Motion Description Languages for Multi-Modal Control in Robotics

Magnus Egerstedt

magnus@ece.gatech.edu
Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332

Abstract. In this paper we outline how motion description languages provide useful tools when designing multi-modal control laws in robotics. Of particular importance is the introduction of the description length as a measure of how complicated a given control procedure is. This measure corresponds to the number of bits needed for coding the input string. Description length arguments can furthermore be invoked for selecting sensors and actuators in a given robotics application, thus providing a unified framework in which a number of major areas of robotics research can coexist.

1 Introduction

When humans instruct each other how to carry out particular tasks, only a limited number of tokenized instructions are used. In contrast to this, classic control theory specifies a control action to be carried out at each time instant. But, in a number of applications, such as semi-autonomous service robots for industrial and domestic use, intelligent appliances, and communication constrained embedded and/or teleoperated devices, the control procedures have a natural, linguistic flavor. In this paper we take the point of view that such tokenized instructions are useful, not only in particular robotics applications, but also for understanding how computer generated inputs to a robotics system should be defined, selected, and coded in order to minimize the number of bits transmitted from the computer to the robot. This should be achieved while guaranteeing that the system meets its specifications. To this end, an information theoretic approach to control theory will be developed, serving as a useful tool not only for source coding of control signals, but also for describing how symbolic instructions should be interpreted and operated on by continuous systems. Questions concerning what sensors and actuators to use in a given robotics application can be addressed quite elegantly within this framework as well.

In order to understand the interactions between these two heterogeneous components, i.e. between the symbolic computer programs and the continuous device dynamics, different *hybrid architectures*, serving as abstractions between continuous and discrete control, have been suggested. In [6] a general

model for such hybrid systems is proposed:

$$\begin{aligned}\dot{x} &= f(x, y, v(\lfloor p \rfloor)) \\ \dot{p} &= g(x, y, v(\lfloor p \rfloor)) \\ y &= h(x, v(\lfloor p \rfloor)),\end{aligned}$$

where x is the continuous state of the system and y is the measured output signal. Moreover, v is the symbolic input string from the motor control program, and the evolution of the scalar p triggers the reading of the string v. Here $\lfloor \cdot \rfloor$ denotes the floor operator, and g is assumed to be nonnegative for all arguments.

In this paper we model the way linguistic control signals affect mechanical devices on this form, and we will combine this model with the notion of a *motion description language* (MDL), which refers to a framework for device control, as proposed for example in [5,14,21]. By combining these two ideas we will construct interpreter mechanisms for generating meaningful control commands from symbolic inputs, and the outline of this paper is as follows: In Section 2, the main objects of study, i.e. the motion description languages, will be introduced for representing strings of idealized motions. Section 3 will consequently investigate how to design control laws using the MDL formalism. A cost criterion for evaluating the control laws will be introduced, corresponding to the description lengths of the control procedures. This cost can be interpreted as the number of bits needed for uniquely coding a given control procedure. In Section 4 we will invoke a complexity argument, similar to that in [22], that provides guidelines for how to choose sensors and actuators for a given robotics application.

2 Motion Description Languages

Given a finite set, or alphabet, A, by $A^\star$ we understand the set of all strings of finite length over A. There is a naturally defined binary operation on this set, namely the concatenation of strings, denoted by $\mathbf{a}_1 \cdot \mathbf{a}_2$, i.e. $\mathbf{a}_1 \cdot \mathbf{a}_2 \in A^\star$ if $\mathbf{a}_1, \mathbf{a}_2 \in A^\star$. Relative to this operation, $A^\star$ is a semigroup. If we include the empty string in $A^\star$ it becomes a monoid, i.e. a semigroup with an identity, and a *formal language* is a subset of a free monoid over a finite alphabet.

Now, by a *motion alphabet* we mean a possibly infinite set of symbols representing different control actions that, when applied to a specific robot, define segments of motion. A MDL is thus given by a set of symbolic strings that represent idealized motions, i.e. a MDL is a subset of a free monoid over a given motion alphabet. Particular choices of MDLs become meaningful only when the languages are defined relative to the physical robot that is to be controlled. In this paper, we let the robot dynamics be given by

$$\begin{aligned}\dot{x} &= F(x, u) \\ y &= H(x),\end{aligned}$$

where $u \in U$ is a control signal, $x \in X \subset \Re^n$ is the state of the system, and H is a measurable mapping from X to Y assigning an output value to each point $x \in X$.[1]

Now, it is quite standard, when structuring the navigation task for autonomous mobile robots, to decompose the task into basic building blocks, e.g. *behaviors* [2,7], and in this paper we refer to such building blocks as *modes*. Each mode should contain a description of the preplanned setpoints (generated by the high-level mission planner), of the way the robot should react to state and environmental changes (reactive control), and of when a new mode of operation should guide the behavior of the robot (e.g. specify the transition or arbitration rules). If we formalize these observations, following the development in [5,14], we say that a mode is given by a triple (u, k, ξ), where $u \in U$ is an *open-loop component*, $k : Y \times U \to U$ is a *closed-loop component*, and $\xi : Y \to \{0, 1\}$ is an *interrupt*.

If, at time t_0, the robot receives the input string $(u_1, k_1, \xi_1), \ldots, (u_p, k_p, \xi_q)$, then x evolves according to

$$\begin{aligned} \dot{x} &= F(x, k_1(y, u_1)); \quad t_0 \le t < T_1 \\ &\vdots \qquad\qquad\quad \vdots \\ \dot{x} &= F(x, k_q(y, u_q)); \; T_{q-1} \le t < T_q, \end{aligned}$$

where T_i denotes the time at which the interrupt ξ_i changes from 0 to 1.

The model of a trigger based hybrid system, as described in the introduction, can moreover reproduce this behavior if symbols are interpreted and operated on as follows: Let the input string to the trigger based hybrid system be such that

$$v(i) = (u_i, k_i, \xi_i), \; i \in \mathbb{Z}^+,$$

and let

$$\begin{aligned} \dot{x} &= f(x, y, v(\lfloor p \rfloor)) = f(x, y, (u_{\lfloor p \rfloor}, k_{\lfloor p \rfloor}, \xi_{\lfloor p \rfloor})) = F(x, k_{\lfloor p \rfloor}(y, u_{\lfloor p \rfloor})) \\ y &= h(x, v(\lfloor p \rfloor)) = h(x, (u_{\lfloor p \rfloor}, k_{\lfloor p \rfloor}, \xi_{\lfloor p \rfloor})) = H(x) \\ \dot{p} &= g(x, y, v(\lfloor p \rfloor)) = g(x, y, (u_{\lfloor p \rfloor}, k_{\lfloor p \rfloor}, \xi_{\lfloor p \rfloor})) = \begin{cases} 0 & \text{if } \xi_{\lfloor p \rfloor}(y) = 0 \\ \delta & \text{if } \xi_{\lfloor p \rfloor}(y) = 1, \end{cases} \end{aligned}$$

where δ is a unit impulse, $x(0) = x_0$, and $p(0) = 1$.

It is clear that we now have a construction that allows continuous machines to operate on linguistic inputs in a way that can be given a meaningful control theoretic interpretation.

2.1 Navigation Example

In order to make matters somewhat concrete, we illustrate these ideas with an example, found in [12]. What makes the control of mobile robots particu-

[1] At this point we will not specify U and Y further but, as will be shown later, different choices of input-output sets have a direct impact on how the control procedures should be selected and coded.

larly challenging is the fact that the robots operate in unknown, or partially unknown environments. Any attempt to model such a system must take this fact into account. We achieve this by letting the robot make certain observations about the environment, and we let the robot dynamics be given by

$$\begin{aligned} &\dot{x} = v,\ x, v \in \Re^2 \\ &y_1 = x,\ y_2 = c_f(x),\ y_1, y_2 \in \Re^2, \end{aligned}$$

where c_f is the contact force from the environment. The contact force could either be generated by tactile sensors in contact with the obstacle or by range sensors such as sonars, lasers, or IR-sensors.

Relative to this robot it is now possible to define the following two MDLs for distinguishing between the open-loop and the closed-loop cases.

Definition 1 (Open-Loop and Closed-Loop MDLs). Let the Open-Loop MDL, $\mathcal{L}_{ol}$, be given by the free monoid over the set

$$\big\{(u, k, \xi) \mid u \in \Re^2,\ k(y_1, y_2, u) = u,\ \xi : \Re^4 \to \{0, 1\}\big\},$$

where ξ is any measurable mapping from $\Re^4$ to $\{0, 1\}$. Consequently, let the Closed-Loop MDL, $\mathcal{L}_{cl}$, be given by the free monoid over the set

$$\big\{(u, k, \xi) \mid u = x_F,\ k(y_1, y_2, u) \in \{\kappa(u - y_1), Dy_2\},\ \xi : \Re^4 \to \{0, 1\}\big\},$$

where $\kappa > 0$ is a given constant, D is any linear mapping from $\Re^2$ to $\Re^2$, and x_F is the given, desired robot position.

By a point-to-point navigation task we understand the problem of moving the robot between given initial and final states in a safe way, and one straightforward question to investigate is how to design short strings of modes in $\mathcal{L}_{ol}$ and $\mathcal{L}_{cl}$ that achieve this? The reason why this is an interesting question is that the paths generated by $\mathcal{L}_{ol}$ on the robot are identical to those paths that are considered in the literature on the complexity of minimum time or shortest path algorithms for robot motion planning in dynamic environments [9].

This problem was studied in [12] and the following two theorems were proved:

Theorem 1 (Open-Loop Complexity). *(Egerstedt [12]) In a convex environment populated by N convex, polygonal obstacles with $M \geq 3$ vertices each, the length of the shortest string (in the worst case) in $\mathcal{L}_{ol}$ that drives the robot between x_0 and x_F is of order $\mathcal{O}(NM)$.*

The proof of this theorem consists of establishing tight bounds on the number of segments necessary for producing a piecewise linear path between x_0 and x_F. This path should furthermore not intersect the interior of any obstacle. To find these bounds is equivalent to finding the shortest string in $\mathcal{L}_{ol}$, since the only paths that can be generated on the particular robot under investigation are piecewise linear when using words in $\mathcal{L}_{ol}$.

In [12] it was furthermore shown how to construct a closed-loop control strategy that requires a lower number of instructions than $\mathcal{O}(NM)$, under the assumption that the contact force from an obstacle in contact with the robot is parallel to the outward normal of the surface of the obstacle.[2]

Theorem 2 (Closed-Loop Complexity). *(Egerstedt [12]) In a convex environment populated by N convex, polygonal obstacles, an upper bound on the length of the shortest (in the worst case) string in $\mathcal{L}_{cl}$ is of order $\mathcal{O}(N)$.*

The proof of Theorem 2 is constructive. When the robot is not in contact with an obstacle, we let $k(y_1, y_2, u) = \kappa(u - y_1)$, where $\kappa > 0$, as in Definition 1. On the other hand, when the robot is in contact with an obstacle it seems reasonable to follow the contour of that obstacle, as suggested in [17]. The control strategy that we propose for this guarantees that the robot reaches the unique global minimum (the point closest to x_F on the obstacle), while committing to a clockwise or counter-clockwise obstacle negotiation, before it leaves the contour of the obstacle. The multi-modal control sequence is thus an element in the set $\sigma_{GA} \cdot (\sigma_{OA} \cdot \sigma_{GA})^\star \subset \mathcal{L}_{cl}$, where GA and OA denotes "goal-attraction" and "obstacle-avoidance" respectively, and where $a^\star = \{\emptyset, a, aa, aaa, ...\}$. The individual modes $\sigma_{GA} = (u_{GA}, k_{GA}, \xi_{GA})$ and $\sigma_{OA} = (u_{OA}, k_{OA}, \xi_{OA})$ are furthermore given by

$$\begin{cases} u_{GA} = x_F \\ k_{GA}(y_1, y_2, u) = \kappa(u - y_1) \\ \xi_{GA}(y_1, y_2) = \begin{cases} 0 \text{ if } \langle y_2, x_F - y_1 \rangle \geq 0 \\ 1 \text{ otherwise} \end{cases} \end{cases}$$

$$\begin{cases} u_{OA} = x_F \\ k_{OA}(y_1, y_2, u) = cR(-\pi/2)y_2 \\ \xi_{OA}(y_1, y_2) = \begin{cases} 0 \text{ if } \langle y_2, x_F - y_1 \rangle < 0 \text{ or } \angle(x_F - y_1, y_2) < 0 \\ 1 \text{ otherwise.} \end{cases} \end{cases}$$

Here $R(\theta)$ is a 2×2 rotation matrix, $c > 0$, and $\angle(\alpha, \beta)$ denotes the angle between the vectors α and β.

An example of using this multi-modal control sequence is shown in Figure 1, and what this result means is that when the sensory information available to us is sufficiently abundant, fewer instructions are necessary in the feedback case than in the open-loop case. However, this way of measuring the complexity of input strings in terms of their *string lengths* is not a very natural complexity measure for the following two reasons:

1. The feedback instructions rely on sensory data, but string lengths do not capture the complexity associated with making these measurements in any meaningful way.

[2] Since no unique normal vector exists when x belongs to a vertex on the boundary on P, we let the output, y_2, take on any value in the *normal cone*, i.e. $y_2 \in N_P(x) = \{h \in \Re^2 \mid \langle h, y - x \rangle \leq 0, \ \forall y \in P\}$.

2. If the input strings were to be transmitted over a communication channel, the number of bits needed for coding the instructions should affect the complexity of the input strings, which is not reflected by the string lengths.

Based on these two observations, we instead introduce the *description length*, i.e. the number of bits needed for describing a given input string, as a natural complexity measure when designing multi-modal control procedures over motion description languages.

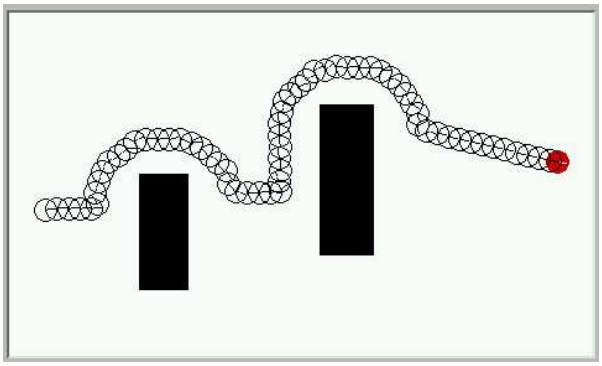

Fig. 1. A multi-modal input string in $\mathcal{L}_{cl}$ is used for negotiating two rectangular obstacles. Depicted is a simulation of a Nomadic Scout in the Nomadic `Nserver` environment.

3 Description Lengths

If we allow the control modes to contain mappings to or from sets of infinite cardinality, it is clear that the description lengths would be infinite as well. Therefore it is necessary to focus our attention on finitely describable control procedures. However, this is not really a restriction. Recall that the robot dynamics was given by $\dot{x}(t) = F(x(t), u(t))$, where u is the control signal and $x \in X \subset \Re^n$ is the state of the system. Based on the choice of actuators, u belongs to some set U. Since all real, physical stepper motors have finite range and resolution, and since the control signals are computer generated, U has finite cardinality, which is the case in the emerging area of *quantized control* as well [3,11,15]. It is also conceivable, as pointed out in [1], that we would like to further restrict the control signal, e.g. $u \in \{u_{\min}, u_{\max}\} = U$, resulting in somewhat more tractable design and verification processes. This would limit the cardinality of U further, and for analogous reasons, we let the observations made by the robot take on values in the finite set Y. This

implies that the input strings take on values in the set $\Sigma^\star$, where Σ is the finite set $\Sigma = U \times U^{Y \times U} \times \{0,1\}^Y$.

Now, assume that we are given a string of modes $\boldsymbol{\sigma} \in \Sigma^\star$. Then the number of bits needed for describing $\boldsymbol{\sigma}$ uniquely is given by the following definition [10,22]:

Definition 2 (Description Length). Consider the finite set Σ. We say that a word $\boldsymbol{\sigma} \in \Sigma^\star$ has description length

$$\mathcal{D}(\boldsymbol{\sigma}, \Sigma) = |\boldsymbol{\sigma}| \log_2(card(\Sigma)),$$

where $|\boldsymbol{\sigma}|$ denotes the length of $\boldsymbol{\sigma}$, i.e. the number of modes in the string, and $card(\cdot)$ denotes cardinality.

The description length thus tells us how complicated $\boldsymbol{\sigma}$ is, i.e. how many bits we need for describing it, and since $\Sigma = U \times U^{Y \times U} \times \{0,1\}^Y$ we directly see that a higher resolution sensor (e.g. laser-scanners) result in a larger Y than what is the case for a lower resolution sensor (e.g. sonars). A better sensor might thus make the control procedures significantly more complicated while only providing marginally better performance. This trade-off between complexity and performance is something that can be capitalized on when designing control laws as well as when choosing what sensors and actuators to use. The idea is simply to pick $(U, Y, \boldsymbol{\sigma} \in \Sigma^\star)$ in such a way as to make the robot behave satisfactory, while minimizing the description lengths of the control procedures.

3.1 Free-Running, Feedback Automata

In order to illustrate how the description length measure can be a useful tool in multi-modal control design, we focus our attention on the finitely describable aspects of finite state machines.

If we let X, U be finite sets, and let $\delta \in X^{X \times U}$, then we can identify (X, U, δ) with a *finite automaton* (see for example [16]), whose operation is given by $x_{k+1} = \delta(x_k, u_k)$. If we add another finite set Y and a mapping $\gamma \in Y^X$ to the definition, we get an *output* automaton $(X, Y, U, \delta, \gamma)$, where $x_{k+1} = \delta(x_k, u_k)$ and $y_k = \gamma(x_k)$.

However, in order to let finite automata read strings of control modes, the model must be modified in such a way that instruction processing is akin to the way in which differential equations "process" piecewise constant inputs. For this, a dynamical system called a *free-running, feedback automaton* (FRF-automaton), was introduced in [14]. The idea is to let such an automaton read an input from a given alphabet, and then advance the state of the automaton repeatedly (free-running property) without reading any new inputs until an interrupt is triggered. Additional structure is furthermore imposed on the input set to allow for feedback signals to be used. Hence a FRF-automaton is a free-running automaton whose input alphabet admits the structure $\Sigma =$

$U \times K \times \Xi$, where, as before, U is a finite set, $K = U^{Y \times U}$, and $\Xi = \{0,1\}^Y$. Hence, the input to a FRF-automaton is a triple (u, k, ξ), where $u \in U$, $k : Y \times U \to U$, and $\xi : Y \to \{0,1\}$.

Definition 3 (Free-Running, Feedback Automaton). Let X, Y, U be finite sets and let $\delta : X \times U \to X$, $\gamma : X \to Y$ be given functions. Let $\Sigma = U \times K \times \Xi$, where U is a finite set, $K = U^{Y \times U}$, and $\Xi = \{0,1\}^Y$. We say that $(X, \Sigma, Y, \delta, \gamma)$ is a free-running, feedback automaton whose evolution equation is

$$x_{k+1} = \delta(x_k, k_{l_k}(y_k, u_{l_k})), \; y_k = \gamma(x_k)$$
$$l_{k+1} = l_k + \xi_{l_k}(y_k),$$

given the input string $(u_1, k_1, \xi_1) \cdots (u_p, k_p, \xi_p) \in \Sigma^\star$.

It should be noted that the free-running property of the FRF-automata implies that they can, in general, be guided along a path using fewer instructions than the classical finite automata. However, since the input set to a finite automaton is simply the finite set U, while the input set to the corresponding FRF-automaton is of the form $U \times K \times \Xi$, the input set has a higher cardinality in the latter of these cases. As already pointed out, any reasonable measure of the complexity of a control procedure must take the size of the input space into account since the number of bits required to code a word over a given alphabet typically depends logarithmically on the size of the alphabet. (See for example [10].) This dependency is captured in a natural way if we define the specification complexity of a control task as the description length of the input sequence.

Definition 4 (Specification Complexity). Consider a FRF-automaton, A, with state space X and input set Σ. Let $\boldsymbol{\sigma}$ be the word of minimal description length over Σ that drives the automaton between two given states $x_0, x_f \in X$. We then say that the task of driving A between x_0 and x_f has specification complexity $\mathcal{C}(A, x_0, x_f) = \mathcal{D}(\boldsymbol{\sigma}, \Sigma)$.

3.2 Feedback Can Reduce the Specification Complexity

We here review the main results from [12] and [14] in order to see how the complexity of the instructions can be reduced when using landmark-based navigation. Since the cardinality of the input set depends on the size of the domain of the feedback mapping, a smaller domain can be expected to reduce the complexity. In order to make this observation rigorous, we need to introduce the notions of ballistic reachability and control-invariant reachability: A set $X_s \subset X$ is *ballistically reachable from* x if there exists a $u \in U$ such that $\delta(x, u^q) \in X_s$ for some $q \in \mathbb{Z}^+$. Furthermore, X_s is ballistically reachable from $X_t \subset X$ if there exists a $u \in U$ such that for all $x \in X_t$ it holds that $\delta(x, u^{q(x)}) \in X_s$ for some $q(x) \in \mathbb{Z}^+$. An element $x \in X_s \subset X$ is said to be *control-invariantly reachable in* X_s if it can be reached from all states in X_s without the trajectory leaving X_s.

Now, in order to compare purely open-loop control, i.e. when no observations are made, with a situation where sensory information is available, we must be able to generate open-loop motions on the FRF-automata. It is clear that the input sequence $\boldsymbol{\sigma}_{ol} = (u_1, k_{ol}, \xi_{ol}) \cdots (u_q, k_{ol}, \xi_{ol}) \in \Sigma^\star$, where $k_{ol}(u, y) = u \ \forall u \in U, \ y \in Y$, $\xi_{ol}(y) = 1 \ \forall y \in Y$ achieves this. However, this word has length q, and it is drawn from the input alphabet $\Sigma = U \times U^{Y \times U} \times \{0,1\}^Y$, and thus the description length is $\mathcal{D}(\boldsymbol{\sigma}_{ol}, \Sigma) = q \log_2(card(\Sigma))$. But, this is clearly not the result we would like to have. Instead we can restrict the input alphabet to be $\Sigma_{ol} = U \times \{k_{ol}\} \times \{\xi_{ol}\}$, which has cardinality $card(U)$. The description length of $\boldsymbol{\sigma}_{ol}$ is now $\mathcal{D}(\boldsymbol{\sigma}_{ol}, \Sigma_{ol}) = q \log_2(card(U))$, relative to the smaller input set Σ_{ol}.

Consider the connected, classical, finite automaton $A = (X, U, \delta)$. We recall that the *backwards eccentricity* of a state, $ecc(A, x)$, denotes the minimum number of instructions necessary for driving the automaton from any other state to x. (See for example [8].) We furthermore let the *radius* of A be given by

$$radius(A) = \min_{x \in X} ecc(A, x).$$

Now, consider the FRF-automaton $\tilde{A}$. If we let

$$\mathcal{C}(\tilde{A}, x) = \max_{x_0 \in X} \mathcal{C}(\tilde{A}, x_0, x),$$

then we directly get that

$$\mathcal{C}(A_{ol}, x) \geq radius(A) \log_2(card(U)),$$

where A_{ol} is the FRF-automaton $(X, Y, \Sigma_{ol}, \delta, \gamma)$, and A is the classical automaton (X, U, δ).

Theorem 3. *(Egerstedt and Brockett [14]) Assume that $card(U) \geq 2$. Suppose that $x_f \in X_f$, where X_f is an observable subset for the finite automaton A, i.e. it is possible to construct an observer that converges in a finite number of steps on the subset X_f. Assume that $card(\gamma(X_f)) < card(X_f)$ and $\gamma(X_f) \cap \gamma(X \backslash X_f) = \emptyset$. If X_f is ballistically reachable from $X \backslash X_f$, and x_f is control-invariantly reachable in X_f, then there exists a FRF-automaton $A_{FRF} = (X, Y, \Sigma', \delta, \gamma)$ such that*

$$\frac{\mathcal{C}(A_{FRF}, x_f)}{\mathcal{C}(A_{ol}, x_f)} \leq \frac{4 card(X_f)}{radius(A)}.$$

The proof of Theorem 3 consists of constructing one closed-loop instruction, consisting of two parts. The first part is given by a ballistic motion, i.e. a long open-loop motion, followed by an observer-based feedback instruction defined on the reduced set X_f, as seen in Figure 2(a). However, goals are seldom final goals. More often they tend to be intermediate goals in a grander scheme. This is for instance the case when mobile robots are navigating using landmarks.

3.3 Navigation Using Landmarks

It is clear that the premises on which the previous theorem is based are too restrictive to capture the *chained* structure that intermediary goals give rise to. Instead we need to extend the trajectories through a series of goal states. This can be achieved by assuming that we work with an automaton where subset-observers can be designed around different states, i.e. the intermediate goals. We also assume that the sets on which the observers are defined are ballistically reachable from each other. We could then use open-loop control for driving the system between these sets on the parts of the state space where the lack of sensory information prevents effective use of feedback. We compliment this with feedback controllers on the subsets where subset-observers can be constructed, as seen in Figure 2(b).

Theorem 4 (Navigation Using Landmarks). *(Egerstedt and Brockett [14]) Assume that* $card(U) \geq 2$. *Let the sets* $X_1, \ldots, X_n$ *be disjoint, observable subsets with cardinality less than or equal to* C, *where* $card(\gamma(X_i)) < C$, $i = 1, \ldots, n$, $\gamma(X_i) \cap \gamma(X \backslash X_i) = \emptyset$, $\gamma(X_i) \cap \gamma(X_j) = \emptyset$, $i \neq j$. *Let* $x_f \in X_n$ *be control-invariantly reachable in* X_n *and let* X_1 *be ballistically reachable from* x_0. *Assume that there exists intermediary goals* $x_i \in X_i$, $i = 1, \ldots, n-1$ *such that* x_i *is control-invariantly reachable in* X_i *and* X_{i+1} *is ballistically reachable from* x_i. *Then there exists a FRF-automaton* $A_{FRF} = (X, Y, \Sigma', \delta, \gamma)$ *such that*

$$\frac{\mathcal{C}(A_{FRF}, x_f)}{\mathcal{C}(A_{ol}, x_f)} \leq \frac{4nC}{radius(A)}.$$

One conclusion to be drawn from Theorem 4 is that the increase in description length, caused by the summation over many intermediate goals, can be counter-acted by making the sets where feedback is effective small. In the mobile robot case, this would correspond to using many easily detectable landmarks as a basis for the navigation system.

4 A Unified Approach to Control and Hardware Design

Another issue that can be tackled quite elegantly within the specification complexity context is how sensor selection for mobile robots affects the description lengths of the control procedures. In other words, we want to be able to determine, in a rigorous manner, which observations to make. But, we furthermore want to do this while simultaneously addressing a number of other potentially interesting questions in robotics. By scanning the proceedings from the *IEEE Conference on Robotics and Automation*, a rough taxonomy over five broad (not necessarily disjoint) areas of research can be identified:

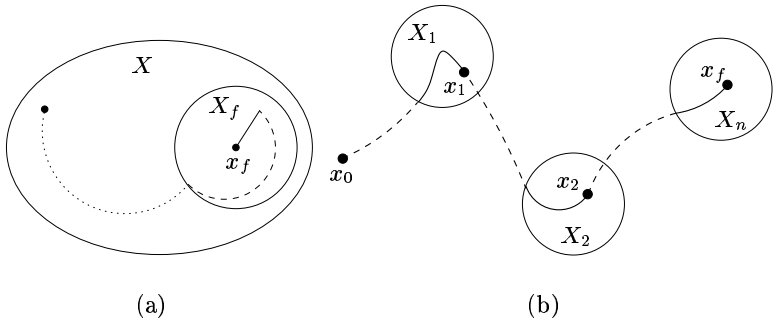

Fig. 2. In the left figure, the observer-based, single instruction, goal-finding procedure in Theorem 3 is depicted. The dotted line is the open-loop part of the evolution, the dashed line defines the part where the observer is converging, and the solid line is the last part of the evolution. The right figure shows the chained extension of this instruction over a collection of landmarks, as in Theorem 4.

i. Electro-mechanical design of robot platforms, sensors, and actuators;
ii. The development of rational methods for collecting, compressing, and analyzing sensory data;
iii. Deliberative, high-level mission planning;
iv. Control design for different classes of dynamical systems; and
v. The construction of appropriate software architectures and data structures for internal representations.

Since these areas of research have been developed somewhat in parallel, questions about software-hardware co-design have not been easy to address (or even to raise). What is novel in this paper is thus a unified model that incorporates some aspects of all of these five research areas and allows us to ask questions simultaneously about hardware design, mission planning, and robot dynamics.

Let us assume that we have access to a collection of sensors $\mathcal{Y}$ (with a corresponding $H : X \to Y$ for each $Y \in \mathcal{Y}$), and a set of possible actuators $\mathcal{U}$. One could thus ask the following question

$$(P): \min_{U \in \mathcal{U}, Y \in \mathcal{Y}} \left\{ \min_{\boldsymbol{\sigma} \in \Sigma^\star} |\boldsymbol{\sigma}| \log_2\big(card(\Sigma)\big) \right\},$$

subject to the constraint that $\boldsymbol{\sigma}$ makes the robot meet the given specifications. By solving P we would thus solve sensor and actuator selection as well as control design on both a low level ("What should the individual σ's look like?") and at a high level ("How should the strings of modes be chosen?") in a unified way. It should be noted that the solution is dependent on the dynamics of the robot as well as on the particular task the robot is asked

to carry out. It is furthermore clear that by solving P we directly address questions residing in research areas **i** and **iv** in the previously established taxonomy. Area **iii** is also touched upon since the open-loop component in the mode description can be thought of as corresponding to setpoints, or landmark locations, and by forming strings over Σ we thus provide a list of landmarks for the robot to move between. Area **ii** is also addressed by solving P through the construction of the output function $H : X \to Y$. Sensor fusion, virtual sensors, and feature extraction all correspond to designing different output functions, and the only remaining area untouched by P is thus area **v**. It can be argued that the use of motion description languages has implications for the software architecture as well [18] (the programmer should specify control triples directly), but we will let this aspect fall outside the scope of this paper.

4.1 Preliminary Results

At the present, we do not have the general solution to P. However, initial work has been conducted along these lines, and the findings are summarized in the following paragraphs:

As already shown in Section 3, the use of the description length as a measure of how complicated it is to specify control procedures can be put to work for answering a question of fundamental significance in control theory, namely that concerning the computational benefits of feedback. The reason why this work is promising is twofold: First, the result can be thought of as a special case of the sensor selection problem since it tells us whether or not sensors should be used at all. Secondly, the many visible and successful applications of feedback mechanisms at work testify to its effectiveness and over the years a variety of arguments have been advanced showing why, in particular settings, it is useful. The models commonly used bring to the fore considerations of sensitivity, uncertainty, etc, and to this list we have now added an argument for the usefulness of feedback in terms of the effect it has on reducing the description lengths of the control procedures.

In [13] it was investigated how to choose the resolution and the scope of the range sensors in order to manage the navigation task successfully at the same time as the description lengths should be kept as short as possible. The main theorem in that work states that if the robot dynamics is evolving on a bounded lattice, where "k-sensors" can detect neighboring states of a distance less than or equal to k from the present state of the system, then there is an optimal choice, $k^\star$, of such a range-sensor in terms of the number of bits needed for coding the control strategy. Furthermore, the number of bits needed for coding the control procedure was found to be convex in k (over the real numbers) which simplifies the design of optimization algorithms for finding $k^\star$ (over the positive integers) significantly, as shown in Figure 3.

This result sets the tone for the development of sensor selection algorithms in more realistic settings. For instance, different choices of k can be thought

of as different range sensors in robotics applications. A larger k, e.g. a laser scanner, can thus be compared to short-range sensors such as infra-red sensors, in terms of the specification complexity. The work also suggests that it should be possible to optimize not only over k, but also over some parameter that describes the accuracy and resolution of the actuators.

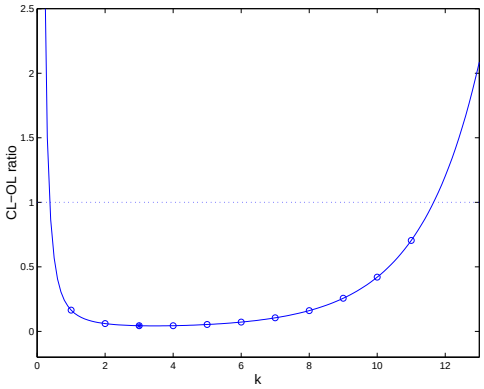

Fig. 3. The ratio between the closed-loop and open-loop specification complexity is shown as a function of k.

4.2 Further Issues

Previously we have seen that given $\boldsymbol{\sigma} \in \Sigma^\star$, a total number of $|\boldsymbol{\sigma}| \log_2(card(\Sigma))$ bits are needed for specifying $\boldsymbol{\sigma}$. Now, if we assume that we have been able to establish a probability distribution over Σ we can use optimal coding schemes, such as the Huffman code [19], for finding the shortest expected number of bits $l^\star(\Sigma)$ needed for coding an element drawn at random from Σ. Shannon's classic source coding theorem [23] tells us that $\mathcal{H}(\Sigma) \leq l^\star(\Sigma) < \mathcal{H}(\Sigma) + 1$, where the *entropy* $\mathcal{H}(\Sigma)$ is given by

$$\mathcal{H}(\Sigma) = -\sum_{i=1}^{card(\Sigma)} p_i \log_2 p_i .$$

Here the interpretation is that the control triple $\sigma_i \in \Sigma$ occurs with probability p_i, and it should be noted that a probability distribution over Σ corresponds to a specification of what modes are potentially useful.

But, to establish such a probability distribution over a structured set, such as the set of modes, is not a trivial task, and four possible routes can be identified:

1. Derive potentially useful modes directly using hybrid control synthesis tools [4,20];
2. Draw inspiration from existing, man-made multi-modal systems, e.g. autopilots for unmanned autonomous vehicles [24], or behaviors in behavior-based robotics [2,7];
3. Observe how human operators instruct mobile robots, or how biological systems are structured, and reconstruct the modes from experimental data; and
4. Apply reinforcement learning techniques in order to find potentially useful modes.

Once an appropriate model has been established for a given robot platform equipped with a collection of sensors and actuators, and a coding strategy has been decided on, we are ready to synthesize multi-modal control laws. As already argued for extensively in this document, we want to keep the mode descriptions short. In other words, assume that we have established a probability distribution over Σ and let $l_C(\boldsymbol{\sigma})$ denote the number of bits used for coding $\boldsymbol{\sigma} \in \Sigma^\star$ under coding strategy C. What one wants to achieve is thus to find $\hat{\boldsymbol{\sigma}} \in \Sigma^\star$ that minimizes $l_C(\boldsymbol{\sigma})$, while making sure that the system meet the specifications, such as driving the robot between given points while avoiding obstacles.

Acknowledgments

The support from NSF through the program EHS NSF-01-161 (grant # 0207411) is gratefully acknowledged. The author also wishes to thank Roger Brockett for many insightful comments in the emerging area of linguistic control.

References

1. R. Alur, J. Esposito, M. Kim, V. Kumar, and I. Lee. Formal Modeling and Analysis of Hybrid Systems: A Case Study in Multirobot Coordination. *Proceedings of the World Congress on Formal Methods*, LNCS 1708, pp. 212–232, Springer, 1999.
2. R.C. Arkin. *Behavior Based Robotics*. The MIT Press, Cambridge, MA, 1998.
3. A. Bicci, A. Margio, and B. Piccoli. On the Reachability of Quantized Control Systems. *IEEE Transactions on Automatic Control*, Vol. 47, No. 4, April 2002.
4. M.S. Branicky. Multiple Lyapunov Functions and Other Analysis Tools for Switched and Hybrid Systems. *IEEE Transactions on Automatic Control*, Vol. 43, No. 4, pp. 475–482, April 1998.
5. R.W. Brockett. On the Computer Control of Movement. In the *Proceedings of the 1988 IEEE Conference on Robotics and Automation*, pp. 534–540, New York, April 1988.
6. R.W. Brockett. Hybrid Models for Motion Control Systems. In *Perspectives in Control*, Eds. H. Trentelman and J.C. Willems, pp. 29–54, Birkhäuser, Boston, 1993.

7. R. Brooks. A Robust Layered Control System for Mobile Robots. *IEEE Journal of Robotics and Automation*, Vol. RA-2, No. 1, pp. 14–23, 1986.
8. F. Buckley and F. Harary. *Distance in Graphs.* Addison-Wesley Publishing Company, New York, 1990.
9. J. Canny. *The Complexity of Robot Motion Planning.* The MIT Press, Cambridge, MA, 1987.
10. T.M. Cover and J.A. Thomas. *Elements of Information Theory*, John Wiley & Sons, Inc., New York, 1991.
11. D.F. Delchamps. The Stabilization of Linear Systems with Quantized Feedback. *Proceedings of the IEEE Conference on Decision and Control*, Austin, TX, Dec. 1988.
12. M. Egerstedt. Linguistic Control of Mobile Robots. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Maui, Hawaii, Oct. 2001.
13. M. Egerstedt. Some Complexity Aspects of the Control of Mobile Robots. *American Control Conference*, Anchorage, Alaska, May, 2002.
14. M. Egerstedt and R.W. Brockett. Feedback Can Reduce the Specification Complexity of Motor Programs. To appear in *IEEE Transactions on Automatic Control*, 2002.
15. N. Elia and S.K. Mitter. Stabilization of Linear Systems With Limited Information. *IEEE Transactions on Automatic Control*, Vol. 46, No. 9, Sept. 2001.
16. A. Ginzburg. *Algebraic Theory of Automata.* ACM Monograph Series, Academic Press, New York, 1968.
17. J.E. Hopcroft and G. Wilfong. Motion of Objects in Contact. *The International Journal of Robotics Research*, Vol. 4, No. 4, pp. 32–46, 1986.
18. D. Hristu and S. Andersson. Directed Graphs and Motion Description Languages for Robot Navigation and Control. *IEEE Conference on Robotics and Automation*, May 2002.
19. D.A. Huffman. A Method for the Construction of Minimum Redundancy Codes. *Proceedings of IRE*, Vol. 40, pp. 1098–1101, 1952.
20. M. Johansson and A. Rantzer. Computation of Piecewise Quadratic Lyapunov Functions for Hybrid Systems. *IEEE Transactions on Automatic Control*, Vol. 43, No. 4, pp. 555–559, April 1998.
21. V. Manikonda, P.S. Krishnaprasad, and J. Hendler. Languages, Behaviors, Hybrid Architectures and Motion Control. In *Mathematical Control Theory*, Eds. Willems and Baillieul, pp. 199–226, Springer-Verlag, 1998.
22. J. Rissanen. *Stochastic Complexity in Statistical Inquiry.* World Scientific Publishing Company, River Edge, NJ, 1989.
23. C.E. Shannon. A Mathematical Theory of Communication. *Bell Systems Technical Journal*, Vol. 27, pp. 379–423, 1948.
24. C. Tomlin, G.J. Pappas, and S. Sastry. Conflict Resolution for Air Traffic Management: A Study in Multi-Agent Hybrid Systems. *IEEE Transactions on Automatic Control*, Vol. 43, No. 4, April 1998.