Using Minimal Communication to Improve Decentralized Conflict Resolution for Non-holonomic Vehicles

Athanasios Krontiris and Kostas E. Bekris

***Abstract*— This work considers the problem of decentralized coordination between multiple non-holonomic vehicles, each navigating to a specified goal. By augmenting the Generalized Roundabout Policy (GRP), which guarantees collision avoidance, this paper improves the performance and liveness characteristics for such problems. These gains are achieved by integrating a second hybrid policy with GRP that updates the desired direction for each vehicle based on a dynamic priority scheme. In this scheme, minimalistic communication between vehicles is employed, such that information is periodically exchanged when changes in the high-level operating mode or prioritization occur. This information exchange is taking place only locally and data are exchanged only between neighboring vehicles. Additionally, each agent selects a control using only this local information and rules established by the two underlying hybrid automata. The proposed technique scales well due to its decentralized nature and as the computational complexity depends on the maximum number of vehicles in communication range for a vehicle. This paper presents simulations which show that the proposed approach can solve problems faster than using GRP alone, as well as solve instances in which GRP fails to find a solution, with minimal communication and computational overhead.**

I. Introduction

Many different application areas can utilize decentralized control solutions for the traffic management of multiple vehicles. Examples include coordinated control of robots, such as UAVs, industrial automation systems, as well as collision avoidance assistance for manually controlled systems in congested regions, e.g., aircraft approaching airports, marine vessels at harbors, etc. Such problems introduce important challenges, including the need to design decentralized control laws with safety and liveness guarantees.

This paper focuses on providing collision avoidance and completeness guarantees for multiple autonomous vehicles operating in the same obstacle-free space. The vehicles have to move from distinct, non-overlapping start configurations to their goals. These goals may be potentially overlapping but once a vehicle reaches its goal, it is no longer part of the coordination problem. For instance, consider a scenario where multiple aircraft coordinate when flying at constant altitude. When an airplane reaches the vicinity of an airport, it initiates its landing maneuver and no longer constrains the remaining aircraft.

This work is supported by NSF CNS 0932423. Any opinions and conclusions expressed in this paper are those of the authors and do not necessarily reflect the views of the sponsor.

A. Krontiris and K. E. Bekris are with the Department of Computer Science and Engineering, University of Nevada, Reno, 1664 N. Virginia St., MS 171, Reno, NV, 89557. Email: tdk.krontir@gmail.com, bekris@cse.unr.edu.

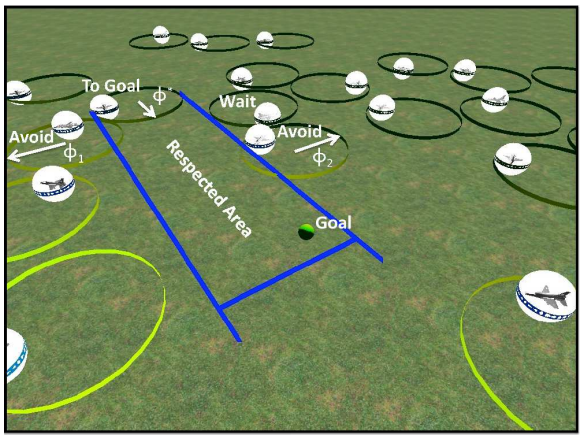

Fig. 1. The different modes of the vehicles in the proposed approach and their desired direction of motion.

One requirement is to consider vehicles with non-holonomic constraints and the inability to stop, such as aircraft. This implies that safety must be guaranteed despite these motion constraints. Similar to related work [1], this paper considers a simplified but realistic kinematic model of aircraft dynamics [2], where the vehicle moves with constant speed subject to curvature bounds.

While it is possible to define complete algorithms for centralized traffic management, such approaches are computationally expensive, and require a central location where information is processed. This implies a single point of failure for the entire system and poor scalability. In a decentralized algorithm, each vehicle plans its own trajectory based on local information. Such approaches have better scalability and faster reactions to unexpected changes. Thus, they are desirable in real-world applications. Proving liveness, however, is more difficult when using decentralized protocols.

In this work, each vehicle uses only information from its local neighborhood. The information corresponds either to the observable position and orientation of nearby vehicles, or the communicated priority and destination of its neighbors. Modes corresponding to a hybrid automaton are employed that define the vehicle's high-level behavior, such as moving towards its goal, waiting for a neighbor to pass or avoiding another vehicle as in the example of Fig. 1. A dynamic priority scheme is used to guarantee that at least one vehicle is always making progress towards its goal. All vehicles make decisions based on the combination of two control policies, one corresponding to the above mentioned hybrid automaton and the other corresponding to an existing safe decentralized controller for non-holonomic vehicles [1].

A. Background

Control-based Approaches: Conflict avoidance problems between multiple aircraft are good benchmarks for studying the control and verification of hybrid systems [3]. The literature on air-traffic control has provided roundabout policies with safety guarantees [4]. A Generalized Roundabout Policy (GRP) was proposed for conflict resolution in multi-vehicle systems, which focused on decentralized, communication-less solutions using a hybrid automaton for systems that cannot stop, such as aircraft [1]. Beyond safety, this work also studies liveness, which is achieved for goal configurations that satisfy a sparseness property. Liveness is not guaranteed in the general case. Hybrid control has also been integrated with prioritized schemes to resolve local conflicts but without liveness guarantees [5]. Navigation functions [6] can be used for decentralized feedback control strategies [7], including techniques for aircraft-like vehicles [8]. There are also methods that utilize a receding horizon approach [9] or solutions to mixed integer linear programs [10].

Reactive Approaches: Reactive obstacle avoidance methods, such as the Dynamic Window Approach [11] and Velocity Obstacles (VOs) [12], enable robots to avoid collisions in unknown or dynamic environments. Reciprocal Velocity Obstacles (RVOs) [13] extend VOs in the case of multiple agents that simultaneously avoid one another. RVOs scale to large numbers of agents and achieve collision avoidance without communication. This method has also been extended to car-like systems [14]. Safety is not guaranteed for kinematically constrained systems and liveness has not been shown yet for the approach. A related control-based method [15] operates on a model of a planar unicycle.

Planning-based Approaches: Multi-robot planning can be addressed either with coupled or decoupled approaches. *Coupled* approaches guarantee completeness but have an exponential dependence on the number of robots. *Decoupled* approaches allow robots to compute their own paths and then resolve conflicts. They are typically incomplete but orders of magnitude faster than coupled alternatives. In prioritized schemes, high priority robots are considered moving obstacles that must be avoided by lower priority ones [16]. The choice of priorities is important and searching the space of priorities can improve performance [17]. Another decoupled approach tunes the velocities of robots along precomputed paths [18], [19]. Coordination graphs can provide safety and reduce the occurrence of deadlocks [20], [21]. Multi-robot planning has also been cast as a search problem in a dynamically changing maze [22]. Hybrid approaches combine both coupled and decoupled planning [23]. For instance, robot networks can opportunistically employ coupled planners when multiple robots define a connected component in the communication network [24].

A variety of AI methods can be used for conflict resolution, such as auctions [25], or distributed constraint satisfaction/optimization [26], [27]. These approaches can typically provide asynchronous coordination mechanisms and dynamic priority schemes that assist in conflict resolution.

B. Contribution

This work builds upon the Generalized Roundabout Policy [1] and provides an extension called Extended Roundabout Policy (ERP), which utilizes ideas from prioritized schemes. ERP provides stronger liveness guarantees, as well as improved performance. Although ERP, in contrast to GRP, requires agents to communicate, the amount of information exchanged is minimal. Results show that for experiments where aircraft recompute controls 100Hz and have a speed of 250 mph, an aircraft transmits short messages at a rate of 3 cycles per 1000 (i.e., 3 messages per 10 seconds). This is a low communication overhead relative to the performance improvements observed. The simulated experiments confirm that ERP can solve problems which GRP cannot. Furthermore, in the same amount of time, ERP can bring more vehicles to the goal than the original method.

To method introduces a second hybrid automaton on top of the original one. The original GRP automaton attempts to move the vehicle towards a direction ϕ, the direction to the goal, while figuring out the valid set of directions that the vehicle can move towards without colliding with its neighbors. The newly introduced automaton in ERP changes the desired direction ϕ, causing vehicles to be in three possible modes: (a) moving towards the goal, in which case, ϕ is again the direction to the goal, or (b) waiting for another agent to pass, in which case the vehicle stays at its current location, or (c) avoiding another agent, in which case ϕ is computed so as to clear the path of the other vehicle. Low priority agens avoid the local paths of higher priority ones. During this process, the lower priority agent acquires the other agent's priority, allowing it to push other vehicles out of its way. This process guarantees that at least the highest priority agent will be making progress towards its goal. Given that priorities are changing dynamically, agents must communicate these changes as soon as they occur. Agents also need to communicate to their neighbors when they change high-level mode and provide their desired destination.

II. Problem Setup

The simple car is a model for non-holonomic robots:

$$\dot{x} = u \cdot \cos\theta(t), \qquad \dot{y} = u \cdot \sin\theta(t), \qquad \dot{\theta} = u \cdot \omega(t), \quad (1)$$

where x, y are Cartesian coordinates for the robot's position and θ is the system's orientation. The controls correspond to the velocity u and ω, where $\omega : \mathbb{R} \to [-\frac{u}{R_C}, \frac{u}{R_C}]$ is a bounded signed curvature control signal. Similar to the GRP work [1], this paper considers homogeneous mobile agents that have the same, constant, linear velocity u and curvature radius R_C. Each agent has a start and a goal position. The initial configuration is described by $c_i(0) = c_{0,i} \in SE(2)$, while the goal configuration by $c_{f,i} \in SE(2)$.

The map $d : SE(2) \times SE(2) \to \mathbb{R}^+$ is the distance between the positions of two agents:

$$d(c_i, c_j) = \|(x_i, y_i) - (x_j, y_j)\|_2 \quad (2)$$

A collision at time t_c between two agents occurs when the agents are closer than a specified safety Euclidean distance d_s: $d(c_i, c_j) < d_s$. Thus, a safety region is defined for each vehicle, which is a circular region with radius $R_s = d_s/2$

centered at the agent's position. A collision will occur when two safety regions overlap. The overall objective, similar to [1], is to devise a decentralized feedback policy that satisfies the following properties:

- **Safety**: Each vehicle follows a safe trajectory, i.e., one without any collisions:

$$\forall t > 0, \forall i, j \in \{1, ..., n\}, i \neq j : d(c_i(t), c_j(t)) \geq d_s \quad (3)$$

- **Liveness**: There is always at least one vehicle which is guaranteed to reach its goal in finite time.

$$\exists t_f \geq 0, \forall i \in \{1, ..., n\} : c_i(t_f) = c_{f,i} \quad (4)$$

If the vehicles do not constrain their fellow-agents after reaching $c_{f,i}$, then the liveness property can be applied recursively to show that all agents will reach their target.

III. Approach

This section proposes an approach described by two hybrid automata. The first one chooses a desired direction for the agent, while the second picks the actual control.

A. High-level operation

The objective of the approach is to select the best controls for the i-th agent so as to reach $c_{f,i}$ safely without collisions. The high-level automaton is using, as input, the direction ϕ^*, the true direction to the goal $c_{f,i}$. It then computes a set of valid desired directions Φ that the vehicle can move towards. Out of this set, the desired direction ϕ for the agent is chosen. The set Φ and control ϕ are used by the second hybrid automaton, which computes a valid set of achievable directions Θ and the control for the agent.

In order to compute the desired direction ϕ the vehicles communicate. Each agent has to transmit priority information and its current desired destination. The desired destination changes depending on the mode of the first automaton. Given the transmitted information, an agent is able to compute the set Φ so as to allow vehicles of higher-priority to make progress towards their goal. The set Θ will guarantee that neighboring vehicles will not collide.

1) Reserved region: Every agent i has a reserved region $\mathcal{RR}_i$ as in Fig. 2, upon which it claims exclusive ownership. Given that the position of neighboring vehicles can be detected, an agent can avoid the neighbors' reserved regions. Let the map $r_c : SE(2) \rightarrow \mathbb{R}^2 : (x, y, \theta) \longmapsto (x_r, y_r)$ associate to agent i's configuration the center of the reserved region $\mathcal{RR}_i$. The reserved region is a circle i defined by the path of the vehicle when it executes a constant control $\omega = -u/R_C$. The center of the reserved region can be easily computed by:

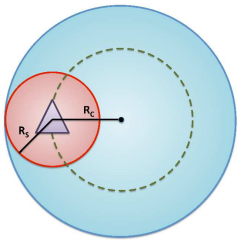

Fig. 2. The reserved region of a vehicle.

$$(x_r, y_r) = r_c(x, y, \theta) = (x + sin(\theta), y - cos(\theta)) \quad (5)$$

The reserved region can be stopped at any time, by setting $\omega = -u/R_C$ and once stopped, it can be moved in any direction, provided one waits long enough for the heading θ of the vehicle to reach the appropriate value.

B. Selecting Desired Directions

Each agent stores the following priority information:

- A static priority, unique for each vehicle.
- A dynamic priority, initially equal to the static one.
- A distance value, which will be explained below.

Beyond the reserved region, this work also defines a "respected area" $\mathcal{RA}$ for all the agents, as shown in Fig. 3. In order to change the mode of the high-level automaton and update the set of valid directions Φ, each vehicle requires information from vehicles that are within its communication range d_a. Specifically, it requires their current priorities and destinations.

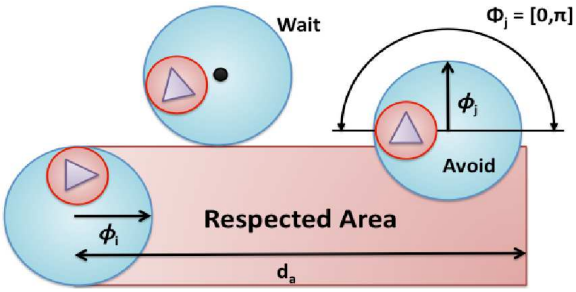

Fig. 3. The set of valid directions that a lower priority agent can use, so as to exit the respected area of an agent with higher priority.

Then, each vehicle can be in the following three modes as illustrated in Figure 4:

(1) *to_goal:* If the agent i is not violating the respected area of its higher dynamic priority neighbors, it can select as its desired direction ϕ the direction towards its goal ϕ^*.

Its dynamic priority in this case is set equal to its static one. When agent i enters into this mode, it transmits to its neighbors its priorities and its goal so that they can estimate the respected area $\mathcal{RA}_i$. Lower priority neighbors will avoid $\mathcal{RA}_i$ and evacuate its path. As agent i moves towards its goal, it may happen that its reserved region $\mathcal{RR}$ will touch one of its neighbors. Then the lower level automaton will take over for collision avoidance purposes. If at any point agent's i region $\mathcal{RR}_i$ enters or touches the area $\mathcal{RA}$ of a higher priority area, then agent i will need to switch to mode "avoid" or "wait", which are described below.

(2) *avoid:* If an agent j is in the respected area of a higher dynamic priority agent i, then j has to clear the path of the higher priority agent i. In this case, half of all directions that would force j to get further into $\mathcal{RA}_i$ are invalidated (e.g., agent j cannot move down in Figure 3). The approach selects as the desired direction ϕ the one that will cause the agent to move away of $\mathcal{RA}_j$ as fast as possible (i.e., the perpendicular direction to the $\mathcal{RA}_j$).

Then agent j acquires the same dynamic priority as agent i. In this way, agent j has higher priority and is now able to push other agents out of its path so as to clear i's respected area. It also acquires the distance value of the higher priority agent and increments it by one. The distance value expresses how many hops away the current agent is from the first agent that caused vehicles to enter into the avoid mode. It serves to differentiate between agents with the same dynamic priority.

When the dynamic priority of two agents is the same, then ties are broken by the distance value (lower distance value results in higher priority). If the distance value is also the same, then ties are broken by the static priority. Once agent j manages to exit $\mathcal{RA}_i$, then it switches to the "wait" mode. (3) *wait:* An agent j enters this mode when its reserved region is touching the respected area of a higher priority agent. The set of valid desired directions is the same as in the "avoid" mode. The dynamic priority of the agent is again set equal to its static one. Upon entering into this mode, the agent transmits to its neighbors its priority. It also transmits its current position as its current destination. Thus, neighboring agents are able to compute that $\mathcal{RA}_i = \mathcal{RR}_i$. While in this mode, the agent rotates with the maximum negative curvature value and remains within the reserved region. Once the goal direction ϕ^* is within the set Φ, then the agent can revert to the "to_goal" mode.

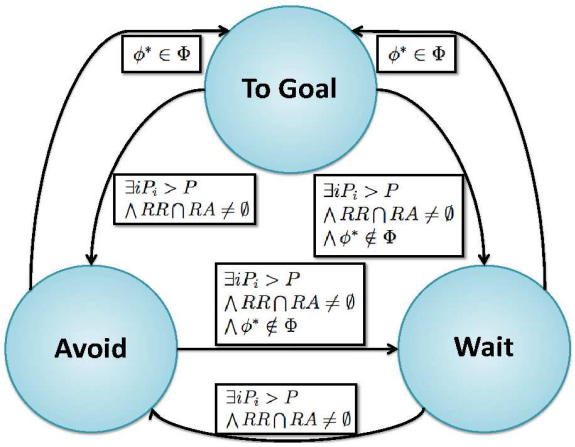

Fig. 4. The high level automaton for selecting the desired direction ϕ.

C. Controlling the Vehicle

Given the direction ϕ computed by the first automaton, the lower level automaton computes a valid, safe direction θ for the vehicle to move. In order to find θ, it is necessary to first find the set Θ containing all the valid directions. This set is constrained by its neighbors' reserved regions, $\mathcal{RR}_j$, which touch $\mathcal{RR}_i$. In particular, the velocity of the i-th reserved region is constrained by the convex cone determined by the intersection of a number of closed half-planes as in Fig. 5. The second automaton is composed of the modes *hold, straight, roll* and *roll2* as in related work [1].

1) Hold: The reserved region of an agent can remain stationary by setting $\omega = -u/R_C$. This is the holding action, where the vehicle can keep rotating and remain safe from collisions.

2) Straight: When the vehicle is in hold mode and the direction ϕ^* to the goal is free, the vehicle can follow this direction. In this case, the policy chooses the direction to the goal for θ (Fig. 6).

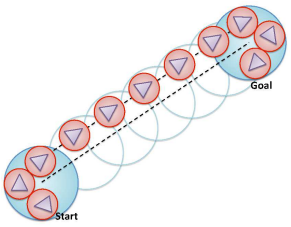

Fig. 6. An agent's trajectory when the direction to the goal is free

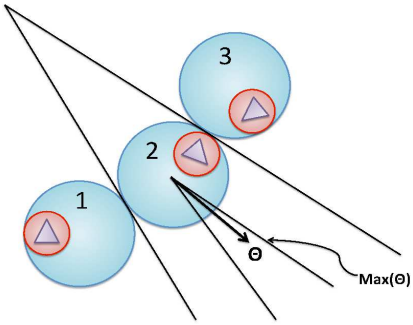

Fig. 5. The set of allowable directions in which the center of the i-th reserved region can move, generated by the contact with the reserved regions of vehicles j and k respectively

3) Roll: In case vehicle i with high priority is moving towards its goal, but another vehicle j is still in its way, then vehicle i will avoid j by rolling around it. This is achieved by rolling around $\mathcal{RR}_j$. Assuming that $\mathcal{RR}_j$ is stationary then the control for the vehicle i is:

$$f(\omega) = \begin{cases} (1 + d_s/2)^{-1} & if\ \Theta \neq \emptyset \text{ and } \theta = max(\Theta) \\ -u/R_C & otherwise \end{cases}$$

In this case, the agent is not turning with maximum curvature. Also, the expression addresses the case in which the agent's motion is constrained by more than one neighbors.

4) Roll2: In general, the reserved region of a neighbor will not necessarily remain stationary while an agent is rolling on it. This may cause the contact between two agents to be lost unexpectedly. In this case a new mode, roll2 is introduced. In this mode the vehicle is turning with the maximum curvature: $\omega = u/R_C$, unless this violates the constraints. By turning faster, the agent attempts to touch the $\mathcal{RR}_j$ of the other agent or find the way to its goal. The mode roll2 can only be entered if the previous state was roll.

D. Properties of the approach

Assume that agent i has to respect agent j. This means that i cannot violate $\mathcal{RA}_j$, or if it is already in this region, then it has to exit. In Fig.3 the avoid situation is depicted. If i is not blocked from other agents, then it can easily depart from $\mathcal{RA}_j$. In the case that an agent k is blocking i's path, agent k must be pushed as well. If agent k has higher priority than i, then i cannot depart $\mathcal{RA}_j$ and a deadlock arises. In order to avoid these situations the agents need more information. This is the reason why, once pushed by j, agent i acquires the same dynamic priority, which is the highest in its neighborhood. In this way, agent i can push other agents out of its way. Agent j still has priority over i, since i will have a higher distance value.

Consider the case in Fig. 7. If two agents (a_2, a_3) are being pushed by the same agent a_1, then both agents will have the same priority. a_3 is pushing agent a_4, which has to move away from $\mathcal{RA}_{a_3}$. In order to achieve this, a_4 has to run into $\mathcal{RA}_{a_2}$. a_4 and a_2 have the same dynamic priority because of the propagation of the priority through the local neighborhood. In this example the agent a_2 has distance 1 while agent a_4 has distance 2. Agent a_4 has to respect a_2

because agent a_2 has a smaller distance from agent a_1 that initiated the departure from its respected area.

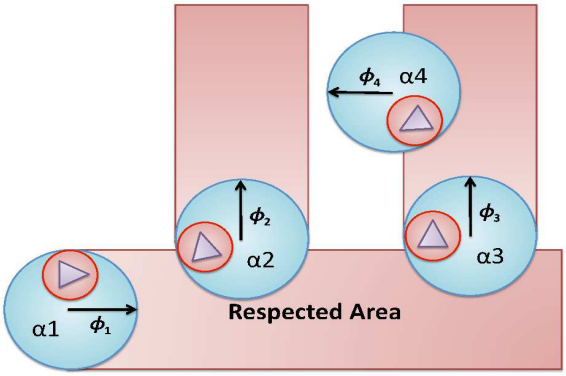

Fig. 7. Agent a_4 will respect $\mathcal{RA}_{a_2}$ as it belongs to an agent with higher priority, since a_2 is being pushed by the agent with the highest priority.

Consider the more complex case where both agents a_2 and a_3 are pushing two other agents a_4 and a_5 as in Figure 8. If the two agents a_4 and a_5 are in the way of a_2 and a_3 respectively, then they must be pushed into each other. In this case, agent with the highest initial priority wins.

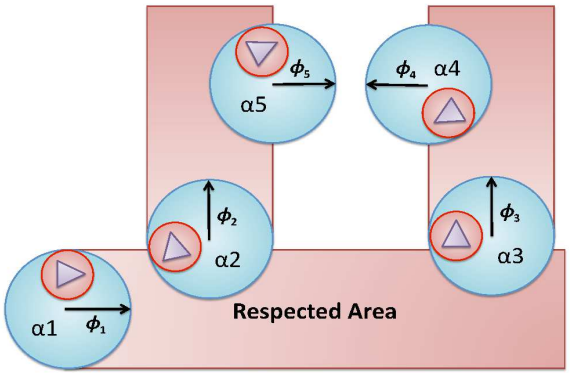

Fig. 8. Agents a_4 and a_5 move one against the other so as to avoid agents a_2 and a_3 respectively. The agent with the highest initial priority will win.

Because the agents are operating in an obstacle free environment, it is certain that all the agents can recursively be pushed away from the highest priority agent. In this way it is possible to avoid deadlocks because there is always an agent with the highest priority, who will be respected by all other agents while it moves toward its goal.

IV. Simulated Experiments

In order to evaluate the approach multiple experiments were performed with different parameters on a simulation engine developed at the authors' institution. The proposed method was compared against the Generalized Roundabout Policy [1]. In the experiments the velocity of the airplanes is 100 miles/hour, the maximum curvature is 0.05, and the control update frequency is 100Hz. The communication range is $d_a = 8 \times Rc + 4 \times ds$. The experiments were executed on a Intel Core 2 Duo 2.66GHz processor with 4GB of memory. Each test is limited to 10 minutes of run time and is parametrized as follows:

- Random goals or one, single goal
- Number of airplanes (20, 40, 60, or 80)

Benchmark Problems: In the presentation of the GRP algorithm [1] a condition is described that is guaranteed to result in a livelock. The proposed technique avoids livelocks under this condition, as depicted in Fig. 9.

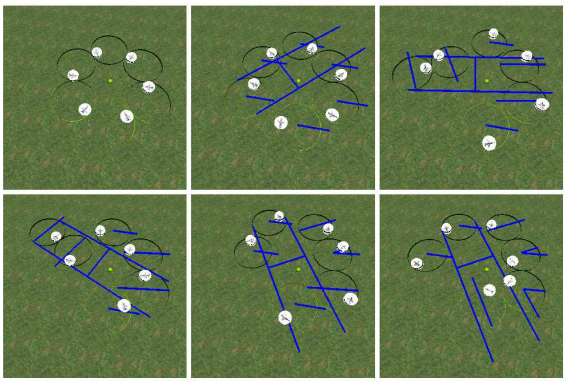

Fig. 9. The solution to the livelock condition as solved by the ERP.

Random Starts - Random Goals: In this case, the starts and goals of the vehicles are random and non-overlapping. For each test, 10 simulations have been run. The results are shown in Table I. Both of the algorithms show a 100% success rate within the 10 minutes, but the average time of ERP is better than GRP in all the experiments.

Random Starts - Single Goal: Here the airplanes have random initial configurations but the same goal. Again, 10 simulations were ran for 20 vehicles. Table II shows that GRP has a 0% success rate within 10 minutes for more than 60 vehicles, while ERP has 100% succes for up to 80 vehicles. The average time that ERP needs to finish the experiments is 19.85 seconds for 10 vehicles, while it is 52.46 seconds for GRP. Experiments with 120 vehicles have been run where GRP cannot get any vehicle to its goal within 10 minutes. ERP lands 25 aircraft in the same amount of time.

Communication Cost: The bandwidth required by the approach is minimal. The average percentage of cycles that the agents have to broadcast information to their neighborhood for the above described experiments is depicted in Table III. The cost of communication amounts to 2 bytes for the priority information. Beyond the static and dynamic priority, a single byte is needed for the distance priority information, which counts how far away the current agent is from the highest priority that initiated the push. Furthermore, a 2-dimensional point needs also to be transmited to describe the destination points.

# vehicles	cycles in which communication occurs
20	0.09413561%
40	0.1610661%
60	0.2439481%
80	0.3754106%

TABLE III
STATISTICS FOR THE COMMUNICATION OVERHEAD

	GRP			ERP		
# vehicles	successful runs	# airplanes landed	average completion time	successful runs	# airplanes landed	average completion time
20	10	20	11.2073	10	20	10.066649
40	10	40	53.3464	10	40	29.94588
60	10	60	96.5554	10	60	74.13597
80	10	80	136.6022	10	80	102.07883

TABLE I

STATISTICS FOR THE RANDOM STARTS AND RANDOM GOALS EXPERIMENTS

	GRP			ERP		
# vehicles	successful runs	# airplanes landed	average completion time	successful runs	# airplanes landed	average completion time
20	10	20	52.46	10	20	19.85
40	9	38.6	429.94	10	40	174.82
60	0	13.4	600	10	60	259.44
80	0	10.2	600	10	80	495.53

TABLE II

STATISTICS FOR THE RANDOM STARTS AND SINGLE GOAL EXPERIMENTS

V. DISCUSSION

This work deals with the problem of decentralized conflict-resolution among multiple non-holonomic vehicles that cannot stop instantaneously, such as aircraft. An existing approach guarantees collision avoidance for this setup but may result in livelocks [1]. The proposed method contributes a solution which improves decentralized conflict resolution by utilizing minimal communication. Simulated experiments show that the new method leads the vehicles to their goals faster and can address scenarios that where not solvable before. Future work will be based on reducing the amount of communication and the dependency on priorities, while providing a formal proof regarding deadlock and livelock avoidance. One way to improve the efficiency of the paths followed by the systems, while still providing safety and liveness, could be the integration with reactive collision avoidance methods (e.g., velocity obstacles [13]).

REFERENCES

[1] L. Pallotino, V. G. Scordio, E. Frazzoli, and A. Bicchi, "Decentralized cooperative policy for conflict resolution in multi-vehicle systems," *IEEE Transactions on Robotics*, vol. 23, no. 6, pp. 1170–1183, 2007.

[2] A. Bicchi and L. Pallotino, "On optimal cooperative conflict resolution for air traffic management systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 4, December 2000.

[3] M. S. Branicky, M. M. Curtiss, J. Levine, and S. Morgan, "Sampling-based Planning, Control and Verification of Hybrid Systems," *IEEE Proceedings on Control Theory and Applications*, vol. 153, no. 5, pp. 575–590, 2006.

[4] C. J. Tomlin, I. Mitchell, and R. Ghosh, "Safety Verification of Conflict Resolution Maneuvers," *IEEE Transactions on Intelligent Transportation Systems*, vol. 2, no. 2, 2001.

[5] K. Azarm and G. Schmidt, "Conflict-free Motion of Multiple Mobile Robots based on Decentralized Motion Planning and Negotiation," in *IEEE Intern. Conference on Robotics and Automation*, 1997, pp. 3526–3533.

[6] E. Rimon and D. E. Koditschek, "Exact Robot Navigation Using Artificial Potential Functions," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 5, pp. 501–508, 1992.

[7] S. Loizu, D. Dimarogonas, and K. Kyriakopoulos, "Decentralized Feedback Stabilization of Multiple Nonholonomic Agents," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2004, pp. 3012–3017.

[8] G. Roussos, D. V. Dimarogonas, and K. J. Kyriakopoulos, "3d navigation and collision avoidance for non-holonomic aircraft-like vehicles," *International Journal of Adaptive Control and Signal Processing*, vol. 24, no. 10, pp. 900–920, 2010.

[9] W. Li and C. Cassandras, "A cooperative receding horizon controller for multi-vehicle uncertain environments," *IEEE Transactions on Automatic Control*, vol. 51, no. 2, pp. 242–257, 2006.

[10] M. Earl and R. D. D'Andrea, "Iterative MILP Methods for Vehicle Control Problems," *IEEE Transactions on Robotics*, vol. 21, pp. 1158–1167, December 2005.

[11] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics and Automation Magazine*, vol. 4, no. 1, 1997.

[12] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *Int. Journal of Robotics Research*, vol. 17, no. 7, 1998.

[13] J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2008.

[14] D. Wilkie, J. van den Berg, and D. Manocha, "Generalized velocity obstacles," *IEEE/RSJ Int. Conf. on Intel. Robots and Systems*, 2009.

[15] E. Lalish and K. A. Morgansen, "Decentralized reactive collision avoidance for multivehicle systems," in *IEEE Conference on Decision and Control*, 2008.

[16] M. Erdmann and T. Lozano-Perez, "On multiple moving objects," in *IEEE Intern. Conference on Robotics and Automation (ICRA)*, 1986, pp. 1419–1424.

[17] M. Bennewitz, W. Burgard, and S. Thrun, "Finding and Optimizing Solvable Priority Schemes for Decoupled Path Planning Techniques for Teams of Mobile Robots," *Robotics and Autonomous Systems*, vol. 41, no. 2, pp. 89–99, 2002.

[18] P. O'Donnell and T. Lozano-Perez, "Deadlock-free and collision-free coordination of two robot manipulators," in *IEEE Int. Conf. Robotics and Automation (ICRA)*, 1989, pp. 484–489.

[19] J. Peng and S. Akella, "Coordinating multiple robots with kinodynamic constraints along specified paths," *Int. Journal of Robotics Research*, vol. 24, no. 4, pp. 295–310, 2005.

[20] Y. Li, K. Gupta, and S. Payandeh, "Motion planning of multiple agents in virtual environments using coordination graphs," in *IEEE Int. Conf. Robotics and Automation (ICRA)*, 2005, pp. 378–383.

[21] K. E. Bekris, K. I. Tsianos, and L. E. Kavraki, "A decentralized planner that guarantees the safety of communicating vehicles with complex dynamics that replan online," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007, pp. 3784–3790.

[22] V. J. Lumelsky and K. R. Harinarayan, "Decentralized Motion Planning for Multiple Mobile Robots: The Cocktail Party Model," *Autonomous Robots*, vol. 4, no. 1, pp. 121–135, 1997.

[23] S. Qutub, R. Alami, and F. Ingrand, "How to solve deadlock situations within the plan-merging paradigm for multi-robot cooperation," in *Proc. of the Inter. Conf. on Intelligent Robots and Systems (IROS)*, vol. 3, 1997, pp. 1610–1615.

[24] C. M. Clark, S. M. Rock, and J.-C. Latombe, "Motion Planning for Multiple Robots using Dynamic Networks," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2003, pp. 4222–4227.

[25] M. Lagoudakis, V. Markakis, D. Kempe, P. Keskinocak, S. Koenig, A. Kleywegt, C. Tovey, M. A., and S. Jain, "Auction-based multi-robot routing," in *Robotics: Science and Systems (RSS)*, 2005, pp. 343–350.

[26] P. J. Modi, W.-M. Shen, M. Tambe, and M. Yokoo, "An asynchronous complete method for distributed constraint optimization," *Artificial Intelligence Journal*, vol. 161, no. 1-2, pp. 149–180, 2005.

[27] M. Yokoo, "Algorithms for distributed constraint satisfaction: A review," *Autonomous Agents and Multi-Agent Systems*, vol. 3, no. 2, pp. 189–212, 2000.