Decomposition of Fundamental Problems for Cooperative Autonomous Mobile Systems

Samiya Souissi*  Xavier Défago*†  Takuya Katayama*

*School of Information Science
Japan Advanced Institute of Science and Technology (JAIST)
1-1 Asahidai, Tatsunokuchi, Ishikawa 923-1292, Japan
†PRESTO, Japan Science and Technology Agency (JST)
E-mail: {ssouissi,defago,katayama}@jaist.ac.jp

Abstract

Mobile computing can be seen as a natural extension of distributed computing, with the difference that hosts can be physically mobile. This results in many interesting new challenges. The most original aspect of mobile computing with respect to traditional distributed computing is when one considers problems whereby the movements of the host must be controlled. In particular, this is a central issue for cooperating autonomous mobile systems.

This paper outlines a specification framework to define recurrent problems for cooperative autonomous mobile systems. The framework consists of four generic properties (two liveness and two safety properties) that can be combined to define many different problems, including those surveyed in the literature. We regard this as a necessary step toward a better understanding of the relationships between problems.

1. Introduction

In recent years, the popularity of mobile systems has sustained an incredible growth, largely supported by recent advances in wireless communication technologies and the increasing pervasiveness of inexpensive mobile communication devices. The introduction of mobility in distributed systems has a strong impact on the applicability of conventional algorithms and protocols, and much research has been aimed at solving conventional problems in spite of mobility (e.g., solving Mutual Exclusion in mobile ad hoc networks [22, 13]). Other researches aim at solving problems where the location is seen as an external parameter, such as group membership depending on the physical proximity of its members [16]. In contrast, we focus on problems where the mobility of the hosts must be controlled. More specifically, we consider the family of problems whereby a group of autonomous mobile systems must coordinate their movements, such as with autonomous transport vehicles (ATV), autonomous mobile base stations for ad hoc networks [3, 14], micro electro-mechanical systems (MEMS), sensor networks [4], nanomachines [6] or large scale robot teams [17].

Problems for cooperative mobile robots A vast amount of researches have been conducted in the field of cooperative mobile robotics (see [2] for a survey). Unfortunately, only few studies address it from a computational standpoint (e.g., [9, 21]), which means that much remains to be done to develop its theoretical foundations. Among other things, it is essential to properly identify and specify the problems that are central to the field. To draw a parallel, problems such as Leader Election and Consensus are central to parallel and distributed systems. In contrast, currently no agreement has been reached on the exact definition of fundamental problems in cooperative mobile robotics. This is a likely consequence of the fact that problems such as gathering or flocking, are notably expressed in a way that depends on details of the system model, or restrict their implementation to specific solutions. For instance, the definition of the flocking problem, as proposed by Principe [20], requires the existence of a single leader robot to determine the path that the group must follow. Hence, fully symmetrical or fault-tolerant solutions of the problem are unfortunately ruled out by their definition. Finally, known problems in

1In this paper, we use the terms “mobile” and “mobility” as referring to the physical mobility of hosts. In particular, we do not discuss the topic of mobile agents as we believe that mobility in that case, is simply a metaphor.

2Gathering: all robots must move and gather at the same location.

3Flocking: all robots must move in such a way that their respective locations always form a given shape.
cooperative mobile robotics have not been defined in a consistent way. This makes it particularly difficult to study their relationships.

We aim at improving the current situation by providing a consistent specification framework for problems involving mobility among cooperative autonomous systems. We think that this will contribute to a better understanding of the problems, and pave the ground for many further developments. We believe that the field will reach its maturity only once its theoretical foundations will have been clearly established; similarly to what happened with conventional distributed systems over the last three decades or so.

Consider for example a collection of robots aiming at mitigating an earthquake disaster. Effective solutions are demanded for several problems: for instance, surveillance of the disaster area inaccessible by human, support of human body search and support of first aid response activities. These problems can be decomposed into various fundamental problems (gathering, geometric pattern formation, flocking, cooperative handling, motion planning) which are common between applications. For example, in case of surveillance of the disaster area, the robots might need to arrange into a specific pattern to oversee the area. This is known as the geometric pattern formation problem. Similarly, for finding casualties, the robots need to gather at some point or location (gathering problem) to handle some cooperative tasks such as raising things out of casualties and carrying them is commonly known as the cooperative handling problem. In the case of first aid response activities, robots can be asked to transport victims to designated first aid stations for treatment in spite of the presence of obstacles; this is known as the motion planning problem. Some victims might have broken bones and cuts, such victims have to be carried in their existing position, this can be seen as an instance of the flocking problem.

**Contribution** The main contribution of this paper is to outline a consistent specification framework whereby several fundamental problems in cooperative mobile robotics can be expressed. More specifically, we define two safety properties and two liveness properties, which can be combined in several ways to define different problems. We discuss some problems presented informally in the literature, and express their specification in terms of the four basic properties mentioned above.

The merit of our approach stems from the fact that we express problems in a common framework, and with a limited set of properties. There are several reasons why this is desirable. Firstly, the exact relationship between problems becomes more apparent. Secondly, problems are decomposed into safety and liveness properties, thus making it easier to prove the correctness of algorithms. Thirdly, some important results can be generalized more easily to whole classes of problems. For instance, assuming that we prove the property of following a path to be impossible to satisfy in a given system model, then we also prove, as a direct consequence, that no path-constrained problem can be solved in that same model.

This is still a work in progress, and much remains to be done. Nevertheless, we see it as an important step toward providing a more general specification framework and the formal definition of the problems in cooperative mobile robotics.

**Structure** The remainder of the paper is structured as follows. Section 2 presents the system model and some basic definitions. Section 3 describes recurrent problems found in the literature on cooperative mobile robotics, and presents some important known results. Section 4 presents four fundamental properties into which all problems previously mentioned can be decomposed. Finally, Section 5 discusses open questions and future directions.

2. System Model and Definitions

In this paper, we make only few assumptions on the system model in order to remain as general as possible. We believe that these assumptions are sufficiently general to cover nearly any reasonable system model for autonomous mobile systems. In particular, our basic assumptions encompass the two important models described later, namely the Suzuki-Yamashita model (SYm) [21] and the CORBA model [20].

**Basic assumptions** We model the world as a distributed environment consisting of a finite set of mobile robots \(\{r_1, r_2, \ldots, r_n\}\) evolving in a \(k\)-dimensional space. To simplify the expression of the problems, we assume that robots are physically represented by mathematical points. Finally, we consider that robots are autonomous, meaning that each of them executes its own program.

**Suzuki-Yamashita model** In the SYm model [21], robots evolve in the Euclidean plane devoid of any landmark. They interact indirectly through their actions on the environment. The robots are anonymous in the sense that they are unable to uniquely identify themselves, neither with a unique identification number nor with some external distinctive mark (e.g., color, flag). They share no common sense of direction, unit distance, or location of the origin of their coordinate system. Time is represented as an infinite sequence of time instants during which each robot can be either active or inactive. Each time instant during which a robot becomes active, the robot observes its environment, computes a new position, and moves toward that position. The activation of robots is unpredictable and unknown to robots, with the
guarantees that (1) every robot becomes active at infinitely many time instants, and (2) at least one robot is active during each time instant. Depending on the context, the model can be further restricted by assuming limited visibility, or that robots are oblivious (i.e., keep no memory of past actions).

CORDA model The CORDA model [20] is similar to the SYm model described above. The most notable difference is that in the CORDA model, there is no synchronization (explicit or implicit) between the robots. This is unlike the SYm model, where activation steps are executed either sequentially or in lock steps (observe-compute-move), thus introducing an implicit synchronization between robots. Prencipe [20] also shows that all problems solvable in the CORDA model are also solvable in the SYm model.

3. Survey of Fundamental Problems

In this section, we survey the most recurrent problems found in the literature on cooperative mobile robotics. For each problem, we give a brief description and present important known results. Whenever possible, we try to illustrate the problem with a concrete example.

3.1. Gathering

With the gathering problem, the robots, initially positioned at arbitrary positions, are required to gather in a not predetermined point. This problem has been studied extensively in the literature, among which only few address the problem from a computational standpoint. Notable exceptions are studied by Suzuki and Yamashita [21] where, along with the definition of their model (see Sect. 2), they propose an algorithm to solve the gathering problem deterministically, in the case where robots have unlimited visibility. Ando et al. [1] propose an algorithm to address the gathering problem in systems wherein robots have limited visibility. Their algorithm converges toward a solution to the problem, but it does not solve it deterministically.

Prencipe [20] study similar issues in his CORDA model. Among other things, one feature the robots must have in order to solve this problem, is the ability to detect multiplicity. The gathering in the case of limited visibility is also discussed by Flochini et al.[10], and by Cielibak and Prencipe [5]. In the limited visibility setting, the proposed algorithm requires that the robots agree on the direction and orientation of both $x$ and $y$ axis.

3.2. Geometric pattern formation

The geometric pattern formation problem is defined as follow: given a group of $n$ robots $\{r_1, r_2, \ldots, r_n\}$ with distinct positions and located arbitrarily on the plane, the robots are required to form a specific geometric shape, where the shape is a set of points (given by Cartesian coordinates) in the plane. The target shape is known beforehand by all the robots in the system, and it can be for instance a circle, a regular polygon, or some other arbitrary pattern. However, the location of the shape, as well as its size and orientation are not specified.

This problem has been investigated extensively in the cooperative robotics literature, with many ad hoc solutions being proposed. From a computational point of view, this problem has only been studied by a few authors.

Suzuki and Yamashita [21] studied the formation of geometric patterns in the plane. They propose an non-oblivious algorithm for circle formation. They show that asymmetric patterns cannot be formed as their consider that robots are anonymous.

In the same model, Défago and Konagaya [7] propose a self-stabilizing algorithm for the circle formation problem. With that algorithm, robots deterministically make a circle, albeit not uniformly. The algorithm converges asymptotically toward a solution whereby robots are uniformly distributed along the circle boundary.

Flochini et al. [8] discuss the problem of arbitrary pattern formation, of which they give an informal definition. They show several important results about this problem, depending on what common knowledge the robots are assumed to share about the coordinate system. The authors give a more formal definition of the problem in their later work [10].

3.3. Flocking

Roughly speaking, flocking is a problem where the robots are required to move in formation, that is, according to some specified pattern or as a flock of birds. More rigorously, given set of $n$ robots $\{r_1, r_2, \ldots, r_n\}$, the robots required to keep a given shape while moving.

In the literature, the flocking problem is generally expressed in a "leader-followers" model [20, 11, 12]. One robot (the leader) is chased by the other robots (the followers). The motion of the leader is not constrained by the problem. In contrast, the followers must follow the leader in such a way that the relative positions of the robots always form a given shape.

To the best of our knowledge, the flocking problem has been studied computationally only in the CORDA model [20, 11, 12]. The authors presented an oblivious algorithm that allow the robots to keep formations that are symmetric.
with respect to the direction of movement of the leader. Ger-
vasti and Pencape [12] simulate their algorithm and present
interesting results.

Solutions to the flocking problem are useful primitives
for larger tasks. For instance box pushing or cooperative
manipulation, where robots can be asked to move heavy
loads.

3.4 Motion planning

Motion planning problem refers to the computational
process of moving from one place to another in the presence
of obstacles. In other words, motion planning must be per-
fomed taking into consideration other robots and the global
environment; this multiple-robot path planning is an intrin-
sically geometric problem in configuration space-time. It is
also known as multi-robot path planning. Motion planning
has an importance historical in the literature. A noteworthy
distributed approach is the one of [23], where each robot ini-
tially attempts a straight-line path to the goal; if an interfe-
ing obstacle is seen, then the robot will scan the visible ven-
exes of the obstacle and move towards the closest one. Dyna-
mmically varying priorities are given to each robot to resolve
path intersection conflicts. Conflicting robots can either
negotiate among themselves or allow a global blackboard
manager to perform this function. Some recent works have
addressed some non-traditional planning problems. For ex-
ample, [15] proposes an algorithm for path planning in teth-
ered robots, and [19] consider the problem of moving while
grasping large objects.

4. Basic Properties

In this section, we propose four properties into which the
problems described in Section 3 can be decomposed. The
advantage of this approach is that problems are expressed in
a more consistent framework, thus making it easier to study
their relationships.

Among the four properties, the first two properties can
be characterized as liveness properties, whereas the other
two are safety properties. The distinction between safety
and liveness properties is often adopted in specification and
design methods for distributed systems. Roughly speaking,
a safety property of a system is one that specifies what the
system should never do (i.e., “bad things never happen”). A
typical example of safety property is given by mutual ex-
clusion in a concurrent system (at most one thread is in the
critical section at any time). Conversely, a liveness prop-
erty is one which specifies that the system must eventually
do something (i.e., “good things eventually happen”) [18].
In general, the requirement that a program must eventually
terminate in finite time is a liveness property.

Following the presentation of the properties, we give a
list of eight problems, including those presented in Sec-
tion 3, and describe them according to a combination of the
four properties.

4.1 Liveness properties

Property 1 (Forming shape) Given some shape X, there
is a time after which the shape defined by the location of all
robots is homomorphic to X.

Informally, the desired shape is formed eventually. In
other words, after a finite number of moves, the final posi-
tions of the robots coincide with the points forming the in-
put shape. Forming shape is a liveness property since while
testing for violation of this property requires looking at in-
finite executions.

Property 2 (Reaching goal) Given some point p, there is
a time after which the center of gravity of all robots is co-
located with p.

Reaching goal property is the ability of a team of robots
to reach a predetermined goal location. At the end of their
computation process, the robots reach to the destination
point at the same time or at different time. In the case when
the robots are asked to attain a goal and preserving a shape
at that destination, the property goal is satisfied when the
center of gravity of the shape forming the robots superpose
with the destination point.

4.2 Safety properties

Property 3 (Keeping shape) Given some shape X, there
is a time since which the shape defined by the location of all
robots is always homomorphic to X.

Roughly speaking, since the robots form the shape, they
are required to keep the same shape in movement. Keeping
shape is a safety property since the violation of this property
can be observed in finite time (time in which the shape is
modified).

Property 4 (Following path) Given some oriented path
p(u), the following two predicates are satisfied.
1. At any time, the center of gravity of the location of all
robots is a point on the path p(u).

2. The center of gravity of the location of all robots pro-
gresses monotonically on the path p(u).

This property expresses the ability of the robots to follow
given path while moving without getting out of that path.
It is satisfied when the center of gravity of the new posi-
tions of robots in each movement is a point on the specified
path and the coordinates of the center of gravity increases
according to the oriented path.
Table 1. Decomposition of common problems into the basic properties

<table>
<thead>
<tr>
<th></th>
<th>forming shape (liveness)</th>
<th>keeping shape (safety)</th>
<th>following path (safety)</th>
<th>reaching goal (liveness)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shape formation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flocking</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Path-constrained flocking</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Motion planning</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cooperative handling (“box-pushing”)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Path-constrained coop. handling</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shape formation w/goal</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Path-constr: shape formation w/goal</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.3. Summary

Table 1 summarizes the decomposition of problems into a combination of the four properties mentioned above. The problems presented in Section 3 are decomposed into the properties. In addition, several other combinations, that yield meaningful problems, are also presented in the table and discussed in the text.

Shape formation The shape formation problem encompasses the two problems of gathering and arbitrary pattern formation mentioned in Section 3. This problem is simply defined by Property 1 (forming shape). In the literature, gathering and pattern formation are treated differently, probably because the former is easier to solve than the latter. Nevertheless, there is no reason to give a completely different definition, and so we combine these two problems as special cases of shape formation. In particular, the robots must eventually form the desired shape, which can be a point (i.e., gathering problem), a circle, a polygon, etc.

Flocking In the literature, the flocking problem has been defined according to a leader-followers approach [20, 12]. In that model, one of the robots is designated as the leader, and the others are followers. The problem requires that the followers follow the path of the leader in such a way that their relative positions maintain a given shape. However, we believe that this definition unnecessarily restricts the problem. In particular, the definition precludes fully symmetrical algorithms. For this reason, we try to provide an alternate definition which does not impose a leader-follower model, thus leaving this as an implementation issue.

Path-constrained flocking In the path-constrained variant of the flocking problem, the path that the robots must follow is given. This results in two safety properties, keeping the shape and following the path.

Motion planning Recall that, in the motion planning problem, the robots need to get from a starting point to an ending point while avoiding collisions with obstacles (or each other). We express this problem by the Property 2 (reaching goal) and the Property 4 (following path). The property of following a path could be used to guarantee that no collisions between robots occur or against obstacles.

Cooperative handling (box-pushing) A problem known as “box-pushing” has been studied in the literature on robotics. In that problem, a pair of small robots are required to collaborate in order to move a large box from one location to another. We can easily express a similar problem by combining flocking and goal reaching. Hence, the cooperative handling problem is defined in terms of one safety property (keeping shape) and one liveness property (reaching goal).

Other problems Several problems can be derived from the problems discussed above: for instance path-constrained cooperative handling is a combination of the problems path-constrained flocking and goal reaching. Therefore, the problem is defined by the two safety properties presented earlier and the liveness property reaching goal. The problem of shape formation with goal (i.e., where the robots are asked to form a shape in a specific location known in advance) is also a combination of the shape forming problem and the goal reaching problem. When the property following path is added to this problem, it is called path-constrained shape formation with goal and in which the two liveness properties and the safety property following path must be fulfilled.

5. Conclusion and Future Directions

We have surveyed some of the most important problems in cooperative mobile robotics identified in the literature so far, and proposed four properties (two safety and two
liveness properties) along which these problems can be decomposed. Other combinations of the properties also yield meaningful problems, and variants thereof, that we did not see discussed in the literature yet. The main benefit of our approach is to make the study of inter-problems relationships easier. As we outline a consistent framework for the definition of problems, it will be easier to generalize future results to classes of problems rather than individual ones.

As mentioned earlier, the work presented in this paper is still in progress, and many interesting research issues remain to be addressed. In particular, we are working on a formal definition of the properties, and hence the problems. We will also need to make sure that our specifications rule out trivial (and useless) solutions to the problems. At last, we will try to extend the framework with additional “model-dependent” safety properties, such as preventing collisions with obstacles or other robots.

In the longer run, we will use our framework to determine the minimal system requirements for solving each problem. We will hence investigate the relationship between the capabilities of robots and their ability to solve the different problems. From a practical standpoint, it is important to know what minimal set of functionalities is necessary to solve a given problem, as a way to reduce development and fabrication costs.

References


