Adaptive Wireless Networks as an Example of Declarative Fractionated Systems

Jong-Seok Choi[2], Tim McCarthy[1], Minyoung Kim[1(✉)], and Mark-Oliver Stehr[1]

[1] SRI International, Menlo Park, CA 94025, USA
{tim.mccarthy,minyoung.kim,markoliver.stehr}@sri.com
[2] Kyungpook National University, Daegu, Korea
choijongseok@knu.ac.kr

Abstract. Adaptive wireless networks can morph their topology and support information gathering and delivery activities to follow high-level goals that capture user interests. Using a case study of an adaptive network consisting of smart phones, robots, and UAVs, this paper extends a declarative approach to networked cyber-physical systems to incorporate quantitative aspects. This is done by distinguishing two levels of control. The temporal evolution of the macroscopic system state is controlled using a logical framework developed in earlier work while the microscopic state is controlled by an optimization algorithm or heuristic. This two-level declarative approach is built on top of a partially-ordered knowledge sharing model for loosely coupled distributed computing and is an example of a so-called fractionated system that can operate with any number of wireless nodes and quickly adapt to changes. Feasibility of the approach is demonstrated simulation and in a hybrid cyber-physical testbed consisting of robots, quadcopters, and Android devices.

Keywords: Cyber-physical systems · Distributed systems · Declarative control · Adaptive networks · MANETs · Swarms · Robots · UAVs

1 Introduction

Adaptive wireless networks are designed to morph in response to changing requirements and conditions. Such networks are becoming increasingly feasible with the emerging convergence of technologies in mobile networking, personal digital assistants (PDAs), smart phones, robotics, smart antennas, and sensors.

At the highest level of abstraction, an *adaptive wireless network* takes into account the user's information needs, adapts while sensing the environment, and tries to find a solution that best satisfies the user's requirements under the given constraints. Adaptive wireless networks are a special case of *networked cyber-physical systems* (NCPS). Robots are wireless nodes equipped with sensors and actuators that allow them to autonomously change location. PDAs carried by humans serve as user interfaces to the adaptive network. Different from traditional mobile ad hoc networks (MANETs), adaptive wireless networks are closer

I. Stojmenovic et al. (Eds.): MOBIQUITOUS 2013, LNICST 131, pp. 549–563, 2014.
DOI: 10.1007/978-3-319-11569-6_43

to content-based networks and solve a more general resource planning and optimization problem by utilizing the computational resources and sensor/actuator capabilities of each node, e.g., the capability to move, turn, take a picture, or sense a sound.

This paper generalizes a *declarative approach to distributed control* of NCPS [18] that distinguishes two levels of control: (1) distributed logic based on user-injected goals controls the system's evolving macroscopic state (e.g., the location of a swarm of UAVs or robots) and (2) an optimization algorithm drives the system's microscopic system (e.g., the formation of the swarm) in a direction that satisfies the specification of the macro-state that, in the most abstract approach, can be declaratively expressed as an objective function.

This paper focuses on a subclass of NCPS with distributed control called *heterogeneous fractionated systems*. Apart from their decentralized nature, they are characterized by the involvement of different types of nodes (PDAs, robots, and UAVs) that, if the same type, are interchangeable for the purposes of this application. One key implication of this model is the independence of the control algorithm from the number of nodes of each type available. Futhermore, both temporary or permanent failure/disconnections or rescaling of the system to adjust its resources can change the number of nodes at runtime. Reference [29] describes additional motivations for the fractionated approach to NCPS.

The unpredictability of the environment that immediately yields the traditional impossibility results for asynchronous systems dictates the use of a sufficiently weak, and hence (wirelessly) implementable, model for loosely-coupled distributed computing. This paper considers fractionated systems as a special case of the *partially ordered knowledge sharing* (POKS) model [17] which does not rely on any form of atomic transactions and only provides eventual consistency. In this declarative setting, *knowledge* refers to all information shared between the nodes, specifically observations (facts) and control actions (goals), as explained in [18] in more detail.

The primary contribution of this paper is the integration of qualitative and quantitative aspects in a unified declarative approach that widens the applicability of the distributed logical framework for NCPS [18]. A second contribution is the extension of the cyber-application software framework to support a hybrid testbed and new set of applications based on this generalized approach.

1.1 Motivating Scenario: Self-organizing Cyber-Physical Ensembles

An example of cyber-physical ensembles is a swarm of programmable ground robots and UAVs that perform a distributed surveillance mission — e.g., to achieve situation awareness during an emergency — by moving or flying to a suspicious location, collecting information, and returning to a base location in a formation that creates an effective sensing grid. This application also involves human-carried computing/communication devices such as smart phones that collect/report sensor data and inject users' interests into the system. Heterogeneous nodes have different capabilities: UAVs can fly and generate an encoded video stream with their front- and bottom-facing cameras. Additional computing

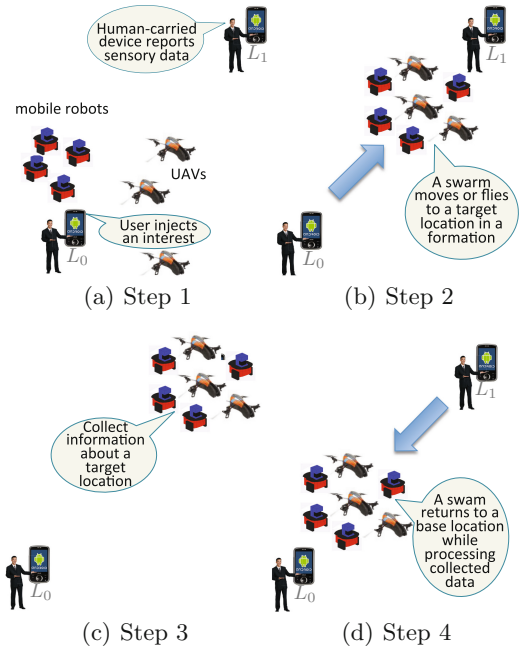


Fig. 1. A user's interest is injected into the network at a location L_0 around which the robots/UAVs are initially clustered randomly. A noise, recorded at a location L_1, triggers the distributed mission.

resources also enable UAVs to extract a snapshot from the encoded video and decode it into JPEG format. Robots can only move on the ground and decode the encoded snapshots from UAVs. The network is adaptive in the sense that it morphs in response to the users' needs.

Figure 1 exemplifies the scenario in which a hybrid swarm moves/flies to a location to investigate a noise reported by an Android phone. The swarm takes a snapshot of the area and returns to the base location. In this scenario, perturbations from the environment (e.g., wind), delayed/incomplete knowledge due to network disruption, and resource contention cause uncertainty. Real-world actions must compensate for temporary network disconnections or failures. These actions can only indirectly control parameters, such as a robot's position, that can be observed by sensors (e.g., GPS). Therefore, a swarm should dynamically configure itself, optimize its resources, and provide robust information dissemination services that adapt to the users' interest in a specific topic. This interest can induce cyber-physical control (e.g., robot positioning, UAV trajectory/formation modifications) and trigger additional sensing (e.g., snapshot of an area).

Goals (e.g., interests and controls) and facts (e.g., sensor readings and computational results) can arrive from the users and the environment at any time, and are opportunistically shared whenever connectivity exists. Programmable robots and UAVs compute their local solution based on local knowledge and exchange

up-to-date information about the progress of their solution. These abilities enable a distributed and cooperative execution approach without the need for global coordination. In a sample mission, the snapshot extracted from the encoded video stream (*abstraction*) may be directly sent to other nodes if the network supports it. If not, the decoded stream (*computation*) is sent to other nodes in the form of knowledge (*communication*). Each node can potentially engage in many *abstractions*, *computations*, and *communications*, which are classes of operations regarded as three dimensions of a distributed computing cube.

The completion of a distributed proof accomplishes the mission. The solution assigns an approximate target region as a subgoal to a swarm's (*macro*-scale controls). The swarm moves or flies to locally realize the subgoal (*micro*-scale controls). The distributed reasoning continues to continuously recompute the local solutions and adjust *macro*-scale controls accordingly. To satisfy the subgoal of *micro*-scale controls, quantitative techniques such as virtual potential fields or artificial physics (Sect. 3) compute the movements of individual UAV/robots.

1.2 Towards Declarative Networked Cyber-Physical Systems

NCPS typically involves a large number of loosely-coupled components that must operate in challenging environments subject to few constraints. Traditional algorithms with problem-specific hardwired execution plans often poorly cover the state space's vast range of possible operating conditions. Declarative approaches rely on executable logical specifications operating on a higher level of abstraction to more directly define the behavior, thereby avoiding micro-management or over-specification of system behavior. Hence, these approaches potentially make the system more adaptive and resilient to changes of many kinds, such as failures, resource constraints, or changes in the number of nodes or the goals.

A *partially ordered knowledge sharing (POKS)* model based on opportunistic information exchange avoids hard constraints on connectivity which is typically wireless and unreliable. As in delay-/disruption-tolerant networks, a local cache partly compensates for the unreliability at each node. For this paper's model, the cache is called a *knowledge base* (KB). Furthermore, the application-dependent model uses a form of semantic networking to exploit the meaning of knowledge in terms of two partial orders referred to as *subsumption* and *replacement* orderings on knowledge units. The subsumption order $k \leq k'$ expresses that k' contains at least as much information as k: k can be discarded in the presence of k'. The replacement order $k \prec k'$ expresses that k is superseded by new information k', and the obsolete k is discarded. In this paper, the former order is used to concisely represent the logical structure (implication) of the current state, while the latter represents the passage of time without introducing logical inconsistencies.

The POKS model naturally expresses in-network computations by generating new knowledge as a function of knowledge received. A typical use of the replacement ordering defines it based on the creation time of the knowledge units, so that in-network transformations and computation can be applied to (unreliable) streams of knowledge, e.g., those generated by sensors.

In declarative networks it is useful to distinguish three classes of operations that can be performed at each node: abstractions, computations, and communications. Since the location at which the operations take place does not matter for the user who simply specifies the information needed in a declarative style, the system has the flexibility to map these operations to different nodes depending on resource availability and connectivity. Adaptive networks can, in addition, morph in response to the user's interest which is represented as a goal.

This paper entertains two approaches to representing and handling goals. Logic may qualitatively represent a goal that is satisfied through a deductive process. The logical framework for NCPS in [18] adopts this approach. It is based on a distributed proof-system defined on top of the POKS model that supports both forward and backward chaining in Horn clause theories. In this framework both facts and goals are uniformly represented as knowledge with suitable orderings and are directly used as an interface to cyber-physical devices (sensors and actuators, respectively).

Alternatively, an objective function may qualitatively represent a goal and measure its degree of satisfaction. This approach, which refers to the objective function as a *virtual potential*, is well known in the multi-agent system community. Related approaches such as artificial physics have been studied as well. Given an objective function, distributed optimization algorithm incrementally improve the satisfaction of the goal. As exemplified in [16], use of the POKS model supports a natural representation of distributed optimization algorithms using a subsumption ordering on the domain of the objective function that captures the degree of satisfaction.

2 Background — Cyber-Framework and Distributed Logic

To implement a distributed computing model based on POKS, the cyber-framework [17] provides a form of semantic networking with caching in a locally event-driven paradigm. The cyber-framework provides communication primitives that can shield applications from the complexities of dealing with dynamic topologies, delays/disruptions, and failures of all kinds. The cyber-framework also supports implementations with simulated and real environments including the network (e.g., wireless/wired) and cyber-physical devices (e.g., sensors, actuators). The knowledge dissemination component implements specific strategies (e.g., deterministic/probabilistic, unicast/broadcast) to propagate knowledge on top of the underlying network layer. The knowledge dissemination protocols make essential use of the POKS by replacing, and hence discarding, a unit of knowledge whenever a higher-ordered unit is received. Note that the kind of knowledge and its frequency of dissemination depends on the specific cyber-application and its objectives as reported in [7].

In the cyber-framework, knowledge is stored redundantly in local KBs that provide the *cyber-applications* with the abstraction of a distributed network cache. Each cyber-application runs on a *cyber-node* that is the smallest managed

computational resource. Cyber-nodes can have attached *cyber-physical devices* (e.g., camera or motors) that observe or control the environment. Cyber-nodes form a hierarchy within a *cyber-engine* and a *cyber-host* that correspond to a specific process and a machine on which cyber-applications are running, respectively. By using the *cyber-API*, the interface provided by the cyber-framework, users can program set-up code that instantiates, composes, and configures cyber-node/engine/host and applications. A cyber-node provides services for posting events (local to the cyber-node) and knowledge (globally shared via dissemination) through the cyber-API — *postEvent* and *postKnowledge*, respectively. The API requires applications that define the local initialization and handling functions — *handleEvent* and *handleKnowledge*.

While the implementation of the cyber-framework provides an operational model for CPS, the declarative view of CPS enables users to describe the mission as a logical theory (e.g., Fig. 2). Distributed reasoning with asynchronous control has been implemented [18] on the cyber-framework with its underlying POKS model. Goals and facts are represented as knowledge that can be shared via POKS. At the level of an individual cyber-physical component, the logic provides a declarative interface for goal-oriented control and feedback through observations that are represented as logical facts. Inference and computation allow facts and goals to interact and form new facts or goals. The opportunity to exchange knowledge with other nodes should lead to cooperation, and the absence of such opportunities should lead to more autonomous behavior.

In this implementation of the logical framework in [18], each node is equipped with a knowledge manager (i.e., an implementation of the POKS model), a reasoner for declarative control, and attached devices. These devices may be considered sub-nodes that exhibit a declarative knowledge-based interface. New goals and facts can arrive at any time and be interleaved with the local inference processes. Two kinds of goals and facts exist: the *cyber-goals*/*cyber-facts* and *goals*/*facts* are the interface with the environment and the basis of inference, respectively. For example, the UAV generates both a cyber-fact of its current location that satisfies a corresponding cyber-goal and a fact/subgoal from forward or backward chaining. The demo scenario (Sect. 1.1) and its implementation (Sect. 4) experimented with a swarm of one to seven robots/UAVs in total and used two Android nodes. These nodes have fixed locations and serve as user access points to the cyber-physical system.

3 Integrating Logic with Quantitative Algorithms

Scalable control of NCPS with resource optimization requires highly decentralized and compositional approaches. Hence, this paper studies a new form of adaptive declarative networking for cyber-physical ensembles by integrating logical inference with quantitative algorithms. In a nutshell, declarative (i.e., logical) specifications serve as executable descriptions that guide the ensemble at the macro-scale, while evolving virtual potentials continuously perform micro-scale controls. Given a macroscopic goal that may have been obtained after decomposition of higher-level mission objectives, satisfaction of this subgoal requires

Forward Clauses:

$F1 : Noise(T, A) \Rightarrow Trigger(T, A).$
$F2 : Motion(T, A) \Rightarrow Trigger(T, A).$

Backward Clauses:

$B1 : Interest(T_I, I, E) \Leftarrow Result(T_I, T_T, 0, I), Deliver(T_I, T_T, 1, I, E).$
$B2 : Deliver(T_I, T_T, N_D, I, E) \Leftarrow Delivered(T_I, T_T, N_D, I, E).$
$B3 : Deliver(T_I, T_T, N_D, I, E) \Leftarrow Position(T_P, E, A), Position(T'_P, E', A'), E' \neq E,$
$\quad MoveTo(T_I, T_T, N_D, 0, \infty, E', A), Deliver(T_I, T_T, N_D, I, E).$
$B4 : Result(T_I, T_T, N_D, I') \Leftarrow EncodedImage(T_I, T_T, N_D, I), I' = Decode(I).$
$B5 : EncodedImage(T_I, T_T, N_D, I) \Leftarrow Trigger(T_T, A), T_I \leq T_T,$
$\quad MoveTo(T_I, T_T, N_D, 0, T_T + \Delta t_{sd}, E, A), TakeSnapshot(T_I, T_T, N_D, T_T + \Delta t_{sd}, A, I).$
$B6 : TakeSnapshot(T_I, T_T, N_D, D, A, I) \Leftarrow Snapshot(T_I, T_T, N_D, T_S, A, I), T_T \leq T_S, T_S \leq D.$
$B7 : MoveTo(T_I, T_T, N_D, D, E, A) \Leftarrow Position(T_P, E, A), T_P \leq D.$
$B8 : MoveTo(T_I, T_T, N_D, D, E, A) \Leftarrow Move(T_I, T_T, N_D, D, E, A).$

Replacement Ordering: (f denotes a fact, g, a goal and x denotes either)

$O1 : f : Position(t_P, e, \ldots) \prec f : Position(t'_P, e, \ldots)$ if $t_P < t'_P$.
$O2 : x : X(t_I, \ldots) \prec g : Interest(t'_I, \ldots)$ if $t_I < t'_I$.
$O3 : x : X(t_I, t_T, n_D, \ldots) \prec f : Result(t_I, t_T, n_D, \ldots)$ if $x : X \neq f : Result$.
$O4 : x : X(t_I, t_D, n_D, \ldots) \prec f : Deliver(t_I, t_D, n_D, \ldots)$ if $x : X \neq f : Deliver$.

Variables: T: time, D: snapshot deadline, A: area, E: ensemble, I: information, N: identifier
Constants: Δt_{sd}: relative snapshot deadline (max. delay from trigger event)

Fig. 2. Logical theory of the motivating scenario in Sect. 1.1 — The trigger condition is specified as forward clause $F1$. Backward clauses are evaluated only when a user or the reasoner injects a goal (or a new subgoal) that unifies with the conclusion of the clause. Orderings are specified to replace inferior facts/goals.

control of the micro-states of individual elements in the ensemble. In Fig. 2, forward and backward rules direct ensembles to accomplish a particular mission. Distributed algorithms optimize the formation objectives of these ensembles by moving or flying the UAVs/robots as a swarm to a specific location and generating appropriate facts (e.g.,*Position*) to satisfy corresponding subgoals (e.g., *MoveTo*).

This paper describes a compositional declarative approach that expresses user objectives in a distributive logic and further decomposes them during translation into real-world actions or quantitative algorithms. Atomic goals are either built-in predicates/functions of the logic or actions directly realized by cyber-physical devices. Non-atomic logical goals for which a corresponding algorithm exists trigger the execution of that algorithm. If no such algorithm exists, the system attempts a distributed proof to simplify and solve the goal as far as possible, potentially injecting new subgoals during the proof's inference. Since each ensemble element may have different capabilities and a limited local view, the continuous evaluation of solutions with respect to the partial order preservers scalability in the context of loosely coupled systems. For example, facts are disseminated in the network to compensate for heterogeneity in the node capabilities and other limitations such as resource and sensor failures.

In Fig. 2, a user injects an *Interest* goal into the system. By processing conditions from left to right, the reasoner attempts to solve a *Result* goal. It fails at first because a corresponding *Result* fact representing a solution does not exist yet. The local reasoner feeds the new *Result* subgoal into the local KB. Some

goals cannot be solved by means of the logical specification, but the local cyber-physical devices may handle the goal through real-world action: for example, by taking an image and adding it to the local KB. The *MoveTo* subgoal will generate a new *Position* fact when it is realized. The clause *B8* is of particular interest, since it initiates a change in the position of a swarm. In this unified approach, quantitative algorithms, e.g. potential-based optimization, realize the cyber-goal *Move*. As a result, a new *Position* cyber-fact will be generated if a swarm manages to move to the desired location within time constraints. Note that a goal can be matched with a fact anywhere in the network. The fully distributed reasoning process allows *Decode* to be solved at all ensemble elements, while *TakeSnapshot* can be only satisfied by ensemble elements with camera devices (UAVs in this scenario).

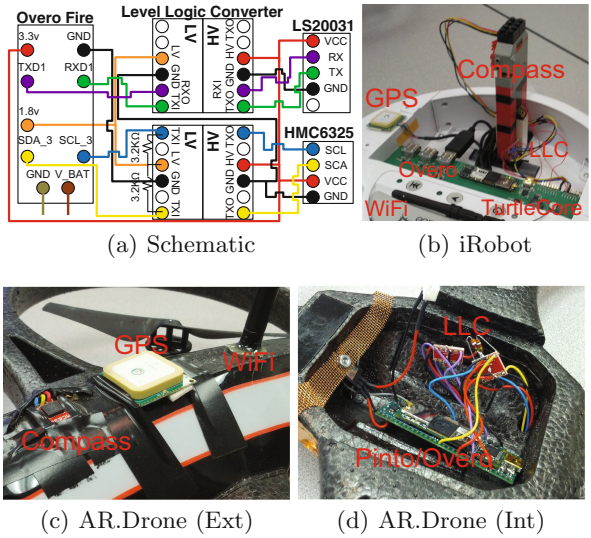
This work leverages the concept of a virtual potential [15] for micro-scale control of ensembles. Each ensemble element is driven by the desire to minimize its perception of, and hence its own contribution to, the virtual potential. In this scenario, the potential is designed to guide ensemble elements into position at a desired pair-wise distance in the formation. The potential also directs the elements to adapt when a new target location is selected as a macro-scale goal. The potential function declaratively captures the quality of a given swarm configuration as a weighted sum, $p = w_f p_f + w_l p_l$, where p_f and p_l are potentials corresponding to formation and desired location, respectively. As an example, the formation potential p_f should be minimized when a swarm creates a hexagonal lattice through the use of one of several possible distributed sensing grids as discussed in [28].

There are two ways to implement the potential-based approach. The most general approach, so far explored only in simulation, allows the user to specify a virtual potential function which is then solved by a generic distributed optimization algorithm. The alternative approach used in the experiments described here is based on the specification of the virtual potential indirectly through a force field, which is also known as *artificial physics* [28]. There are two kinds of forces, namely *formation forces* and *goal forces*, that define p_f and p_l in the above equation, respectively. Formation forces can be repulsive or attractive. Each ensemble element repels others closer than a desired distance R while attracting those further than R. Force balance must be maintained against p_f to map the goal force into p_l and keep the formation during movement to target location. Enforcement of an upper bound on the goal force (as explained in [28]) and tuning of the weights of composing potential achieves this balance.

4 Real World Deployment on Cyber-Physical Ensembles

The underlying cyber-framework was extended to support various abstractions and APIs for programming a heterogeneous testbed based on the Parrot AR.Drone 2.0 [22], iRobot Create [3] platforms, and various Android devices.

Heterogeneous Testbed: The AR.Drones and Creates were equipped with additional computing resources (Gumstix modules [12]) and sensors (GPS and



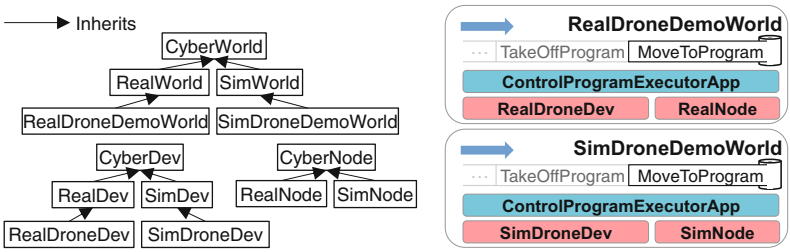
(a) Schematic (b) iRobot

(c) AR.Drone (Ext) (d) AR.Drone (Int)

Fig. 3. AR.Drones and Create robots were equipped with Gumstix Overo Fire [9] via a Pinto-TH [10] (AR.Drone) or TurtleCore [11] (Create) expansion board. Overo Fire is augmented with a GPS unit [20] via serial port and a digital compass [13] via I^2C. In between the sensors and the expansion board sit a pair of logic level converters [27].

digital compass) shown in Fig. 3 to increase their autonomy and improve capabilities such as localization in physical space. The Gumstix computer is also equipped with a WiFi antenna which provides communication via a peer-to-peer ad-hoc network. The testbed uses the Google Nexus S running CyanogenMod 7.2 with built-in digital compass, GPS, and camera as a standard Android device. As for the software side, the AR.Drones (firmware ver. 2.3.3) are controlled via WiFi, using a modified version of JavaDrone (ver. 1.3) [14] for sending commands and interpreting information from the drone's onboard sensors and video cameras. JavaDrone was modified to support video reception from the AR.Drone 2.0 by extracting video frames from the stream and passing them directly to the framework rather than attempting to decode them. This encoded frame data can be decoded later when it is deemed necessary, possibly on another device with more available computing resources. The Creates are controlled via a direct serial port connection from the Gumstix using the cyber-API which supports sending movement and audio commands.

Cyber-Framework Extension: Because it is written in Java, the cyber-framework runs directly on the ARM-based Gumstix module without modification. A variety of abstractions were implemented over the available devices to make writing applications easier and more consistent. To allow testing of the same code with both real and simulated scenarios, the cyber-framework supports two different device implementations for each device that share a common



(a) Class Hierarchy (b) Execution Hierarchy

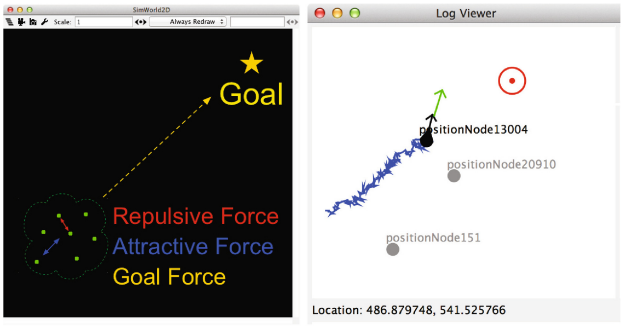
Fig. 4. Class/Execution hierarchy enables the execution of same application code without modification. The Executor cyber-application controls a device via a sequence of ControlPrograms such as MoveToProgram.

interface as seen in Fig. 4. Each device is assigned a behavior which can be real (controlling a real device in the physical world) or simulated (controlling a simulated device in a virtual world). Applications can be run entirely on a simulated platform such as Stage [8] or Mason [4] without any code modification.

As a further abstraction over hardware devices, virtual Camera and Position devices that can be used independently of the underlying hardware (e.g., regardless of whether the hardware in use is a drone, robot, or phone) were implemented. The Camera device supports a single command that takes a snapshot of a given area. The Position device reports its location, and supports a variety of movement methods (e.g., waypoint-based or velocity-based).

For lower level programming of drone/robot actions, a *ControlProgram* interface allows users to write programs that will run periodically to send new commands to devices based on their current situation. These programs, which provide notification to their caller via events on completion, can be asynchronously launched and terminated at will. An example of such a program is the *MoveToProgram* which implements waypoint-based movement. The *Position* device uses *MoveToProgram* to implement the physical movement. *MoveToProgram* runs with device-specific frequency and at each invocation checks the current position of the device. It then issues direct movement commands to move closer to the desired destination. If the destination has been reached, *MoveToProgram* terminates.

Mission Deployment: To demonstrate the autonomous outdoor mission described in Sect. 1.1 and further elaborated in Fig. 2, GPS was used for localization without infrastructural support such as motion-tracking camera setups. To compensate for errors in GPS readings and facilitate simpler position and trajectory calculations, the testbed used differential GPS (10 Hz) [1] and a Cartesian ENU (East-North-Up) coordinate system [2]. For the AR.Drone, which lacks the capability to reliably stop at a fixed position and wait for its GPS readings to stabilize, a trajectory prediction filter was implemented to sit between the GPS sensor and the Drone API. This filter uses a least squares regression to fit movement to a polynomial curve to predict future positions.



(a) Mason Simulation (b) Mission Log

(c) Final Deployment

Fig. 5. Mission is tested on simulation and further fine-tuned towards real deployment. In the mission log, the robot is represented by a pointed black dot. The blue path represents the trajectory so far. The green arrow is a force acting on the robot. The bullseye is the swarm's goal, and grey dots represent the robot's belief about the positions of other robots.

Figure 5 shows the progress towards real world deployment. A swarm mission can be run entirely on a simulated platform to evaluate the algorithm's correctness and performance. In Fig. 5(a), a swarm of seven mobile nodes forms a hexagonal lattice maintained by artificial forces. The formation force dictates a 9 m distance between each pair of nodes, while the goal force attempts to push the entire swarm towards a target location. In Fig. 5(b), the same code is then deployed on a real device and the trajectory is recorded for further debugging and fine-tuning since the simulation lacks fidelity to accurately model the physical environment (e.g., movement delay, wind). The demo mission is deployed on two Android phones, three AD.Drones, and four Creates in a 70×35-m parking lot in Fig. 5(c). One Android phone generates a fact that represents an observation, e.g., a sensed noise, which triggers the distributed execution of a mission with the user's location as target area.

Once the swarm has reached at the target location, AR.Drones take encoded video frames via their front-facing camera, and extract an encoded frame. Creates then decode the frame and transmit it back to the phones.

5 Related Work

Declarative techniques are increasingly used in networking, in particular in data-centric sensor networks and recently in information-centric approaches for wired and wireless networks. NDlog [21], a variant of Datalog, refocuses the programming tasks on high-level issues of networking by providing a declarative language and dataflow framework that executes the compiled specifications. Declarative querying of sensor networks [32] composes services on the fly and in a goal-driven fashion using the concept of semantic streams. Meld [5] extends the ideas from declarative sensor networks to modular robots, i.e., ensembles of robots, by providing an abstract view of a system as a single asset. The flexibility and efficiency of the declarative paradigm can be further improved by correlating the logical and physical properties of the system and by integrating the logical and quantitative approaches for programming NCPS.

References [6,30] studied decentralized control of robotic or aerial swarms with a limited view of the system caused by intermittent connectivity and resource constraint. In particular, Swarmorph [24] utilizes network morphing to create a specific global structure of a heterogeneous swarm for collective responses to different tasks as a single entity. Swarms of micro aerial vehicles [19,23] have been studied as a newly emerging class of mobile sensor networks. However, in most cases the high-level goal is controlled by the base station with high-quality vision-based localization. This control approach requires infrastructure and makes the swarm less robust to failure/disconnections of specific elements, unlike the fully distributed and autonomous control of fractionated systems proposed here.

Most applications of virtual potential fields have considered single agents or localized unstructured ensembles. A notable exception is [26] which develops a compositional approach that results in a single potential function that coordinates a complex mission. A similar approach to aggregate motion control is inspired by flock behavior based on a distributed behavioral model [25] in which each simulated bird navigates according to its local perception of the dynamic environment, the laws of simulated physics, and a set of programmed behaviors. The computational field model [31] is based on concurrent objects in an open distributed environment guided by a metric space defined by mass, distance, gravitational force, repulsive force, and inertia of objects. Its communication model is based on object migration, an atomic primitive that is too strong for the kinds of environments targeted by this paper.

6 Conclusions and Future Directions

Generalizing earlier work, this paper investigates the use of virtual potential functions to extend a logical framework that combines qualitative and quantitative paradigms. After the general approach, which is based on a generic reasoning and optimization framework, was studied in simulation, the distributed logical framework was deployed for the first time on a newly developed hybrid NCPS

testbed which consists of phones, robots, and drones. As a simplification, the approach describe here used an artificial-physics-based algorithm to implement the virtual potential instead of a generic distributed optimization algorithm. The deployment of the latter is in progress and has already been tested in simulations.

Adaptive wireless networks that can physically morph in response to mission requirements are an ideal application for the declarative approach proposed here because the distributed state space is too complex for traditional hard-wired algorithmic solutions, especially in the absence of strong assumptions on the environment. The case study in this paper illustrates the basic idea by combining robots, drones, and phones, but there is potential for other applications with newly emerging technologies such as pico-satellite constellations, networked balloons, and networks of buoys. In addition to physical control over position and orientation, there are other dimensions of control associated with wireless technologies such as transmission power control and beam forming that could be investigated in future work. A core challenge of all these distributed control problems is that the coordination required to find acceptable solutions must rely on the same network that is continuously morphed and potentially disrupted by unforeseen events. Hence, a conceptual framework and tools to evaluate the stability and robustness of such fractionated systems constitute another important and challenging direction for future work.

Acknowledgments. We thank Walter Alvarez, Kyle Leveque, and Bryan Klofas at SRI International for their contributions to building the testbed. Support from National Science Foundation Grant 0932397 (A Logical Framework for Self-Optimizing Networked Cyber-Physical Systems) and Office of Naval Research Grant N00014-10-1-0365 (Principles and Foundations for Fractionated Networked Cyber-Physical Systems) is gratefully acknowledged. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF or ONR.

References

1. Differential GPS. http://en.wikipedia.org/wiki/Differential_GPS
2. Geodetic Datum. http://en.wikipedia.org/wiki/Geodetic_system
3. iRobot Create. http://store.irobot.com/shop/index.jsp?categoryId=3311368
4. Mason multiagent simulation lib. http://cs.gmu.edu/eclab/projects/mason/
5. Ashley-Rollman, M.P., Goldstein, S.C., Lee, P., Mowry, T.C., Pillai, P.: Meld: a declarative approach to programming ensembles. In: IEEE International Conference on Intelligent Robots and Systems (2007)
6. Berman, S., Kumar, V., Nagpal, R.: Design of control policies for spatially inhomogeneous robot swarms with application to commercial pollination. In: ICRA, pp. 378–385. IEEE (2011)
7. Choi, J.-S., McCarthy, T., Yadav, M., Kim, M., Talcott, C., Gressier-Soudan, E.: Application patterns for cyber-physical systems. In: IEEE International Conference on Cyber-Physical Systems, Networks, and Applications, CPSNA'13 (2013)
8. Gerkey, B., Vaughan, R.T., Howard, A.: The player/stage project: tools for multi-robot and distributed sensor systems. In: 11th International Conference on Advanced Robotics (ICAR) (2003)

9. Gumstix. Overo Fire COM. https://www.gumstix.com/store/product_info.php?products_id=227
10. Gumstix. Pinto-TH. https://www.gumstix.com/store/product_info.php?products_id=239
11. Gumstix. TurtleCore. https://www.gumstix.com/store/product_info.php?products_id=280
12. Gumstix Inc. Gumstix. https://www.gumstix.com/
13. Honeywell Inc. HMC6352. http://magneticsensors.com/magnetometers-compasses.php
14. JavaDrone. http://code.google.com/p/javadrone/
15. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. Int. J. Rob. Res. **5**, 90–98 (1986)
16. Kim, J., Kim, M., Stehr, M.-O., Oh, H., Ha, S.: A parallel and distributed metaheuristic framework based on partially ordered knowledge sharing. J. Parallel Distrib. Comput. **72**(4), 564–578 (2012)
17. Kim, M., Stehr, M.-O., Kim, J., Ha, S.: An application framework for loosely coupled networked cyber-physical systems. In: IEEE/IFIP International Conference Embedded and Ubiquitous Computing, EUC'10, pp. 144–153 (2010)
18. Kim, M., Stehr, M.-O., Talcott, C.: A distributed logic for networked cyber-physical systems. Sci. Comput. Program. **78**, 2453–2467 (2013)
19. Kushleyev, A., Mellinger, D., Kumar, V.: Towards a swarm of agile micro quadrotors. In: Robotics: Science and Systems, July 2012
20. LOCOSYS Technology Inc. LS20031. http://www.locosystech.com/product.php?zln=en&id=20
21. Loo, B.T., Condie, T., Garofalakis, M., Gay, D.E., Hellerstein, J.M., Maniatis, P., Ramakrishnan, R., Roscoe, T., Stoica, I.: Declarative networking. Commun. ACM **52**(11), 87–95 (2009)
22. Parrot SA. AR. Drone 2.0. http://ardrone2.parrot.com/
23. Purohit, A., Sun, Z., Mokaya, F., Zhang, P.: Sensorfly: controlled-mobile sensing platform for indoor emergency response applications. In: International Conference on Information Processing in Sensor Networks (IPSN) (2011)
24. O'Grady, M.D.R., Christensen, A.L.: Swarmorph: multi-robot morphogenesis using directional self-assembly. IEEE Trans. Robot. **25**(3), 738–743 (2009)
25. Reynolds, C.W.: Flocks, herds and schools: a distributed behavioral model. In: 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '87, pp. 25–34 (1987)
26. Sentis, L., Mintz, M., Ayyagari, A., Battles, C., Ying, S., Khatib, O.: Large scale multi-robot coordination under network and geographical constraints. In: Proceedings of the IEEE International Symposium on Industrial, Electronics (2009)
27. Spark Fun Electronics. Logic Level Converter. https://www.sparkfun.com/products/8745
28. Spears, W.M., Heil, R., Zarzhitsky, D.: Artificial physics for mobile robot formations. In: IEEE Internatioanl Conference on Systems, pp. 2287–2292 (2005)
29. Stehr, M.-O., Talcott, C., Rushby, J., Lincoln, P., Kim, M., Cheung, S., Poggio, A.: Fractionated software for networked cyber-physical systems: research directions and long-term vision. In: Agha, G., Danvy, O., Meseguer, J. (eds.) Formal Modeling: Actors, Open Systems, Biological Systems. LNCS, vol. 7000, pp. 110–143. Springer, Heidelberg (2011)
30. Swarmanoid Project. http://www.swarmanoid.org

31. Tokoro, M.: Computational field model: toward a new computing model/methodology for open distributed environment. In: Proceedings of the 2nd IEEE Workshop on Future Trends in Distributed Computing Systems (1990)
32. Whitehouse, K., Zhao, F., Liu, J.: Semantic streams: a framework for composable semantic interpretation of sensor data. In: Römer, K., Karl, H., Mattern, F. (eds.) EWSN 2006. LNCS, vol. 3868, pp. 5–20. Springer, Heidelberg (2006)