# Parallel Simulation Testbed for Swarming MAVs

Ge Li

Institute for Automation, National University of Defense Technology
Changsha, Hunan, China 410073
geli@nudt.edu.cn

**Abstract.** The objective of the research presented in this paper is to develop an applicable parallel simulation testbed for studying the behavior of autonomous agents and their collective interaction with relatively high fidelity and speedup. A parallel simulation testbed for swarming Micro Air Vehicles (MAVs) is built based on a parallel simulation engine developed by us called KD-PARSE. The framework is described and object-oriented models are discussed. Issues in Parallel Discrete-Event Simulation, such as speed-up and processor distributions, are discussed and analyzed. Finally scenarios are executed in MAV simulation. Results show parallel simulation cannot guarantee high speed-up, and our solution is provided.

**Keywords:** Parallel simulation, Testbed, Swarm, MAV

## 1 Introduction

Swarms of MAVs(Micro Air Vehicle) provide an added level of robustness, fault tolerance, and flexibility over individuals, as the failure of one MAV does not result in failure of the task, as long as the remaining MAVs can redistribute and share the tasks of failed MAV.The complex and highly interdependent nature of the behavior of MAVs makes it difficult to be simulated. To achieve accuracy and high fidelity, complex models are often used in simulating the behavior of MAVs [3] [4] [6]. As the number of MAVs to be simulated increase, the performance suffers drastically from the computation. More often, researchers choose Parallel Discrete-event Simulation (PDES) paradigm when dealing with large simulation applications. PDES is concerned with execution on multiprocessor computing platforms containing multiple processors that interact frequently. The fundamental challenge of PDES is to efficiently process events concurrently on multiple processors while preserving the overall causality of the system as it advances in simulated time. This problem is solved using synchronization mechanism that ensures simulations on all processors being synchronized at any time when they must interact or exchange data [1]. A Parallel Simulation Engine(KD-PARSE) developed by us provides an excellent platform to distribute this multiple-MAV simulation across clusters.

## 2 Parallel Simulation Testbed for Swarming MAVs

The KD-PARSE framework is an object-oriented PDES Framework. KD-PARSE supports multiple time management approaches to guarantee logically correct time management in a distributed simulation. Supported approaches utilize lookahead and rollback in various schemes to provide many forms of optimistic time managements. Disabling the rollback mechanism at run time permits conservative time management to synchronize event processing. KD-PARSE serves as the environment for developing PDES simulation of MAVs.

### 2.1 Object-oriented MAV Simulation Framework

MAV simulation framework should defining state variables of MAV, defining events of MAVs, and events handlers to each event. We try to separate the simulation framework and the MAV models and make the framework extendable to new models and parameters.

Each MAV is an independent individual, which receives information, manipulates according to some control rules, and publishes information to others. They interact with each other to perform some collective missions. The MAV simulation framework is shown in Figure 1.

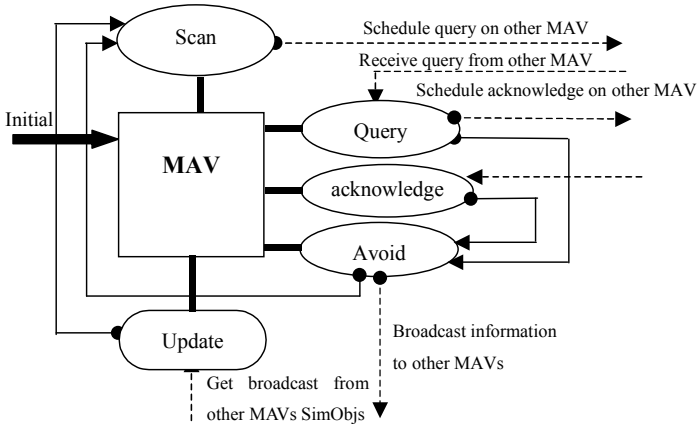


Fig. 1. MAV simulation framework

MAV object has five event handlers. In an update event, information from other MAV is stored in a table. A scan event is scheduled to process this information.

In the scan event, one MAV checks the possibility of encountering other MAVs based the MAV information stored in the table. If another MAV falls into its detection region (either at present time or some time in future), a query event will be scheduled to that MAV at that certain point in time.

In a query event, MAV verifies an encounter with the MAV by which this query event is scheduled. An acknowledge event with the verification result is scheduled back to the MAV. An avoid event is scheduled to start avoidance motion, if the result is positive.

In an acknowledge event, MAV receives verification result from other MAVs, and schedule an avoid event to itself if the result is positive.

In the avoid event, MAV manipulates the avoidance movement until it collides with other MAV or no other MAVs remains in its detection region. Based on a predefined set of control rules, MAV adjusts thrust and/or bank angle, thus changing its speed and heading angle. Upon the end of avoidance motion, MAV publishes its information to others and schedules a scan event to itself.

### 2.2 MAV model

MAV models include motion dynamics and behavior dynamics model. MAV behavior dynamics model focus on the collective behavior of swarms of MAVs or other robots. Wu et al [5] utilized Genetic Algorithm in an evolved controller to dynamically distribute a group of MAVs appropriately in a surveillance area for maximum coverage. Spears et al [7] explored the performance of rule-based control strategies on multi-asset surveillance. Lin et al [2] also studied the behavior control of the Unmanned Combat Air Vehicle's swarm. They used Monte Carlo simulation in conjunction with GA to evolve the robust control when wind-gust disturbance exists. MAV models in those works have less fidelity because the battlefield is composed of grid points, and the MAV moves on the grip points only with fixed step size.

**Dynamic Model.**

In this research MAVs are defined to move freely in the x-y plane. A MAV is set to have a constant mass of 0.211 kilograms, a maximum thrust of 1.0 Newton, and a negative drag force from the static air whose amplitude is related to the MAV's speed. The flat-earth two-dimensional point-mass airplane equations of motion and the kinematics equations are given below, where, $m$ is mass, $V$ is (absolute) speed, $T$ is thrust, $D$ is drag, $\phi$ is heading angle, $\psi$ is bank angle.

$$\mathrm{d}\,V/\mathrm{dt} = (\,\mathrm{T} - \mathrm{D}\,) / \mathrm{m}\,. \qquad (1)$$

$$\mathrm{d}\phi/\mathrm{dt} = \mathrm{G}\tan\psi / V\,.$$

**Behavior Model.**

We suppose every MAV possesses an array of eight optical sensors, each of which covers a 45-degree range, for detecting neighboring region. Since MAVs have simple and primitive sensors, we assume that those sensors can only perceive the presence of objects (other MAVs, boundaries, etc.) within the detecting sector. Sensor cannot detect the number or distance of objects within its range.

Furthermore, we will divide the neighborhood into a collision region and a detection region. We define the collision region as a 0.5-meter-radius circle around MAV. Whenever two MAVs detect each other in collision region, we will label them

as “dead” and then take them off the simulation. The radius of detection region is chosen to be 20 meters in the simulation.

The surroundings are divided into 8 sections; they are Front, Left, Rear, Right, Front-left, Rear-left, Rear-right, and Front-right. Whenever other MAVs fall into the surroundings, MAV will adjust its thrust and bank-angle, thus have the right speed and heading-angle to avoid crashing. Otherwise, it has zero bank-angle and a certain thrust, which will make its heading-angle and speed constant. According to the signals from detecting sensors, one or more of rules can apply at the same time:

- Rule 1. If there is MAV(s) in Front section, it will decrease its thrust by 10% and decrease bank-angle 10 degrees.
- Rule 2. If there is MAV(s) in Front-right, Right, Rear-right section, it will decrease its bank-angle by 5 degrees.
- Rule 3. If there is MAV(s) in Front-left, Left, Rear-left section, it will increase its bank-angle by 5 degrees.
- Rule 4. If there is MAV(s) in Front-left, Front, Front-right section, it will decrease its thrust by 5%.
- Rule 5. If there is MAV(s) in Rear-left, Rear, Rear-right section, it will increase its thrust by 5%.
- Rule 6. If there is MAV(s) in Rear section, it will increase its thrust by 5%.

# 3 Case Studies

## 3.1 Speedup VS. Number of MAVs

In this case MAVs are simulated in a 500-meter by 500-meter square field for duration of 500 seconds. As the number of MAVs in the simulation increases from 5, to10, 20, 30, 40, and 50, the time cost will increase too, but exponentially. This increment of CPU time cost is due to the increasing number of MAVs, as well as the increasing possibilities of MAV encountering others(Figure 2).

These runs are repeated on two processors using Time Warp synchronization approach. The simulation CPU time cost, which also increases exponentially as number of MAVs increases, is less than the time cost of simulation running sequentially. Speedup of each run is calculated and plotted in Figure 2. Speedup varies as number of MAVs increases, and the maximum speedup occurs when number of MAVs is about 20.

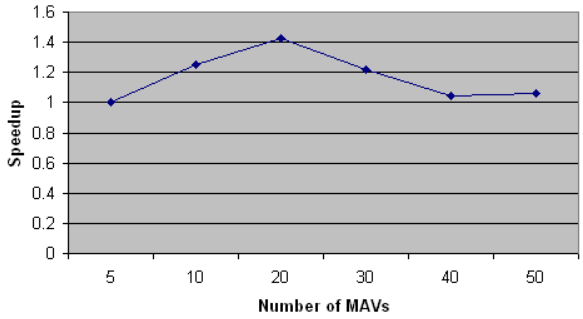


Fig. 2. Speedup vs. number of MAVs

### 3.2 Speedup

Speedup is defined as the serial execution time divided by the parallel execution time. Running simulation on multiple processors will not always shorten the time cost. When more processors are involved in simulation, more efforts are needed to get every processor synchronized, which will offset the expected speedup.

Scenario 2 is applied in this test: 24 MAVs flying in a 1200-meter by 600-meter area for duration of 120 seconds are simulated on 1, 2, 3, 4, 5, 6 processor(s) respectively. Since the simulation time cost may vary randomly, we repeat each case and use the average CPU time cost to calculate speedup.

To reduce the interaction between MAVs, the field is divided into six 200-meter by 600-meter subareas. MAVs are label from 0 to 23 and can only fly within their respective subareas (Figure 3).

| | | | | | | |
|---|---|---|---|---|---|---|
| 600m | 0 | 1 | 2 | 3 | 4 | 5 |
| | 6 | 7 | 8 | 9 | 10 | 11 |
| | 12 | 13 | 14 | 15 | 16 | 17 |
| | 18 | 19 | 20 | 21 | 22 | 23 |
| 0m | 200m | 400m | 600m | 800m | 1000m | 1200m |

Fig. 3. MAVs distribution in subareas

The speedup is plotted in Figure 4. It is obvious that when 24 MAVs are distributed to 2, 3, and 6 nodes, the speedup almost reaches its best performance. Actually we can see from Figure 3 that MAVs within the same subareas are distributed in the same node in these conditions, thus the interactions are constrained locally in one node. As there are no interactions between nodes, good speedups are achieved. On the other hand, when MAVs are distributed to 4, and 5 nodes, MAVs within the same subareas are distributed in different nodes, large amount of interactions and rollbacks show up so that reduce the speedup.

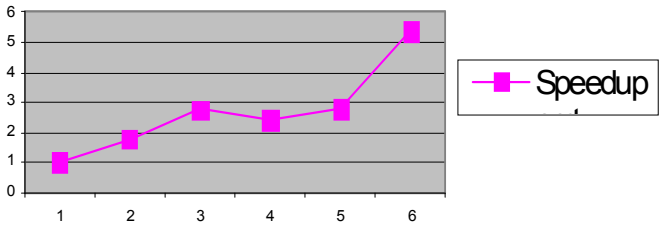


Fig. 4. Speedup

## 4 Conclusions

The parallel simulation testbed is feasible to study the motion behavior of multiple MAV. It separate the simulation framework and the MAV models and make the framework extendable to new models and parameters. Case studies also show that elaborate design of the MAV applications is crucial to good performance.

**Acknowledgments.** The work is partially supported by Chinese National Natural Science Foundation Grant No. 61074108.

## References

1. Li, G., Lin, K., Huang, K.: An Architecture to Facilitate the Development of Parallel and Distributed Simulation System. Advances in Modeling & Analysis. D, vol.10 No.1, AMSE 1-16 (2005)
2. Lin, K., Yu, H., Zhou, L., Xia, Z., Sisti, A., Alexander, S.: Robust Control of A Swarm of UCAVs. In: Proceedings of SPIE, Vol. 4716, pp.108-115(2002).
3. Corner, J. J., Lamont G. B.: Parallel Simulation of UAV Swarm Scenarios. In: Proceedings of the 2004 Winter Simulation Conference, pp.355-363(2004)
4. Russell, M. A., Lamont G. B., Melendez K.: On Using Speedes As A Platform For A Parallel Swarm Simulation. In: Proceedings of the 2005 Winter Simulation Conference, pp.1129-1137(2005)
5. Wu, A. S., Schultz, A. C., Agah, A.: Evolving control for distributed micro air vehicles. In: Proceedings of the 1999 IEEE International Symposium on Computational Intelligence in Robotics and Automation, pp.174-179 (1999)
6. Xia, Z., Lin, K.: Distributed Combined Discrete-Continuous Simulation for Multiple MAVs Motion Analysis. In: Proceedings of The 2003 International Symposium on Collaborative Technologies and Systems, Simulation Series, Vol. 35, Num. 1, pp.162-167 (2003)
7. Spears, W. M., Zarzhitsky, D., Hettiarachchi S., Kerr W.: Strategies for Multi-Agent Surveillance, http://www.cs.uwyo.edu/~suranga/research/pack.pdf