Maintaining Team Coherence under the Velocity Obstacle Framework

Andrew Kimmel
University of Nevada, Reno
1664 N. Virginia St., MS171
Reno, NV 89557
akimmel@cse.unr.edu

Andrew Dobson
University of Nevada, Reno
1664 N. Virginia St., MS171
Reno, NV 89557
dobsona@cse.unr.edu

Kostas Bekris
University of Nevada, Reno
1664 N. Virginia St., MS171
Reno, NV 89557
bekris@cse.unr.edu

ABSTRACT

Many multi-agent applications may involve a notion of spatial coherence. For instance, simulations of virtual agents often need to model a coherent group or crowd. Alternatively, robots may prefer to stay within a pre-specified communication range. This paper proposes an extension of a decentralized, reactive collision avoidance framework, which defines obstacles in the velocity space, known as Velocity Obstacles (VOs), for coherent groups of agents. The extension, referred to in this work as a Loss of Communication Obstacle (LOCO), aims to maintain proximity among agents by imposing constraints in the velocity space and restricting the set of feasible controls. If the introduction of LOCOs results in a problem that is too restrictive, then the proximity constraints are relaxed in order to maintain collision avoidance. If agents break their proximity constraints, a method is applied to reconnect them. The approach is fast and integrates nicely with the Velocity Obstacle framework. It yields improved coherence for groups of robots connected through an input constraint graph that are moving with constant velocity. Simulated environments involving a single team moving among static obstacles, as well as multiple teams operating in the same environment, are considered in the experiments and evaluated for collisions, computational cost and proximity constraint maintenance. The experiments show that improved coherence is achieved while maintaining collision avoidance, at a small computational cost and path quality degradation.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Algorithms, Design, Experimentation

Keywords

AAMAS proceedings, multi-agent, collision avoidance, team coherence, communication constraints

Appears in: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), June, 4–8, 2012, Valencia, Spain.

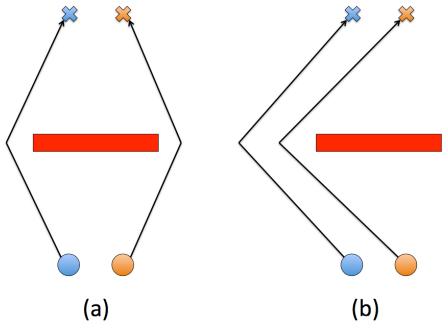


Figure 1: Two agents navigating around a static obstacle: (a) the agents split to reach their goals without collisions, while in (b) the agents move together as a coherent team. The second behavior must be achieved in a decentralized manner.

1. INTRODUCTION

Many practical applications of decentralized collision avoidance may involve a secondary objective, where teams of agents need to maintain a certain level of coherence. Coherence often implies that the agents should remain within a certain distance of one another. In games and simulations, the agents may need to remain within a certain distance because of implied social interactions, or because they need to reach their destination together so as to be more effective in completing an objective at their goal. For instance, a team of agents in a game will be more effective in attacking an enemy if all the units move together against the opponent and do not split into multiple groups. In mobile sensor networks, robots may have to respect radial communication limits.

The Velocity Obstacle (VO) formulation [8, 27, 22] is a framework for reactive collision avoidance. It is fast as it operates directly in the velocity space of each agent. It is also a decentralized approach as each agent reasons independently about its controls, as long as it can compute the position and velocity of its neighbors. Current work in Velocity Obstacles does not directly address the issue of coherence. Consider the situation in Figure 1, where two agents are required to avoid an obstacle and reach their desired destination. An application of the basic VO framework may result in the two agents splitting and passing the obstacle from opposite ends. It would be desirable, however, for the agents to select, in a decentralized manner, a single direction to follow so as to avoid the obstacle and reach their goal while maintaining

coherence. The decentralized nature of the solution will be especially helpful in robotic applications because no communication will be required between robots. It will also provide improved scalability in simulations and games.

This paper proposes a method for maintaining team coherence within the VO framework in a decentralized manner. The desired coherence for a problem can be defined as a graph of dependencies between agents. An edge in the graph implies that the corresponding agents should remain within a predefined distance as they move towards their goal. Given this input graph, an agent constructs an additional obstacle in the velocity space for each neighbor with which it wants to maintain connectivity. This construction is referred to in this work as a Loss of Communication Obstacle (LOCO).

Building on top of the VO framework allows LOCOs to focus on coherence maintenance, rather than obstacle avoidance. In order to construct the LOCO, the assumption is that a neighbor will maintain its current control, as in the original VO framework. Then, a LOCO defines the set of velocities that will lead the two agents to be separated beyond a desired distance within a certain time horizon. A scheme is proposed for integrating information from the multiple VOs defined for collision avoidance and the LOCOs defined for coherence maintenance. If the set of velocities that satisfy both constraints is not empty for a satisfactory time horizon, then the velocity in this set that brings the agent closer to its goal is chosen. A valid velocity implies that, given the neighbor does not change control, following this velocity will not lead the agent into a collision or violation of proximity constraints for the given horizon. If the set of valid velocities is empty, then the objective of maintaining coherence is dropped in favor of guaranteeing collision avoidance, which should be satisfiable, unless oscillations appear in the selected controls of agents. If two agents that are supposed to retain proximity end up violating the distance constraint, the proposed method makes them move in a direction that will allow them to reconnect.

This paper describes the LOCO method and provides simulations which show that, by reasoning about distance constraints, it is possible to solve decentralized collision avoidance problems while improving the coherence of a team. The approach is built on top of and compared against a formulation of velocity obstacles proposed in the literature for teams of agents that execute the same protocol, referred to as Reciprocal Velocity Obstacles [27, 22]. The experiments consider disk-shaped agents that move with a constant speed in a holonomic manner, i.e., the agents can freely choose to follow any direction instantaneously. First, a series of static environments are tested with a single team of agents navigating while maintaining coherence. Different types of formations between the teams are considered. Then, tests for multiple teams of agents navigating in the same environment while maintaining coherence are performed.

The organization of the paper is as follows. First, in Section 2, the relevant work to this problem is reviewed. Section 3 provides a formal description of the problem as well as the applicable notation used throughout the manuscript. The Velocity Obstacle framework is outlined in Section 4 together with the proposed approach for maintaining team coherence. Section 5 describes relevant results on various experimental setups. Lastly, Section 6 concludes the paper with a discussion of the technique and possible future work.

2. BACKGROUND

2.1 Virtual Agent Applications

The need to move multiple agents as a coherent team arises in many virtual agent applications [25, 18], ranging from crowd simulation [17], to pedestrian behavior analysis [16, 19], to shepherding and flocking behaviors [15, 29]. Many methods make use of "steering behaviors", with the objective of having agents navigate in a life-like and improvisational manner [20]. These steering behaviors can be combined to achieve higher-level goals, such as "get to the goal while avoiding obstacles" or "join a group of characters". Similar is the objective of the social force model [10].

2.2 Coupled Multi-Robot Path Planning

There is also extensive literature on motion coordination and collision avoidance in robotics. The multi-robot path planning problem can be approached by either a coupled approach or a decoupled one [12, 13]. The coupled approach plans for the composite robot, which has as many degrees of freedom as the sum of degrees of freedom of each individual robot. Integrated with complete/optimal planners, the coupled algorithm achieves completeness/optimality. Nevertheless, it becomes intractable due to its exponential dependency on the number of degrees of freedom.

2.3 Decoupled Multi-Robot Path Planning

Decoupled approaches plan for each agent individually. In prioritized schemes, paths are computed sequentially and high-priority agents are treated as moving obstacles by low-priority ones [6]. Searching the space of priorities can assist in performance [4]. Such decoupled planners tend to prune states in which higher priority agents allow lower priority ones to progress, which may eliminate the only viable solutions. Search-based decoupled approaches consider dynamic prioritization and windowed search [21], as well as spatial abstraction for improved multi-agent heuristic computation [24, 30]. In particular, mobile robotic sensor networks require that robots move while maintaining communication. Techniques which attempt to tackle this issue have to balance a trade-off between centralized and decentralized planning. There are techniques that create networks of robots to compute plans in a centralized manner across distributed systems [5] or in a fully decentralized manner [3].

2.4 Formations

There is a significant amount of work on formation control, which is a way of moving multiple agents as a coherent team. One direction is to use a virtual rigid body structure to define the shape of a formation, and then plan for this rigid body [14]. Other techniques attempt to have more flexible structures, where interactions between robots are modeled as flexible joints [1]. An alternative is to first generate a feasible trajectory for the group's leader according to its constraints and then use feedback controllers for the followers [7, 2].

2.5 Reactive Obstacle Avoidance

Many techniques attempt to solve the problem of collision avoidance using reactive methods. One technique used in robotics is the Dynamic Window approach, which operates directly in the velocity space of a robot, reasoning over the achievable velocities within a small time interval [9]. An

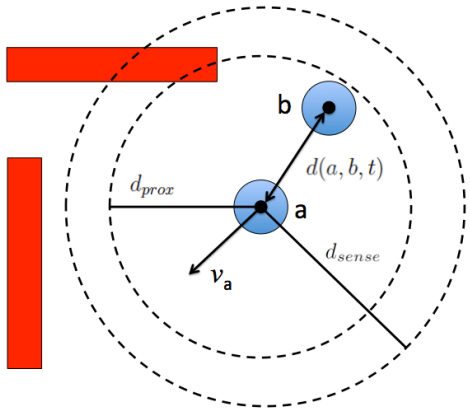


Figure 2: Agents a and b share a proximity constraint d_{prox}. Agent a moves with velocity v_a where $\|v\| = s$ and can sense all agents within d_{sense}.

alternative approach, known as the Velocity Obstacle (VO), assumes that neighboring agents will keep following their current control. Based on this assumption, it defines conic regions in velocity space, which are invalid to follow, as they lead to a collision with the neighbor at some time in the future [8]. If the future trajectory of other robots is known, non-linear VOs can be constructed [11]. The basic VO formulation can result in oscillations in motion when multiple agents execute the same algorithm. The reciprocal nature of other robots can be taken into account in order to avoid these oscillations, which leads to the definition of Reciprocal Velocity Obstacles [27]. Using this idea of reciprocity, the robots can attempt to optimally steer out of collision courses with other robots using an extension of this framework called Optimal Reciprocal Collision Avoidance (ORCA) [26]. This technique was extended to 3D cases using simple-airplane systems [23]. Further work extends the VO formulation to work with acceleration constraints as well as many kinematically and dynamically constrained systems [28].

2.6 Contribution

This work uses a reactive technique to define new obstacles in the velocity space, called Loss of Communication Obstacles (LOCO). LOCOs are computed quickly and, when integrated with Velocity Obstacles, allow robots to reactively avoid static and dynamic obstacles while maintaining a better sense of coherence. The new technique does not impose communication requirements, yet maintains connectivity in a decentralized manner. The approach still requires that robots are able to sense the position and velocity of other robots in the scene, as well as the position of static obstacles.

3. PROBLEM SETUP

Consider n planar, holonomic disks moving with constant speed s. Let the set of all agents be A. Each disk agent $a \in A$ has radius r_a and can instantaneously move with a velocity vector v_a that has magnitude s. Agents are assumed to be capable of sensing the position and velocity of other agents in the environment within a sensing radius d_{sense}. Furthermore, the agents have available a map M of the environment that includes the static obstacles. The configuration space for each agent is $Q = \mathbb{R}^2$, and it can be partitioned into two sets, Q_{free} and Q_{obst}, where Q_{free} represents the obstacle free part of the space, and Q_{obst} is the part of the space with obstacles. Each agent follows a trajectory $q_a(t)$, where t is time. Initially an agent is located at a configuration $q_a(0) = q_a^{init}$ and has a goal location q_a^{goal}.

Consider the distance $d(a, b, t)$ between agents a and b at time t (Figure 2). If $d(a, b, t) < r_a + r_b$, then agents a and b are said to be in collision at time t. Collisions with static obstacles occur when $q_a(t) \in Q_{obst}$. An input graph $G(A, E)$ is provided that specifies which agents need to be in close proximity. The vertices of graph G correspond to the set of agents A and an edge (a, b) implies that agents a and b must satisfy $d(a, b, t) \leq d_{prox}$, where d_{prox} is a proximity constraint.

The objective is for the agents to move from q^{init} to q^{goal} without any collisions with obstacles or among them, while satisfying as much as possible the proximity constraints specified in the input graph G. More formally, all agents should follow trajectories $q_a(t)$ for $0 \leq t \leq t_{final}$, so that $\forall a \in A$:

- $q_a(0) = q_a^{init}$,
- $q_a(t_{final}) = q_a^{goal}$,
- $q_a(t) \in Q_{free}$, $\forall t \in [0, t_{final}]$,
- and $\forall b : r_a + r_b < d(a, b, t) < D$, where
  - $D = d_{prox}$, if $\exists (a, b) \in E$ of graph G,
  - and $D = \infty$ otherwise.

4. APPROACH

4.1 Velocity Obstacle Framework

Velocity obstacles are defined in the relative velocity space of two agents. $VO^{\infty}_{a|b}$ can be geometrically constructed as in Figure 3. Agent a's geometry is reduced to a point by performing the Minkowski sum of agent a and agent b: $a \oplus b$. Then, tangent lines to the Minkowski sum disk $a \oplus b$ are constructed from agent a. These tangent lines bound a conic region. This region represents the space of all velocities v_a of agent a that would eventually lead into collisions with agent b assuming that b has zero velocity. Given that agent b has a velocity v_b, the conic region needs to be translated by the vector v_b. This construction assumes an infinite time horizon. In practice, it is often helpful to truncate the VO based on a finite time horizon τ. This VO represents all velocities v_a, which will lead into a collision within time $t \leq \tau$, given that agent b keeps moving with velocity v_b.

This work adopts a modification of the basic VO framework, which deals with the case that the two agents are reciprocating [27]. Reciprocal Velocity Obstacles (RVOs) are created by translating the VO according to a weighted average of the agents' velocities, $(\alpha \cdot v_A) + ((1 - \alpha) \cdot v_B)$ where α is a parameter representing the level of reciprocity between agents A and B. If both agents are equally reciprocal, then $\alpha = 0.5$.

Given this framework, an algorithm for calculating valid velocities for decentralized collision avoidance can be defined. Agent a will have a set of reachable velocities and the task is to select one such velocity, which is collision-free. For holonomic disk agents moving at constant speed s, the set of reachable velocities would be $V_{reach} = \{v | \|v\| = s\}$. Let the set of velocities which are invalid according to all the VOs for neighbors of a be defined as follows: $V_{inv} = \{v | \exists\ \mathtt{VO}_{a|b}\ s.t.\ v \in \mathtt{VO}_{a|b}, b \in A, a \neq b\}$. Then, the set of feasible velocities which are reachable but not in the invalid set is defined as $V_{feas} = V_{reach} \setminus V_{inv}$. The selected veloc-

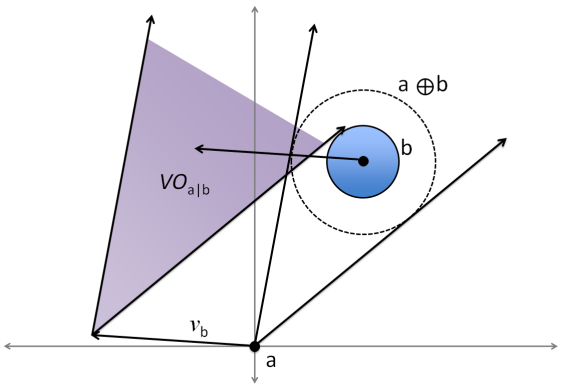


Figure 3: Construction of $VO^{\infty}_{a|b}$ for an infinite time horizon. The Minkowski sum of a and b, $a \oplus b$ is used to define a cone in velocity space, which is then translated by v_b. The shaded region represents all velocities v_a, which lead a into collision with b.

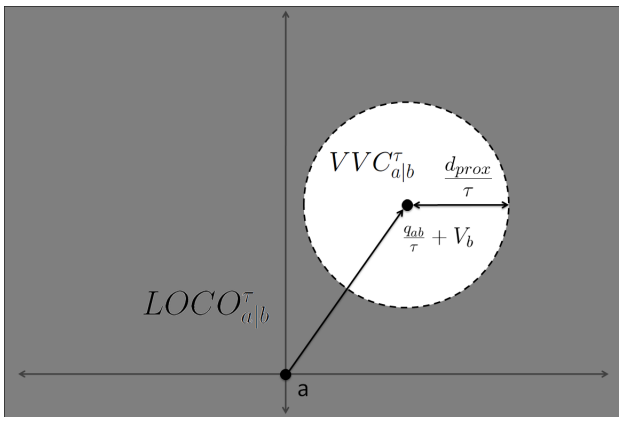


Figure 4: Construction of $\texttt{LOCO}^{\tau}_{a|b}$ and $\texttt{VVC}^{\tau}_{a|b}$. The circular region represents the viable velocities to maintain $d(a,b,t) \leq d_{prox}$ for time horizon τ. The shaded region outside the disk represents invalid velocities for agent a.

ity should be feasible, $v \in V_{feas}$, and typically minimizes a metric relative to the preferred velocity v_a^{pref}, e.g., a velocity vector that points to the goal. More details will be provided regarding the specific velocity selection scheme used in this work, in section 4.4.

4.2 Loss of Communication Obstacles

The proposed approach extends the VO framework by creating new obstacles in the velocity space. These obstacles aim to prevent loss of communication, or more generally, to satisfy proximity constraints between agents in the form $d(a,b,t) \leq d_{prox}$ for agents a and b for at least a finite time horizon τ. The LOCO imposed by agent b on agent a for a horizon τ, denoted as $\texttt{LOCO}^{\tau}_{a|b}$, is the set

$$\texttt{LOCO}^{\tau}_{a|b} = \{v | \ \forall \ t \in [0,\tau] : d(a,b,t) \leq d_{prox}\},$$

under the assumption that agent b follows its current velocity v_b for at least time τ.

Assume agents a, $b \in A$ for which $(a,b) \in E$ of the input graph G. The relative position of agent b for agent a will be denoted as $q_{ab} = q_b - q_a$. If the relative position at time t is $q_{(ab)}(t)$, then at time $t+\tau$ the relative position of the two agents is going to be:

$$q_{ab}(t+\tau) = q_{(ab)}(t) + \tau * (V_b - V_a).$$

For the two agents to be able to communicate at time $t+\tau$, it has to be that:

$$(q^X_{ab}(t+\tau))^2 + (q^Y_{ab}(t+\tau))^2 \leq d^2_{prox} \Rightarrow$$

$$(q^X_{ab}(t)+\tau*(V^X_b - V^X_a))^2 + (q^Y_{ab}(t)+\tau*(V^Y_b - V^Y_a))^2 \leq d^2_{prox} \Rightarrow$$

$$(\frac{q^X_{ab}(t)}{\tau} + V^X_b - V^X_a)^2 + (\frac{q^Y_{ab}(t)}{\tau} + V^Y_b - V^Y_a)^2 \leq \frac{{d^{prox}}^2}{\tau^2} \Rightarrow$$

$$(V^X_a - \frac{q^X_{ab}(t)}{\tau} - V^X_b)^2 + (V^Y_a - \frac{q^Y_{ab}(t)}{\tau} - V^Y_b)^2 \leq (\frac{d_{prox}}{\tau})^2$$

The last expression implies that the velocity V_a of agent a has to be within a circle with center $(\frac{q_{ab}}{\tau} + V_b)$ and radius $\frac{d_{prox}}{\tau}$. This circle will be referred to as the valid velocity circle ($\texttt{VVC}^{\tau}_{a|b}$) and is the complement of the $\texttt{LOCO}^{\tau}_{a|b}$. Figure 4 gives an example of a VVC circle.

An agent a, however, may have multiple neighbors leading to the definition of multiple LOCOs and VVCs. A choice that is made for simplicity is to consider the same time horizon τ for the definition of all the LOCOs for all the neighbors. Then, the set of velocities v_a that will not allow a to maintain connectivity with at least one neighbor b in the graph G is the union of individual LOCOs:

$$\texttt{LOCO}^{\tau}_{a} = \bigcup_{\forall b \ | \ \exists (a,b) \in E} \texttt{LOCO}^{\tau}_{a|b}$$

There are two complications arising from this definition. Firstly, it is not straightforward to compute the longest horizon for which this union is not the entire plane, i.e., the longest horizon for which the intersection of VVCs is not empty. The problem is that both the centers and the radii of the VVCs are changing for different time horizons. Secondly, the resulting region is rather complex to describe, as it corresponds to multiple circle intersections. This representation can impose significant computational overhead when the LOCOs are integrated with VOs.

Instead of computing the exact intersection of VVCs, this paper proposes a conservative approximation that is easier to represent and beneficial for computational purposes. The approximation of the valid set of velocities corresponds to a circle inside the intersection of VVCs. Figure 5 illustrates the procedure for two and three neighbors. Given two circles with centers C_b and C_c and radii r_b and r_c, the inscribed circle of their intersection has the following radius and center:

$$r_{bc} = \frac{r_b + r_c - ||C_b, C_c||}{2}$$

$$C_{bc} = C_b + (r_b - r_{bc})\frac{C_c - C_b}{||C_b, C_c||}$$

The procedure works in an incremental manner. First it computes the inscribed circle (C_{bc}, r_{bc}) of the intersection of two VVCs, and then computes the inscribed circle of (C_{bc}, r_{bc}) with another VVC and so on.

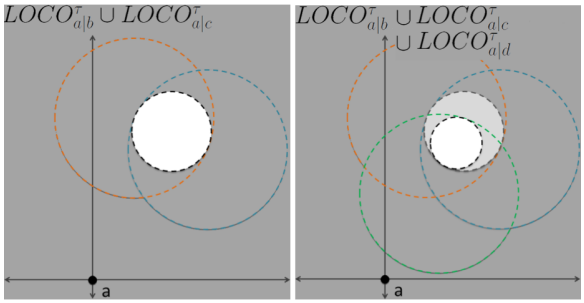


Figure 5: The conservative approximation of $\texttt{VVC}^{\tau}_{a}$ in the velocity space of agent a for two (left) and three (right) neighbors. The white circle corresponds to velocities that are guaranteed to maintain connectivity with the neighbors for time τ.

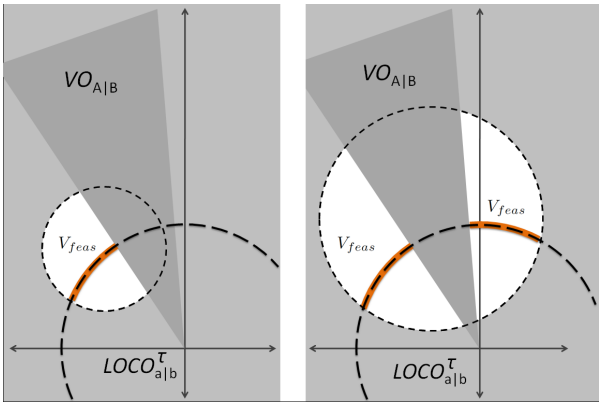


Figure 6: An example of how changing the horizon affects the set of feasible controls. The left image has a larger value for τ while the right has a smaller value of τ. Larger values of τ provide stronger guarantees for communication maintenance, but make finding a feasible control more difficult.

4.3 Integration of VOs and LOCOs

The final step for computing the $\texttt{LOCO}^{\tau}_{a}$ is to select a suitable time horizon. A tuning approach over consecutive simulation steps is used. Given the horizon τ from the previous time step, the $\texttt{LOCO}^{\tau}_{a}$ is computed as in Figure 5. Then the set $V_{valid} = V_{reach} \setminus \texttt{LOCO}^{\tau}_{a}$ is computed, which considers the reachable controls that do not violate the LOCO constraints. This operation can be done in an efficient manner especially for systems with constant speed s, as it corresponds to a circle to circle intersection. In the general case for systems with varying velocity there are still computational advantages, as the circular representation of the VVC greatly increases the speed in which V_{valid} can be computed. Once V_{valid} is available for a given τ, the measure of $|V_{valid}|$ is compared against a predefined threshold $|V_{thresh}|$. If $|V_{valid}| < |V_{thresh}|$, then τ is decreased and V_{valid} is recomputed. A smaller τ implies that a bigger set of valid velocities for proximity maintenance will be returned that provides guarantees for a shorter horizon. Alternatively, if $|V_{valid}| > |V_{thresh}|$, then τ is increased. This will return a smaller set of valid velocities but will provide guarantees for a longer horizon. This tuning process continues until V_{valid} comes close to V_{thresh}, but this takes place over multiple simulation steps and adapts on the fly to changes in the relative configuration of agents. The effects of tuning τ can be seen in Figure 6.

Once $\texttt{LOCO}^{\tau}_{a}$ has been computed, it must then be included in the list of constraints in order to correctly compute V_{feas}, which is now redefined to be $V_{feas} = V_{valid} \setminus V_{inv}$, as in Figure 6. Sometimes, the additional constraints imposed by LOCOs, cause V_{feas} to become empty. In this situation, the LOCO constraints are ignored, as attempting to maintain communication may be perilous to the agent's safety.

4.4 Velocity Selection

Each agent has a preferred velocity it would like to follow, denoted as v^{pref}_{a} for agent a. Ignoring the proximity constraints and in obstacle-free environment, the preferred velocity should be in the direction of the goal configuration $v_{goal} = q^{goal}_{a} - q_a$. If there are obstacles, however, setting the goal velocity in the same manner can lead the agents into local minima, causing the agent to become stuck behind the obstacle. This work avoids this issue by computing a discrete wave-front function in environments with obstacles. The goal velocity is computed as the direction to the cell with the minimum distance to the goal within a 3×3 region from the current cell of the agent.

In order to account for agents violating proximity constraints, a weighted velocity selection scheme is employed that takes neighbors into account. For some agent a, let d_i be the distance from agent a to another agent i and X_i be the state of agent i, where agent i is part of the proximity graph of agent a. Then, for n agents in the proximity graph of agent a, the average weighted configuration can be computed as:

$$q_{avg} = \frac{\sum_i^n \frac{d_i}{d_{prox}} q_i}{\sum_i^n \frac{d_i}{d_{prox}}}$$

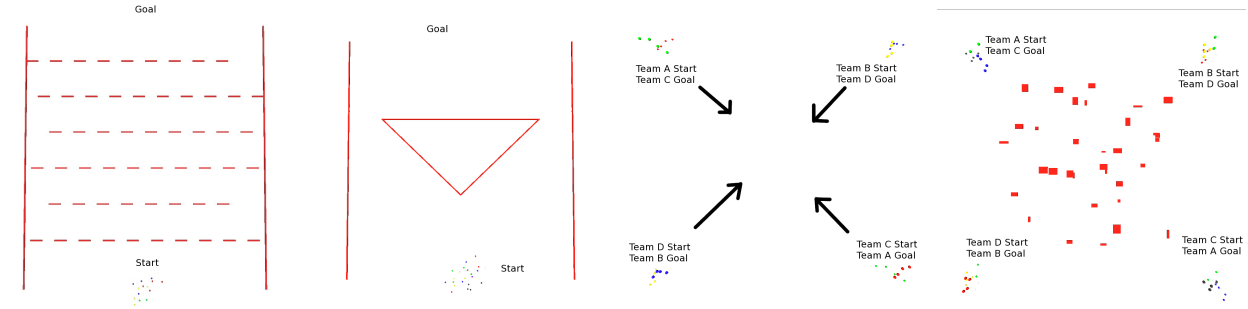
Then, $d_{avg} = ||q_a - q_{avg}||$ is the distance between agent a and q_{avg}. Let $v_{avg} = q_{avg} - q_a$ be the vector pointing to q_{avg}, and let v_{goal} be the vector towards the goal computed through the wavefront. Then, agent a's preferred velocity is computed as follows:

$$v^{pref}_{a} = \frac{d_{avg}}{d_{prox}} v_{avg} + \frac{d_{prox} - d_{avg}}{d_{prox}} v_{goal}$$

Thus, agents which are farther away from their proximity constraints (i.e., have violated constraints) will be inclined to shorten this distance, whereas agents that have not violated any proximity constraints move towards their goals.

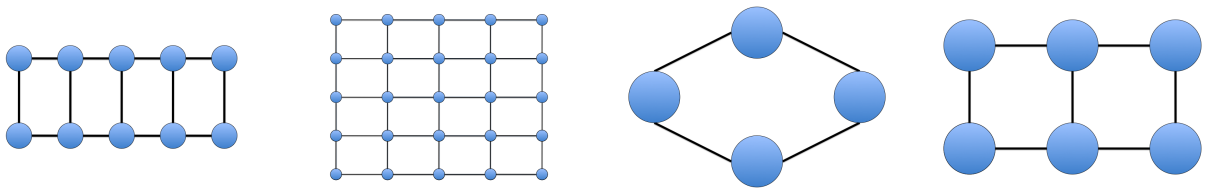
Once v^{pref}_{a} is computed, it can be checked for validity given the set V_{feas}. If v^{pref}_{a} is feasible, then agent a will use it. In the case that v^{pref}_{a} is not in V_{feas}, a different feasible control must be computed. In the general case, the region V_{feas} defines an area in velocity space which must be searched to find a control. The control found is the velocity $v \in V^{a}_{feas}$ which minimizes distance between v and v^{pref}_{a}.

The distance metric used in this approach is the Euclidean distance in the velocity space. Other metrics for finding the distance between velocities in this space are possible [28]. A list of intersections of the boundaries of the RVOs with V^{a}_{reach} is generated and a rotational sweep algorithm determines which points are valid. These points define the boundaries of the V^{a}_{feas} regions. They are checked to find which one of them minimizes the distance to the preferred velocity :



(a) PACHINKO environment (b) WEDGE environment (c) CROSSROADS environment (d) Random environment

Figure 7: The environments on which the experiments were executed



(a) (b) (c) (d)

Figure 8: Examples of input proximity graphs used in the experiments.

$min_{v \in V^a_{feas}}[d(v, v^{pref}_a)]$. In the general case, V^a_{feas} will be a non-convex region or possibly disjoint non-convex regions in the velocity space where a variant of the simplex algorithm can be used to find the optimum.

In the event that there is no valid velocity available, the agent will select its current velocity. The reasoning behind this is that in the VO framework, agents assume that their neighbors will keep using their current control. Thus, choices that keep the current control are preferable for this scheme.

5. RESULTS

The approach was implemented using a simulation software platform. Experiments were run on computers with a 3.06 GHz Intel Core 2 Duo processor and 4GB of RAM. The experiments are organized in the following manner. First, experiments were conducted using a single team of agents moving in an environment with obstacles. Then, experiments with multiple teams were run both in an obstacle-free world and in an environment with obstacles. A team of agents corresponds to a connected component of the input graph G. The experiments conducted compare the LOCO formulation against RVOs. The reason for comparing against RVOs, rather than other formation techniques, is due to the decentralized nature of the LOCO algorithm, which requires no additional information outside the VO framework. In addition, LOCOs utilize the notion of coherence, which is more abstract in nature than formations. Thus, RVOs are the most relevant technique to experiment against. Each experiment was measured with regards to the following metrics:

- total number of collisions during the entire experiment,
- computation time per frame,
- total time to solve the problem,
- average ratio of respected proximity links per frame,
- and number of successful runs.

For each variation of the environment, input graph G and algorithm, there were 10 runs executed. Each run had a random initial q_{init} and goal configuration q_{goal}, which, satisfied the proximity link in the graph G.

5.1 Single Team

For the single team scenarios, two different environments were tested with varying numbers of agents. The PACHINKO environment uses a series of walls with small gaps (Figure 7(a)), and a team of 10 agents was considered in this case. The proximity graph imposed on the team is shown in Figure 8(a). The WEDGE environment (Figure 7(b)) attempts to split the agents into two lanes. The proximity graph imposed on the team is shown in Figure 8(b).

	time per step (s)		collisions		maintained links		steps		success	
	LOCO	RVO	LOCO	RVO	LOCO	RVO	LOCO	RVO	LOCO	RVO
wedge grid	0.03964	0.03488	0.1	0	92%	65%	4279.1	3376.9	100%	100%
pachinko train	0.04474	0.0355	0	0	96%	46%	3881.2	3555	100%	100%

Table 1: Results for a single team.

Table 1 shows that there is a significant improvement in terms of the percentage of links maintained by the LOCO-based approach relative to RVOs. Another interesting statistic to examine is the coherence of agents over time, shown in Figure 9, which the LOCO-based approach also improves. This comes at the cost of a slightly increased computational cost and increased time taken by the algorithm to bring all of the agents to their goals. An example of the paths taken by 24 agents in the PACHINKO environment is shown in Figure 10. Both approaches were able to solve all of the problems and without any collision, with the exception of one experiment for the WEDGE environment, where a single collision was reported.

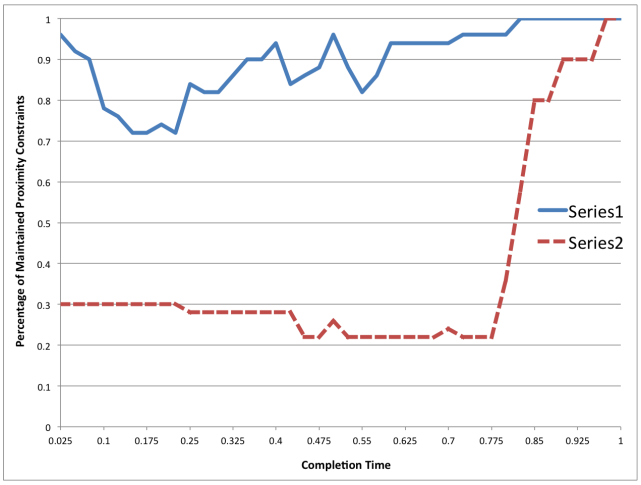


Figure 9: Coherence over time for 24 agents in the wedge grid experiment.

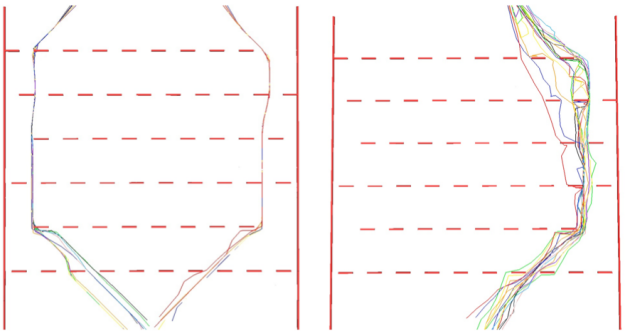


Figure 10: An example of the paths taken by 24 agents in the PACHINKO environment for the RVO (left) and LOCO (right) approach.

5.2 Multiple Teams

The performance of LOCOs was evaluated against RVOs in a scenario involving four teams of agents navigating in an obstacle-free environment as shown in Figure 7(c) (CROSS-ROADS) and for the connectivity graph for each team shown in Figure 8(c). There were four agents in each team. The last setup involved again four teams of agents, each one of which had six agents connected as in Figure 8(d). This time the environment involved a random set of rectangular obstacles as in Figure 7(d). For each of the 10 runs the rectangles were randomly placed. The results are shown in Table 2.

In both the random and crossroads examples, LOCOs outperform RVOs as far as connections are concerned, though oftentimes LOCOs take more time to solve the specified problem. On different problems, both approaches can take some extra time to complete the problem for two different reasons. The VOs take extra time as agents will sometimes get caught on obstacles or are pushed away from their goals by agents on other teams. LOCOs may take extra time as they spend effort navigating agents in densely-packed situations that occur at the center of the environment as they cross over. Furthermore, LOCOs exhibit a flocking behavior which sometimes causes agents to overshoot their goals.

	time per step (s)		collisions		maintained links		steps		success	
	LOCO	RVO	LOCO	RVO	LOCO	RVO	LOCO	RVO	LOCO	RVO
random	0.03005	0.02919	0	0.1	96%	72%	2840.78	2416.3	90%	100%
crossroads	0.00564	0.0034	0	0	98%	71%	2871.1	2596	100%	100%

Table 2: Results for multiple teams.

In the random obstacle environment, LOCOs imposed a very small computational overhead. The crossroads scenario resulted in a more significant increase, because all agents meet in the center of the environment nearly at the same point in time. This increases computational cost, as LOCOs must tune their horizon and reconnect broken proximities more frequently. In both environments, LOCOs and RVOs resulted in no collisions, with the exception of one run of RVOs in the random environment. The main focus of these results, however, is the improvement in maintained links. The LOCO-based approach maintained 95% of the proximity links, which was an improvement over RVOs. LOCOs cause a small degradation in path quality, which is measured in the number of steps it took for agents to solve the environment. In addition to this, LOCO failed to solve one of the randomly generated environments, because agents tend to spread to the edge of their proximity constraints, which may cause problems when agents move around obstacles. Overall, the results seem to validate the initial approach, as the goal of LOCOs is to maintain safety while providing better proximity maintenance is experimentally supported.

6. DISCUSSION

This work provides the formal definition for a new kind of obstacle in the velocity space of moving agents, referred to as a Loss of Communication Obstacle (LOCO), which correspond to proximity constraints with neighboring agents. These obstacles can be easily computed and integrated into the existing framework of Velocity Obstacles for decentralized collision avoidance. Additionally, an approach for tuning the time horizon parameter for these obstacles over multiple simulation steps is provided. These additional constraints increase the overall coherence of teams of agents while navigating through environments with static obstacles and other moving bodies. LOCOs can be dropped if it is determined that it is too difficult to maintain proximity without jeopardizing safety. The implementation of the technique shows improved coherence for agents who share communication links without sacrificing safety at a small computational overhead.

A natural extension is to consider more challenging systems, including kinematically and dynamically constrained systems. Adapting LOCOs to work in these cases is possible, as there have already been extensions on using VOs with dynamics. Since these extensions are in an orthogonal direction to the LOCO algorithm, it is easy so as to achieve decentralized coherence maintenance for dynamic systems. Further extending the work to applications of real robots will require introducing a method for handling sensor errors, as well creating a more robust reconnection strategy built on this error model. One drawback of reactive techniques is that they may get stuck in local minima. It is interesting to better study the integration with the wavefront approach or another global planner so as to guarantee that agents will make progress towards their goal while guaranteeing safety and connectivity. A global planner can also improve the performance of the reconnection strategy. Currently, the direction to the agent with which connectivity has been lost

ignores the existence of local obstacles. Another interesting direction is to study reciprocity tuning. More constrained agents, such as those who create a bridge for two otherwise disconnected agents, may be less able to reciprocate than others. Further reducing the computational overhead of the technique would also have great benefits, since larger-scale tests could be performed and have practical application areas, as in crowd simulation and video games.

7. REFERENCES

[1] T. Balch and M. Hybinette. Social potentials for scalable mulit-robot formations. pages 73–80, 2000.

[2] T. D. Barfoot and C. M. Clark. Motion planning for formations of mobile robots. *Journal of Robotics and Autonomous Systems*, 46(2), Feb. 2004.

[3] K. E. Bekris, K. I. Tsianos, and L. E. Kavraki. Safe and distributed kinodynamic replanning for vehicular networks. *Mobile Networks and Applications*, 14(3), February 2009.

[4] 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*, 41(2):89–99, 2002.

[5] 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)*, pages 4222–4227, 2003.

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

[7] R. Fierro, C. Belta, J. Desai, and V. Kumar. On controlling aircraft formations. In *Decision and Control, 2001. Proceedings of the 40th IEEE Conference on*, volume 2, pages 1065 –1070 vol.2, 2001.

[8] P. Fiorini and Z. Shiller. Motion planning in dynamic environments using velocity obstacles. *Int. Journal of Robotics Research*, 17(7), 1998.

[9] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, 4(1), 1997.

[10] D. Helbing, L. Buzna, A. Johansson, and T. Werner. Self-organized pedestrian crowd dynamics: Experiments, simulations and design solutions. *Transportation Science*, 2005.

[11] F. Large, C. Laugier, and Z. Shiller. Navigation among moving obstacles using the NLVO: Principles and applications to intelligent vehicles. *Autonomous Robots*, 19(2), 2005.

[12] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[13] S. M. LaValle. *Planning Algorithms*. Cambridge, 2006.

[14] M. A. Lewis and K.-H. Tan. High precision formation control of mobile robots using virtual structures. *Auton. Robots*, 4:387–403, October 1997.

[15] J. M. Lien, S. Rodriguez, J. P. Malric, and N. Amato. Spepherdin behaviors with multiple shepderds. In *Intern. Conf. on Robotic and Automation*, 2005.

[16] S. Paris, J. Pettre, and S. Donikian. Pedestrian reactive navigation for crowd simulation: A predictive approach. *Computer Graphics Forum*, September 2007.

[17] N. Pelechano, J. Allbeck, and N. Badler. Controlling individual agents in high-density crowd simulation. In *ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA)*, volume 3, pages 99–108, San Diego, CA, 2007.

[18] N. Pelechano, J. Allbeck, and N. Badler. *Virtual Crowds: Methods, Simulation and Control*. Morgan and Claypool Publishers, 2008.

[19] J. Pettre, J. Ondrej, A.-H. Olivier, A. Cretual, and S. Donikian. Experiment-based modeling, simulation and validation of interactions between virtual walkers. In *Symposium on Computer Animation*. ACM, 2009.

[20] C. W. Reynolds. Steering behaviors for autonomous characters. In *Proc. of the Game Developers Conference (GDC)*, pages 763–782, San Jose, CA, 1999.

[21] D. Silver. Cooperative pathfinding. In *The 1st Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE'05)*, pages 23–28, 2005.

[22] J. Snape, S. J. Guy, J. van den Berg, S. Curtis, S. Patil, M. C. Lin, and D. Manocha. Independent navigation of multiple robots and virtual agents. In *Proc. of the 9th Int. Conf. on Autonomous Agents and Multiagents Systems (AAMAS 2010)*, Toronto, Canada, May 2010.

[23] J. Snape and D. Manocha. Navigating multiple simple-airplanes in 3d workspace. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, May 2010.

[24] N. Sturtevant and M. Buro. Improving collaborative pathfinding using map abstraction. In *The Second Artificial Intelligence for Interactive Digital Entertainment Conference (AIIDE'06)*, pages 80–85, 2006.

[25] D. Thalmann. Populating virtual environments with crowds. In *Int. Conf. on Virtual Reality Continuum and Its Applications*. ACM, 2006.

[26] J. van den Berg, S. J. Guy, M. Lin, and D. Manocha. Reciprocal n-body collision avoidance. In *Proc. Int. Symposium of Robotics Research*. International Foundation on Robotics Research, aug. 2009.

[27] 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.

[28] J. Van den Berg, J. Snape, S. Guy, and D. Manocha. Reciprocal Collision Avoidance with Acceleration-Velocity Obstacles. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, May 2011.

[29] C. Vo, J. F. Harrisong, and J. M. Lien. Behavior-based motion planning for group control. In *Intern. Conf. on Intelligent Robots and Systems*, St. Louis, Mo, 2009.

[30] K.-H. C. Wang and A. Botea. Fast and Memory-Efficient Multi-Agent Pathfinding. In *International Conference on Automated Planning and Scheduling (ICAPS)*, pages 380–387, Sydney, Australia, 2008.