2014 International Conference on Unmanned Aircraft Systems (ICUAS)
May 27-30, 2014. Orlando, FL, USA

Optimal Reciprocal Collision Avoidance with Mobile and Static Obstacles for Multi-UAV Systems

D. Alejo, J. A. Cobano, G. Heredia and A.Ollero

Abstract— **This paper presents a collision avoidance algorithm for multiple aerial vehicle systems to be applied in real-time. The proposed algorithm is based on the 3D-Optimal Reciprocal Collision Avoidance (ORCA) algorithm. Several improvements have been implemented such as considering dynamic constraints of the UAV model and static obstacles, so it can be used in realistic environments. The algorithm has been integrated in ROS framework and tested with up to eight Unmanned Aerial Vehicles (UAVs). Several simulations in a realistic environment have been performed, including long endurance cooperative missions. A comparison with the ORCA algorithm is presented to analyze the improvements done.**

I. INTRODUCTION

Cooperative missions of multi-vehicle systems have been extensively studied in the last years because of the advantages that present with respect to a single robot mission. The main advantages are noticeable in applications such as surveillance [1], structure assembly [2], fire detection and monitoring [3], etc. The coordination and collision avoidance play a crucial role in these applications as the number of robots increases. In these cases, trajectory planning algorithms should be used to compute the initial plans of the mission and reactive methods should compute solutions in real time whenever a potential collision is detected. Moreover, a good scalability of the methods used is essential.

The ARCAS FP7 European Project is developing a cooperative free-flying robot system for assembly and structure construction [2] (see Figure 1). The ARCAS system uses helicopters and quadrotors with multi-link manipulators for assembly tasks [4]. The aerial vehicles carry structure parts that will be assembled at the target destination. An important part in ARCAS is cooperative assembly planning and safe trajectory generation to perform the coordinated missions, assuring that neither the aerial vehicles nor the manipulators or the objects carried collide with each other.

The work presented in [5] presents a systems for assembly and structure construction with multiple aerial vehicles which automatically identifies conflicts among them. The system can be applied in real time but it computes collision-free trajectories whose quality improves when available computation time increases. Thus, a feasible but non-optimal initial solution is quickly computed. This system presents deficiencies for reactive collision avoidance. Therefore, a reactive method ensuring a better or optimal solution should be added.

Most of works published in the literature on trajectory planning are not good candidates for this kind of applications because a low computational load is needed to compute the solution in real time. These methods include Rapidly-exploring Random Trees (RRT) [6], evolutionary techniques [7][8][9], particle swarm optimization [10], multi-objective evolutionary algorithms [11], graph search like A* and D* [12], etc.

Fig. 1. Assembly and structure construction in the ARCAS project.

Many works have been also published related to the conflict resolution and detection problem but, in general, they present the same limitation [13] [14] [15][16].

Among the reactive collision avoidance methods, a decentralized reactive collision avoidance based on potential fields for multi-vehicle systems by changing the speed and using a planar unicycle model is used in [17]. Also, a decentralized control-free control by using dipolar navigation is presented in [18].

The method known as Reciprocal Collision Avoidance (RCA)[19] considers multiple vehicles navigating in a common environment and each vehicle takes half the responsibility of avoiding pairwise collisions. Most approaches solve the problem by considering zero order planning, that is, generate paths in the position space. On the other hand, RCA is formulated in the velocity space, so this is a first order planning procedure. Thus, RCA easily handles moving obstacles and also kinematic and dynamics constraints of the vehicle. The latter constraints can easily be taken into account by reducing the set of velocities that a vehicle can reach considering its current velocity. Different works on RCA have been applied to robots with holonomic dynamics [20], non-holonomic dynamics [21] and car-like dynamics [22].

The Optimal Reciprocal Collision Avoidance (ORCA) [20]

978-1-4799-2376-2/14/$31.00 ©2014 IEEE

method is based on the work presented in [23] and it improves the RCA behavior ensuring collision avoidance. Each robot avoids a collision by using information on the relative position and the relative velocity of the rest of robots. A centralized method by using ORCA for collision avoidance among multiple agents has been presented in [24].

The work presented in this paper has been carried out in the context of the ARCAS FP7 European project [2] to improve the system presented in [5]. Thus, a real time collision avoidance algorithm based on ORCA has been implemented in order to assure that neither the aerial vehicles nor the manipulators or the carried objects collide with each other nor with the static obstacles during the mission. This algorithm computes an optimal solution, so it considerably improves the quick solution computed by the system presented in [5].

The main novelty of the proposed algorithm with respect to the works already published is that it considers both mobile and static obstacles in a 3D environment. It is based on the RVO2-3D [25] implementation of 3D-ORCA algorithm with some improvements regarding to 3D static obstacles, dynamic constraints and ellipsoid agent shapes. The algorithm has been integrated in ROS (Robot Operating System) framework. Realistic simulations have been performed in different scenarios of the ARCAS project with a dynamic quadrotor model based on the implemented in the Hector-quadrotor ROS package [26].

The paper is organized into six sections. The description of the problem of Multi-UAV collision avoidance is presented in Section II. The proposed system is described in Section III. Section IV presents the simulations performed in ROS. Finally the conclusions are detailed in Section V.

II. Muti-UAV collision avoidance

This paper addresses the problem of collision avoidance with multiple UAVs to perform the coordinated missions proposed by the ARCAS project. Conflicts among UAVs or with static obstacles could be detected during the execution of the mission. In this case, a reactive method should compute an optimal solution by ensuring that the separation is greater than a given safety distance. It is assumed that velocity changes are allowed to solve the conflicts, that is, changes of the heading and speed of each vehicle. The information that the system needs in order to solve the problem is the following:

1) Initial spatial trajectory of each UAV.
2) Parameters of the model of each UAV
3) Location of each UAV

Let the system be composed of two robots R_A and R_B, which are located on $\boldsymbol{p}_A$ and $\boldsymbol{p}_B$ and with radius r_A and r_B (see Figure 2(a)). These robots are on collision course, that is, if none of their velocities is changed a collision will take place before time τ. The velocity obstacle, VO^{τ}, is the set of relative velocities, $\boldsymbol{v}$, that will lead them to collision before τ.

Consider $D(p,r)$ as an open disc of radius r centered at p:

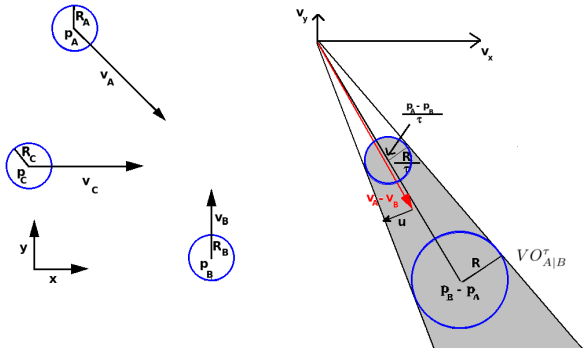

Fig. 2. On the left side, a scenario involving three robots (A, B and C) on collision course is represented. This scenario leads to $VO^{\tau}_{A|B}$ which is represented on the right side. The minimum reaction robots A and B have to perform in order to avoid their potential collision is represented by u.

$$D(\boldsymbol{p}, r) = \{\boldsymbol{q} | \|\boldsymbol{q} - \boldsymbol{p}\| < r\} \quad (1)$$

Then, $VO^{\tau}_{A|B}$, which is the velocity obstacle for A induced by B within time τ, can be defined as (see Figure 2(b)):

$$VO^{\tau}_{A|B} = \{\boldsymbol{v} | \exists t \in [0,\tau] :: t\boldsymbol{v} \in D(\boldsymbol{p}_B - \boldsymbol{p}_A, r_A + r_B)\} \quad (2)$$

Let $\boldsymbol{v}_A$ and $\boldsymbol{v}_B$ be the velocity of robots A and B, respectively. In order to get a collision-free situation, the relative velocity, $\boldsymbol{v}_A$-$\boldsymbol{v}_B$ or $\boldsymbol{v}_B$-$\boldsymbol{v}_A$, should be outside the velocity obstacle $VO^{\tau}_{A|B}$ or $VO^{\tau}_{B|A}$, respectively. There are a lot of pairs of sets of allowed velocities $\boldsymbol{v}_A$ and $\boldsymbol{v}_B$ but the pair that maximizes the allowed velocities close to preferred velocities, $\boldsymbol{v}_A^{pref}$ and $\boldsymbol{v}_B^{pref}$, should be chosen. These preferred velocities are given by the navigation modules of robots A and B, to encourage the minor deviation from the current trajectories. A reaction takes place when the current velocities and the preferred velocities have to be different. Let $\boldsymbol{u}$ be the vector from $\boldsymbol{v}_A^{pref}$-$\boldsymbol{v}_B^{pref}$ to the closest point on the boundary of the velocity obstacle (see Figure 2(b)). The minimum is represented by $\boldsymbol{u}$. As collaborative agents are considered, each one of them takes half of this reaction. Thus, the total reaction will be collision-free and the agents will not overreact. ORCA defines a half-space of collision-free velocities $ORCA^{\tau}_{A|B}$ as the set of velocities:

$$ORCA^{\tau}_{A|B} = \{\boldsymbol{v} | (\boldsymbol{v} - (\boldsymbol{v}_A + 0.5\boldsymbol{u})) \cdot \boldsymbol{u} \geq 0\} \quad (3)$$

Each agent computes the half-spaces of collision-free velocities taking into account the relative position and relative velocity of the rest of agents. Then, the intersection of all half-spaces is computed and a new collision-free velocity is selected that minimizes the next function (see Figure 3):

$$ORCA^{\tau}_{A} = D(0, v_A^{max}) \bigcap \left(\bigcap_{B \neq A} ORCA^{\tau}_{A|B} \right) \quad (4)$$

$$\boldsymbol{v}_A^{ORCA} = \min_{\boldsymbol{v} \in ORCA^{\tau}_A} \|\boldsymbol{v} - \boldsymbol{v}_A^{pref}\| \quad (5)$$

where v_A^{max} is the maximum velocity for the agent A. This problem can be efficiently solved by using quadratic programming.

Note that, in some densely packaged situations, this problem may become unfeasible. In these cases, a new problem is generated by relaxing the conditions of the ORCA planes by decreasing the time τ in which the collision is ensured. For more details, please refer to [20].

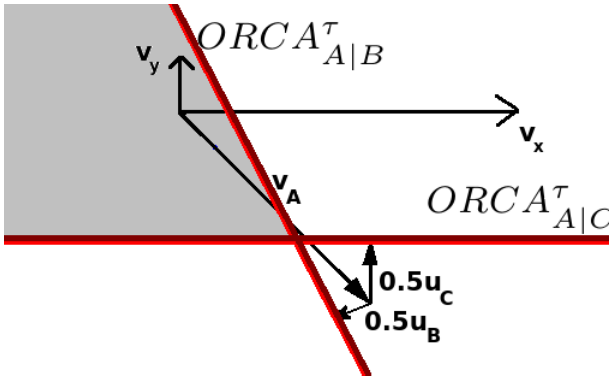

Fig. 3. The ORCA half-planes $ORCA^{\tau}_{A|B}$ and $ORCA^{\tau}_{A|C}$ that robots B, C induce in robot A are represented. The region of allowed velocities robot A can take is given by the intersection of these half-planes. This region is filled in light gray.

III. Method based on ORCA

Figure 4 shows the block diagram of the system considered in ARCAS project to ensure safe trajectories during the mission. This section describes the new added block, ORCA, to improve the system presented in [5]. The rest of the system is described in [5].

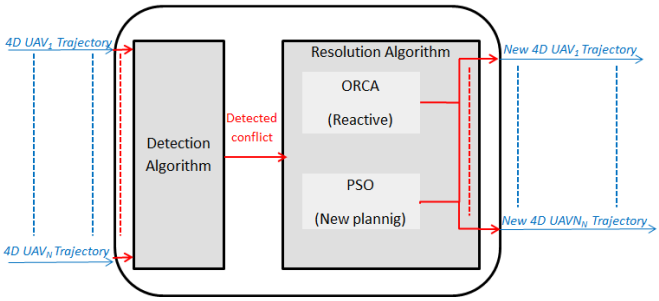

Fig. 4. Description of the proposed system with the addition of the new reactive block based on ORCA.

The proposed method is based on ORCA. It is a good candidate to efficiently carry out the coordinated mission of the ARCAS project. The experimental scenario of the project is the multi-UAV aerial testbed of the Center for Advanced Aerospace Technologies (CATEC) which is equipped with a VICON localization system that provides estimation of the position of the vehicles with a precision of few millimeters in real-time. Therefore, all the vehicles can obtain information of the position of the rest of vehicles and the minimum separation distance could be defined as a sphere around of the vehicle because the uncertainties are not relevant so they are not considered in this paper.

However, several improvements to the original ORCA algorithm are necessary in order to adapt them to the realistic environments as the ones proposed in the ARCAS project. In this section, the most important improvements are described.

A. Dynamics constraints handling

The proposed method considers dynamics constraints. We use a similar approach as the one proposed in [27]. Another constraint is added considering the current velocity of the vehicle $\boldsymbol{v}(t)$ and the maximum acceleration $\boldsymbol{a}_{max}$. Let T_s be the sample rate of the algorithm, then the inequation relating $\boldsymbol{v}^{ORCA}$ and $\boldsymbol{v}(t)$ is given by:

$$\|\boldsymbol{v}^{ORCA} - \boldsymbol{v}(t)\| \leq a_{max} T_s \tag{6}$$

This will force the velocity given by ORCA module to be reachable by the controller on-board the quadrotors.

B. Considering 3D obstacles

Another important requirement of the project is that the navigation is performed in scenarios with the presence of static obstacles. For example, the structure which is being built in the ARCAS project.

A 3D-map of the environment is assumed to be known. It has to be saved as a set of mesh files. Note that in real scenarios unexpected or unmodeled obstacles might appear. For this reason, this information could be enriched with the inclusion of vision or range sensors in order to detect them. However, this is beyond the scope of this paper.

The PQP library [28] has been used in order to calculate the distance between the position of the aerial vehicle and the static obstacles. This library not only checks for collision between two 3D meshes with triangular faces, but also returns the distance vector between these meshes, $\boldsymbol{d}$.

When applying the algorithm in a determinate time-step, only each obstacle's closest point to the agent is considered. This is done for two main reasons: the first is to decrease computational load and the second is to not over-constrain the QP problem. Once this closest point to an obstacle is calculated, its velocity obstacle is calculated by only considering this closest point. In consecutive computations this point seems to be moving slowly (see Figure 5), allowing the algorithm to smoothly react to the shape of the obstacle. Besides, it is a natural approach that resembles the behavior of humans when piloting a vehicle in a scenario with complex obstacles.

However, concave obstacles can make the closest point between the quadrotor and the obstacle not to be time continuous (see Figure 6). For this reason, we have to include only convex obstacles in the 3D meshes file that represents the static obstacles. There exist many methods for splitting concave obstacles into multiple convex obstacles, such as [29]. They can be applied offline in a preprocessing step to force the meshes to be composed of convex obstacles.

Note that the environment in the proposed application is dynamic, that is: new obstacles can be added. In our implementation, each different obstacle is saved as an independent 3D-model. Actually each convex part of each obstacle is saved as an independent 3D-model. These models can be deleted or added online.

Figure 5 represents an obstacle, O, and a robot, R_A, which lies in position p_A and has radius r_A. The velocity obstacle is a cone constructed by the union of the position of the robot and the closest point from the robot to the obstacle. In a similar development as indicated in [20], the velocity obstacle can be defined as:

$$VO^{\tau}_{A|O} = \{\boldsymbol{v} | \exists t \in (0, \tau] :: t\boldsymbol{D}(\boldsymbol{p}_O, r_A)\} \tag{7}$$

where $\boldsymbol{p}_O$ is the closest point from the robot A to the obstacle. Once the velocity obstacle has been constructed, the minimum reaction can be calculated as indicated in section II. Then, the ORCA half-plane is obtained taking into account that the robot should perform the whole reaction. This is indicated in the next equation.

$$ORCA^{\tau}_{A|O} = \{\boldsymbol{v} | (\boldsymbol{v} - (\boldsymbol{v}_A + \boldsymbol{u})) \cdot \boldsymbol{u} \geq 0\} \tag{8}$$

The last consideration is that the constraints due to static obstacles are not relaxed when an unfeasible problem is detected [20].

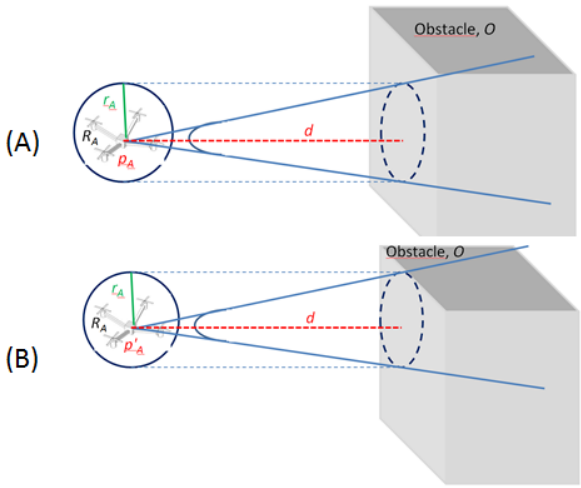

Fig. 5. Velocity obstacles induced by obstacle O to an agent in two different instants.

C. Safety region

The original 3D-ORCA algorithm assumes that the robots have spherical shapes. However, the shapes may vary among vehicles. For example, the shapes of the quadrotors that will be used in simulation is not covered uniformly with an sphere. In this case, the minimum horizontal separation distance should be greater than the vertical one, which is better approximated by an ellipsoid with the vertical semi-axis smaller than the other two. Therefore, a simple coordinate transform to the distances between vehicles and between vehicles and static obstacles is applied.

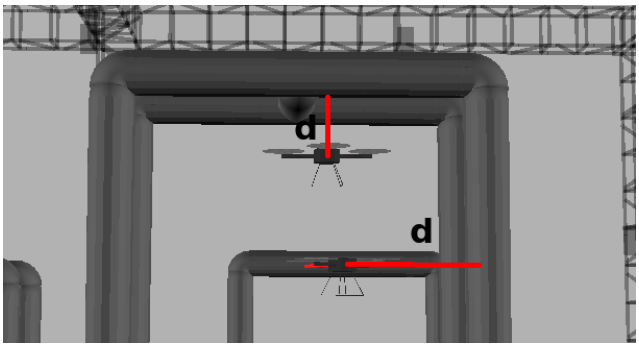

Fig. 6. Concave obstacles can break the time continuity of the distance vector. They have to be decomposed into several convex obstacles in a preprocessing step. d is the horizontal or vertical separation between a quadrotor and the closer static obstacle.

$$x' \leftarrow x \tag{9}$$

$$y' \leftarrow x \tag{10}$$

$$z' \leftarrow \alpha z \tag{11}$$

where $\alpha = \frac{r_{xy}}{r_z}$. Figure 7 represents the original ORCA volume and the proposed in this paper. Note that, the distances among the z' coordinate seem larger than the distances among z, so the real vehicle shape becomes an ellipsoid. These calculations will allow the quadrotor to get closer while performing vertical collision avoidance maneuvers than when performing horizontal maneuvers.

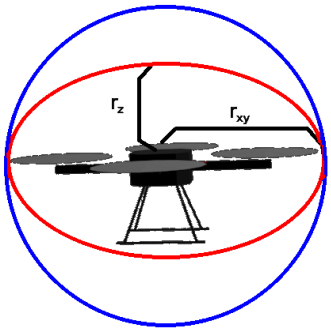

Fig. 7. Comparison of spherical shaped region of radius r_{xy} (in blue) and the proposed with different minimum horizontal , r_{xy}, and minimum vertical ,r_z (in red), separation distances.

Last but not least, the dimensions of the safety region can be reconfigured in real-time in order to model the UAV shape with the arm extended and contracted.

IV. Simulations

Many simulations have been carried out in a realistic environment that simulates the multi-UAV testbed of the CATEC where the experiments will be carried out. ROS framework

is used to test the proposed method. The dimensions of the scenario are $15x15x4m^3$ (see Figure 8).

The proposed algorithm has been run in a Toshiba^{TM}Satellite L-735 equipped with an Intel^{TM}Core i5 processor, 4 GB of RAM and NVIDIA^{TM}Corporation GT218M [GeForce 315M] graphical card. The operating system used was Kubuntu 12.04 Linux. The code was written in C++ language and integrated with ROS fuerte distribution. The dynamic quadrotor model used is based on the Hector-quadrotor ROS package [26].

It is important to point out that the testbed of CATEC is also integrated with ROS and that the same ROS node architecture is used for both simulation and experimentation. This makes the transition between simulation and experimentation straightforward as it diminishes the possible faults in the process.

One scenario with static obstacles with up to eight quadrotors is considered (see Figures 9). The simulations have been performed in the same machine. The videos show the collision avoidance in real time. The videos of the different experiments are available at `http://www.youtube.com/0grvc0`.

Fig. 8. Multi-UAV testbed placed in the CATEC facilities.

Next, the studies and results obtained in three different scenarios are presented. The safety distance parameters that have been considered are the following: radius of the ellipsoid surface surrounding the UAVs are $r_{xy} = 0.5m$ and $r_z = 0.3m$ (see section III). The separations among the center of the quadrotors can be easily calculated just by doubling these radiuses. The minimum separation between a quadrotor and obstacles is $r_{xy} = 0.6m$ and $r_z = 0.45$.

A. Safety

In this section we will study the separation between pairs of UAVs and between UAVs and the static obstacles in a simulation with three UAVs (Q1, Q2 and Q3). These UAVs fly the trajectories in which several head-to-head conflicts have to be solved (see Figure 10) .

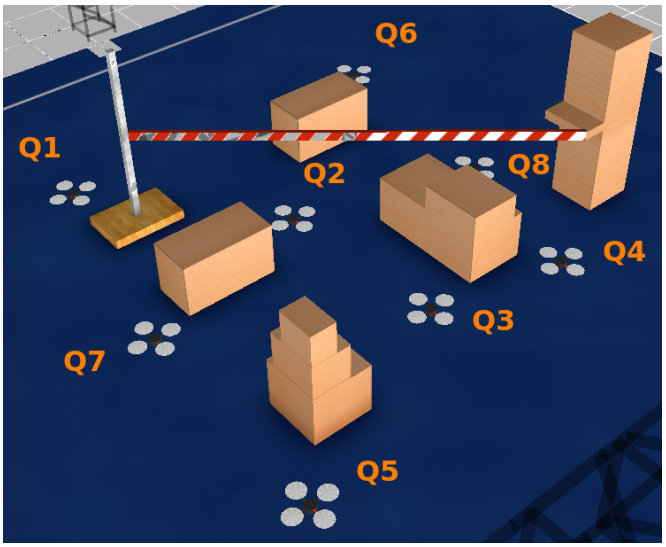

Fig. 9. Simulation scenario with up to eight quadrotors and static obstacles.

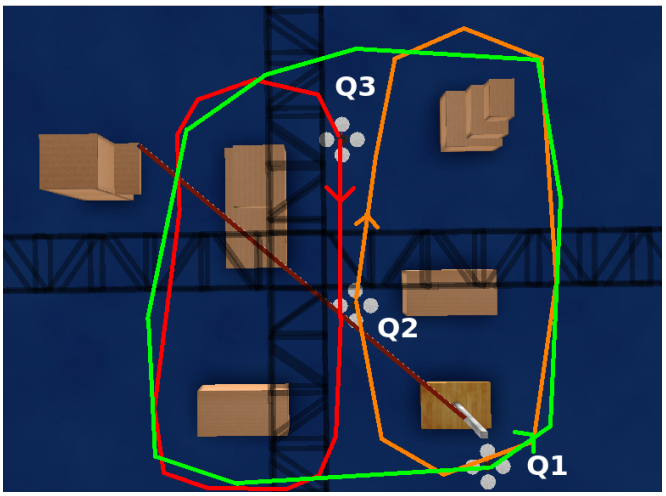

Fig. 10. Initial trajectories of quadrotors Q1 (green), Q2 (orange) and Q3 (red). Q4, Q7 and Q8 fly the same path as Q1 but starting from different positions. Q5 and Q6 follow the same paths as Q2 and Q3, respectively.

Figure 11 shows the vertical and horizontal separation between each pair of UAVs. All conflicts amongst UAVs have been successfully solved as the vertical and horizontal separations have not been violated at the same instants. In addition, the type of maneuver that the UAVs perform is automatically selected depending on the desired trajectories of the UAVs and the static obstacles surrounding them. Figure 12 shows the separation of each UAV to the closest obstacle. This separation also fulfills the requirements.

B. Scalability

Figure 13 shows the distribution of the computation time for calculating the collision-free ORCA velocity for one agent in the execution of simulations from 3 to 8 UAVs. Note that each agent only takes into account the agents that are closer than the neighboring distance. In this case, this

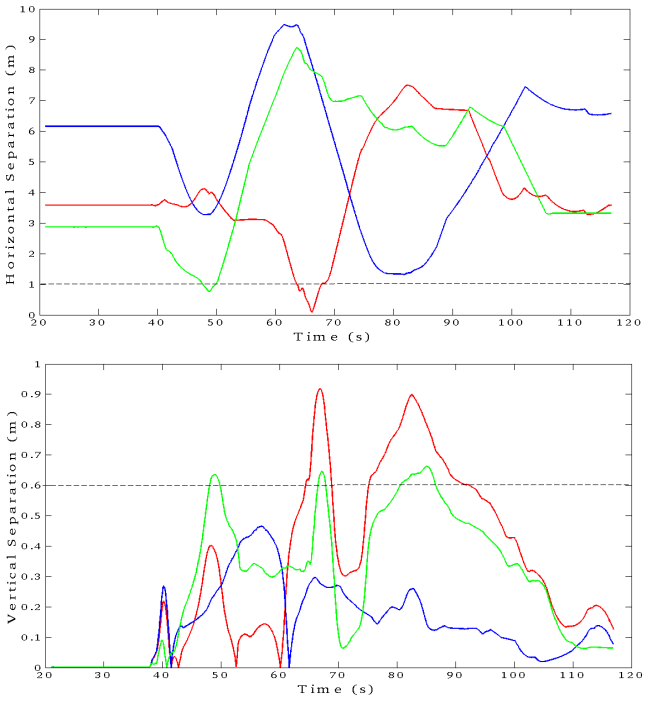

Fig. 11. Separation between each pair of UAVs. Red plot represents the separation between Q1 & Q2, blue plot represent the separation between Q1 & Q3 and green plot represents the separation between Q2 & Q3. The plot on the top represents the horizontal separation and the plot on the bottom represents the vertical separation. Dashed lines represent the minimum allowed separations.

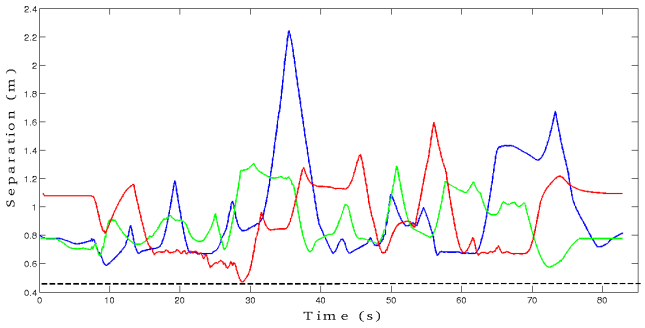

Fig. 12. Separation between static obstacles and Q1(blue), Q2 (green) and Q3 (red). The minimum separation is represented in dashed line.

distance was set to $4m$. Note that the preprocessing step is done offline, so its execution time has not been taken into account because it does not affect the real-time performance of the system.

These results show that the computation time in calculating the ORCA velocity for each agent was far below $1ms$ in more than the 97% of the cases. Moreover, the computation time grows very slowly with number of UAVs: it was confined between 0.3 and $0.5ms$ in the case of 3 UAVs and between 0.4 and $0.6ms$ with 8 UAVs.

The Collision Avoidance module was computing collision-free velocities at a rate of $20Hz$ for each quadrotor, that is one for each $50ms$. This has allowed us to perform

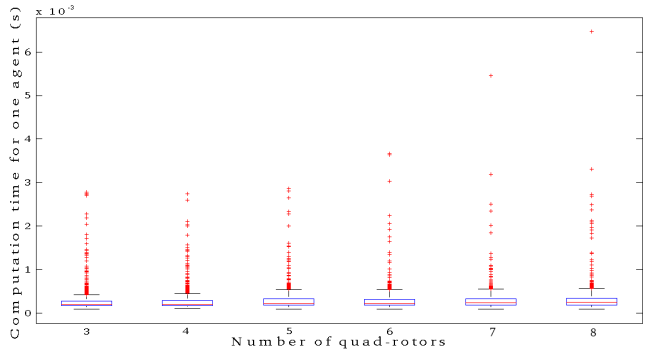

Fig. 13. Distribution of the execution computation time in proposed algorithm for one agent with the number of UAVs in the system. The median of each distribution is indicated in red, the blue box represent the 25th and 75th percentiles and the 3rd and 97th percentiles are indicated in black. Red marks represent the outliers.

simulations with up 8 quadrotors in real time and in the same machine. Taking into account these results, it is expected that we can raise this number to more than 50 without experiencing flaws. Furthermore, the computations can be easily distributed among several PCs thanks to the ROS integration. Therefore, there exist no limits in the number of robots this method can handle in theory.

C. Stability and reliability

Finally, a ten minutes long simulation in order to test the stability and reliability of the proposed system has been carried out. Main results of the simulation are detailed in table I. Minimum separation distance between pairs of vehicles and between pairs the vehicles and the scenario is listed. These minimum separation distances fulfill the requirements imposed in the first part of this section IV demonstrating that even in long simulations the systems keeps being stable.

The maneuvers this algorithm has performed throughout the simulation have been also listed. We consider that a vehicle is performing a maneuver whenever its desired velocity and the ORCA velocity are distant enough. Mathematically this can be expressed as:

$$\|\boldsymbol{v}^{pref} - \boldsymbol{v}^{ORCA}\| \geq v_{thres} \tag{12}$$

where v_{thres} is set to $0.1\frac{m}{s}$. In this simulation, forty-one maneuvers involving from one to three quadrotors have been performed. The mean time a maneuver has lasted is $6.56s$.

V. Conclusions

A new real-time collision avoidance method based on 3D-ORCA algorithm is presented in this paper. It is added to the system presented in [5]. Several improvements have been implemented in order to make the algorithm work in realistic scenarios considering the dynamics of the quadrotors. Coordinated missions have been considered and the computational load for each single agent is below $1ms$. This shows that the algorithm can be run in real time in the same computer even in simulations with great number of UAVs.

TABLE I

RESULTS OBTAINED IN THE RELIABILITY SIMULATION.

Characteristics	Quad-rotors	Value
# maneuvers	-	41
Avg Duration	-	$6.56s$
Avg Vehicles	-	1.75
Separation w. obstacles	QR1	0.60
	QR2	$0.62m$
	QR3	$0.53m$
Vertical separation	QR1-QR2	$0.77m$
	QR1-QR3	$0.84m$
	QR2-QR3	$0.75m$

The main novelty with respect to the works published [20][21] [22] is that this considers 3D static obstacles. This was necessary in order to apply it in realistic environment as the proposed in ARCAS project [2]. Moreover, the algorithm has been integrated in ROS framework with the same ROS node architecture used in the multi-UAV testbed of CATEC. Thus, it is easy to perform real experiments in this testbed.

Several simulations demonstrate both the safety and reliability of the system. The vehicles automatically perform the best maneuvers taking into account their relative motion and the obstacles surrounding them.

Also, scalability is analyzed considering up to eight vehicles. The distribution of the computation time of proposed algorithm is shown as the number of vehicles in the system increases. Results show that more than 97% of the calculations were carried out in less than a millisecond. This allows the proposed method to handle with systems composed of tenths of vehicles in the same machine. Furthermore, calculations can be distributed among several machines thanks to the ROS integration, so there are no theoretical limits in the size of the system this algorithm can be applied in.

Future efforts will include real experiments with up to 4 Hummingbird quadrotors that will be performed in the multi-UAV testbed of CATEC in March 2014. Also, it will be interesting to include non-holonomic constraints in order to apply this algorithm to fixed-wing UAVs.

REFERENCES

[1] R. W. Beard, T. McLain, D. Nelson, D. Kingston, and D. Johanson, "Decentralized cooperative aerial surveillance using fixed-wing miniature UAVs ," *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1306–1324, 2006.

[2] A. Ollero, "Aerial robotics cooperative assembly system (ARCAS): First results," in *Aerial Physically Acting Robots (AIRPHARO) workshop, IROS 2012*, Vilamoura, Portugal, October 7-12 2012.

[3] L. Merino, F. Caballero, J. M. de Dios, I. Maza, and A. Ollero, "An unmanned aircraft system for automatic forest fire monitoring and measurement," *Journal of Intelligent and Robotic Systems*, vol. 65, no. 1, pp. 533–548, 2012. [Online]. Available: http://dx.doi.org/10.1007/s10846-011-9560-x

[4] A. E. Jimenez-Cano, J. Martin, G. Heredia, R. Cano, and A. Ollero, "Control of an aerial robot with multi-link arm for assembly tasks," in *IEEE Int. Conf. Robotics and Automation (ICRA)*, Karlsruhe, Germany, MAy 6-10 2013.

[5] D. Alejo, J. A. Cobano, G. Heredia, and A. Ollero, "Collision-free 4D trajectory planning in Unmanned Aerial Vehicles for assembly and structure construction," *Journal of Intelligent and Robotic Systems*, vol. 73, pp. 783–795, 2014.

[6] S. M. Lavalle, J. J. Kuffner, and Jr., "Rapidly-Exploring Random Trees: Progress and Prospects," in *Algorithmic and Computational Robotics: New Directions*, 2000, pp. 293–308.

[7] J. A. Cobano, R. Conde, D. Alejo, and A. Ollero, "Path planning based on genetic algorithms and the monte-carlo method to avoid aerial vehicle collisions under uncertainties," in *Proc. IEEE Int Robotics and Automation (ICRA) Conf*, 2011, pp. 4429–4434.

[8] R. Vivona, D. Karr, and D.Roscoe, "Pattern-based genetic algorithm for airborne conflict resolution," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, Keystone, Colorado, August 2006.

[9] N. Durand and J. Alliot, "Optimization resolution of en route conflicts," in *First USA/Europe Air Traffic Management Research and Development Seminar (ATM1997)*, Saclay (France), 17-19 June 1997.

[10] M. Pontani and B. A. Conway, "Particle Swarm Optimization Applied to Space Trajectories," *Journal of Guidance Control and Dynamics*, vol. 33, pp. 1429–1441, 2010.

[11] A. J. Pohl and G. B. Lamont, "Multi-objective UAV mission planning using evolutionary computation," in *Winter Simulation Conference*, 2008, pp. 1268–1279. [Online]. Available: http://dx.doi.org/10.1109/WSC.2008.4736199

[12] A. Stentz and I. C. Mellon, "Optimal and Efficient Path Planning for Unknown and Dynamic Environments," *International Journal of Robotics and Automation*, vol. 10, pp. 89–100, 1993.

[13] A. Vela, S. Solak, W. Singhose, and J.-P. Clarke, "A mixed integer program for flight-level assignment and speed control for conflict resolution," in *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, dec. 2009, pp. 5219 –5226.

[14] I. Hwang and C. J. Tomlin, "C.: Protocol-based conflict resolution for air traffic control. air traffic control quarterly 15(1," 2007.

[15] N. Durand and J. Alliot, "Ant colony optimization for air traffic conflict resolution," in *Proceedings of the Eighth USA/Europe Air Traffic Management Research and Development Seminar (ATM2009)*, Napa, (CA, USA), 2009.

[16] J. A. Cobano, D. Alejo, A. Ollero, and A. Viguria, "Efficient conflict resolution method in air traffic management based on the speed assignment," in *Proceedings of the 2nd International Conference on Application and Theory of Automation in Command and Control Systems*, ser. ATACCS '12. Toulouse, France, France: IRIT Press, 2012, pp. 54–61. [Online]. Available: http://dl.acm.org/citation.cfm?id=2325676.2325684

[17] E. Lalish and K. A. Morgansen, in *Proceedings of the 47nd IEEE Conference on Decisionn and Control*, Cancun, Mexico.

[18] G. Roussos, G. Chaloulos, K. Kyriakopoulos, and J. Lygeros, in *Proceedings of the 47nd IEEE Conference on Decisionn and Control*, Cancun, Mexico.

[19] J. P. van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation." in *ICRA*. IEEE, 2008, pp. 1928–1935. [Online]. Available: http://dblp.uni-trier.de/db/conf/icra/icra2008.html#BergLM08

[20] J. van den Berg, S. J. Guy, M. C. Lin, and D. Manocha, "Reciprocal n-body Collision Avoidance," in *INTERNATIONAL SYMPOSIUM ON ROBOTICS RESEARCH*, 2009.

[21] J. Alonso-Mora, A. Breitenmoser, M. Rufli, P. Beardsley, and R. Siegwart, "Optimal Reciprocal Collision Avoidance for Multiple Non-Holonomic Robots," in *Proc. of the 10th International Symposium on Distributed Autonomous Robotic Systems (DARS)*, A. Martinoli and F. Mondada, Eds. Berlin: Springer Press, November 2010.

[22] J. Alonso-Mora, A. Breitenmoser, P. Beardsley, and P. Siegwart, in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Saint Paul, MN, USA.

[23] P. Fiorini and Z. Shiller, "Motion Planning in Dynamic Environments using Velocity Obstacles," *International Journal of Robotics Research*, vol. 17, pp. 760–772, 1998.

[24] J. Alonso-Mora, M. Rufli, P. Siegwart, and P. Beardsley, in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany.

[25] J. van der Berg, S. J. Guy, J. Snape, M. C. Lin, and D. Manocha, "RVO2 Library: Reciprocal Collision Avoidance for Real-Time Multi-Agent Simulation." [Online]. Available: http://gamma.cs.unc.edu/RVO2

[26] J. Meyer, "Hector quadrotor ros package website," 2014, [Accessed 5-February-2014]. [Online]. Available: http://wiki.ros.org/hector_quadrotor

[27] J. van den Berg, J. Snape, S. Guy, and D. Manocha, "Reciprocal collision avoidance with acceleration-velocity obstacles," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, May 2011, pp. 3475–3482.

[28] M. C. L. Eric Larsen, Stefan Gottschalk and D. Manocha., "Proximity query package website," 2014, [Accessed 5-February-2014]. [Online]. Available: http://gamma.cs.unc.edu/SSV/

[29] M. Ghosh, N. M. Amato, Y. Lu, and J.-M. Lien, "Fast approximate convex decomposition using relative concavity," *Computer-Aided Design*, in press 2012, also appear in Proc. of Symposium on Solid and Physical Modeling, Dijon, France, Oct. 2012.