Coordinated control strategies for networked vehicles: an application to Autonomous Underwater Vehicles

João Borges de Sousa and Fernando Lobo Pereira
Departamento de Engenharia Electrotécnica e de Computadores
Faculdade de Engenharia da Universidade do Porto
Rua Dr. Roberto Frias
4200-465, Porto
Portugal
{jtasso, flp}@fe.up.pt

Abstract

The specification and design of coordinated control strategies for networked vehicles and systems is discussed. A strategy to find the local minimum of an oceanographic scalar field with networked autonomous underwater vehicles (AUV) is presented. The strategy consists in coordinating the motions of the AUVs to implement a modified version of the simplex optimization algorithm. In the original algorithm, the scalar field is given by a function. In the modified version, the scalar field is given by the phenomenon itself. The AUVs sample the phenomenon to calculate the directions of descent, and to minimize the phenomenon along each direction of descent. The strategy is discussed in the more general context of coordination and control of networked vehicles and systems.

1 Introduction

Today, and partly due to our involvement in the design and implementation of networked vehicles and systems, we have a better understanding of the issues arising in the coordination and control of these systems [9]. In order to encompass all of these issues we need to consider potential applications from diverse fields, each presenting its own unique challenges to our efforts to generalize and formalize. Recognizing that we are still taking the first steps in this direction, we have been actively involved with the potential users of networked vehicles and systems. This enables us to envision concepts for the operation of systems which could not have been imagined before.

In this paper, we present some concepts for the operation of autonomous underwater vehicles, and formulate the corresponding coordination and control problems in the setting of dynamic optimization. In this setting, we express complex requirements, such as the disjunction of joint-state constraints and relative motion coordination, in terms of invariance and of level sets of value functions. We focus on mapping set-valued requirements, expressed within the language of set theory and logic, onto problem formulations where these requirements are expressed in terms of set-valued operations.

The paper is organized as follows. In section 2, we discuss a preview of a mission involving autonomous underwater vehicles to provide a concrete referent for our overview. In section 3, we discuss the patterns of coordination and control arising in this mission and show that they are quite general, and not specific to this application. In section 4, we motivate and illustrate our specification and design approach by reference to this mission. In section 5, we draw some conclusions and discuss future work.

2 Preview of an oceanographic mission

Let us start with a preview of an oceanographic mission conceivable in a near future. Imagine two teams of Autonomous Underwater Vehicles (AUVs) that must coordinate their motions to find the local minimum of some oceanographic phenomenon.

- The first team, denoted as LPS, provides a Local Positioning System (LPS) service to other team. This system can be viewed as an underwater version of the Global Positioning Service (GPS).

- The second team, denoted as S, uses the LPS service for localization and provides a search service.

The Local Positioning System works as follows. Each LPS vehicle has a GPS receiver and an acoustic transponder – the vehicle is required to operate at the surface to receive the GPS signal. The transponder emits regularly, with a known frequency, an acoustic ping encoding the time t when it was emitted, and the name and location of the emitter at time t. The time t and the position of the emitter are given by the GPS receiver.

The AUVs from the S team are equipped with an acoustic system. This system detects the acoustic pings, and decodes them to extract the position and name of the emitter, and the time when the acoustic ping was emitted. This information together with the time of arrival of the ping is used to calculate the distance between the AUV and the emitter. To calculate its absolute position the AUV needs to receive pings from at least three sources – vehicles from LPS. In order to minimize the calculation error we require these three sources to be the vertices of a triangle. Due to attenuation, the LPS service is only available within a neighborhood P(t) of the LPS team.

The search service works as follows. The S team implements a modified version of the simplex optimization algorithm to find the local minimum of the phenomenon. Each vehicle has a suite of oceanographic sensors – to sample the phenomenon – and a low-bandwidth underwater acoustic communication system – to implement the search strategy in coordination with the rest of the team and with the LPS team.

In terms of motion coordination the S team assumes the role of the leader. The LPS team controls the motions of its vehicles in order to keep all the elements of the S team inside P(t). It does this based on the information exchanged with the S vehicles.

Hereafter we designate this mission preview as the oceanographic mission, or simply the mission.

3 Coordinated control of vehicles and systems

3.1 Why motion coordination?

From the analysis of the oceanographic mission we may infer some of the reasons why we need to coordinate the motions of heterogeneous vehicles:

Functional complementarity. It is generally the case that space is at premium in autonomous vehicles, whether for sea, air, or land applications. Moreover, sensing and sampling strategies may require the spatial distribution not only of sensors, but also of different components of the same sensing system. In both cases we need to distribute capabilities – multiple sensors or different components of the same sensor – among different vehicles.

Spatial-temporal distribution of services. Some services, for example communication services, have to be distributed among multiple vehicles to cover a given area.

Sensing and acting on the world. Sensing involves establishing spatial relations between the vehicle where the sensor is mounted, and the object or phenomena being sensed. The same happens with vehicles with the capability to act upon the world. One such example arises with Unmanned Combat Air Vehicles (UCAV), that are capable of launching attacks with missiles. For the attack to be effective, we require the UCAV to satisfy a predicate on the distance and azimuth from the target before a missile is launched.

Algorithmic motion specifications. At a certain level of abstraction, vehicles are points in the 3D space. The motion requirements for several applications, for example in the oceanographic mission, are expressed as an algorithm, that may, or may not, be implementable with the motions of those points.

3.2 Patterns of coordination and control

We design teams to provide services that cannot be provided by a single vehicle. This means that vehicles within a team, and teams within interacting teams, have to coordinate their actions – motions and the utilization of their capabilities – to provide those services. This is done according to patterns of coordination and control. For example, the vehicles in our oceanographic mission exhibit patterns of coordination and control that are quite general, as we will see.

Satisfaction of joint state and capability constraints. A service requires the satisfaction of predicates on the capabilities and on the relative motions of the vehicles providing the service.

Team as a specific entity. A team comes into existence through the coordinated actions of its vehicles. This means that, with respect to the other teams, each team acts as a single unit, thus engaging, as a single unit, in interactions with those teams. For example, the LPS follows the S team in order to keep S inside P(t).

Coordination of teams. Services may build on other services. In fact, we may need to recruit the utilization of several teams to deliver a service. This requires the nesting of services, of constraints, and of controllers. For example, the LPS and the S team jointly provide a search service. Note that the modes of coordination at this level are richer than the modes of coordination at the intra-team level. First, teams form spatial entities whose shape and evolution we may want to control. Second, it may be possible, and desirable, to transfer assets among teams.

Interactions with externalities. Teams are designed to interact with the world, and to intervene in the world through sensing and actuation. This means that we close some of the control loops with the external environment. This relates to the next issue.

Algorithmic based activities. Teams have to interact with other teams, or with the world. In the absence of models of the world that are based on the principles of physics, we have to write specifications, and their implementations, as algorithms. For example, the S team is seeking to find the local minima of an oceanographic phenomenon.

Mobility of links. In order to coordinate their activities, the vehicles within a team (e.g. the S team), and groups of teams (e.g. the S and LPS teams), interact among themselves, and with other entities. To do this, they establish and destroy links among them. This means that they form a system with an evolving structure. The structure evolves when links change. Hence, establishing or destroying a link is a control action that may result in a different behavior for the structure.

We can find the same patterns of interactions in other problem domains, for example in applications involving Unmanned Air Vehicles (UAV)s or Unmanned Combat Air Vehicles (UCAV) (see [8] for an extensive survey on applications of these vehicles). UAVs and UCAVs are in high demand for military, scientific and civilian applications. Military operations present the most challenging scenarios. For military operations UAVs are tasked to "dirty","dull", and "dangerous" operations. "Dirty" refers to reconnoitering areas that may be contaminated, "dull" applies to surveillance, zone interdiction or sentry duty, while "dangerous" is related to obvious threats, such as those posed by the suppression of enemy air defenses (SEAD). In SEAD missions we have spatial and temporal rendezvous where vehicles form teams. In zone interdiction missions we have teams of UCAVs that coordinate their motions to maximize the rate of coverage. In reconnaissance and target finding missions we have search-based algorithms with integration of data from different sensors mounted on different vehicles. Moreover, with Unmanned Combat Air Vehicles, we may want to emulate the behavior of fighter pilots, where all types of engagements are guided by algorithmic procedures, or tactics. Tactics, also called plays, are used in robotic games involving antagonistic teams of robots. The collection of all tactics, or plays, is called the *play-book*.

4 Illustration of the approach

In this section we motivate and illustrate our specification and design approach by reference to the oceanographic mission.

4.1 Introduction

The potentially rich behavior of networked vehicles and systems results from the way agents – vehicles, controllers, service providers, and devices – are connected, and also from the way connections among these agents evolve with time, i.e. from the control architecture. This is why we turn our attention to architectural and specification issues[1].

Our approach addresses the issue of formal specification, and treats the design problem as a refinement of the specification.

Consider the mission description from section 2. First, it is not *formal*, i.e., it lacks a vocabulary of relevant concepts and rules that determine how they can be used. Second, it is not *complete*, i.e., it does not contain all components, connections, etc., intended to be true at all levels of detail. Or, equivalently, from it we cannot assert which properties will hold in the implementation, and which properties should not be present in the implementation. Third, we cannot reason about the description, or prove facts about it.

We address these issues in the following way:

- We represent specifications as logical theories. We introduce a vocabulary of the relevant components and well formedness axioms that determine how they can be used. We use set-theoretic constructs that are amenable to mathematical manipulation at the design stage.

- We write open specifications for components. These specify the component itself, and not the complete system containing it.

[1] The architectures of large systems are often described by a hierarchy of related architectures. A hierarchy of architectures is a linear sequence of individual architectures that may differ with respect to the number of components and connections among them.

The current level of informality is one of the problems with architectural design. For example, quite often there are no formal mappings between adjacent architectures in the hierarchy. Hence, it is not possible to assert that one architecture is an implementation of a more abstract one. To be able to answer this question we need to define equivalent behaviors, and we need to study under which conditions are behaviors preserved under those mappings. This means that we need more than syntactic checks; we need to check for the semantic properties of an architecture.

This problem has been addressed by the computer science community under the designation of "architecture refinement" (see for example, [14]).

- We define invariants, that we require the implementation to satisfy. The invariant captures the essence of what makes an implementation correct. In practice the invariants define the set of behaviors that satisfy the predicates associated with the specified requirements.

We formulate the coordination and control problems in the setting of dynamic optimization. In this setting we express complex requirements, such as the disjunction of joint-state constraints and relative motion coordination, in terms of invariance, of level sets of value functions, and of reachability. We focus on mapping the specification, expressed within the language of set theory and logic, onto problem formulations where these specifications are expressed in terms of set-valued operations. In doing this, we are able to derive conditions under which the invariants will be true, and synthesize controllers that ensure invariance. Basically, the design is a refinement of the specification.

4.2 Specification

In this section we introduce the main concepts and sketch the specification for our mission.

4.2.1 Main concepts

We use simple concepts for specification: *components* and *connectors*. A component has an internal structure, and a port. A connector has two or more ports, and specifies how to components can be linked together. We can build components from other components, using connectors. This allows for hierarchical organization. A particular arrangement of components and connectors is termed a *configuration*. We impose rules on the way components are connected. This defines a *configuration style* – a vocabulary of design elements, well formedness constraints that determine how they can be used, and a semantic definition of components associated with the style. The interface of a component defines available services and conditions under which the service is provided. We connect components with connectors defining relations among the interfaces of those components. Components, interfaces, and connectors are treated as *first-class objects* – i.e., they have a name and they are refinable. Abstract architectural objects can be decomposed, aggregated, or eliminated on a concrete architecture. The semantics of components is not considered part of an architecture, but the semantics of connectors is.

4.2.2 Intra and inter-team specification

Algorithm. The S team implements a modified version of the simplex algorithm that is described next. We could have used another algorithm. This one suffices to illustrate our approach in spite of its simplicity.

Consider, for the sake of simplicity, a scalar field $f(x) : \Re^2 \rightarrow \Re$ evolving in the horizontal plane with a unique local minimum in the region of interest. We are interested in finding this minimum.

At the beginning of each iteration i, we have three points A_i, B_i, C_i, and three values of the scalar field $f(A_i), f(B_i), f(C_i)$, where we have labelled the points so that $f(A_i) \leq f(B_i) \leq f(C_i)$. The sequence of computations is described next.

Algorithm 1 (Modified simplex). *Set $i := 1$. Consider three points A_1, B_1, C_1 forming a triangle. Repeat until finding minimum.*

1. *Take the segment joining A_i and B_i. Define the midpoint of this segment as z_i. Define the cone $K(z_i)$ with apex at z_i of all the unit vectors from C_i to $z_i + \delta \times B$, where δ is a parameter and B the unit ball.*

2. *Define the set of feasible directions at z_i as $K(z_i)$.*

3. *Select one vector v from $K(z_i)$.*

4. *If, at z_i, and for any direction w such that $\langle w, v\rangle \geq 0$, the directional derivative of f along w is non-negative, i.e., $f'(z_i; w) \geq 0$, then stop. In this situation, the point of minimum is located within the triangle with vertices A_i, B_i, and C_i. If not then, starting at z_i, find the minimum of f in the direction of v. Denote the point where the minimum is attained as A_{i+1}. Then $f(A_{i+1}) \leq f(A_i) \leq f(B_i) \leq f(C_i)$.*

5. *Rename A_i and B_i as B_{i+1} and C_{i+1}.*

6. *Set $i := i + 1$.*

This algorithm is scalable with respect to the number of vehicles used to implement it. With one vehicle, the ideal implementation would require this vehicle to jump from point A_{i+1} to point z_{i+1} at the beginning at each new iteration. With two vehicles, the ideal implementation would require one of them to be at position z_{i+1} when the other reaches the point A_{i+1}.

Available assets. Consider the following assets.

1. A set $\mathcal{SV}$, of n_{SV} identical surface vehicles with a GPS receiver, a transponder, a radio, and an acoustic modem. The ranges of the radio, transponders, and acoustic modems are respectively r_r, r_t and r_a.

2. A set $\mathcal{AUV}_u$, of n_u identical AUVs with a Conductivity Temperature Depth (CTD) sensor, an acoustic modem, and a navigation acoustic system. The range of the acoustic modem is r_a.

3. A set $\mathcal{AUV}_s$, of n_s identical AUVs mounting the same devices as the surface vehicles, plus the sensor pack mounted on all the vehicles from $\mathcal{AUV}_u$. These multi-role vehicles may be assigned to the LPS, or to the S teams.

We use the surface vehicles to implement the LPS team, and the AUVs to implement the LPS, and S teams. The LPS team is composed of two sets of vehicles, $\mathcal{LPS}_{SV}$ and $\mathcal{LPS}_{AUV}$, where $\mathcal{LPS}_{SV} = \mathcal{SV}$, and $\mathcal{LPS}_{AUV} \subseteq \mathcal{AUV}_s$. The S team is also composed of two sets of vehicles, $\mathcal{S}_u$ and $\mathcal{S}_v$, where $\mathcal{S}_u = \mathcal{AUV}_u$, and $\mathcal{S}_v \subseteq \mathcal{AUV}_s$.

We label each vehicle in $S(LPS)$ with a number $i(j)$, where $i(j)$ ranges from 1 to $n_S(t)$ $(n_{LPS}(t)))$, and $n_S(t), (n_{LPS}(t))$ varies with time. We denote the (x,y,z) position of the $i-th(j-th)$ vehicle from $S(LPS)$ by $X_{S_i}(t), (X_{LPS_i}(t))$.

Distance function. In what follows we express the Euclidean distance between two points X, Y in $\Re^3$ as $d(X, Y)$.

First, we specify the behavior for each team, and then the required inter-team behavior.

LPS team. The LPS team provides a positioning service to other vehicles. At time t, the service is available at all locations X such that there are at least three vehicles from LPS within distance r_t – the range of the transponder – from X. The set of all such points is denoted P(t).

$$P(t) = \{X \in \Re^3 : \exists i, j, k \quad (k \neq i) \wedge (i \neq j) \wedge (j \neq k) \wedge (d(X_{LPS_j}, X) \leq r_t) \wedge (d(X_{LPS_i}, X) \leq r_t) \wedge (d(X_{LPS_k}, X) \leq r_t)\} \tag{4.1}$$

The vehicles in LPS must satisfy the following motion constraints[2].
Service constraints. At least three vehicles from LPS are required to form a triangle. We express a relaxed version of this requirement using the distance function d as follows:

$$\exists i, j, k \in LPS : (j \neq i) \wedge (i \neq k) \wedge (k \neq j) \wedge (d_1 \leq d(X_{LPS_i}, X_{LPS_j}) \leq d_2) \wedge (d_1 \leq d(X_{LPS_i}, X_{LPS_k}) \leq d_2 \wedge (d_1 \leq d(X_{LPS_k}, X_{LPS_j}) \leq d_2) \tag{4.2}$$

Structural constraints. The LPS vehicles have to coordinate their motions to satisfy the service constraints, and to follow the S team. This is why we require the LPS vehicles to form a communication network where every two distinct vehicles should be able to communicate between them. We express the requirement as graph connectedness. To express graph

[2]There are two distinct types of constraints. The ones required for the team to coordinate its operations and to maintain its integrity, let us call them the *structural constraints*, and the ones required for the team to provide services, let us call them the *service constraints*. The structural constraints have precedence over the service constraints. The violation of the former implies the collapse of the team, while the violation of the latter degrades the way the service is delivered. From the above we conclude that we are in the presence of two levels of dynamic behavior. The first one ensures that the structural constraints are an invariant set for the operation of the team, the *internal dynamics*. The second one ensures service delivery, the *external dynamics*. The two levels of dynamic behavior are obviously coupled. We want to control this coupling so that the team is able to respond to service requests as fast as possible. This property is called *flexibility*. The specification of both types of constraints should be scalable to accommodate the addition, or deletion, of vehicles to and from the team.

connectedness formally we need some terminology and notions from graph theory[3]. Define the graph T as follows. Each vehicle in LPS is represented by a vertex. There is an edge between two vertices whenever the distance between the corresponding vehicles is less than the radio communication range r_r. The communication constraints are expressed as follows:

$$\text{The graph T is connected} \tag{4.3}$$

S team. The vehicles from the S team implement a modified version of the simplex algorithm. The implementation of this algorithm requires permanent communication among the vehicles. Here, again, we need graph connectedness in a graph K defined as follows. To each vertex in V there corresponds a vehicle in S. There is an edge between two vertices whenever the distance between the corresponding vehicles is less than r_a. The constraint is expressed as:

$$\text{The graph K is connected} \tag{4.4}$$

LPS-S coordination. There are two coordination requirements:

1. The vehicles from the S team should remain inside the set P(t):

$$\forall i \in S : X_{S_i}(t) \in P(t) \tag{4.5}$$

2. The vehicles in both teams should be able to communicate among themselves. If r_a is the maximum range of communication we express this constraint as follows:

$$\forall t, \forall i \in S, \forall j \in LPS : min \quad d(X_{LPS_j}(t)), X_{S_i}(t)) \leq r_a \tag{4.6}$$

Task. Control the motions and services of the two teams to find the minimum of the temperature f in a given zone of the ocean in minimal time, and using the simplex algorithm.

4.2.3 Remarks

Now, rather informally, let us consider this specification in the light of the concepts introduced before, i.e., components and connectors.

Consider the S team. We specify this team as a component. The interface of the team includes, as outputs, the search service, the set $C_S(t)$ – the convex closure of the locations of the members of the team) – and the set $D_S(t)$ – the set of locations where other vehicles can communicate with this team – and, as inputs, a localization service for the whole team. The LPS team is treated analogously. The interface includes, as outputs, the positioning

[3]A graph G is a finite nonempty set V together with an irreflexive symmetric relation R on V. V is the vertex set. We denote by E the set of symmetric pairs in R. Each element in E is called and edge, and the set E is called the edge set of G. *A u-v walk* in G is an alternating sequence of vertices and edges of G, beginning with u and ending with v, such that every edge joins the vertices immediately preceding it and following it. Two vertices u and v in a graph G are connected if $u = v$, or if $u \neq v$ and a u-v path exists in G. A graph is *connected* if every two vertices of G are connected.

service, the set P(t), and the set $D_{LPS}(t)$ – the set of locations where other vehicles can communicate with the team – and, as inputs, the locations of vehicles requesting the LPS service.

Consider now the coordination conditions. These can be simply expressed as 1) $C_S(t) \subset P(t)$; 2) $D_{LPS}(t) \cap D_S(t) \neq \emptyset$[4]; 3) the locations of vehicles requesting the LPS service come from S, and 4) the positioning service for S comes from LPS. In fact, we are expressing the coordination conditions are a connector.

We can use the same approach for the specification of the intra-team behavior.

4.3 Coordination and control

In this section we discuss the control and coordination problems arising in this mission specification. We show how the specification can be mapped onto control formulations within the framework of dynamic optimization [1, 10, 4, 5, 7, 6, 11, 12, 13], and how the intra and the inter team coordination problems can be formulated as nested problems of invariance. Finally, we briefly discuss the conceptual solution methodology.

The S team plays the role of the leader in the coordinated operation of both teams. The leading role stems from the implementation of the modified simplex algorithm – the reason why we are combining both teams. Conditions 4.3, 4.2 and 4.4 are invariance requirements respectively for the vehicles of the S team, and for the vehicles of the LPS team. Likewise, the coordination conditions 4.5 and 4.6 are an invariance requirement for the LPS team.

4.3.1 Implementation of the simplex algorithm

To each iteration i, there corresponds a set of data points A_i, B_i, C_i, and to each data point, there corresponds a value of the function f we are trying to minimize. Each iteration starts with the discovery, at time t_{i+1}, of the next point A_{i+1} by some vehicle k from S. At t_{i+1}, the position of vehicle k coincides with A_{i+1}, i.e. $X_{S_k}(t_{i+1}) = A_{i+1}$.

Consider we are at the beginning of iteration i. The problem is now to allocate vehicles to motion patterns. A scalable implementation of this algorithm requires two basic motion patterns for each iteration. One, let us call it the *leader pattern*, requires one vehicle, the leader, to enter the cone of descent directions $K(z_i)$ at the apex z_i, and to move in one of those directions with maximum speed until a maximum is found. The other, let us call it the *competition for leadership pattern*, requires the other vehicles to compete to become the leader in the next iteration. The competition consists in finding the vehicle that will reach, in minimal time, the apex z_{i+1} of the cone $K(z_{i+1})$ of descent directions for iteration $i + 1$.

The simplex algorithm is implemented as the solution of three problems.

Find the leader. Let $T_k^*(i)$ be the minimum time required for the vehicle k to reach a ball B of radius δ centered on z_i – the midpoint between A_i and B_i and the apex of cone $K(z_i)$ –

[4] This is basically a re-statement of condition 4.6

and starting at its current position $X_{S_k}(t_i)$. Then, the leader for the next iteration is given by:

$$argmin_k\{T_k^*(i) : X_{S_k}(T_k^*(i)) \in B_\delta(z_i)\} \tag{4.7}$$

Actually we want the vehicle to penetrate the cone. Hence, we have to add the following constraint to the previous minimization.

$$\dot{X}_{S_k}(T_k^*(i)) \in K(z_i) - \{z_i\}$$

Motion of the leader. This is basically a tracking problem with state constraints.

Competition for leadership. This is simultaneously a tracking and a reachability problem with state constraints. Let us assume that, at each time $t \geq t_{i+1}$, the position of the leader is known, and let us call it $\overline{A}_{i+1}$. Define $\overline{z}_{i+1}$ and $\overline{K}(\overline{z}_{i+1})$ as before. The objective for each vehicle, with the exception of the current leader, is to move with respect to cone $\overline{K}(\overline{z}_{i+1})$ with apex $\overline{z}_{i+1}$ in such a way that it can become the leader in the next competition. In addition all of these vehicles have to satisfy the invariance condition 4.4.

4.3.2 Inter-team motion coordination

Consider the set $C_S(t)$ given by the convex closure of the locations of the vehicles from the S team. The coordination and control problems for the LPS team consist in finding a controller such that the communication, service, and structural constraints, given respectively by equations 4.6, 4.2 and 4.3, are satisfied simultaneously, and the following is true:

$$\forall t, C_S(t) \subset P(t) \tag{4.8}$$

This is basically a problem of controlled invariance, where the LPS team controls the motions and shape of the set P(t).

There are at least three basic formulations for this coordination and control problem: 1) constraining the motions of the LPS vehicles; 2) constraining the motions of the S vehicles; 3) constraining both the motions of the S and LPS vehicles.

The selection of the right formulation for a specific instance of this problem depends on the following considerations.

- *Information structures* – The last formulation requires a two-way information flow between the two teams. This comes for free in our example.
- *Controllability* – The first two formulations assume that one the teams is able to solve the problem – these are the non-cooperative formulations. This means that no additional constraints are placed on the motions of the vehicles from the other team. But it may not be possible to leave all the coordination burden to one of the teams. This is why we need to consider the third formulation – the cooperative formulation. In this case we are in the presence of the lack of local controllability of the motions of one team with respect to the motions of the other team.

In the next section we formulate this problem under the following assumptions: 1) the state of the team S is known and, 2) S does not cooperate with LPS, i.e., the S team evolves independently of the LPS team. This places the coordination and control burden on the vehicles of LPS team.

The S team broadcasts its state to make it available to the LPS team. The LPS team receives this information since it satisfies the communication condition 4.6. At each time t where $t_i < t < t_{i+1}$, the state of S is described by z_i and $K(z_i)$, $\overline{A_{i+1}}$ and $\overline{K}(\overline{z}_{i+1})$, $n_S(t)$, and $X_{S_i}(t)$ for all vehicles in S.

4.3.3 Intra and inter-team invariance

The coordination and control requirements are expressed as intra-team and intra-team invariance problems. In this section, we formulate both types of invariance problems in the framework of value functions as presented in [11, 12, 13].

S intra-team invariance. Consider the $i = 1, \ldots, n_{LPS}(t)$ vehicles from the LPS team. The dynamics of vehicle i are given by the following equation:

$$\dot{x}_{LPS_i}(t) = f_{LPS_i}(x_{LPS_i}(t), u_{LPS_i}(t)), \quad u_{LPS_i}(t) \in U_{LPS_i}(x_{LPS_i}(t)), \quad x_{LPS_i}(t) \in \Re^k \tag{4.9}$$

We can group the differential equations in 4.9 as follows:

$$\dot{x}_{LPS}(t) = f_{LPS}(x_{LPS}(t), u_{LPS}(t)), \quad u_{LPS}(t) \in U_{LPS}(x_{LPS}(t)), \quad x_{LPS}(t) \in \Re^{k \times n_{LPS}(t)}$$
$$x_{LPS}(t) = \{x_{LPS_1}(t), \ldots, x_{LPS_{n_{LPS}(t)}}(t)\} \tag{4.10}$$

The team must satisfy two types of state constraints: service constraints; and structural constraints. We specify these constraints as the disjunction of $i = 1, \ldots, I$ sets of state constraints in $X = \Re^{k \times n_{LPS}(t)}$. Each set of state constraints i is expressed as an inequality:

$$\varphi_i(t, x_{LPS}(t)) \leq 1, \quad \varphi_i^0(x_{LPS}(0)) \leq 1, i = 1, \ldots, I \tag{4.11}$$

In our illustrative oceanographic mission we are interested in constraining just the (x,y,z) components of the state of all vehicles. The general problem formulation is the following:

Problem 4.1. *Find the largest invariant set $\mathcal{WI}_{LPS}(t)$ such that, if the initial position of the system lies in this set, it is always possible to find a control $u_{LPS}(t)$ that prevents the state of the system to violate condition 4.11.*

This is a problem of weak invariance since we are interested in the existence of at least one control law.

Consider the following value function:

$$V(\tau, z) = min_{u_{LPS}(t_0,\tau)} \Big\{ min\Big\{ max\big\{\{max_{t_0 \leq t \leq \tau} \quad \varphi_1(t, x_{LPS}(t))\}, \varphi_1^0(x_{LPS}(t_0))\big\}, \ldots,$$
$$max\big\{\{max_{t_0 \leq t \leq \tau} \quad \varphi_I(t, x_{LPS}(t))\}, \varphi_I^0(x_{LPS}(t_0))\big\}\Big\} | \quad x_{LPS}(\tau) = z) \Big\} \tag{4.12}$$

Then, following the arguments from [13], we can prove that the set $\mathcal{WI}_{LPS}$(t) is given by:

$$\mathcal{WI}_{LPS}(t) = \{z : V(t, z) \leq 1\} \tag{4.13}$$

The patterns of motion for the S vehicles are constrained to belong to this set.

We can obtain similar results for strong invariance if we substitute the first minimization in equation 4.12 by a maximization.

Inter-team invariance. Consider the inter-team motion coordination problems from the previous section.

Consider the $i = 1, \ldots, n_S(t)$ vehicles from the S team. The dynamics of vehicle i are given by the following equation:

$$\dot{x}_{S_i}(t) = f_i(x_{S_i}(t), u_{S_i}(t)), u_{S_i}(t) \in U_{S_i}(x_{S_i}(t)), x_{S_i}(t) \in \Re^k \tag{4.14}$$

Analogously, the motions of the S team vehicles are described by the following differential equation:

$$\begin{aligned} \dot{x}_S(t) = f_S(x_S(t), u_S(t)), \quad u_S(t) \in U_S(x_S(t)), \quad x_S(t) \in \Re^{k \times n_S(t)} \\ x_S(t) = \{x_{S_1}(t), \ldots, x_{S_{n_S(t)}}(t)\} \end{aligned} \tag{4.15}$$

The S team also satisfies service and structural constraints. The corresponding invariant set is $\mathcal{WI}_S$(t).

Let us express the coordination constraints. Consider the set-valued map h:

$$h : X \rightarrow \Re^3 \tag{4.16}$$

We want the vehicles in S to satisfy this additional constraint. Note that, at time t, this constraint can also be expressed as P(t).

We extend the set-valued map h to a set-valued map $\overline{h}$ defined on the whole space $\Re^k$ as follows:

$$\overline{h}(x) : h(x) \times \Re^{k-3} \tag{4.17}$$

Let us define another set-valued map J, as the cross product of these $n_{LPS}(t)$ set-valued maps. In order to write this state constraint in the form of an inequality such as 4.11 we consider the following "distance" function $d_A(B)$ between the closed set A and the closed set B both in $\Re^k$:

$$d_A(B) := Max\{d_A(x) : x \in B\} \tag{4.18}$$

where $d_A(x)$ measures the Euclidean distance between the point x and the set A.

Now define:

$$\varphi_{LPS}(t, x_{LPS}(t)) = d_{\mathcal{WI}_S(t)}(J(x_{LPS}(t))) + 1 \tag{4.19}$$

We can express this additional constraint on the motions of the LPS vehicles as follows:

$$\varphi_{LPS}(t, x_{LPS}(t)) \leq 1 \tag{4.20}$$

In practice, the LPS team defines a state constraint for all the vehicles in S. According to the previous section, we want the S vehicles to satisfy this constraint without additional control effort from the elements of this team. Hence, we want to solve the following problem:

Problem 4.2. *Find the largest invariant set $\mathcal{WI}_{LPS_S}(t)$ such that if the initial positions of the LPS vehicles lie in this set, it is always possible to find controls $u_{LPS}(t)$ and $u_S(t)$ such that condition 4.20 is never violated.*

See [1] for a discussion of a related problem.

5 Conclusions

The specification and design of coordinated control strategies for networked vehicles and systems poses significant challenges to control, computation, and networking.

In this paper we focus on the issues of specification, and of coordinated control design. We sketch a specification framework where we introduce a vocabulary of relevant concepts and the rules that determine how they can be used, and where we can reason about the specification and prove facts about it. We define invariants that we require the implementation to satisfy – the invariant captures the essence of what makes an implementation correct. We sketch the formulation of the coordination and control problems in the setting of dynamic optimization. We focus on mapping the specification, expressed within the language of set theory and logic, onto problem formulations where these specifications are expressed in terms of set-valued operations.

We drawn the attention to the importance of methods and results from computer science and from computer engineering, and also to the fact that control engineering provides a rich pool of results for the, at least conceptual, synthesis of coordinated control strategies.

Future work involves defining refinement relations to map the specifications onto problem formulations, synthesizing control laws that are implementable, and formulating the problem of intra and inter-team dynamic reconfiguration as one of switching controls [3, 2, 15].

Acknowledgment: Research supported in part by Fundação da Ciência e Tecnologia project Cordyal and by Fundação das Universidades Portuguesas project COOP. The authors thank Professors Pravin Varaiya and Alexander Kurzhanski for stimulating discussions and valuable comments, insights and contributions.

References

[1] Jean-Pierre Aubin. *Viability theory.* Birkhauser, 1991.

[2] Martino Bardi and I. Capuzzo-Dolcetta. *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*. Birkhauser, 1997.

[3] I. Capuzzo-Dolcetta and L. C. Evans. Optimal switching for ordinary differential equations. *SIAM J. Control and Opt.*, 22(1):143–161, 1984.

[4] F. H. Clarke. *Optimization and Nonsmooth Analysis*. SIAM, 1990.

[5] F. H. Clarke. A proximal characterization of the reachable set. *Systems & Control Letters*, 27(3):195–7, 1996.

[6] F. H. Clarke, Y.S. Ledyaev, and A.I. Subbotin. The synthesis of universal feedback pursuit strategies in differential games. *SIAM Journal on Control and Optimization*, vol.35(no.2):552–61, 1997.

[7] F. H. Clarke and P. R. Wolenski. Control of systems to sets and their interiors. *Journal of Optimization Theory and Applications*, 88(1):3–23, 1996.

[8] D. Van Cleave. Trends and technologies for uninhabited autonomous vehicles. In T. Samad and G. Balas, editors, *Software-Enabled Control: Information Technology for Dynamical Systems*. IEEE Press/John Wiley and Sons, 2002.

[9] J. Borges de Sousa and Raja Sengupta. Cdc tutorial on autonomous and semi-autonomous networked multi-vehicle systems. December 2001.

[10] F. H. Clarke et. al. *Nonsmooth Analysis and Control Theory*. Springer, 1998.

[11] A. N. Krasovskii. *Control under lack of information*. Birkhauser, 1995.

[12] A. B. Kurzhanskii. *Advances in nonlinear dynamics and control : a report from Russia*. Birkhauser, 1993.

[13] A. B. Kurzhanskii. *Ellipsoidal calculus for estimation and control*. Birkhauser, 1997.

[14] M. Moriconi, Xialei Qian, and R. A. Riemenschneider. Correct architecture refinement. *IEEE Transactions on Software Engineering*, 21(4):356–372, April 1995.

[15] F. Lobo Pereira and J. Borges de Sousa. A differential game with graph constrained dynamic switching strategies. In *Proceedings of Decision and Control Conference*. IEEE, 2001.