AUCTION-BASED FAULT-TOLERANT MULTI-ROBOT COOPERATION

Muhammad T. Khan*, Clarence W. de Silva**, Afzal Khan*
*University of Engineering and Technology Peshawar, Pakistan
**University of British Columbia, Vancouver, BC, Canada
*tahir, afzalkhan@nwfpuet.edu.pk
**desilva@mech.ubc.ca

ABSTRACT
This paper presents a market driven approach for robust multi-robot cooperation. The example task that is considered is object transportation from an original location to a goal location. The developed methodology relies on market-based decision making, which uses auction as a process of assigning a task to a robot by offering it up for a bid. In the event of robot failure during the execution of a task, the task is re-allocated to another suitable robot using the same auctioning process. This paper addresses two possible robot malfunctions—partial and full failures, which can happen any time during the execution of a task. The practicability of the developed methodology is demonstrated by implementing an auction-based approach on a team of simulated robots that cooperatively execute a task.

KEY WORDS
Multi-robot systems, robot cooperation, fault-tolerant robots, auction-based method.

1. Introduction

With recent innovation and progress in robotic technologies, mobile multi-robot systems have received increased attention among researchers. This research interest is further supported by the advantages of multi-robot systems, as they are: more efficient and time effective as they can work in parallel, robust and reliable as the failure of one robot will not completely halt the function of the entire system, and cost effective as it is more economical to build several robots having different capabilities to work cooperatively rather than having one complex robot with all the capabilities. These advantages come with numerous challenges, primary among which are robustness and fault tolerance of multi-robot systems. The need for automation has become more widespread in this technological age. With the increased deployment of robots, the systems responsible for making them work have also become complex. With the increasing interest in multi-robot systems, fault tolerance of these systems has become correspondingly important.

The failures in a multi-robot system can be of two types: partial failure and full failure. In partial failure, a robot loses its ability to use some of its resources but maintains the ability to use others. In this case the robot may be able to reallocate its task to another robot. In full failure of a robot, none of its subsystem will function and the robot will be unable to carry out any task, even partially. Detecting full failure is more challenging since a fully-failed robot will neither be able to detect its own failure nor will it be able to reallocate its task to another robot.

The failure of a robot in a multi-robot system can have an adverse effect on the performance of the overall system. To address this problem, in the present paper an auction-based approach is developed for multi-robot cooperation that is robust and fully fault tolerant. The team members are able to respond to failures of individual robots during the mission in order to efficiently complete the task and hence gracefully downsize the team performance.

2. Related Work


Robustness and fault tolerance are very crucial for a multi-robot system, particularly when operating in a dynamic and unknown environment. Robot teams require some level of robustness to individual failures as malfunctioning of a robot will have an adverse effect on the performance of the entire system.

Parker [4] proposed a behavior-based software architecture called ALLIANCE which facilitated fault-tolerant cooperative control of heterogeneous mobile robots performing a mission composed of loosely-coupled sub-tasks that may have ordering dependencies.
Bererton and Khosla [5] presented and analyzed a reliability-based model of a team of repairable robots. Gerkey and Mataric [6, 7] presented an auction-based fault-tolerant dynamic task allocation method for a group of robots. In this method an auctioneer robot auctioned a task to the group of robots in the system. The robot that won the task based on some metrics was awarded a time-limited contract to execute the task. The time-limited contract provided a built-in time out that could trigger fault detection and recovery. Dias et al. [8] studied three possible failures of robot; i.e., communication failure, partial failure, and full failure in a multi-robot team. Christenson et al. [9] presented a distributed approach that allowed the robots in a swarm robotic system to detect the failure among them. They adopted the firefly approach to detect the failure. Specifically, they presented a synchronizing flashing protocol as an exogenous fault detection tool.

3. Proposed Approach

3.1 Autonomous Market-based Multi-Robot Cooperation

Tahir-Khan et al. [10] developed a market-(auction-) based approach for autonomous multi-robot cooperation. A brief overview of the developed methodology is presented here. Consider a team of heterogeneous robots searching for a task in an unknown, dynamic and unpredictable environment. When a task is found, if the robot has the necessary capabilities to execute it, it attempts to perform the task alone. However, if the robot realizes that it is unable to complete the task alone, it auctions off the task. Other robots in the communication range submit their bids to auctioneer robot. The auctioneer robot carefully evaluates all the bids based on the bidding function. Finally the auctioneer robot declares the winner and assigns the task to the winner robot. For the process of choosing the winner robot, coordination among the robots is essential. The winner robot approaches the task to help the auctioneer robot in order to complete the task cooperatively. During cooperative task execution, the auctioneer and the winner robots coordinate with each other until the task is completed. Hence, for cooperative task execution, coordination among robots is essential before and during cooperation. Various algorithms for fault tolerance before and during cooperative task execution are presented now.

3.2 Algorithm for Failure of Winner Robot while Approaching the Task

1. \( R_w \) fails while approaching the task
2. \( \text{if} (R_w \text{ failure} == \text{Partial Failure } ) \)
3. \( R_w \text{ Notify}("R_w \text{ fails"}, R_x) \)
4. \( R_x \text{ ally} = \text{NULL}. \)
5. \( \text{Auction}(R_x, t_i) \)
6. \( \text{END} \)

7. \( \text{else if} (R_w \text{ failure} == \text{Full Failure}) \)
8. \( \text{After } R_x \text{ reaches threshold,} \)
9. \( \text{Auction}(R_x, t_i) \)
10. \( \text{END} \)

When a winner robot \( R_w \) is selected by the auctioneer robot \( R_y \), the winner robot sends messages at regular intervals to the auctioneer robot so that the auctioneer robot can monitor the movements of the winner. The winner robot may fail while approaching the task. In case of partial failure, the winner robot notifies the auctioneer robot about its failure. If the auctioneer robot does not receive any messages from the winner robot and the threshold is reached, the auctioneer robot assumes that the winner robot has failed completely. Upon this, the auctioneer robot will auction the task once again to pick another robot for cooperation. The considered specific example is the transportation of an object from its original location to a goal location.

3.3 Algorithm When Auctioneer Robot Fails After Announcing the Winner

1. \( R_x \) fails after winner is selected and before winner arrival
2. \( \text{if} (R_y \text{ failure} == \text{Partial Failure}) \)
3. \( R_y \text{ Notify}("R_y \text{ fails"}, R_w) \)
4. \( R_w \text{ ally} = \text{NULL}. \)
5. \( \text{else if} (R_y \text{ failure} == \text{Full Failure}) \)
6. \( R_w \text{ continuously sends approach messages.} \)
7. \( R_w \text{ arrives. } R_w \text{ Send}(A, R_y, t_i) \)
8. \( R_x \text{ declare } R_w \text{ failed after reaching threshold.} \)
9. \( \text{END} \)

An auctioneer robot can fail after choosing the winner robot. If it is a partial failure, the auctioneer robot notifies the winner robot about its failure. If the auctioneer robot does not respond at all and the threshold is reached, the winner robot declares that the other robot has failed completely. After detecting the failure and upon reaching the task, the winner robot becomes the auctioneer robot and chooses another robot for cooperation.

3.4 Algorithm for Failure of Auctioneer (Leader) Robot during Cooperative Task Execution

1. \( R_x \) can fail at any time during cooperative task execution
2. \( \text{if} (R_x \text{ failure} == \text{Partial Failure} ) \)
3. \( R_x \text{ Notify}("R_x \text{ fails"}, R_y) . \)
4. \( R_y = \text{leader.} \)
5. \( \text{Auction}(R_y, t_i, \{R\}) \)
6. \( \text{END} \)
7. \( \text{else if} (R_x \text{ failure} == \text{Full Failure}) \)
8. \( R_y \text{ reaches threshold on no progress} \)
9. \( R_y = \text{leader.} \)
10. \( \text{Auction}(R_y, t_i, \{R\}) \)
11. \( \text{END} \)
During object transportation, the auctioneer becomes leader robot \( R_x \) and the winner becomes the follower robot \( R_y \). The leader robot guides the follower robot from time to time during transportation. The follower robot does not take any decision by itself, but rather follows instructions given by the leader robot, such as: stop, turn, wait, and so on. The leader robot or the follower robot may fail at any time during cooperative execution of a task. If the failure of the leader robot is of the type partial failure, the leader robot notifies the follower robot about the failure. Then the follower robot becomes the auctioneer robot and announces the task to the group. If the leader robot does not perform at all or does not help in the task execution, the follower robot will know of the problem and will declare that the leader robot has completely failed.

The same algorithm with minor changes may be applied to the situation of failure of a follower robot. If the follower robot does not perform as expected, the leader robot declares that the follower has failed, and re-auctions the task.

4. Simulation Results and Discussion

4.1 Simulation Platform

The developed methodology is validated by simulating a team of mobile robots which transport a variety of objects to a goal location in an unknown and dynamic environment. The multi-robot test environment is developed in Java (Fig.1). It consists of black colored robots and brown colored objects to be transported. Robots may or may not require cooperation to transport objects to a goal location. The red color represents randomly-distributed static obstacles. The robots themselves are dynamic obstacles to other robots. The yellow color represents the goal location.

4.2 Results and Discussion

Figs 2-5 present typical results obtained from the simulation studies. In all the cases shown here, the simulations are run 1000 times for a specific number of robots, and the average time (steps) computed is shown on the label of the Y-axis in the figures.

The issue of fault tolerance in cooperative task execution is considered now. Fig.2 shows the result of failure when the winner robot fails during coordination prior to transportation before it reaches the object that needs to be transported cooperatively. Figure shows the time (steps) incurred in different types of failures. The time (steps) taken in partial failure is less since the malfunctioned winner robot communicates its failure to the auctioneer robot. However, in full failure, the time (steps) taken is greater since the auctioneer robot has to wait longer until the motivation level reaches the threshold value, before declaring that the winner robot has failed completely. As shown in the figure, the time (steps) increases with failure. However, the system manages to

![Fig.1: The multi-robot multi-object simulation platform.](image-url)
cope with all the failures at the expense of extra time (steps).
Fig. 3 shows the results for failure of the auctioneer robot during coordination before transportation. The multi-robot system took 389 time steps to transport all objects to the goal location as compared to full failure when the system spent 870 time steps to complete the mission. The auctioneer robot monitors the movements of the winner robot until it reaches the task location. During the movement of the winner robot towards the task, it does not know anything about the health of the auctioneer robot. Once it reaches the object, the winner robot sends an arrival message to the auctioneer robot in order to start the transportation process. Upon not receiving a reply or not observing the requested action, the winner robot assumes that either the auctioneer robot has failed completely or two robots are not adequate to transport the object. In either case, the winner robot becomes the auctioneer robot and re-auctions the task. It is evident from the figure that in case of partial failure, the time (steps) taken is less than for full failure, since the auctioneer robot is able to communicate its failure to the winner robot.

Fig. 4 shows a case of failure of the follower robot during cooperative transportation of an object. Here it is pertinent to mention that one of the cooperating robots is the leader and the other is the follower. The leader robot guides the follower whenever necessary until the task is completed (i.e., until the object is transported to a goal location). All communication will be one way, from the leader to the follower. When the follower robot does not follow the leader robot’s commands and the motivation level reaches the threshold, the leader robot declares that the follower robot has failed completely. Then the leader robot re-auctions the task to get the help of another follower robot. Again, as expected the system takes more time steps due to full failure as compared to partial failure.

Fig. 2: Effect on execution time (steps) due to partial or full failure of winner robot.

Fig. 3: Effect on execution time (steps) due to partial or full failure of auctioneer robot.

Fig. 4: Effect on execution time (steps) due to partial or full failure of follower robot.
The spikes and overshoots seen in the figures are due to the fact that the robots in the “explore” state wander in the work environment, causing interference from time to time during transportation of an object. If a wandering robot comes into the transportation path, it is perceived by the cooperating robots as a dynamic obstacle. As the robot movement is random and unpredictable, it takes increased coordination and time (steps) between the two cooperating robots to avoid the obstacle. Moreover, due to failures, when the malfunctioned robot is replaced by a healthy one, time (steps) taken by the healthy robot to reach the task increases, as is seen in the form of overshoots in the figures. Overshoots also arise when a robot gets stuck at corners or in between robots.

5. Conclusion

In this paper, algorithms for fault tolerant multi-robot cooperation were presented, which employed a market-(auction-) based approach. In the present work both partial and full failures were introduced into the robots during different stages of task execution, it is perceived by the cooperating robots as a dynamic obstacle. As the robot movement is random and unpredictable, it takes increased coordination and time (steps) between the two cooperating robots to avoid the obstacle. Moreover, due to failures, when the malfunctioned robot is replaced by a healthy one, time (steps) taken by the healthy robot to reach the task increases, as is seen in the form of overshoots in the figures. Overshoots also arise when a robot gets stuck at corners or in between robots.

Acknowledgements

This work was supported by research grants from the Natural Sciences and Engineering Research Council (NSERC) of Canada, the Canada Foundation for Innovation (CFI), Canada Research Chair (CRC) program, British Columbia Knowledge Development Fund (BCKDF), and the Higher Education Commission (HEC) of the Government of Pakistan.

References


