Brief Announcement: Certified Universal Gathering in $\mathbb{R}^2$ for Oblivious Mobile Robots

Pierre Courtieu
Cédric – CNAM
Paris, France
Pierre.Courtieu@cnam.fr

Lionel Rieg
Collège de France
Paris, France
Lionel.Rieg@college-de-france.fr

Sébastien Tixeuil
UPMC Sorbonne Universités,
LIP6-CNRS 7606,
Institut Universitaire de France
Paris, France
Sebastien.Tixeuil@lip6.fr

Xavier Urbain
ENSIIE,
LRI, CNRS UMR 8623,
Université Paris-Sud,
Université Paris-Saclay
Orsay, France
Xavier.Urbain@lri.fr

1. INTRODUCTION

Designing and proving mobile robot protocols is notoriously difficult. Since its initial presentation [13], this computing model has grown in popularity[1] and many refinements have been proposed (see [10] for a recent state of the art). The diversity of model variants makes it extremely onerous to check whether a particular property of a robot protocol holds in a particular setting. Even worse, checking whether a property that holds in a particular setting also holds in another setting that is not strictly contained in the first one often requires a completely new proof, even if the proof argument is very similar. The lack of proof reusability between model variants is a major problem for investigating the viability of new solutions or implementations of existing protocols (that are likely to execute in a more concrete execution model). Also, oblivious mobile robot protocols are mostly based on observing geometric constructions and deriving invariants from those observations. As the protocols are typically written in an informal high level language, assessing whether they conform to a particular model setting is particularly cumbersome, and may lead to hard to find mismatches. Hence, solely relying on handcrafted protocols, models and proofs is likely to introduce subtle errors that eventually lead to catastrophic failures when the system is actually deployed. Formal methods encompass a long-lasting path of research that is meant to overcome errors of human origin. Not surprisingly, this mechanised approach to protocol correctness was successively used in the context of mobile robots [5, 9, 2, 12, 7, 3, 4].

Model-checking proved useful to find bugs in existing literature [4] and assess formally published algorithms [9, 4], in a simpler setting where robots evolve in a *discrete space* where the number of possible positions is finite. Automatic program synthesis (for the problem of perpetual exclusive exploration in a ring-shaped discrete space) is due to Bonnet *et al.* [5], and can be used to obtain automatically algorithms that are "correct-by-design". The approach was refined by Millet *et al.* [12] for the problem of gathering in a discrete ring network. As all aforementioned approaches are designed for a discrete setting where both the number of positions and the number of robots are known, they cannot be used in the continuous space where robots positions take values in a set that is not enumerable, and they cannot permit to establish results that are valid for any number of robots.

Developed for the Coq proof assistant,[2] the Pactole framework enabled the use of higher-order logic to certify impossibility results [2] for the problem of convergence: for any positive ε, robots are required to reach locations that are at most ε apart. Another classical impossibility result that was certified using the Pactole framework is the impossibility of gathering starting from a bivalent configuration [7]. While the proof assistant approach seems a sensible path for establishing certified results for mobile robots that evolve in a continuous space, until this paper there exists no *positive* certified result in this context. Expressing mobile robot protocols in a formal framework that permits certification poses a double challenge: how to express the protocol (which can make use of complex geometric abstractions that must be properly defined within the framework), and how to write the proof?

Our contribution.

Our first contribution is a unified formal framework for expressing mobile robots models, protocols, and proofs. This framework is motivated by the fact that many of the observed errors in published papers come from a mismatch between

[1]The 2016 SIROCCO Prize for Innovation in Distributed Computing was awarded to Masafumi Yamashita for this line of work.

[2]http://coq.inria.fr

PODC'16, July 25–28, 2016, Chicago, IL, USA.

ACM ISBN 978-1-4503-3964-3/16/07.
DOI: http://dx.doi.org/2933057.2933070

the advertised model and the model that is actually used for writing the proofs. For example, some dining philosophers protocols were expressed and proved in a high-level atomicity model, but advertised as working in a lower-level atomicity model, revealed to be incorrect in the lower-level atomicity model (see the work of Adamek *et al.* [1] and references herein). Sometimes, the mismatch between the proof and the advertised model is more subtle: a perpetual exclusive exploration protocol the proof of which did not consider all possible behaviours in the advertised model ASYNC was used to exhibit a counter example in such a setting (See the work of Berard *et al.* [4] and references therein). A unified formalisation whose consistency can be mechanically assessed is a huge asset for designing correct solutions, whose correctness can be certified. As we used a subset of the same framework for certifying impossibility results [2, 7], consistency between negative and positive results is also guaranteed.

Our second contribution is a protocol design/proof methodology dedicated to mobile robots. We advocate the joint development of both the mobile robot protocol and its correctness proof, by taking advantage of the Coq proof assistant features. The proof assistant is typically able to check whether the proof of a particular theorem/lemma/corollary is valid. So replacing particular clauses of those theorems/lemmas/corollaries statements makes the proof assistant check whether the proof still is acceptable for the new statement. We used this feature to lift a preliminary version of this work (uni-dimensional setting [6]) to a Euclidean bi-dimensional space: the proof assistant checked which arguments were still valid in the new setting. This feature also proved useful when slightly changing parts of the algorithm: the impact of the changes on the proofs were immediate. Also, it becomes easy to remove or weaken hypotheses from the protocol, as the proof assistant makes it obvious if they are not used in the proof arguments. Finally, our methodology includes a formal way to guarantee whether the "global" view of the system (as seen from the protocol prover point of view) is effectively realisable given the hypotheses assumed in the model.

We instantiate our framework and methodology to actually design and prove correct a new protocol for the oblivious mobile robot universal gathering problem, that can be informally defined as follows: robots have to reach in a finite number of steps the same location, not known beforehand. In more details, *we present a new gathering algorithm* for robots operating in a continuous space that *(i)* can start from any configuration that is not bivalent (that is, the robots are not initially equally placed in exactly two locations, since gathering is impossible in this case), *(ii)* does not put restriction on the number of robots, *(iii)* does not assume that robots share a common chirality (no common notion of "left" and "right"). To our knowledge, this is the first certified positive (and constructive) result in the context of oblivious mobile robots.

2. OUR ALGORITHM

The protocol we propose uses multiplicity to build the set of towers of maximal height. If there is a unique tower of maximal height, i.e., a unique location of highest multiplicity, this location is the destination of each activated robot. Otherwise, the inhabited locations on the smallest enclosing circle (SEC) are taken into account to define a *target*.

The spectrum of a configuration is the multiset of all its robots' locations. It is *clean* if inhabited locations are either on the SEC or at target. When it is not clean (*dirty*), robots on SEC (or at target) stay where they are, the others move to the target, thus cleaning the spectrum. In a clean spectrum, any activated robot moves to the target. A configuration is said to be clean if and only if its spectrum is clean.

The important operation is thus to define a convenient target. Our target depends on how many inhabited locations are on the SEC. If there is only one, then the whole spectrum is reduced to a single location and all robots are gathered. When the number of towers on the SEC is not equal to 3, the target is the center of the SEC. Critical situations occur when towers on the SEC define a triangle. If this triangle is *equilateral*, the target is the center of the SEC (which is also the triangle's barycenter). If it is *isosceles* and not equilateral, the target is the vertex opposite to its base. Finally if the triangle is *scalene*, the target is the vertex opposite to its longest side.

A rephrase of that description in informal pseudo-code is presented as Figure 1. For a spectrum s, let support(s) be the set of locations in s, let max(s) be the set of locations of maximal multiplicity in s, and let SEC(s) be the smallest enclosing circle of s. Let *dest* be the destination to be computed. Remember that $(0, 0)$ is always the location of a robot in its own frame of reference.

Approach.

To certify results and to guarantee the soundness of theorems, we use Coq, a Curry-Howard-based interactive proof assistant enjoying a trustworthy kernel.

Developing a proof in a proof assistant may nonetheless be tedious, or require expertise from the user. To make this task easier, we are actively developing (under the name Pactole) a formal model, as well as lemmas and theorems, to specify and certify results about networks of autonomous mobile robots. It is designed to be robust and flexible enough to express most of the variety of assumptions in robots network, for example with reference to the considered space: discrete or continuous, bounded or unbounded... We want to stress that the framework eases the developer's task.

The Distributed Computing community is known to have fundamental algorithms tightly coupled with their proof of correctness. The mobile robot setting is no exception, as the minimal hypotheses a protocol must make to solve a given problem are extremely difficult to identify without actually writing the corresponding correctness proofs (that is, an intuitive approach is often detrimental to the correctness of the result to be established, as recent errors found in the literature proved [1]). In a formal proof approach to obtain mechanically certified protocols, our framework and methodology clearly contributes to two main phases in a verified development.

Firstly the *specification* phase, where all objects, definitions, algorithms, statements and expected properties are expressed without any ambiguity, in a higher order type theoretic functional environment. The lack of ambiguity is a key feature to enable the early detection of inconsistencies between the problem specification, the algorithmic proposal, and the execution model. We emphasise the fact that there is no need to be an expert with the Coq proof assistant to use our framework in this phase. Clear and unequivocal specifications are indeed a fundamental step towards correct algorithms.

```
if max(s) = ∅ then dest := (0,0)                                   (* absurd case *)
else   if max(s) = {p} then dest := p
else   begin                          (* first compute target then dest depending on cleanliness *)
       if support(s) ∩ SEC(s) = ∅ then dest := (0,0)                (* absurd case *)
       else if   support(s) ∩ SEC(s) = {p} then target := p          (* already gathered *)
       else if   support(s) ∩ SEC(s) = {p1,p2,p3} then              (* triangle cases *)
                 if equilateral(p1,p2,p3) then target := barycenter(p1,p2,p3)
                 else if isosceles(p1,p2,p3) then target := opposite of base(p1,p2,p3)
                 else target := opposite of longest(p1,p2,p3)
       else target := center(SEC(s));
       if ∀p ∈ s, p ∈ SEC(s) or p = target  then dest := target     (* clean ⇒ go to target *)
       else if   (0,0) ∈ SEC(s) or (0,0) = target  then dest := (0,0) (* dirty ⇒ clean config *)
            else dest := target
       end
```

Figure 1: Our algorithm pseudocode

Secondly the *proof* phase, where properties are proved to hold for the relevant executions. This phase is of course more demanding on the expertise side, so our goal when constructing the framework was to provide useful libraries and proof techniques that can be reused in other contexts, enabling more automation to the protocol designer. Considering reusability, useful assets brought by the current work are the notions of gathering, SSYNC demons, etc., developments on geometry in $\mathbb{R}^2$ and smallest enclosing circles, as well as the proof that forbidden configurations can be reached from already forbidden configurations only [6]. Those will most likely prove useful in future developments. When developing the protocol for our case study, we decided to modify the protocol code several times, either to fix a newly discovered bug, or to ease the writeup of the proofs. This classical design stage was streamlined by the use of a formal language based on the Curry-Howard isomorphism [11] where both activities can be done in a uniform way. In such a setting, correcting the algorithm amounts to modifying the algorithm definition, and replaying the proofs certification process after adapting the proof scripts written previously. The mechanised verification of the proofs makes this process fast and trustworthy, compared to a purely handcrafted approach.

Resources.

A research report [8] describing our approach as well as the actual development and its `html` documentation are available from the project's webpage: http://pactole.lri.fr

3. REFERENCES

[1] J. Adamek, M. Nesterenko, and S. Tixeuil. Evaluating and optimizing stabilizing dining philosophers. In *11th European Dependable Computing Conference, EDCC 2015, Paris, France, September 7-11, 2015*, pages 233–244. IEEE, 2015.

[2] C. Auger, Z. Bouzid, P. Courtieu, S. Tixeuil, and X. Urbain. Certified Impossibility Results for Byzantine-Tolerant Mobile Robots. In T. Higashino, Y. Katayama, T. Masuzawa, M. Potop-Butucaru, and M. Yamashita, editors, *Stabilization, Safety, and Security of Distributed Systems - 15th International Symposium (SSS 2013)*, volume 8255 of *Lecture Notes in Computer Science*, pages 178–186, Osaka, Japan, Nov. 2013. Springer-Verlag.

[3] B. Bérard, P. Courtieu, L. Millet, M. Potop-Butucaru, L. Rieg, N. Sznajder, S. Tixeuil, and X. Urbain. Formal Methods for Mobile Robots: Current Results and Open Problems. *International Journal of Informatics Society*, 7(3):101–114, 2015. Invited Paper.

[4] B. Bérard, P. Lafourcade, L. Millet, M. Potop-Butucaru, Y. Thierry-Mieg, and S. Tixeuil. Formal verification of Mobile Robot Protocols. *Distributed Computing*, 2016.

[5] F. Bonnet, X. Défago, F. Petit, M. Potop-Butucaru, and S. Tixeuil. Discovering and assessing fine-grained metrics in robot networks protocols. In *33rd IEEE International Symposium on Reliable Distributed Systems Workshops, SRDS Workshops 2014, Nara, Japan, October 6-9, 2014*, pages 50–59. IEEE, 2014.

[6] P. Courtieu, L. Rieg, S. Tixeuil, and X. Urbain. A Certified Universal Gathering Algorithm for Oblivious Mobile Robots. *CoRR*, abs/1506.01603, 2015.

[7] P. Courtieu, L. Rieg, S. Tixeuil, and X. Urbain. Impossibility of Gathering, a Certification. *Information Processing Letters*, 115:447–452, 2015.

[8] P. Courtieu, L. Rieg, S. Tixeuil, and X. Urbain. Certified Universal Gathering Algorithm in $\mathbb{R}^2$ for Oblivious Mobile Robots. *CoRR*, abs/1602.08361, 2016.

[9] S. Devismes, A. Lamani, F. Petit, P. Raymond, and S. Tixeuil. Optimal Grid Exploration by Asynchronous Oblivious Robots. In A. W. Richa and C. Scheideler, editors, *Stabilization, Safety, and Security of Distributed Systems - 14th International Symposium (SSS 2012)*, volume 7596 of *Lecture Notes in Computer Science*, pages 64–76, Toronto, Canada, Oct. 2012. Springer-Verlag.

[10] P. Flocchini, G. Prencipe, and N. Santoro. *Distributed Computing by Oblivious Mobile Robots*. Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool Publishers, 2012.

[11] W. A. Howard. The formulae-as-types notion of construction. In J. R. H. Jonathan P. Seldin, editor, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 479–490. Academic Press, London, 1980.

[12] L. Millet, M. Potop-Butucaru, N. Sznajder, and S. Tixeuil. On the synthesis of mobile robots algorithms: The case of ring gathering. In P. Felber and V. K. Garg, editors, *Stabilization, Safety, and Security of Distributed Systems - 16th International Symposium, (SSS 2014)*, volume 8756 of *Lecture Notes in Computer Science*, pages 237–251, Paderborn, Germany, sep 2014. Springer-Verlag.

[13] I. Suzuki and M. Yamashita. Distributed Anonymous Mobile Robots: Formation of Geometric Patterns. *SIAM Journal of Computing*, 28(4):1347–1363, 1999.