehicles and calibrated, realistic simulations to get an insight of the effect of noise on collision avoidance. Our results show that although algorithms that use velocity as input are better in minimizing velocity variation and generally produces more efficient trajectories, they are less robust to perception noise. On the other hand, position-based algorithms that typically generate slower and longer avoidance maneuvers, become competitive at high levels of sensor noise.

Key words: Collision avoidance, unmanned aerial vehicle, robustness

1 Introduction

Over the past years, the field of small Unmanned Aerial Vehicles (UAVs) has grown significantly and more applications appear as time goes on, ranging from surveil-

Steven Roelofsen and Alcherio Martinoli
Distributed Intelligent Systems and Algorithms Laboratory (DISAL), School of Architecture, Civil and Environmental Engineering, École Polytechnique Fédérale de Lausanne (EPFL), Station 2, 1015 Lausanne, Switzerland
e-mail: steven.roelofsen@epfl.ch, alcherio.martinoli@epfl.ch

Steven Roelofsen and Denis Gillet
Coordination and Interaction System Group (REACT), School of Engineering, École Polytechnique Fédérale de Lausanne (EPFL), Station 9, 1015 Lausanne, Switzerland
e-mail: steven.roelofsen@epfl.ch, denis.gillet@epfl.ch

lance and mapping to delivering of goods, and require always more autonomy for the UAVs. Up to date, research activities have focused mainly on self-localization [12], path planning [6] and navigation [14, 13]. Importantly, ensuring safety is one of the remaining challenges that needs to be overcome in order to achieve fully autonomous UAVs. More specifically, there is yet no Sense And Avoid (SAA) system reliable enough to allow for fully autonomous UAVs.

The robustness of a SAA system is not only determined by the individual robustness of its components (i.e., sensors, estimators or control algorithms) but also the interaction among them. Imperfections in one component of the system (e.g., sensing) can lower the performance and robustness of another one (e.g., control). An example of the effect of sensor imperfection having significant effect on the actuation is presented in [9].

Several collision avoidance algorithms exist in the literature and most of them fall in one of the two following categories. First, the algorithms based on Velocity Obstacle (VO) [5]. VO-based algorithms allow for collision avoidance while minimising the change in velocity and are applicable to a large set of systems [1]. Second, the algorithms derived from Potential Fields (PF) [7]. Those algorithms allow for more diverse behaviors than VO but usually at the expense of optimality in the velocity space (i.e. minimizing changes in velocity). The algorithms are also less generalizable than those of the VO class, but allow for integration of a large range of both actuation [8] and sensing constraints [10].

In this paper we present our effort to assess the robustness of different collision avoidance algorithms in presence of sensor noise by leveraging both real experiments and simulations. We performed real experiments for calibration and validation of simulation tools. We use high-fidelity simulation to go beyond our experimental facility's space limitations and systematically assess the performance avoidance algorithms in more generalizable scenarios.

1.1 Collision Avoidance Algorithms

In this work, two algorithms are investigated as case study: those that only use position information and those that use both position and velocity information of the other aircraft. So far we implemented two collision avoidance algorithms.

ORCA The Optimal Reciprocal Collision Avoidance [2] is based on the concept of Velocity Obstacle (VO) which is the set of all velocities that will lead to a collision. Based on the VO, ORCA builds one half plane of forbidden velocities per obstacle. Putting all the half-planes together results in a convex polygon of allowed velocities from which the optimal velocity (i.e., the closest to the desired velocity) is computed. ORCA also relies on reciprocity where each quadrotor only performs half of the velocity change that would be needed if they would not cooperate. ORCA has already been implemented on real quadrotors in [4].

FOVA The Field Of View Avoidance algorithm was presented in [10]. The algorithm uses a virtual potential field to navigate in its environment and smoothly

switches to a turning behavior when another quadrotor approaches. Because the motion of the quadrotor is constrained to always go forward by design, the quadrotors will move away from the collision point as soon as the quadrotor turned enough for the collision point to be out of its field of view (i.e., rear). The turn rate increases as the distance between the quadrotor decreases in order to guarantee avoidance. Once the collision point is out of the field of view of the quadrotor, the quadrotor goes back to its navigation function. A boundary zone makes for a smooth transition. Contrary to ORCA, this algorithm relies only on position information. It has been proven to avoid collision even with constraints on the sensor's field of view. FOVA has already been implemented on real quadrotors in [9].

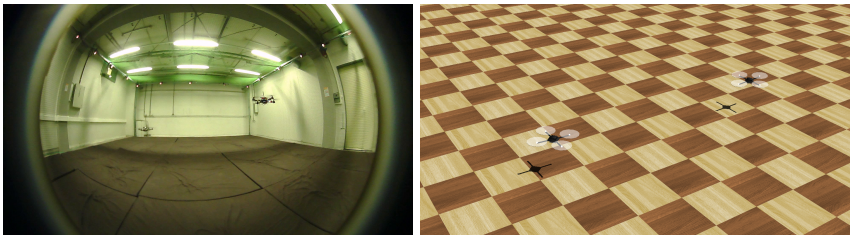
2 Technical Approach

In this work, the experiments are carried out with two quadrotors, highly agile platforms limiting the impact of the dynamical constraints of the vehicles on the results. To be as generic as possible, the sensory input is emulated, leveraging the millimetric precision of an external Motion Capture System (MCS). Such a solution also allows us to set the level of sensing noise with ease and precision.

To allow for fair comparison between the collision avoidance algorithms, the noise level needs to be equivalent, despite the fact that the two algorithm use different inputs. Both algorithms use position, but only ORCA uses velocity. To remain fair, the errors in position and velocity need to be linked to a single parameter. This is done by considering a sensor that only returns a position measurement that is affected by some Gaussian noise. The velocity is derived from the position using a Kalman filter. This way, both algorithms have access to the same information (filtered position information) but may not use all of the information available (FOVA does not use the velocity information that is contained in the position measurements).

For a meaningful comparison, the algorithms are implemented to take into consideration the effect of sensor uncertainty. All algorithms have been made noise-resistant similarly, using similar techniques to what is presented in [11]. More specifically, in both algorithms, the radius of the other UAV is increased by one standard deviation of the estimated position error given by the Kalman filter's covariance matrix. This increased radius is the only modification needed for the FOVA algorithm. For ORCA, the VO is also shifted by half a standard deviation of the velocity error as described in [11].

Finally, because the FOVA algorithm uses the sensor range as a key parameter, the sensing range is limited in the same way for both algorithms. This also guarantees that the quadrotors do only sense each other in a collision avoidance situation (i.e., not in standby mode). Additionally to a limited range, the field of view of the quadrotors for the FOVA algorithm is 220°, a parameter explained in [10]. This pa-



(a) Experimental facility (b) Two quadrotors in Webots

Fig. 1 On the left: photo of the experimental facility with two quadrotors flying (the second quadrotor is in the lower left corner of the flying arena). The bright redish dots all around the flying arena are the cameras of the MCS. On the right: two quadrotors simulated in the high-fidelity robotic simulator.

rameter is needed as it is key in the FOVA algorithm. The ORCA algorithm has no such FOV limitation.

2.1 Experimental Setup: Reality and Simulation

All algorithms are experimentally evaluated with off-the-shelf Hummingbird quadrotors from Ascending Technologies. The quadrotors are equipped with a Gumstix Airstorm computation module, providing a Linux-based operating system and a wireless communication link. The Kalman filter, the collision avoidance algorithms, and most of the control run on the embedded Gumstix. They are implemented using the Robotic Operating System (ROS) framework, leveraging the recording functionality of rosbag. Part of the low-level control (e.g., motor speed control) runs on dedicated, proprietary hardware of the Hummingbird quadrotor and is thus not directly accessible. We used a similar technique for estimating both the parameters of the proprietary, low-level control software as well as the physical parameters of our vehicles (e.g., thrust coefficient of the propellers). The Gumstix and the Hummingbird are interfaced using the *asctec_hl_interface* ROS package. The localization was provided by a MCS, providing millimetric precision pose information. The experiments were performed in a room of size $6\times8\times3$ *m*. A picture of the experimental facility with two quadrotors flying can be seen in Figure 1a.

Due to the limited size of our experimental facility, the number and variety of possible scenarios is limited. For a more thorough study, simulations are leveraged to provide a larger palette of scenarios. Our simulations are performed using Webots, a realistic sub-microscopic simulator. The simulator is able to interface with ROS, allowing it to run the same code as the one implemented on our quadrotors. A screenshot of the simulation environment can be seen in Figure 1b.

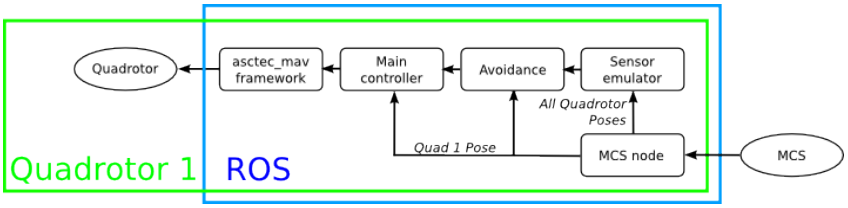


Fig. 2 Interaction between the different rosnodes on one of the two real quadrotors used in the experiments.

2.2 Implementation

Several software modules are necessary in order to perform the experiments presented in this paper: the MCS management software, a Kalman filter, the avoidance algorithm, and both the high-level and the low-level control algorithms. The MCS management and the low-level control software are proprietary modules and each one has a dedicated rosnode that allows to communicate with them. The Kalman filter node gets position data from the MCS (which is assumed to be noiseless) at 100 Hz, adds artificial Gaussian noise, and filter the data using a Kalman filter with optimized gain to obtain both a position and velocity estimate. Both position and velocity are forwarded to the collision avoidance algorithm, which computes a desired velocity and heading that both avoids collision and brings the aircraft to the goal. ORCA has been implemented leveraging the existing library [3], as the FOVA algorithm has been implemented by us based on the available literature. The desired velocity and heading are sent to the high-level control node, which translate then to desired thrust and orientation of the quadrotor. The control loop is performed at 20 Hz. Those commands are sent to the low-level control node that is in charge of setting the motors speeds of the quadrotor to ensure convergence to the desired state. The data flow in the quadrotor is shown in Figure 2.

To automatize the simulations, we added two more types of ROS nodes. Each simulated quadrotor has a Flight Manager node responsible to send commands to other rosnodes (e.g., command to take off the quadrotor). The Flight Manager nodes are supervised by a Scenario Manager common to all quadrotors; the Scenario Manager coordinates the quadrotor on how and when the maneuver should be performed. The MCS is replaced with a Webots supervisor that is able to retrieve the position of the simulated quadrotors and send the data to other rosnodes. The interactions between the rosnodes in simulation is presented in Figure 3.

2.3 Simulation Calibration

To obtain simulation results comparable to reality, the simulation needs to be calibrated. Physical parameters such as weight are directly measured on the aircraft.

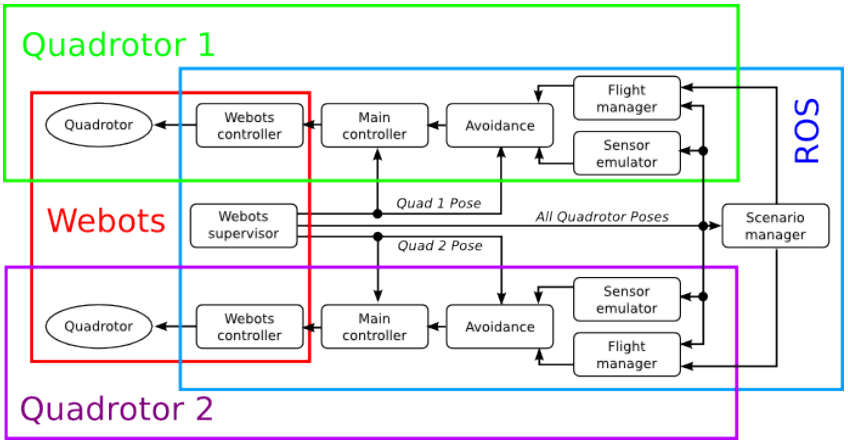


Fig. 3 Interaction between the different rosnodes in the simulation. The simulation framework uses the same avoidance and sensor emulation code that is implemented on the quadrotors.

Because the internal structure of the low-level control (responsible to control the motor speeds to bring the quadrotor to the correct attitude) is unknown to us, its parameters need also to be estimated. The unknown control scheme is assumed to be a PID controller and to have the structure described by Equations 1, 2 and 3:

$$M_x = K_a(\phi_t - \phi) - K_{da}\Omega_\phi \tag{1}$$

$$M_y = K_a(\theta_t - \theta) - K_{da}\Omega_\theta \tag{2}$$

$$M_z = K_{dy}(\Omega_{\psi,t} - \Omega_\psi) - K_{ddy}\frac{d\Omega_\psi}{dt} - K_{Iy}\int \Omega_\psi \tag{3}$$

with ϕ, θ and ψ being roll, pitch and yaw respectively, and Ω_ϕ, Ω_θ and Ω_ψ their respective angular rates. ϕ_t and θ_t are target roll and pitch respectively. $\Omega_{\psi,t}$ is a target yaw rate. M_x, M_y and M_z and the desired torques to apply along the x, y and z axis in order to get the quadrotor to the desired state. K_a, K_{da}, K_{dy}, K_{ddy} and K_{Iy} are the control parameters that need to be calibrated.

The five control parameters are optimized to minimize the difference between trajectories obtained from simulation and real experiments. The optimization was carried out using Particle Swarm Optimization (PSO), where each particle is a vector containing the five control parameters. Their fitnesses are defined as the RMS difference between the average trajectory obtained through real experiments and the average trajectory generated in simulation. The real experiment data set is composed of a quadrotor trying to follow a predefined trajectory eleven times. The simulation trajectory is obtained by simulating the quadrotor following the same predefined trajectory. The simulations are performed four times to average out the timing variability of the ROS framework (i.e., messages do not arrive with a deterministic timing).

While performing the experiments, it was observed that the quadrotors were affected by the airflow they were generating, resulting in a not perfectly steady flight.

To replicate this disturbance in simulation, we added a first order Gauss-Markov process on the thrust and torques in roll, pitch and yaw. The disturbances for each time step are described as:

$$T_d[n+1] = 0.995T_d[n] + 0.005(W(0,2)) \tag{4}$$
$$M_{x,d}[n+1] = 0.995M_{x,d}[n] + 0.005(W(0,0.12)) \tag{5}$$
$$M_{y,d}[n+1] = 0.995M_{y,d}[n] + 0.005(W(0,0.12)) \tag{6}$$
$$M_{z,d}[n+1] = 0.995M_{z,d}[n] + 0.005(W(0,0.02)) \tag{7}$$

with $W(\mu,\sigma)$ a Gaussian process of mean μ and standard deviation σ. The time step is 10 *ms*. The parameters have been set empirically so that the average acceleration between experimental and simulated data is similar. The validation on the simulator's faithfulness to reality is presented in Section 3.2.

2.4 Scenarios

The scenario for the real experimental setup is rather simple because of the limited space available; it is a head-on collision between two quadrotors, where each one starts on one side of the room and tries to get to other quadrotor's position. The initial position of the quadrotors are $[0,1.6,1]$ *m* and $[0,-1.6,1]$ *m*, the initial inter-vehicle distance is therefore of 3.2 *m*. The desired speed was set to 0.3 *m/s* for all algorithms. The experiments have been carried out with Gaussian noise levels of 0.01 *m*, 0.03 *m*, 0.1 *m*, 0.3 *m* and 1.0 *m* (standard deviation). Table 1 presents the error in position and velocity after the emulated measurements went through the Kalman filter. The values reported in Table 1 are directly used in the avoidance algorithms to make them resistant to noise. For simplicity, it is assumed that both algoritms know the radius, without considering noise, of the other quadrotor to be 0.35 *m*. For the real experiments, over 20 collision avoidance maneuvers have been carried out for each algorithm and noise level combination, for a total of more than 200 data points.

Table 1 Noise levels expressed with their standard deviations, and their corresponding errors in position and velocity obtained after the Kalman filter.

Noise level [m]	0.01	0.03	0.1	0.3	1
Position RMSE [m]	0.0071	0.0174	0.0448	0.1044	0.2614
Velocity RMSE [m/s]	0.0770	0.1052	0.1454	0.1936	0.2640

In simulation, we are not limited by spatial constraints, allowing for more extended scenarios. There are two main differences between simulated and real scenarios. First, the initial distance between the quadrotors is 8 *m* for a head-on configuration, or put differently, each quadrotor starts 4 *m* away from the collision point and try to reach a point 8 *m* in front of them. Second, the angle between the desired

directions of the quadrotors is changed. The initial and final positions are modified in order to have the lines defined by the two points to have the desired angle. The angles are 180° (head-on), 150°, 120°, 90°, 60°. For each possible angle, 20 collision avoidance maneuvers are performed. The plots presented in Figures 6, 7 and 8 report aggregated results for the five angles defined above. As a result, each curve of Figures 6, 7 and 8 represents 500 avoidance maneuvers for each algorithm (5 noise levels, each with 5 different angles, each angle and noise level combination repeated 20 times). Other parameters (e.g., desired speed of 0.3 *m/s*) are left the same, unless explicitly mentioned.

2.5 Metrics

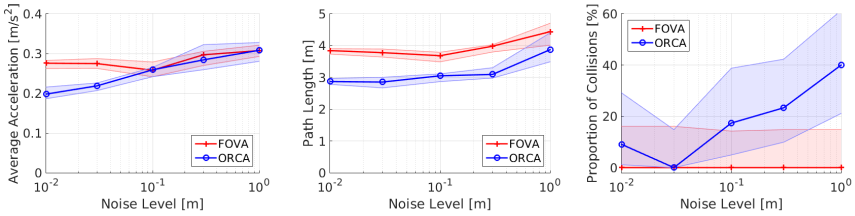
To compare the algorithms, we use three metrics: first, we compare the proportion of collisions as an indicator of safety; we consider a collision occurs when the horizontal distance between the two quadrotors is below twice their radius, or 0.7 *m*. Second, we also compare the path length as an approximation of the energy consumption and as a result an indication of the efficiency of the collision avoidance algorithms. Finally, we compare the average acceleration during the avoidance maneuver, as an indicator of overreaction due to noisy sensing. For all plots in Figures 4, 6, 7 and 8, the thick solid lines represent the median over the multiple runs of the same experiment. For the average acceleration and path length metrics, the color patches represent the interval between the upper and lower quartiles while for the proportion of collision they represent the 95% confidence intervals computed with the Clopper-Pearson method.

3 Results

In this section, we first present the results of the real experiments followed by simulation and a related discussion on how they compare with reality.

3.1 Real Experiment Results

Figure 4 shows the results obtained for the two algorithms when deployed on real robots. Clearly, the performance of ORCA degrades more as the sensor noise increases. The most significant difference is in the proportion of collisions. Where the FOVA algorithm never had a collision, ORCA has collisions for higher noise levels. The non-zero proportion of collisions for ORCA at the lowest noise level is due to actuation perturbations. In that case, ORCA avoids with such a small distance margin that any actuation perturbation (e.g., turbulent airflow) experienced



(a) Average acceleration (b) Path length (c) Proportion of collisions

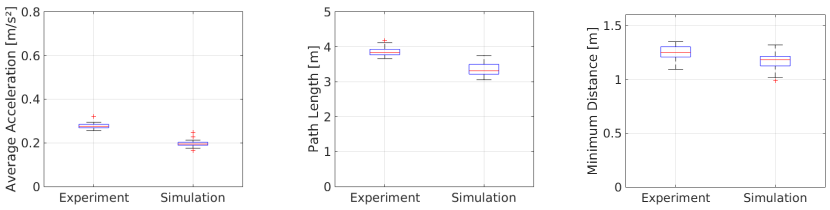
Fig. 4 The evolution of different metrics as function of sensor noise. The data were acquired using real quadrotors. a) The average acceleration for the two algorithms as function of sensor noise. b) The path length as function of the sensor noise. c) The proportion of collisions for different noise levels.

by the quadrotors can bring them below the collision distance. This is not the case for the FOVA algorithm, which, similarly to most of PF-based approaches, always avoids with a distance larger than the strict minimum. In order for ORCA to become totally noise-resistant, an additional margin on the obstacle radius needs to be considered. However, doing so will lower its performance in the other metrics. For the average acceleration metric, the FOVA algorithm starts with the highest acceleration but remains relatively constant compared to ORCA, the latter increasing as noise increases. Finally, both algorithms show longer path lengths as noise increases, partially due to the increase of the collision radius, correspondingly a priori implemented in the algorithms (see Section 2). Again, the difference of performance between position and velocity-based algorithms decreases as sensor noise increases.

3.2 Simulation Results

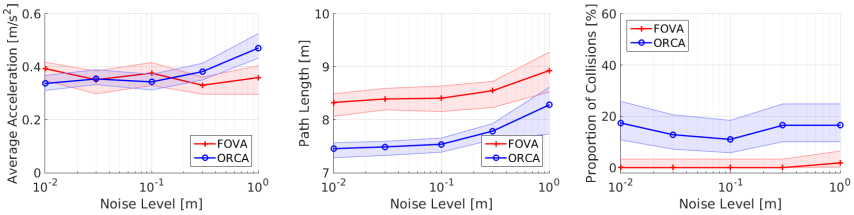
To validate the calibration of our simulator, we compared the performance of the three metrics described above for the FOVA algorithm in simulation and with real data, both with a Gaussian noise level of 0.01 m. The resulting data are shown in Figure 5. Contrary to all other simulations that use the extended scenario, the simulations carried out for Figure 5 faithfully reproduced the scenario used in reality (i.e., same initial position and heading as used for the real experiments). We see that for all the metrics considered (acceleration, path lenght, and minimum distance between the quadrotors) the simulated data assume smaller values than those gathered through experiments. This is probably due to the airflow generated by a quadrotor that tends to push away the other one, an effect that is, only roughly approximated in simulation (i.e. the approximation does not simulate that the airflow push the quadrotors away from each other).

Figure 6 presents the evolution of the three metrics for different sensory noise levels. For the FOVA metric, experimental and simulation data show similar trends. On the contrary, ORCA shows differences between simulation and experimental



(a) Average acceleration (b) Path length (c) Minimum distance

Fig. 5 Comparison between simulation and real experiments.

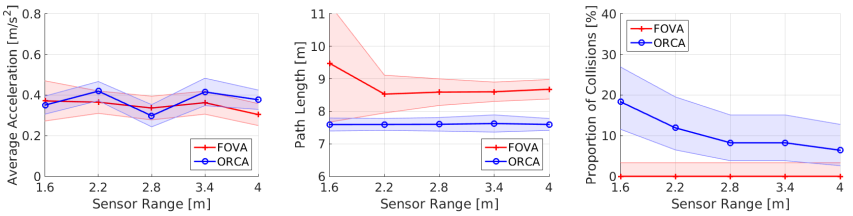


(a) Average acceleration (b) Path length (c) Proportion of collisions

Fig. 6 The evolution of different metrics as function of sensor noise. The data was obtained in simulation. a) The average acceleration for the two algorithms as function of sensor noise. b) The path length as function of the sensor noise. c) The proportion of collisions for different noise levels.

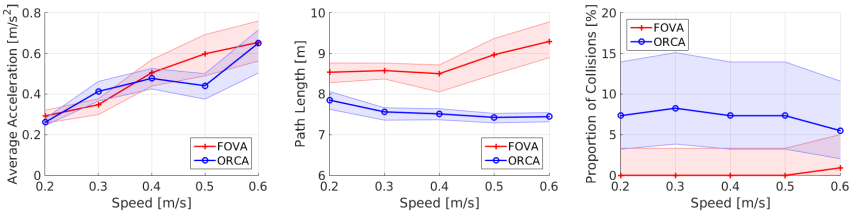
results, especially in the metric concerned with the proportion of collisions. The explanation for such differences is two-fold. First, the scenario is spatially more extended and does not only consists of head-on collisions. Head-on collision is harder to avoid as it is the configuration where the relative speed between the quadrotors is maximal, giving them less time to avoid and therefore explaining why experiments show a higher proportion of collision at high sensor noise levels. Second, the real quadrotors are pushing each other away with their airflow, which is not the case in simulation, explaining the flat curve over different levels of noise of Figure 6c.

The effect of sensor range on avoidance capability for both algorithms was also investigated in simulation (see Figure 7). For this investigation, the noise level was kept constant at 0.1 m. From Figure 7, the sensor range does not appear to have a significant effect on both algorithms, except for two aspects. First, the path length for the FOVA algorithm at small sensor range has a large variance. This is because with such a small sensor range, the FOVA algorithm has to perform very aggressive turns to avoid. Due to its inertia, the quadrotor overshoots its desired yaw angle, doing a full 360° turn. Because the FOVA avoids by only turning in one direction, both quadrotors spin quickly without being able to move away from each-other. When they eventually are able to separate (after some long time), all the spinning sums up to a long distance. The second notable effect of sensor range is that the proportion of collisions decreases as the sensor range increases for the ORCA algorithm. The reason is that the ORCA algorithm does not prefer a specific side on which to avoid; both quadrotors need thus to converge on which side the avoidance



(a) Average acceleration (b) Path length (c) Proportion of collisions

Fig. 7 The evolution of different metrics as function of sensor range. The data were obtained in simulation. a) The average acceleration for the two algorithms as function of sensor noise. b) The path length as function of the sensor noise. c) The proportion of collisions for different noise levels.



(a) Average acceleration (b) Path length (c) Proportion of collisions

Fig. 8 The evolution of different metrics as function of vehicles' maximum speed. The data were obtained in simulation. a) The average acceleration for the two algorithms as function of sensor noise. b) The path length as function of the sensor noise. c) The proportion of collisions for different noise levels.

will be done. With sensing noise, the ORCA algorithm will oscillate between sides, a phenomenon also known as reciprocal dance. A larger sensor range provides the ORCA algorithm more time to converge to a stable solution.

The effect of the quadrotor's maximal speed was also investigated. As for the sensor range, the simulations were performed with a noise level of 0.1 *m*. The results of the simulations are shown in Figure 8. The ORCA algorithm is not significantly affected an increased speed of the vehicles. However, it is notable an increase of the average acceleration for higher speeds due to the corresponding need for more aggressive maneuvers. The FOVA algorithm is more affected by an increase in speed. Besides an increase of the average acceleration for higher speed, the path length is also increased and even suffered from a collision when a maximal speed of 0.6 *m/s* was allowed. This is the result of the FOVA algorithm relying on turning, which is a less effective type of maneuver on a quadrotor when compared to ORCA's acceleration sideways .

4 Conclusion

In this work, we compare the performance of two collision avoidance algorithms in presence of noisy sensing. In particular, we evaluate the impact of the noise level, the sensing range, and the speed of the vehicles using a combination of calibrated simulation tools and real robot experiments. One algorithm is based on a Potential Field approach and only relies on positional data to perform avoidance, as the second algorithm based on Velocity Obstacles also needs velocity information. For fairness, the sensor only provides position measurements, velocity being acquired through Kalman filtered successive position measurements.

From this work, we draw two main lessons. First, the VO approach, due to its optimal nature, will show better performance in fuel-consumption-related metrics (in this case path length). Second, VO is significantly more collision prone because the algorithm tries to avoid as close as possible the obstacles on the vehicle's path. On the other hand PF-based algorithms stay at a safer distance from encountered obstacles.

Future work will include additional experimental scenarios and a thorough theoretical analysis. It will also consider the performance of the algorithms under specific sensor and actuator constraints (e.g., limited field of view, acceleration limits, different vehicle dynamics).

Acknowledgements This work has been financially supported by Honeywell, and has benefited of the administrative and technical coordination of the EPFL Transportation Center.

References

1. Bareiss, D., van den Berg, J.: Generalized reciprocal collision avoidance. The International Journal of Robotics Research **34**(12), 1501–1514 (2015)
2. van den Berg, J., Guy, S.J., Lin, M., Manocha, D.: Reciprocal n-body collision avoidance. In: Proc. Int. Symp. on Robotics Research 2009, ser. Springer Tracts in Advanced Robotics, pp. 3–19. Springer (2011)
3. van den Berg, J., J. Guy, S., Snape, J., C. Lin, M., Manocha, D.: RVO2 Library: Reciprocal Collision Avoidance for Real-Time Multi-Agent Simulation. `http://gamma.cs.unc.edu/RVO2/` (2008–2015)
4. Conroy, P., Bareiss, D., Beall, M., van den Berg, J.: 3-D reciprocal collision avoidance on physical quadrotor helicopters with on-board sensing for relative positioning. arXiv preprint arXiv:1411.3794 (2014)
5. Fiorini, P., Shiller, Z.: Motion planning in dynamic environments using velocity obstacles. The International Journal of Robotics Research **17**(7), 760–772 (1998)
6. Forster, C., Faessler, M., Fontana, F., Werlberger, M., Scaramuzza, D.: Continuous on-board monocular-vision-based elevation mapping applied to autonomous landing of micro aerial vehicles. In: IEEE International Conference on Robotics and Automation, pp. 111–118 (2015)
7. Kim, J.O., Khosla, P.K.: Real-time obstacle avoidance using harmonic potential functions. IEEE Transactions on Robotics and Automation **8**(3), 338–349 (1992)
8. Panyakeow, P., Mesbahi, M.: Decentralized deconfliction algorithms for unicycle UAVs. In: American Control Conference, pp. 794–799 (2010)

9. Roelofsen, S., Gillet, D., Martinoli, A.: Reciprocal collision avoidance for quadrotors using on-board visual detection. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 4810–4817 (2015)
10. Roelofsen, S., Martinoli, A., Gillet, D.: Distributed deconfliction algorithm for unmanned aerial vehicles with limited range and field of view sensors. In: American Control Conference, pp. 4356–4361 (2015)
11. Snape, J., van den Berg, J., Guy, S.J., Manocha, D.: The hybrid reciprocal velocity obstacle. IEEE Transactions on Robotics **27**(4), 696–706 (2011)
12. Weiss, S., Achtelik, M.W., Lynen, S., Chli, M., Siegwart, R.: Real-time Onboard Visual-Inertial State Estimation and Self-Calibration of MAVs in Unknown Environments. In: IEEE International Conference on Robotics and Automation, pp. 957–964 (2012)
13. Yang, S., Scherer, S.A., Schauwecker, K., Zell, A.: Autonomous landing of mavs on an arbitrarily textured landing site using onboard monocular vision. Journal of Intelligent & Robotic Systems pp. 27–43 (2013)
14. Zufferey, J., Beyeler, A., Floreano, D.: Autonomous flight at low altitude with vision-based collision avoidance and GPS-based path following. In: IEEE International Conference on Robotics and Automation, pp. 3329–3334 (2010)