Multi-Agent Group Behaviors using Roadmap-Based Techniques

Daniel Latypov, Andres Medina, Karen Poblete, Colton Revia,
Sam Rodriguez, Andrew Giese, Jory Denny, Nancy Amato
Department of Computer Science, Texas A&M University, College Station, Texas 77840, USA
daniel.latypov@gmail.com, nrxus@tamu.edu, kpoblete@cse.tamu.edu, crevia1@gmail.com
smallsor8786@neo.tamu.edu, agiese@cse.tamu.edu, jdenny@cse.tamu.edu, amato@cse.tamu.edu

***Abstract*—In robotics, multiple robots frequently collaborate together in order to accomplish a common goal that might normally be too difficult or time-consuming for individual agents. In this paper, we present a novel framework for modeling group behaviors using a set of abstract components. On the highest level a simulator coordinates the behaviors and the transition between states. The behaviors in turn specify a goal and movement for each agent. We utilize roadmap-based planners to provide a path for each agent through the environment. This path may span over simple planar or multilevel environments such as an office building. The local controller applies the correct controls to the agent to approximately follow this path while avoiding obstacles and other agents. We then take these modified controls and apply motion models, an abstraction of agent motion to move the agent's state. The agents have knowledge of their state through sensors which publish data to information sources. We demonstrate three common behaviors in this framework: reciprocal velocity obstacles, frontier search, and large scale evacuation in simple and multilevel environments.**

I. Introduction

Multi-agent systems are commonly found in nature and are the subjects of ongoing research [6], both in trying to simulate them (graphics) or trying to mimic them (robotics). Imagine a search and rescue situation. Instead of sending just one agent (whether a human or a robot) to search through a large area, sending multiple agents will make the search for survivors as fast as possible if properly coordinated, making them highly important in scenarios where time is critical. The question remains, however, of how to optimally search an environment with a given number of agents. To help study this and a myriad of other problems, we present a scalable framework for modeling such multi-agent systems both virtually and physically.

In robotics, multiple agents frequently collaborate as in the example above to accomplish a common goal which might normally be too difficult or time-consuming for any one agent.

In graphics, it is common to try to simulate the behavior of real-world group such as a flock of birds,a herd of sheep, or a group of evacuees. The method for simulation needs to be efficient and scalable so that it is able to handle large amount of agents being simulated at once (e.g., 500 birds flying). These models provide information which can be used in applications ranging from games and computer graphics to traffic simulation and building design or even training for emergency personnel.

The goal of our work is to provide a scalable framework using an agent-centric approach and roadmap-based methods in order to replicate authentic real-world scenarios of arbitrary complexity such as multi-level environments. Our framework will be used to model different kind of behaviors including flocking [2], searching [11], caravanning [3], and evacuation [8].

II. Related Work

Just as the motivation for this framework comes from robotics and graphics, we apply two important concepts from these fields in our research. From robotics, we borrow roadmap-based motion planning techniques which enable robots to find their way through an environment. From graphics, we borrow the concept of flocking which is a scalable technique that enables us to efficiently handle large number of agents.

1) Roadmap-Based Path Planning with PRMs [5]: The motion planning problem is to find a feasible path that takes an agent from a start state to a goal state. As mentioned before, our approach utilizes probabilistic roadmaps to provide the overall path which the agents have to try to follow while avoiding collision with their own local controllers.

Briefly, PRMs work by randomly sampling points from the robots configuration space (C-space) [7], and retaining those that satisfy certain feasibility requirements (e.g., they must correspond to collision-free configurations of the movable object). Then, these points are connected to form a weighted graph, or roadmap, using some simple local planning method to connect nearby points, if possible. During query processing, the start and goal are connected to the roadmap and a path connecting them is extracted from the roadmap using standard graph search techniques.

Our framework is configurable to use different motion planners, such Medial-Axis PRM [10], Obstacle PRM [1], etc.

2) Flocking: Flocking behaviors are common in nature and there are ongoing research efforts to simulate such behaviors in computer animation and robotics applications. Generally, such work only considers behaviors that can be determined independently by each flock member solely by observing its local environment, e.g., the speed an agent has, without

any form of communication. Since flock members are not assumed to have global information about the environment, only very simple navigation and planning techniques have been considered for such flocks [2].

III. Framework

In order to model several classes of behaviors efficiently and generally, we present a new framework of eight abstract components. These abstractions allow us to easily handle a variety of behaviors acting on both physical and virtual agents. Our framework makes use of a simulator that controls the update of each behavior. Behaviors are whatever you wish to simulate, whether it is searching, swapping places with other agents, moving in circles, exiting, etc. Each behavior owns a set of agents, giving the behavior control over the agents' motion and knowledge. The remaining components of our framework can be divided into two parts: motion components and information components. The motion components let the agent know how to move, while the information components help the behavior decide the next goal for the agent. Following is a more detailed description of each component in our framework.

- **Simulator:** Coordinates the transition between timesteps by calling the behaviors to update and resolves collisions.
- **Behavior:** Defines how the robots acts and what it wants to do by specifying a goal state and reweighting the roadmap's edges as necessary.
- **Global Planner:** Uses a roadmap to compute a path for the agent to get to the goal specified by its behavior.
- **Local Controller:** Chooses how the robot gets from the place to place as specified by the Global Planner while avoiding obstacle and interagent collisions.
- **Motion Model:** Describes mathematically how the robot moves in response to the controls applied by the Local Controller. This abstraction allows us to work with both holonomic and nonholonomic agents or other agents with limited controls (e.g. Dubin's cars) in a general way.
- **Agent:** The individual robot or actor in the simulation. It is composed of a state and knowledge. It uses its sensors to gain knowledge of the environment.
- **Sensor:** Anything that can collect and publish information to one or more Information Sources
- **Information Source:** A piece of named data about an Agent that can be requested and used by other components in the simulation, e.g. the agent's nearest neighbors, visible nodes in the roadmap, etc. This is also known as the knowledge of the agent.

To demonstrate our framework, we show an example of Shakey's Sense Plan Act model in Figure 1. In our framework, the simulator resolves collisions and coordinates the transition between timesteps by calling the behaviors to update their agents. The behavior provides the agent a goal to reach in the planning stage, and the global planner, a tool the behavior uses, provides a path to the goal. The local controller modifies the agents trajectory slightly to try to avoid collisions with other agents and obstacles while staying as true as possible to the path. This modified trajectory is then given to the agent's motion model, which describes the motion (physics) of the agent. For example, it allows our framework to work equally well with holonomic and nonholomic robots such as cars. The global planner, local controller, and motion model interact to constitute the "Act" stage of the loop. After updating the state, the "Sense" stage begins with the sensors collecting information about the new state and publishing them to the information sources. The information sources are accessible by any other component and store the knowledge of each agent. The behavior then uses this information from the sensing stage to reassess the situation and provides the agent with a new goal. This completes the sense-plan-act loop and is re-started whenever the simulator calls for an update of the behavior.

IV. Example Behaviors

Our framework is able to support multiple classes of behaviors. In this section we are present three major behaviors that represent the capabilities of our framework.

A. Frontier Exploration

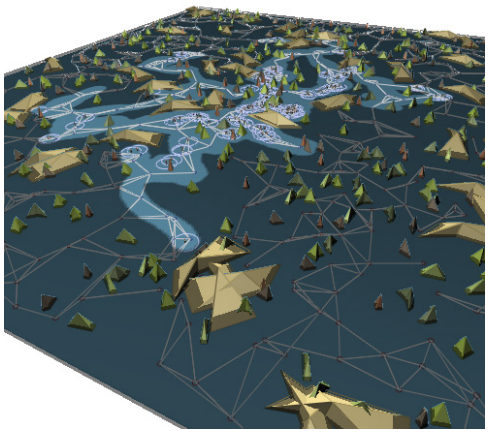

Fig. 2. A group of agents exploring an environment. The dark areas represent the unexplored regions.

The goal of any exploration behavior is to explore a given environment in a way that is efficient and fast. Frontier exploration makes use of an implicit way of communication by "marking" nodes as uncleared, cleared or frontier. **Uncleared** nodes are the nodes in the roadmap that have not been seen by any agent. **Cleared** nodes are the nodes that the agent has seen with its vision sensor. **Frontier** nodes are those in between cleared and uncleared nodes. In our implementation we define frontier nodes as the uncleared successor nodes to the cleared nodes.

The map starts with all the nodes uncleared. Each agent will independently call for an update of its sensor and will mark and clear the nodes that it can see and set nodes as frontier and add them to the list. The list of frontier nodes is owned

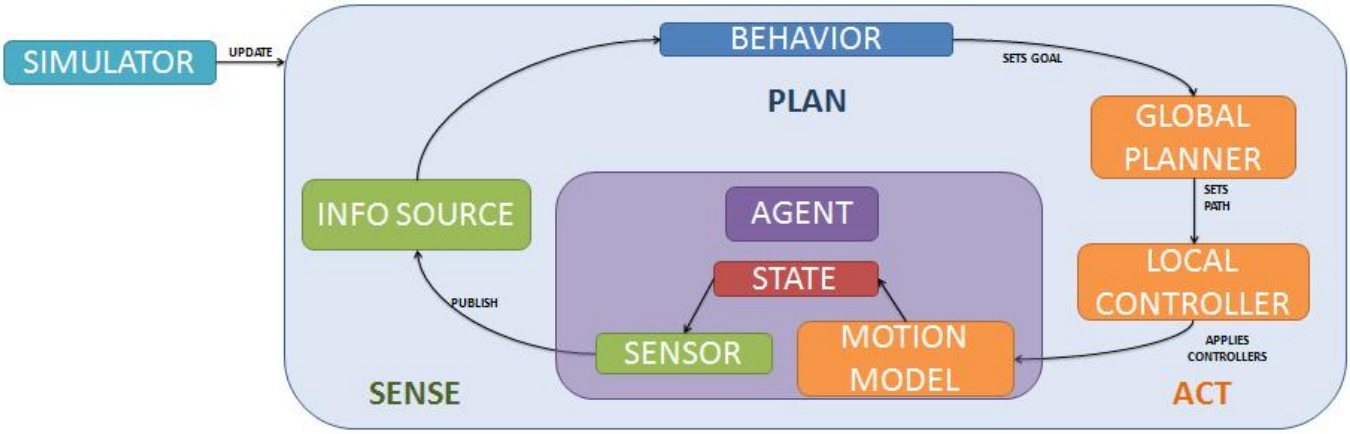

Fig. 1. Here is a simple example of the standard sense-plan-act loop in our framework.

by the behavior and therefore is "shared" information for all the agents. The agent will then choose to go to its closest frontier node. Once the frontier node has been cleared, the agent will pick a different cleared node. This process will be repeated until all nodes have been cleared. Figure 1 shows a partially explored map. The bright areas represent the cleared portions and the frontier is the border between the uncleared and cleared sections.

B. Large scale Evacuation

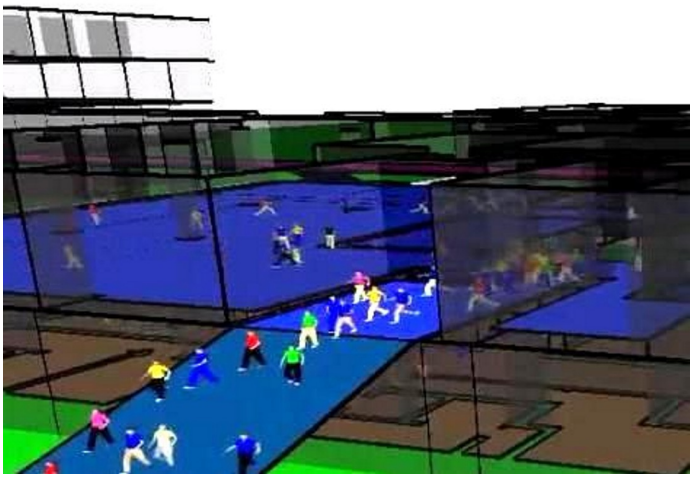

Fig. 3. Agents evacuate from a multi-level environment.

The goal of evacuation behavior is to simulate how some n number of agents arrive to a **safe area** from a set of m safe areas using known **exits**. In our implementation, the safe areas as well as the exits are information sources which are shared for all the agents. The agent also has information on which safe area it is trying to go to (once it has chosen a safe route) and whether it has reached it or not. Once an agent reaches a safe area it finds a random point within that same safe area to go to so as to not be an obstacle to other agents going to the same point. In our current implementation we assume that the exits are neccesary to reach a safe area. In the future, we would like to improve this method by having the agent figure out if an exit is neccessary to reach the safe area or not.

C. Reciprocal Velocity Obstacles

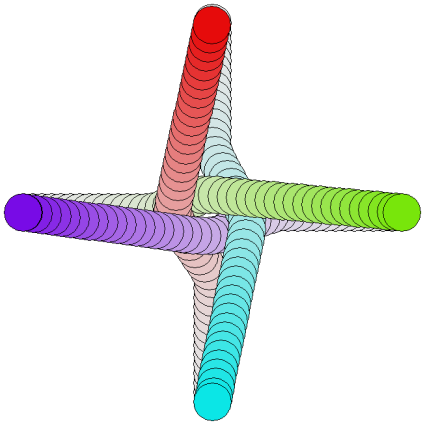

Fig. 4. An example of four agents exchanging anitpodal positions. Notice the symmetry inherent in the algorithm manifests itself in the paths, leading to fairness.

Reciprocal velocity obstacles are an extension of the original work of Fiorni and Shiller on velocity obstacles by van der Burg, et al. The approach views the problem of local collision avoidance between heterogeneous agents as a constrained optimization problem. The algorithm must select the velocity closest to a desired velocity that will not result in a collision between the agent within some input time horizon. Geometrically, this is understood as applying linear programming on the polytope defined by the set of velocities resulting in collision between the agent an its neighbor (a velocity obstacle) to choose the next velocity [4]. The agents can expect the others

to reciprocate their behavior and average the selected velocity and its current, optimal velocity [9].

V. Discussion

While implementing the behaviors previously discussed we found acertain similarities between all of them. Most behaviors need information, whether it is what can the agent now see, what areas are safe for the agent to go, what agents are nearby and at what velocity, etc. This fits well within our framework due to the abstraction of Information Sources which can be publicly known throughout all agents or private to each agent. The behavior then sets what the goal is to "accomplish" the desired behavior. The behavior doesn't need to know how to get to that goal. This separation between goal and path allows for the abstraction of a separate global planner that is distinct from the behavior. In the future we would like to explore more complex behaviors to see if the design our sense-plan-act loop fits well with a broader range of behaviors.

VI. Conclusion

We present a scalable roadmap-based framework for modeling group behaviors. We have implemented different classes of behaviors in our framework to test its efficacy in adding new behaviors. In the future we plan to model more complex and interesting behaviors to further challenge our framework. We also plan to do a study to compare it against the previous framework used in our lab.

References

[1] Nancy M. Amato, O. Burchan Bayazit, Lucia K. Dale, Christopher Jones, and Daniel Vallejo. Obprm: an obstacle-based prm for 3d workspaces. In *Proceedings of the third workshop on the algorithmic foundations of robotics on Robotics : the algorithmic perspective: the algorithmic perspective*, WAFR '98, pages 155–168, Natick, MA, USA, 1998. A. K. Peters, Ltd.

[2] O. B. Bayazit, J.-M. Lien, and N. M. Amato. Roadmap-based flocking for complex environments. pages 104–113, Oct 2002.

[3] Jory Denny, Andrew Giese, Aditya Mahadevan, Arnaud Marfaing, Rachel Glockenmeier, Colton Revia, Samuel Rodriguez, and Nancy M. Amato. Multi-robot caravanning. November 2013. To appear.

[4] Paolo Fiorini and Zvi Shillert. Motion planning in dynamic environments using velocity obstacles. *International Journal of Robotics Research*, 17:760–772, 1998.

[5] L.E. Kavraki, P. Svestka, J.-C. Latombe, and M.H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *Robotics and Automation, IEEE Transactions on*, 12(4):566–580, 1996.

[6] Jyh-Ming Lien, Samuel Rodriguez, Xinyu Tang, John Maffei, and Arnaud Masciotra. Composable group behaviors. Technical Report TR05-006, 2005.

[7] T. Lozano-Pérez. Spatial planning: A configuration space approach. *Computers, IEEE Transactions on*, C-32(2):108–120, 1983.

[8] Samuel Rodriguez and Nancy M. Amato. Behavior-based evacuation planning. pages 350–355, 2010.

[9] J. van den Berg, Ming Lin, and D. Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1928–1935, 2008.

[10] S. A. Wilmarth, N. M. Amato, and P. F. Stiller. MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space. volume 2, pages 1024–1031, 1999.

[11] Brian Yamauchi. Frontier-based exploration using multiple robots. In *International Conference on Autonomous Agents (Agents '98)*, pages 47–53, 1998.